Python Tutorial

Learn Python

Python is a popular programming language.

Python can be used on a server to create web applications.

[Start learning Python now »](https://www.w3schools.com/python/python_intro.asp)

Learning by Examples

With our "Try it Yourself" editor, you can edit Python code and view the result.

Example

print("Hello, World!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_default)

**Click on the "Try it Yourself" button to see how it works.**

Python File Handling

In our File Handling section you will learn how to open, read, write, and delete files.

[Python File Handling](https://www.w3schools.com/python/python_file_handling.asp)

Python Database Handling

In our database section you will learn how to access and work with MySQL and MongoDB databases:

[Python MySQL Tutorial](https://www.w3schools.com/python/python_mysql_getstarted.asp)

[Python MongoDB Tutorial](https://www.w3schools.com/python/python_mongodb_getstarted.asp)

Python Exercises

Top of Form

Exercise:

Insert the missing part of the code below to output "Hello World".

("Hello World")

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_syntax1)

Bottom of Form

Python Examples

Learn by examples! This tutorial supplements all explanations with clarifying examples.

[See All Python Examples](https://www.w3schools.com/python/python_examples.asp)

Python Quiz

Test your Python skills with a quiz.

[Python Quiz](https://www.w3schools.com/python/python_quiz.asp)

Python Reference

You will also find complete function and method references:

[Reference Overview](https://www.w3schools.com/python/python_reference.asp)

[Built-in Functions](https://www.w3schools.com/python/python_ref_functions.asp)

[String Methods](https://www.w3schools.com/python/python_ref_string.asp)

[List/Array Methods](https://www.w3schools.com/python/python_ref_list.asp)

[Dictionary Methods](https://www.w3schools.com/python/python_ref_dictionary.asp)

[Tuple Methods](https://www.w3schools.com/python/python_ref_tuple.asp)

[Set Methods](https://www.w3schools.com/python/python_ref_set.asp)

[File Methods](https://www.w3schools.com/python/python_ref_file.asp)

[Python Keywords](https://www.w3schools.com/python/python_ref_keywords.asp)

[Python Exceptions](https://www.w3schools.com/python/python_ref_exceptions.asp)

[Python Glossary](https://www.w3schools.com/python/python_ref_glossary.asp)

[Random Module](https://www.w3schools.com/python/module_random.asp)

[Requests Module](https://www.w3schools.com/python/module_requests.asp)

[Math Module](https://www.w3schools.com/python/module_math.asp)

[CMath Module](https://www.w3schools.com/python/module_cmath.asp)

Python Introduction

What is Python?

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

* web development (server-side),
* software development,
* mathematics,
* system scripting.

What can Python do?

* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

* Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
* Python has a simple syntax similar to the English language.
* Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
* Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
* Python can be treated in a procedural way, an object-oriented way or a functional way.

Good to know

* The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
* In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

* Python was designed for readability, and has some similarities to the English language with influence from mathematics.
* Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
* Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Example

print("Hello, World!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_default)

Python Getting Started

Python Install

Many PCs and Macs will have python already installed.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

C:\Users\*Your Name*>python --version

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

python --version

If you find that you do not have Python installed on your computer, then you can download it for free from the following website: <https://www.python.org/>

Python Quickstart

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

C:\Users\*Your Name*>python helloworld.py

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

helloworld.py

print("Hello, World!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_helloworld)

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

C:\Users\*Your Name*>python helloworld.py

The output should read:

Hello, World!

Congratulations, you have written and executed your first Python program.

The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

C:\Users\*Your Name*>python

Or, if the "python" command did not work, you can try "py":

C:\Users\*Your Name*>py

From there you can write any python, including our hello world example from earlier in the tutorial:

C:\Users\*Your Name*>python  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("Hello, World!")

Which will write "Hello, World!" in the command line:

C:\Users\*Your Name*>python  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("Hello, World!")  
Hello, World!

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

exit()

Python Syntax

Execute Python Syntax

As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

>>> print("Hello, World!")  
Hello, World!

On this page

[Execute Python Syntax](https://www.w3schools.com/python/python_syntax.asp#execute_python_syntax)[Python Indentation](https://www.w3schools.com/python/python_syntax.asp#python_indentation)[Python Variables](https://www.w3schools.com/python/python_syntax.asp#python_variables)[Python Comments](https://www.w3schools.com/python/python_syntax.asp#python_comments)[Exercises](https://www.w3schools.com/python/python_syntax.asp#exercises)

Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:

C:\Users\*Your Name*>python myfile.py

Python Indentation

Indentation refers to the spaces at the beginning of a code line.

Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important.

Python uses indentation to indicate a block of code.

Example

if 5 > 2:  
  print("Five is greater than two!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_indentation)

Python will give you an error if you skip the indentation:

Example

Syntax Error:

if 5 > 2:  
print("Five is greater than two!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_indentation_test)

The number of spaces is up to you as a programmer, the most common use is four, but it has to be at least one.

Example

if 5 > 2:  
 print("Five is greater than two!")   
if 5 > 2:  
        print("Five is greater than two!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_indentation2)

You have to use the same number of spaces in the same block of code, otherwise Python will give you an error:

Example

Syntax Error:

if 5 > 2:  
 print("Five is greater than two!")  
        print("Five is greater than two!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_indentation2_error)

Python Variables

In Python, variables are created when you assign a value to it:

Example

Variables in Python:

x = 5  
y = "Hello, World!"

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_syntax_variables)

Python has no command for declaring a variable.

You will learn more about variables in the [Python Variables](https://www.w3schools.com/python/python_variables.asp) chapter.

Comments

Python has commenting capability for the purpose of in-code documentation.

Comments start with a #, and Python will render the rest of the line as a comment:

Example

Comments in Python:

#This is a comment.  
print("Hello, World!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_comment)

Top of Form

Exercise:

Insert the missing part of the code below to output "Hello World".

("Hello World")

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_syntax1)Bottom of Form

Python Comments

Comments can be used to explain Python code.

Comments can be used to make the code more readable.

Comments can be used to prevent execution when testing code.

Creating a Comment

Comments starts with a #, and Python will ignore them:

Example

#This is a comment  
print("Hello, World!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_comment1)

Comments can be placed at the end of a line, and Python will ignore the rest of the line:

Example

print("Hello, World!") #This is a comment

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_comment2)

A comment does not have to be text that explains the code, it can also be used to prevent Python from executing code:

Example

#print("Hello, World!")  
print("Cheers, Mate!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_comment3)

Multi Line Comments

Python does not really have a syntax for multi line comments.

To add a multiline comment you could insert a # for each line:

Example

#This is a comment  
#written in  
#more than just one line  
print("Hello, World!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_comment4)

Or, not quite as intended, you can use a multiline string.

Since Python will ignore string literals that are not assigned to a variable, you can add a multiline string (triple quotes) in your code, and place your comment inside it:

Example

"""  
This is a comment  
written in  
more than just one line  
"""  
print("Hello, World!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_comment5)

As long as the string is not assigned to a variable, Python will read the code, but then ignore it, and you have made a multiline comment.

Top of Form

Exercise:

Comments in Python are written with a special character, which one?

This is a comment

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_comments1)

Python Variables

Variables

Variables are containers for storing data values.

Creating Variables

Python has no command for declaring a variable.

A variable is created the moment you first assign a value to it.

Example

x = 5  
y = "John"  
print(x)  
print(y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables1)

Variables do not need to be declared with any particular *type*, and can even change type after they have been set.

Example

x = 4       # x is of type int  
x = "Sally" # x is now of type str  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables2)

Casting

If you want to specify the data type of a variable, this can be done with casting.

Example

x = str(3)    # x will be '3'  
y = int(3)    # y will be 3  
z = float(3)  # z will be 3.0

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_casting)

Get the Type

You can get the data type of a variable with the type() function.

Example

x = 5  
y = "John"  
print(type(x))  
print(type(y))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_type)

You will learn more about [data types](https://www.w3schools.com/python/python_datatypes.asp) and [casting](https://www.w3schools.com/python/python_casting.asp) later in this tutorial.

Single or Double Quotes?

String variables can be declared either by using single or double quotes:

Example

x = "John"  
# is the same as  
x = 'John'

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables7)

Case-Sensitive

Variable names are case-sensitive.

Example

This will create two variables:

a = 4  
A = "Sally"  
#A will not overwrite a

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_cs)

Python - Variable Names

Variable Names

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total\_volume). Rules for Python variables:

* A variable name must start with a letter or the underscore character
* A variable name cannot start with a number
* A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_ )
* Variable names are case-sensitive (age, Age and AGE are three different variables)

Example

Legal variable names:

myvar = "John"  
my\_var = "John"  
\_my\_var = "John"  
myVar = "John"  
MYVAR = "John"  
myvar2 = "John"

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variable_names_legal)

Example

Illegal variable names:

2myvar = "John"  
my-var = "John"  
my var = "John"

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variable_names_error)

Remember that variable names are case-sensitive

Multi Words Variable Names

Variable names with more than one word can be difficult to read.

There are several techniques you can use to make them more readable:

Camel Case

Each word, except the first, starts with a capital letter:

myVariableName = "John"

Pascal Case

Each word starts with a capital letter:

MyVariableName = "John"

Snake Case

Each word is separated by an underscore character:

my\_variable\_name = "John"

Python Variables - Assign Multiple Values

Many Values to Multiple Variables

Python allows you to assign values to multiple variables in one line:

Example

x, y, z = "Orange", "Banana", "Cherry"  
print(x)  
print(y)  
print(z)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables8)

**Note:** Make sure the number of variables matches the number of values, or else you will get an error.

One Value to Multiple Variables

And you can assign the *same* value to multiple variables in one line:

Example

x = y = z = "Orange"  
print(x)  
print(y)  
print(z)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables6)

Unpack a Collection

If you have a collection of values in a list, tuple etc. Python allows you to extract the values into variables. This is called *unpacking*.

Example

Unpack a list:

fruits = ["apple", "banana", "cherry"]  
x, y, z = fruits  
print(x)  
print(y)  
print(z)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_unpack)

Python - Output Variables

Output Variables

The Python print() function is often used to output variables.

Example

x = "Python is awesome"  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_print)

In the print() function, you output multiple variables, separated by a comma:

Example

x = "Python"  
y = "is"  
z = "awesome"  
print(x, y, z)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables3)

You can also use the + operator to output multiple variables:

Example

x = "Python "  
y = "is "  
z = "awesome"  
print(x + y + z)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables4)

Notice the space character after "Python " and "is ", without them the result would be "Pythonisawesome".

For numbers, the + character works as a mathematical operator:

Example

x = 5  
y = 10  
print(x + y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables5)

In the print() function, when you try to combine a string and a number with the + operator, Python will give you an error:

Example

x = 5  
y = "John"  
print(x + y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_test)

The best way to output multiple variables in the print() function is to separate them with commas, which even support different data types:

Example

x = 5  
y = "John"  
print(x, y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_comma)

Python - Global Variables

Global Variables

Variables that are created outside of a function (as in all of the examples above) are known as global variables.

Global variables can be used by everyone, both inside of functions and outside.

Example

Create a variable outside of a function, and use it inside the function

x = "awesome"  
  
def myfunc():  
  print("Python is " + x)  
  
myfunc()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_global)

If you create a variable with the same name inside a function, this variable will be local, and can only be used inside the function. The global variable with the same name will remain as it was, global and with the original value.

Example

Create a variable inside a function, with the same name as the global variable

x = "awesome"  
  
def myfunc():  
  x = "fantastic"  
  print("Python is " + x)  
  
myfunc()  
  
print("Python is " + x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_global2)

The global Keyword

Normally, when you create a variable inside a function, that variable is local, and can only be used inside that function.

To create a global variable inside a function, you can use the global keyword.

Example

If you use the global keyword, the variable belongs to the global scope:

def myfunc():  
  global x  
  x = "fantastic"  
  
myfunc()  
  
print("Python is " + x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_global3)

Also, use the global keyword if you want to change a global variable inside a function.

Example

To change the value of a global variable inside a function, refer to the variable by using the global keyword:

x = "awesome"  
  
def myfunc():  
  global x  
  x = "fantastic"  
  
myfunc()  
  
print("Python is " + x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_variables_global4)

Python - Variable Exercises

Test Yourself With Exercises

Now you have learned a lot about variables, and how to use them in Python.

Are you ready for a test?

Try to insert the missing part to make the code work as expected:

Top of Form

Exercise:

Create a variable named carname and assign the value Volvo to it.

 = ""

Submit Answer »

Bottom of Form

Go to the Exercise section and test all of our Python Variable Exercises:

[Python Variable Exercises](https://www.w3schools.com/python/exercise.asp?filename=exercise_variables1)

Python Data Types

Built-in Data Types

In programming, data type is an important concept.

Variables can store data of different types, and different types can do different things.

Python has the following data types built-in by default, in these categories:

|  |  |
| --- | --- |
| Text Type: | str |
| Numeric Types: | int, float, complex |
| Sequence Types: | list, tuple, range |
| Mapping Type: | dict |
| Set Types: | set, frozenset |
| Boolean Type: | bool |
| Binary Types: | bytes, bytearray, memoryview |
| None Type: | NoneType |

Getting the Data Type

You can get the data type of any object by using the type() function:

Example

Print the data type of the variable x:

x = 5  
print(type(x))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_type)

Setting the Data Type

In Python, the data type is set when you assign a value to a variable:

|  |  |  |
| --- | --- | --- |
| **Example** | **Data Type** | **Try it** |
| x = "Hello World" | str | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_str) |
| x = 20 | int | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_int) |
| x = 20.5 | float | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_float) |
| x = 1j | complex | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_complex) |
| x = ["apple", "banana", "cherry"] | list | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_list) |
| x = ("apple", "banana", "cherry") | tuple | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_tuple) |
| x = range(6) | range | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_range) |
| x = {"name" : "John", "age" : 36} | dict | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_dict) |
| x = {"apple", "banana", "cherry"} | set | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_set) |
| x = frozenset({"apple", "banana", "cherry"}) | frozenset | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_frozenset) |
| x = True | bool | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_bool) |
| x = b"Hello" | bytes | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_bytes) |
| x = bytearray(5) | bytearray | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_bytearray) |
| x = memoryview(bytes(5)) | memoryview | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_memoryview) |
| x = None | NoneType | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_nonetype) |

Setting the Specific Data Type

If you want to specify the data type, you can use the following constructor functions:

|  |  |  |
| --- | --- | --- |
| **Example** | **Data Type** | **Try it** |
| x = str("Hello World") | str | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_str2) |
| x = int(20) | int | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_int2) |
| x = float(20.5) | float | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_float2) |
| x = complex(1j) | complex | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_complex2) |
| x = list(("apple", "banana", "cherry")) | list | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_list2) |
| x = tuple(("apple", "banana", "cherry")) | tuple | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_tuple2) |
| x = range(6) | range | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_range2) |
| x = dict(name="John", age=36) | dict | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_dict2) |
| x = set(("apple", "banana", "cherry")) | set | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_set2) |
| x = frozenset(("apple", "banana", "cherry")) | frozenset | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_frozenset2) |
| x = bool(5) | bool | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_bool2) |
| x = bytes(5) | bytes | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_bytes2) |
| x = bytearray(5) | bytearray | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_bytearray2) |
| x = memoryview(bytes(5)) | memoryview | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_type_memoryview2) |

Top of Form

Exercise:

The following code example would print the data type of x, what data type would that be?

x = 5

print(type(x))



Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_datatypes1)

Python Numbers

Python Numbers

There are three numeric types in Python:

* int
* float
* complex

Variables of numeric types are created when you assign a value to them:

Example

x = 1    # int  
y = 2.8  # float  
z = 1j   # complex

To verify the type of any object in Python, use the type() function:

Example

print(type(x))  
print(type(y))  
print(type(z))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers)

Int

Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length.

Example

Integers:

x = 1  
y = 35656222554887711  
z = -3255522  
  
print(type(x))  
print(type(y))  
print(type(z))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_int)

Float

Float, or "floating point number" is a number, positive or negative, containing one or more decimals.

Example

Floats:

x = 1.10  
y = 1.0  
z = -35.59  
  
print(type(x))  
print(type(y))  
print(type(z))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_float)

Float can also be scientific numbers with an "e" to indicate the power of 10.

Example

Floats:

x = 35e3  
y = 12E4  
z = -87.7e100  
  
print(type(x))  
print(type(y))  
print(type(z))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_float2)

Complex

Complex numbers are written with a "j" as the imaginary part:

Example

Complex:

x = 3+5j  
y = 5j  
z = -5j  
  
print(type(x))  
print(type(y))  
print(type(z))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_complex)

Type Conversion

You can convert from one type to another with the int(), float(), and complex() methods:

Example

Convert from one type to another:

x = 1    # int  
y = 2.8  # float  
z = 1j   # complex  
  
#convert from int to float:  
a = float(x)  
  
#convert from float to int:  
b = int(y)  
  
#convert from int to complex:  
c = complex(x)  
  
print(a)  
print(b)  
print(c)  
  
print(type(a))  
print(type(b))  
print(type(c))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_convert)

**Note:** You cannot convert complex numbers into another number type.

Random Number

Python does not have a random() function to make a random number, but Python has a built-in module called random that can be used to make random numbers:

Example

Import the random module, and display a random number between 1 and 9:

import random  
  
print(random.randrange(1, 10))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_random)

In our [Random Module Reference](https://www.w3schools.com/python/module_random.asp) you will learn more about the Random module.

Top of Form

Exercise:

Insert the correct syntax to convert x into a floating point number.

x = 5

x = (x)

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_numbers1)

Bottom of Form

Python Casting

Specify a Variable Type

There may be times when you want to specify a type on to a variable. This can be done with casting. Python is an object-orientated language, and as such it uses classes to define data types, including its primitive types.

Casting in python is therefore done using constructor functions:

* int() - constructs an integer number from an integer literal, a float literal (by removing all decimals), or a string literal (providing the string represents a whole number)
* float() - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)
* str() - constructs a string from a wide variety of data types, including strings, integer literals and float literals

Example

Integers:

x = int(1)   # x will be 1  
y = int(2.8) # y will be 2  
z = int("3") # z will be 3

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_casting_int)

Example

Floats:

x = float(1)     # x will be 1.0  
y = float(2.8)   # y will be 2.8  
z = float("3")   # z will be 3.0  
w = float("4.2") # w will be 4.2

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_float)

Example

Strings:

x = str("s1") # x will be 's1'  
y = str(2)    # y will be '2'  
z = str(3.0)  # z will be '3.0'

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string)

Python Strings

Strings

Strings in python are surrounded by either single quotation marks, or double quotation marks.

'hello' is the same as "hello".

You can display a string literal with the print() function:

Example

print("Hello")  
print('Hello')

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_literal)

Assign String to a Variable

Assigning a string to a variable is done with the variable name followed by an equal sign and the string:

Example

a = "Hello"  
print(a)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_var)

Multiline Strings

You can assign a multiline string to a variable by using three quotes:

Example

You can use three double quotes:

a = """Lorem ipsum dolor sit amet,  
consectetur adipiscing elit,  
sed do eiusmod tempor incididunt  
ut labore et dolore magna aliqua."""  
print(a)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_multi)

Or three single quotes:

Example

a = '''Lorem ipsum dolor sit amet,  
consectetur adipiscing elit,  
sed do eiusmod tempor incididunt  
ut labore et dolore magna aliqua.'''  
print(a)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_multi2)

**Note:** in the result, the line breaks are inserted at the same position as in the code.

Strings are Arrays

Like many other popular programming languages, strings in Python are arrays of bytes representing unicode characters.

However, Python does not have a character data type, a single character is simply a string with a length of 1.

Square brackets can be used to access elements of the string.

Example

Get the character at position 1 (remember that the first character has the position 0):

a = "Hello, World!"  
print(a[1])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string1)

Looping Through a String

Since strings are arrays, we can loop through the characters in a string, with a for loop.

Example

Loop through the letters in the word "banana":

for x in "banana":  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_string)

Learn more about For Loops in our [Python For Loops](https://www.w3schools.com/python/python_for_loops.asp) chapter.

String Length

To get the length of a string, use the len() function.

Example

The len() function returns the length of a string:

a = "Hello, World!"  
print(len(a))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_len)

Check String

To check if a certain phrase or character is present in a string, we can use the keyword in.

Example

Check if "free" is present in the following text:

txt = "The best things in life are free!"  
print("free" in txt)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_is_in)

Use it in an if statement:

Example

Print only if "free" is present:

txt = "The best things in life are free!"  
if "free" in txt:  
  print("Yes, 'free' is present.")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_if_in)

Learn more about If statements in our [Python If...Else](https://www.w3schools.com/python/python_conditions.asp) chapter.

Check if NOT

To check if a certain phrase or character is NOT present in a string, we can use the keyword not in.

Example

Check if "expensive" is NOT present in the following text:

txt = "The best things in life are free!"  
print("expensive" not in txt)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_is_not_in)

Use it in an if statement:

Example

print only if "expensive" is NOT present:

txt = "The best things in life are free!"  
if "expensive" not in txt:  
  print("No, 'expensive' is NOT present.")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_if_not_in)

Python - Slicing Strings

Slicing

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

Example

Get the characters from position 2 to position 5 (not included):

b = "Hello, World!"  
print(b[2:5])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string2)

**Note:**The first character has index 0.

Slice From the Start

By leaving out the start index, the range will start at the first character:

Example

Get the characters from the start to position 5 (not included):

b = "Hello, World!"  
print(b[:5])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_slice_start)

Slice To the End

By leaving out the *end*index, the range will go to the end:

Example

Get the characters from position 2, and all the way to the end:

b = "Hello, World!"  
print(b[2:])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_slice_end)

Negative Indexing

Use negative indexes to start the slice from the end of the string:

Example

Get the characters:

From: "o" in "World!" (position -5)

To, but not included: "d" in "World!" (position -2):

b = "Hello, World!"  
print(b[-5:-2])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_negativeindex)

Python - Modify Strings

Python has a set of built-in methods that you can use on strings.

Upper Case

Example

The upper() method returns the string in upper case:

a = "Hello, World!"  
print(a.upper())

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_upper)

Lower Case

Example

The lower() method returns the string in lower case:

a = "Hello, World!"  
print(a.lower())

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_lower)

Remove Whitespace

Whitespace is the space before and/or after the actual text, and very often you want to remove this space.

Example

The strip() method removes any whitespace from the beginning or the end:

a = " Hello, World! "  
print(a.strip()) # returns "Hello, World!"

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_strip)

Replace String

Example

The replace() method replaces a string with another string:

a = "Hello, World!"  
print(a.replace("H", "J"))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_replace)

Split String

The split() method returns a list where the text between the specified separator becomes the list items.

Example

The split() method splits the string into substrings if it finds instances of the separator:

a = "Hello, World!"  
print(a.split(",")) # returns ['Hello', ' World!']

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_split)

Learn more about Lists in our [Python Lists](https://www.w3schools.com/python/python_lists.asp) chapter.

String Methods

Learn more about String Methods with our [String Methods Reference](https://www.w3schools.com/python/python_ref_string.asp)

Python - String Concatenation

String Concatenation

To concatenate, or combine, two strings you can use the + operator.

Example

Merge variable a with variable b into variable c:

a = "Hello"  
b = "World"  
c = a + b  
print(c)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_concat)

Example

To add a space between them, add a " ":

a = "Hello"  
b = "World"  
c = a + " " + b  
print(c)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_concat2)

Python - Format - Strings

String Format

As we learned in the Python Variables chapter, we cannot combine strings and numbers like this:

Example

age = 36  
txt = "My name is John, I am " + age  
print(txt)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_format_error)

But we can combine strings and numbers by using the format() method!

The format() method takes the passed arguments, formats them, and places them in the string where the placeholders {} are:

Example

Use the format() method to insert numbers into strings:

age = 36  
txt = "My name is John, and I am {}"  
print(txt.format(age))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_format1)

The format() method takes unlimited number of arguments, and are placed into the respective placeholders:

Example

quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want {} pieces of item {} for {} dollars."  
print(myorder.format(quantity, itemno, price))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_format2)

You can use index numbers {0} to be sure the arguments are placed in the correct placeholders:

Example

quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want to pay {2} dollars for {0} pieces of item {1}."  
print(myorder.format(quantity, itemno, price))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_format3)

Learn more about String Formatting in our [String Formatting](https://www.w3schools.com/python/python_string_formatting.asp) chapter.

Python - Escape Characters

Escape Character

To insert characters that are illegal in a string, use an escape character.

An escape character is a backslash \ followed by the character you want to insert.

An example of an illegal character is a double quote inside a string that is surrounded by double quotes:

Example

You will get an error if you use double quotes inside a string that is surrounded by double quotes:

txt = "We are the so-called "Vikings" from the north."

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_escape_error)

To fix this problem, use the escape character \":

Example

The escape character allows you to use double quotes when you normally would not be allowed:

txt = "We are the so-called \"Vikings\" from the north."

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_escape)

Escape Characters

Other escape characters used in Python:

|  |  |  |
| --- | --- | --- |
| **Code** | **Result** | **Try it** |
| \' | Single Quote | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_escape2) |
| \\ | Backslash | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_backslash) |
| \n | New Line | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_newline) |
| \r | Carriage Return | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_r) |
| \t | Tab | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_t) |
| \b | Backspace | [Try it »](https://www.w3schools.com/python/showpython.asp?filename=demo_string_b) |
| \f | Form Feed |  |
| \ooo | Octal value | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_octal) |
| \xhh | Hex value | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_hex) |

Python - String Methods

String Methods

Python has a set of built-in methods that you can use on strings.

**Note:** All string methods return new values. They do not change the original string.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [capitalize()](https://www.w3schools.com/python/ref_string_capitalize.asp) | Converts the first character to upper case |
| [casefold()](https://www.w3schools.com/python/ref_string_casefold.asp) | Converts string into lower case |
| [center()](https://www.w3schools.com/python/ref_string_center.asp) | Returns a centered string |
| [count()](https://www.w3schools.com/python/ref_string_count.asp) | Returns the number of times a specified value occurs in a string |
| [encode()](https://www.w3schools.com/python/ref_string_encode.asp) | Returns an encoded version of the string |
| [endswith()](https://www.w3schools.com/python/ref_string_endswith.asp) | Returns true if the string ends with the specified value |
| [expandtabs()](https://www.w3schools.com/python/ref_string_expandtabs.asp) | Sets the tab size of the string |
| [find()](https://www.w3schools.com/python/ref_string_find.asp) | Searches the string for a specified value and returns the position of where it was found |
| [format()](https://www.w3schools.com/python/ref_string_format.asp) | Formats specified values in a string |
| format\_map() | Formats specified values in a string |
| [index()](https://www.w3schools.com/python/ref_string_index.asp) | Searches the string for a specified value and returns the position of where it was found |
| [isalnum()](https://www.w3schools.com/python/ref_string_isalnum.asp) | Returns True if all characters in the string are alphanumeric |
| [isalpha()](https://www.w3schools.com/python/ref_string_isalpha.asp) | Returns True if all characters in the string are in the alphabet |
| [isdecimal()](https://www.w3schools.com/python/ref_string_isdecimal.asp) | Returns True if all characters in the string are decimals |
| [isdigit()](https://www.w3schools.com/python/ref_string_isdigit.asp) | Returns True if all characters in the string are digits |
| [isidentifier()](https://www.w3schools.com/python/ref_string_isidentifier.asp) | Returns True if the string is an identifier |
| [islower()](https://www.w3schools.com/python/ref_string_islower.asp) | Returns True if all characters in the string are lower case |
| [isnumeric()](https://www.w3schools.com/python/ref_string_isnumeric.asp) | Returns True if all characters in the string are numeric |
| [isprintable()](https://www.w3schools.com/python/ref_string_isprintable.asp) | Returns True if all characters in the string are printable |
| [isspace()](https://www.w3schools.com/python/ref_string_isspace.asp) | Returns True if all characters in the string are whitespaces |
| [istitle()](https://www.w3schools.com/python/ref_string_istitle.asp) | Returns True if the string follows the rules of a title |
| [isupper()](https://www.w3schools.com/python/ref_string_isupper.asp) | Returns True if all characters in the string are upper case |
| [join()](https://www.w3schools.com/python/ref_string_join.asp) | Joins the elements of an iterable to the end of the string |
| [ljust()](https://www.w3schools.com/python/ref_string_ljust.asp) | Returns a left justified version of the string |
| [lower()](https://www.w3schools.com/python/ref_string_lower.asp) | Converts a string into lower case |
| [lstrip()](https://www.w3schools.com/python/ref_string_lstrip.asp) | Returns a left trim version of the string |
| [maketrans()](https://www.w3schools.com/python/ref_string_maketrans.asp) | Returns a translation table to be used in translations |
| [partition()](https://www.w3schools.com/python/ref_string_partition.asp) | Returns a tuple where the string is parted into three parts |
| [replace()](https://www.w3schools.com/python/ref_string_replace.asp) | Returns a string where a specified value is replaced with a specified value |
| [rfind()](https://www.w3schools.com/python/ref_string_rfind.asp) | Searches the string for a specified value and returns the last position of where it was found |
| [rindex()](https://www.w3schools.com/python/ref_string_rindex.asp) | Searches the string for a specified value and returns the last position of where it was found |
| [rjust()](https://www.w3schools.com/python/ref_string_rjust.asp) | Returns a right justified version of the string |
| [rpartition()](https://www.w3schools.com/python/ref_string_rpartition.asp) | Returns a tuple where the string is parted into three parts |
| [rsplit()](https://www.w3schools.com/python/ref_string_rsplit.asp) | Splits the string at the specified separator, and returns a list |
| [rstrip()](https://www.w3schools.com/python/ref_string_rstrip.asp) | Returns a right trim version of the string |
| [split()](https://www.w3schools.com/python/ref_string_split.asp) | Splits the string at the specified separator, and returns a list |
| [splitlines()](https://www.w3schools.com/python/ref_string_splitlines.asp) | Splits the string at line breaks and returns a list |
| [startswith()](https://www.w3schools.com/python/ref_string_startswith.asp) | Returns true if the string starts with the specified value |
| [strip()](https://www.w3schools.com/python/ref_string_strip.asp) | Returns a trimmed version of the string |
| [swapcase()](https://www.w3schools.com/python/ref_string_swapcase.asp) | Swaps cases, lower case becomes upper case and vice versa |
| [title()](https://www.w3schools.com/python/ref_string_title.asp) | Converts the first character of each word to upper case |
| [translate()](https://www.w3schools.com/python/ref_string_translate.asp) | Returns a translated string |
| [upper()](https://www.w3schools.com/python/ref_string_upper.asp) | Converts a string into upper case |
| [zfill()](https://www.w3schools.com/python/ref_string_zfill.asp) | Fills the string with a specified number of 0 values at the beginning |

Python - String Exercises

Test Yourself With Exercises

Now you have learned a lot about Strings, and how to use them in Python.

Are you ready for a test?

Try to insert the missing part to make the code work as expected:

Top of Form

Exercise:

Use the len method to print the length of the string.

x = "Hello World"

print()

Submit Answer »

Bottom of Form

Go to the Exercise section and test all of our Python Strings Exercises:

[Python String Exercises](https://www.w3schools.com/python/exercise.asp?filename=exercise_strings1)

Python Booleans

Booleans represent one of two values: True or False.

Boolean Values

In programming you often need to know if an expression is True or False.

You can evaluate any expression in Python, and get one of two answers, True or False.

When you compare two values, the expression is evaluated and Python returns the Boolean answer:

Example

print(10 > 9)  
print(10 == 9)  
print(10 < 9)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans1)

When you run a condition in an if statement, Python returns True or False:

Example

Print a message based on whether the condition is True or False:

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

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans2)

Evaluate Values and Variables

The bool() function allows you to evaluate any value, and give you True or False in return,

Example

Evaluate a string and a number:

print(bool("Hello"))  
print(bool(15))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans3)

Example

Evaluate two variables:

x = "Hello"  
y = 15  
  
print(bool(x))  
print(bool(y))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans4)

Most Values are True

Almost any value is evaluated to True if it has some sort of content.

Any string is True, except empty strings.

Any number is True, except 0.

Any list, tuple, set, and dictionary are True, except empty ones.

Example

The following will return True:

bool("abc")  
bool(123)  
bool(["apple", "cherry", "banana"])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans5)

Some Values are False

In fact, there are not many values that evaluate to False, except empty values, such as (), [], {}, "", the number 0, and the value None. And of course the value False evaluates to False.

Example

The following will return False:

bool(False)  
bool(None)  
bool(0)  
bool("")  
bool(())  
bool([])  
bool({})

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans6)

One more value, or object in this case, evaluates to False, and that is if you have an object that is made from a class with a \_\_len\_\_ function that returns 0 or False:

Example

class myclass():  
  def \_\_len\_\_(self):  
    return 0  
  
myobj = myclass()  
print(bool(myobj))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans7)

Functions can Return a Boolean

You can create functions that returns a Boolean Value:

Example

Print the answer of a function:

def myFunction() :  
  return True  
  
print(myFunction())

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_boolean_return)

You can execute code based on the Boolean answer of a function:

Example

Print "YES!" if the function returns True, otherwise print "NO!":

def myFunction() :  
  return True  
  
if myFunction():  
  print("YES!")  
else:  
  print("NO!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_boolean_return2)

Python also has many built-in functions that return a boolean value, like the isinstance() function, which can be used to determine if an object is of a certain data type:

Example

Check if an object is an integer or not:

x = 200  
print(isinstance(x, int))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_booleans8)

Top of Form

Exercise:

The statement below would print a Boolean value, which one?

print(10 > 9)



Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_booleans1)

Python Operators

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:

Example

print(10 + 5)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper)

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 »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_add) |
| - | Subtraction | x - y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_sub) |
| \* | Multiplication | x \* y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_mult) |
| / | Division | x / y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_div) |
| % | Modulus | x % y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_mod) |
| \*\* | Exponentiation | x \*\* y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_exp) |
| // | Floor division | x // y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_floordiv) |

Python Assignment Operators

Assignment operators are used to assign values to variables:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Example** | **Same As** | **Try it** |
| = | x = 5 | x = 5 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass1) |
| += | x += 3 | x = x + 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass2) |
| -= | x -= 3 | x = x - 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass3) |
| \*= | x \*= 3 | x = x \* 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass4) |
| /= | x /= 3 | x = x / 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass5) |
| %= | x %= 3 | x = x % 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass6) |
| //= | x //= 3 | x = x // 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass7) |
| \*\*= | x \*\*= 3 | x = x \*\* 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass8) |
| &= | x &= 3 | x = x & 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass9) |
| |= | x |= 3 | x = x | 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass10) |
| ^= | x ^= 3 | x = x ^ 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass11) |
| >>= | x >>= 3 | x = x >> 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass12) |
| <<= | x <<= 3 | x = x << 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass13) |

Python Comparison Operators

Comparison operators are used to compare two values:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Try it** |
| == | Equal | x == y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_compare1) |
| != | Not equal | x != y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_compare2) |
| > | Greater than | x > y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_compare4) |
| < | Less than | x < y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_compare5) |
| >= | Greater than or equal to | x >= y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_compare6) |
| <= | Less than or equal to | x <= y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_compare7) |

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 »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_logical1) |
| or | Returns True if one of the statements is true | x < 5 or x < 4 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_logical2) |
| not | Reverse the result, returns False if the result is true | not(x < 5 and x < 10) | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_logical3) |

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:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **Example** | **Try it** |
| is | Returns True if both variables are the same object | x is y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_identity1) |
| is not | Returns True if both variables are not the same object | x is not y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_identity2) |

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 »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_membership1) |
| not in | Returns True if a sequence with the specified value is not present in the object | x not in y | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_membership2) |

Python Bitwise Operators

Bitwise operators are used to compare (binary) numbers:

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

Top of Form

Exercise:

Multiply 10 with 5, and print the result.

print(10  5)

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_operators1)

Bottom of Form

Python Lists

mylist = ["apple", "banana", "cherry"]

List

Lists are used to store multiple items in a single variable.

Lists are one of 4 built-in data types in Python used to store collections of data, the other 3 are [Tuple](https://www.w3schools.com/python/python_tuples.asp), [Set](https://www.w3schools.com/python/python_sets.asp), and [Dictionary](https://www.w3schools.com/python/python_dictionaries.asp), all with different qualities and usage.

Lists are created using square brackets:

Example

Create a List:

thislist = ["apple", "banana", "cherry"]  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list)

List Items

List items are ordered, changeable, and allow duplicate values.

List items are indexed, the first item has index [0], the second item has index [1] etc.

Ordered

When we say that lists are ordered, it means that the items have a defined order, and that order will not change.

If you add new items to a list, the new items will be placed at the end of the list.

**Note:** There are some [list methods](https://www.w3schools.com/python/python_lists_methods.asp) that will change the order, but in general: the order of the items will not change.

Changeable

The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.

Allow Duplicates

Since lists are indexed, lists can have items with the same value:

Example

Lists allow duplicate values:

thislist = ["apple", "banana", "cherry", "apple", "cherry"]  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_duplicates)

List Length

To determine how many items a list has, use the len() function:

Example

Print the number of items in the list:

thislist = ["apple", "banana", "cherry"]  
print(len(thislist))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_len)

List Items - Data Types

List items can be of any data type:

Example

String, int and boolean data types:

list1 = ["apple", "banana", "cherry"]  
list2 = [1, 5, 7, 9, 3]  
list3 = [True, False, False]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_datatypes)

A list can contain different data types:

Example

A list with strings, integers and boolean values:

list1 = ["abc", 34, True, 40, "male"]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_datatypes2)

type()

From Python's perspective, lists are defined as objects with the data type 'list':

<class 'list'>

Example

What is the data type of a list?

mylist = ["apple", "banana", "cherry"]  
print(type(mylist))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_datatypes3)

The list() Constructor

It is also possible to use the list() constructor when creating a new list.

Example

Using the list() constructor to make a List:

thislist = list(("apple", "banana", "cherry")) # note the double round-brackets  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_constructor)

Python Collections (Arrays)

There are four collection data types in the Python programming language:

* **List** is a collection which is ordered and changeable. Allows duplicate members.
* [**Tuple**](https://www.w3schools.com/python/python_tuples.asp) is a collection which is ordered and unchangeable. Allows duplicate members.
* [**Set**](https://www.w3schools.com/python/python_sets.asp) is a collection which is unordered, unchangeable\*, and unindexed. No duplicate members.
* [**Dictionary**](https://www.w3schools.com/python/python_dictionaries.asp) is a collection which is ordered\*\* and changeable. No duplicate members.

\*Set *items* are unchangeable, but you can remove and/or add items whenever you like.

\*\*As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

When choosing a collection type, it is useful to understand the properties of that type. Choosing the right type for a particular data set could mean retention of meaning, and, it could mean an increase in efficiency or security.

Python - Access List Items

Access Items

List items are indexed and you can access them by referring to the index number:

Example

Print the second item of the list:

thislist = ["apple", "banana", "cherry"]  
print(thislist[1])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_access)

**Note:**The first item has index 0.

Negative Indexing

Negative indexing means start from the end

-1 refers to the last item, -2 refers to the second last item etc.

Example

Print the last item of the list:

thislist = ["apple", "banana", "cherry"]  
print(thislist[-1])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_negative_index)

Range of Indexes

You can specify a range of indexes by specifying where to start and where to end the range.

When specifying a range, the return value will be a new list with the specified items.

Example

Return the third, fourth, and fifth item:

thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]  
print(thislist[2:5])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_range)

**Note:** The search will start at index 2 (included) and end at index 5 (not included).

Remember that the first item has index 0.

By leaving out the start value, the range will start at the first item:

Example

This example returns the items from the beginning to, but NOT including, "kiwi":

thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]  
print(thislist[:4])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_range2)

By leaving out the end value, the range will go on to the end of the list:

Example

This example returns the items from "cherry" to the end:

thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]  
print(thislist[2:])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_range3)

Range of Negative Indexes

Specify negative indexes if you want to start the search from the end of the list:

Example

This example returns the items from "orange" (-4) to, but NOT including "mango" (-1):

thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]  
print(thislist[-4:-1])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_range_negative)

Check if Item Exists

To determine if a specified item is present in a list use the in keyword:

Example

Check if "apple" is present in the list:

thislist = ["apple", "banana", "cherry"]  
if "apple" in thislist:  
  print("Yes, 'apple' is in the fruits list")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_in)

Python - Change List Items

Change Item Value

To change the value of a specific item, refer to the index number:

Example

Change the second item:

thislist = ["apple", "banana", "cherry"]  
thislist[1] = "blackcurrant"  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_change)

Change a Range of Item Values

To change the value of items within a specific range, define a list with the new values, and refer to the range of index numbers where you want to insert the new values:

Example

Change the values "banana" and "cherry" with the values "blackcurrant" and "watermelon":

thislist = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]  
thislist[1:3] = ["blackcurrant", "watermelon"]  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_change_range)

If you insert *more* items than you replace, the new items will be inserted where you specified, and the remaining items will move accordingly:

Example

Change the second value by replacing it with *two* new values:

thislist = ["apple", "banana", "cherry"]  
thislist[1:2] = ["blackcurrant", "watermelon"]  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_change2)

**Note:** The length of the list will change when the number of items inserted does not match the number of items replaced.

If you insert *less* items than you replace, the new items will be inserted where you specified, and the remaining items will move accordingly:

Example

Change the second and third value by replacing it with *one* value:

thislist = ["apple", "banana", "cherry"]  
thislist[1:3] = ["watermelon"]  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_change3)

Insert Items

To insert a new list item, without replacing any of the existing values, we can use the insert() method.

The insert() method inserts an item at the specified index:

Example

Insert "watermelon" as the third item:

thislist = ["apple", "banana", "cherry"]  
thislist.insert(2, "watermelon")  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_change2_insert)

**Note:** As a result of the example above, the list will now contain 4 items.

Python - Add List Items

Append Items

To add an item to the end of the list, use the append() method:

Example

Using the append() method to append an item:

thislist = ["apple", "banana", "cherry"]  
thislist.append("orange")  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_append)

Insert Items

To insert a list item at a specified index, use the insert() method.

The insert() method inserts an item at the specified index:

Example

Insert an item as the second position:

thislist = ["apple", "banana", "cherry"]  
thislist.insert(1, "orange")  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_insert)

**Note:** As a result of the examples above, the lists will now contain 4 items.

Extend List

To append elements from *another list* to the current list, use the extend() method.

Example

Add the elements of tropical to thislist:

thislist = ["apple", "banana", "cherry"]  
tropical = ["mango", "pineapple", "papaya"]  
thislist.extend(tropical)  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_add_extend)

The elements will be added to the *end* of the list.

Add Any Iterable

The extend() method does not have to append *lists*, you can add any iterable object (tuples, sets, dictionaries etc.).

Example

Add elements of a tuple to a list:

thislist = ["apple", "banana", "cherry"]  
thistuple = ("kiwi", "orange")  
thislist.extend(thistuple)  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_add_extend_tuple)

Python - Remove List Items

Remove Specified Item

The remove() method removes the specified item.

Example

Remove "banana":

thislist = ["apple", "banana", "cherry"]  
thislist.remove("banana")  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_remove)

Remove Specified Index

The pop() method removes the specified index.

Example

Remove the second item:

thislist = ["apple", "banana", "cherry"]  
thislist.pop(1)  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_pop2)

If you do not specify the index, the pop() method removes the last item.

Example

Remove the last item:

thislist = ["apple", "banana", "cherry"]  
thislist.pop()  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_pop)

The del keyword also removes the specified index:

Example

Remove the first item:

thislist = ["apple", "banana", "cherry"]  
del thislist[0]  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_del)

The del keyword can also delete the list completely.

Example

Delete the entire list:

thislist = ["apple", "banana", "cherry"]  
del thislist

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_del2)

Clear the List

The clear() method empties the list.

The list still remains, but it has no content.

Example

Clear the list content:

thislist = ["apple", "banana", "cherry"]  
thislist.clear()  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_clear)

Python - Loop Lists

Loop Through a List

You can loop through the list items by using a for loop:

Example

Print all items in the list, one by one:

thislist = ["apple", "banana", "cherry"]  
for x in thislist:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_loop)

Learn more about for loops in our [Python For Loops](https://www.w3schools.com/python/python_for_loops.asp) Chapter.

Loop Through the Index Numbers

You can also loop through the list items by referring to their index number.

Use the range() and len() functions to create a suitable iterable.

Example

Print all items by referring to their index number:

thislist = ["apple", "banana", "cherry"]  
for i in range(len(thislist)):  
  print(thislist[i])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_loop2)

The iterable created in the example above is [0, 1, 2].

Using a While Loop

You can loop through the list items by using a while loop.

Use the len() function to determine the length of the list, then start at 0 and loop your way through the list items by referring to their indexes.

Remember to increase the index by 1 after each iteration.

Example

Print all items, using a while loop to go through all the index numbers

thislist = ["apple", "banana", "cherry"]  
i = 0  
while i < len(thislist):  
  print(thislist[i])  
  i = i + 1

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_while_loop)

Learn more about while loops in our [Python While Loops](https://www.w3schools.com/python/python_while_loops.asp) Chapter.

Looping Using List Comprehension

List Comprehension offers the shortest syntax for looping through lists:

Example

A short hand for loop that will print all items in a list:

thislist = ["apple", "banana", "cherry"]  
[print(x) for x in thislist]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_loop_comprehensive)

Learn more about list comprehension in the next chapter: [List Comprehension](https://www.w3schools.com/python/python_lists_comprehension.asp).

Python - List Comprehension

List Comprehension

List comprehension offers a shorter syntax when you want to create a new list based on the values of an existing list.

Example:

Based on a list of fruits, you want a new list, containing only the fruits with the letter "a" in the name.

Without list comprehension you will have to write a for statement with a conditional test inside:

Example

fruits = ["apple", "banana", "cherry", "kiwi", "mango"]  
newlist = []  
  
for x in fruits:  
  if "a" in x:  
    newlist.append(x)  
  
print(newlist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_without_comprehension)

With list comprehension you can do all that with only one line of code:

Example

fruits = ["apple", "banana", "cherry", "kiwi", "mango"]  
  
newlist = [x for x in fruits if "a" in x]  
  
print(newlist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension)

The Syntax

newlist = [*expression* for *item* in *iterable* if *condition* == True]

The return value is a new list, leaving the old list unchanged.

Condition

The *condition* is like a filter that only accepts the items that valuate to True.

Example

Only accept items that are not "apple":

newlist = [x for x in fruits if x != "apple"]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension_if)

The condition if x != "apple"  will return True for all elements other than "apple", making the new list contain all fruits except "apple".

The *condition* is optional and can be omitted:

Example

With no if statement:

newlist = [x for x in fruits]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension_noif)

Iterable

The *iterable* can be any iterable object, like a list, tuple, set etc.

Example

You can use the range() function to create an iterable:

newlist = [x for x in range(10)]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension_range)

Same example, but with a condition:

Example

Accept only numbers lower than 5:

newlist = [x for x in range(10) if x < 5]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension_range2)

Expression

The *expression* is the current item in the iteration, but it is also the outcome, which you can manipulate before it ends up like a list item in the new list:

Example

Set the values in the new list to upper case:

newlist = [x.upper() for x in fruits]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension_upper)

You can set the outcome to whatever you like:

Example

Set all values in the new list to 'hello':

newlist = ['hello' for x in fruits]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension_hello)

The *expression* can also contain conditions, not like a filter, but as a way to manipulate the outcome:

Example

Return "orange" instead of "banana":

newlist = [x if x != "banana" else "orange" for x in fruits]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_comprehension_if_else)

The *expression* in the example above says:

*"Return the item if it is not banana, if it is banana return orange".*

Python - Sort Lists

Sort List Alphanumerically

List objects have a sort() method that will sort the list alphanumerically, ascending, by default:

Example

Sort the list alphabetically:

thislist = ["orange", "mango", "kiwi", "pineapple", "banana"]  
thislist.sort()  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort)

Example

Sort the list numerically:

thislist = [100, 50, 65, 82, 23]  
thislist.sort()  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort_num)

Sort Descending

To sort descending, use the keyword argument reverse = True:

Example

Sort the list descending:

thislist = ["orange", "mango", "kiwi", "pineapple", "banana"]  
thislist.sort(reverse = True)  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort_desc)

Example

Sort the list descending:

thislist = [100, 50, 65, 82, 23]  
thislist.sort(reverse = True)  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort_num_desc)

Customize Sort Function

You can also customize your own function by using the keyword argument key =*function*.

The function will return a number that will be used to sort the list (the lowest number first):

Example

Sort the list based on how close the number is to 50:

def myfunc(n):  
  return abs(n - 50)  
  
thislist = [100, 50, 65, 82, 23]  
thislist.sort(key = myfunc)  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort_key)

Case Insensitive Sort

By default the sort() method is case sensitive, resulting in all capital letters being sorted before lower case letters:

Example

Case sensitive sorting can give an unexpected result:

thislist = ["banana", "Orange", "Kiwi", "cherry"]  
thislist.sort()  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort_case_sensitive)

Luckily we can use built-in functions as key functions when sorting a list.

So if you want a case-insensitive sort function, use str.lower as a key function:

Example

Perform a case-insensitive sort of the list:

thislist = ["banana", "Orange", "Kiwi", "cherry"]  
thislist.sort(key = str.lower)  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort_lower)

Reverse Order

What if you want to reverse the order of a list, regardless of the alphabet?

The reverse() method reverses the current sorting order of the elements.

Example

Reverse the order of the list items:

thislist = ["banana", "Orange", "Kiwi", "cherry"]  
thislist.reverse()  
print(thislist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_sort_reverse)

Python - Copy Lists

Copy a List

You cannot copy a list simply by typing list2 = list1, because: list2 will only be a *reference* to list1, and changes made in list1 will automatically also be made in list2.

There are ways to make a copy, one way is to use the built-in List method copy().

Example

Make a copy of a list with the copy() method:

thislist = ["apple", "banana", "cherry"]  
mylist = thislist.copy()  
print(mylist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_copy)

Another way to make a copy is to use the built-in method list().

Example

Make a copy of a list with the list() method:

thislist = ["apple", "banana", "cherry"]  
mylist = list(thislist)  
print(mylist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_copy2)

Python - Join Lists

Join Two Lists

There are several ways to join, or concatenate, two or more lists in Python.

One of the easiest ways are by using the + operator.

Example

Join two list:

list1 = ["a", "b", "c"]  
list2 = [1, 2, 3]  
  
list3 = list1 + list2  
print(list3)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_concat1)

Another way to join two lists is by appending all the items from list2 into list1, one by one:

Example

Append list2 into list1:

list1 = ["a", "b" , "c"]  
list2 = [1, 2, 3]  
  
for x in list2:  
  list1.append(x)  
  
print(list1)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_concat2)

Or you can use the extend() method, which purpose is to add elements from one list to another list:

Example

Use the extend() method to add list2 at the end of list1:

list1 = ["a", "b" , "c"]  
list2 = [1, 2, 3]  
  
list1.extend(list2)  
print(list1)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_list_concat3)

Python - List Methods

List Methods

Python has a set of built-in methods that you can use on lists.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [append()](https://www.w3schools.com/python/ref_list_append.asp) | Adds an element at the end of the list |
| [clear()](https://www.w3schools.com/python/ref_list_clear.asp) | Removes all the elements from the list |
| [copy()](https://www.w3schools.com/python/ref_list_copy.asp) | Returns a copy of the list |
| [count()](https://www.w3schools.com/python/ref_list_count.asp) | Returns the number of elements with the specified value |
| [extend()](https://www.w3schools.com/python/ref_list_extend.asp) | Add the elements of a list (or any iterable), to the end of the current list |
| [index()](https://www.w3schools.com/python/ref_list_index.asp) | Returns the index of the first element with the specified value |
| [insert()](https://www.w3schools.com/python/ref_list_insert.asp) | Adds an element at the specified position |
| [pop()](https://www.w3schools.com/python/ref_list_pop.asp) | Removes the element at the specified position |
| [remove()](https://www.w3schools.com/python/ref_list_remove.asp) | Removes the item with the specified value |
| [reverse()](https://www.w3schools.com/python/ref_list_reverse.asp) | Reverses the order of the list |
| [sort()](https://www.w3schools.com/python/ref_list_sort.asp) | Sorts the list |

Python List Exercises

Test Yourself With Exercises

Now you have learned a lot about lists, and how to use them in Python.

Are you ready for a test?

Try to insert the missing part to make the code work as expected:

Top of Form

Exercise:

Print the second item in the fruits list.

fruits = ["apple", "banana", "cherry"]

print()

Submit Answer »

Bottom of Form

Go to the Exercise section and test all of our Python List Exercises:

[Python List Exercises](https://www.w3schools.com/python/exercise.asp?filename=exercise_lists1)

Python Tuples

mytuple = ("apple", "banana", "cherry")

Tuple

Tuples are used to store multiple items in a single variable.

Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are [List](https://www.w3schools.com/python/python_lists.asp), [Set](https://www.w3schools.com/python/python_sets.asp), and [Dictionary](https://www.w3schools.com/python/python_dictionaries.asp), all with different qualities and usage.

A tuple is a collection which is ordered and **unchangeable**.

Tuples are written with round brackets.

Example

Create a Tuple:

thistuple = ("apple", "banana", "cherry")  
print(thistuple)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple)

Tuple Items

Tuple items are ordered, unchangeable, and allow duplicate values.

Tuple items are indexed, the first item has index [0], the second item has index [1] etc.

Ordered

When we say that tuples are ordered, it means that the items have a defined order, and that order will not change.

Unchangeable

Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple has been created.

Allow Duplicates

Since tuples are indexed, they can have items with the same value:

Example

Tuples allow duplicate values:

thistuple = ("apple", "banana", "cherry", "apple", "cherry")  
print(thistuple)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_duplicates)

Tuple Length

To determine how many items a tuple has, use the len() function:

Example

Print the number of items in the tuple:

thistuple = ("apple", "banana", "cherry")  
print(len(thistuple))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_len)

Create Tuple With One Item

To create a tuple with only one item, you have to add a comma after the item, otherwise Python will not recognize it as a tuple.

Example

One item tuple, remember the comma:

thistuple = ("apple",)  
print(type(thistuple))  
  
#NOT a tuple  
thistuple = ("apple")  
print(type(thistuple))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_one_item)

Tuple Items - Data Types

Tuple items can be of any data type:

Example

String, int and boolean data types:

tuple1 = ("apple", "banana", "cherry")  
tuple2 = (1, 5, 7, 9, 3)  
tuple3 = (True, False, False)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_datatypes)

A tuple can contain different data types:

Example

A tuple with strings, integers and boolean values:

tuple1 = ("abc", 34, True, 40, "male")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_datatypes2)

type()

From Python's perspective, tuples are defined as objects with the data type 'tuple':

<class 'tuple'>

Example

What is the data type of a tuple?

mytuple = ("apple", "banana", "cherry")  
print(type(mytuple))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_datatypes3)

The tuple() Constructor

It is also possible to use the tuple() constructor to make a tuple.

Example

Using the tuple() method to make a tuple:

thistuple = tuple(("apple", "banana", "cherry")) # note the double round-brackets  
print(thistuple)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple3)

Python Collections (Arrays)

There are four collection data types in the Python programming language:

* [**List**](https://www.w3schools.com/python/python_lists.asp) is a collection which is ordered and changeable. Allows duplicate members.
* **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
* [**Set**](https://www.w3schools.com/python/python_sets.asp) is a collection which is unordered, unchangeable\*, and unindexed. No duplicate members.
* [**Dictionary**](https://www.w3schools.com/python/python_dictionaries.asp) is a collection which is ordered\*\* and changeable. No duplicate members.

\*Set *items* are unchangeable, but you can remove and/or add items whenever you like.

\*\*As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

When choosing a collection type, it is useful to understand the properties of that type. Choosing the right type for a particular data set could mean retention of meaning, and, it could mean an increase in efficiency or security.

Python - Access Tuple Items

Access Tuple Items

You can access tuple items by referring to the index number, inside square brackets:

Example

Print the second item in the tuple:

thistuple = ("apple", "banana", "cherry")  
print(thistuple[1])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple1)

**Note:** The first item has index 0.

Negative Indexing

Negative indexing means start from the end.

-1 refers to the last item, -2 refers to the second last item etc.

Example

Print the last item of the tuple:

thistuple = ("apple", "banana", "cherry")  
print(thistuple[-1])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_negative_index)

Range of Indexes

You can specify a range of indexes by specifying where to start and where to end the range.

When specifying a range, the return value will be a new tuple with the specified items.

Example

Return the third, fourth, and fifth item:

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")  
print(thistuple[2:5])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_range)

**Note:** The search will start at index 2 (included) and end at index 5 (not included).

Remember that the first item has index 0.

By leaving out the start value, the range will start at the first item:

Example

This example returns the items from the beginning to, but NOT included, "kiwi":

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")  
print(thistuple[:4])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_range2)

By leaving out the end value, the range will go on to the end of the list:

Example

This example returns the items from "cherry" and to the end:

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")  
print(thistuple[2:])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_range3)

Range of Negative Indexes

Specify negative indexes if you want to start the search from the end of the tuple:

Example

This example returns the items from index -4 (included) to index -1 (excluded)

thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")  
print(thistuple[-4:-1])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_range_negative)

Check if Item Exists

To determine if a specified item is present in a tuple use the in keyword:

Example

Check if "apple" is present in the tuple:

thistuple = ("apple", "banana", "cherry")  
if "apple" in thistuple:  
  print("Yes, 'apple' is in the fruits tuple")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_in)

Python - Update Tuples

Tuples are unchangeable, meaning that you cannot change, add, or remove items once the tuple is created.

But there are some workarounds.

Change Tuple Values

Once a tuple is created, you cannot change its values. Tuples are **unchangeable**, or **immutable** as it also is called.

But there is a workaround. You can convert the tuple into a list, change the list, and convert the list back into a tuple.

Example

Convert the tuple into a list to be able to change it:

x = ("apple", "banana", "cherry")  
y = list(x)  
y[1] = "kiwi"  
x = tuple(y)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_change)

Add Items

Since tuples are immutable, they do not have a build-in append() method, but there are other ways to add items to a tuple.

1. **Convert into a list**: Just like the workaround for*changing*a tuple, you can convert it into a list, add your item(s), and convert it back into a tuple.

Example

Convert the tuple into a list, add "orange", and convert it back into a tuple:

thistuple = ("apple", "banana", "cherry")  
y = list(thistuple)  
y.append("orange")  
thistuple = tuple(y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_add_orange)

2. **Add tuple to a tuple**. You are allowed to add tuples to tuples, so if you want to add one item, (or many), create a new tuple with the item(s), and add it to the existing tuple:

Example

Create a new tuple with the value "orange", and add that tuple:

thistuple = ("apple", "banana", "cherry")  
y = ("orange",)  
thistuple += y  
  
print(thistuple)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_add_tuple)

**Note:** When creating a tuple with only one item, remember to include a comma after the item, otherwise it will not be identified as a tuple.

Remove Items

**Note:** You cannot remove items in a tuple.

Tuples are **unchangeable**, so you cannot remove items from it, but you can use the same workaround as we used for changing and adding tuple items:

Example

Convert the tuple into a list, remove "apple", and convert it back into a tuple:

thistuple = ("apple", "banana", "cherry")  
y = list(thistuple)  
y.remove("apple")  
thistuple = tuple(y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_remove)

Or you can delete the tuple completely:

Example

The del keyword can delete the tuple completely:

thistuple = ("apple", "banana", "cherry")  
del thistuple  
print(thistuple) #this will raise an error because the tuple no longer exists

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_del)

Python - Unpack Tuples

Unpacking a Tuple

When we create a tuple, we normally assign values to it. This is called "packing" a tuple:

Example

Packing a tuple:

fruits = ("apple", "banana", "cherry")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_packing)

But, in Python, we are also allowed to extract the values back into variables. This is called "unpacking":

Example

Unpacking a tuple:

fruits = ("apple", "banana", "cherry")  
  
(green, yellow, red) = fruits  
  
print(green)  
print(yellow)  
print(red)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_unpacking)

**Note:** The number of variables must match the number of values in the tuple, if not, you must use an asterisk to collect the remaining values as a list.

Using Asterisk\*

If the number of variables is less than the number of values, you can add an \* to the variable name and the values will be assigned to the variable as a list:

Example

Assign the rest of the values as a list called "red":

fruits = ("apple", "banana", "cherry", "strawberry", "raspberry")  
  
(green, yellow, \*red) = fruits  
  
print(green)  
print(yellow)  
print(red)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_unpacking_asterix1)

If the asterisk is added to another variable name than the last, Python will assign values to the variable until the number of values left matches the number of variables left.

Example

Add a list of values the "tropic" variable:

fruits = ("apple", "mango", "papaya", "pineapple", "cherry")  
  
(green, \*tropic, red) = fruits  
  
print(green)  
print(tropic)  
print(red)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_unpacking_asterix2)

Python - Loop Tuples

Loop Through a Tuple

You can loop through the tuple items by using a for loop.

Example

Iterate through the items and print the values:

thistuple = ("apple", "banana", "cherry")  
for x in thistuple:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_loop)

Learn more about for loops in our [Python For Loops](https://www.w3schools.com/python/python_for_loops.asp) Chapter.

Loop Through the Index Numbers

You can also loop through the tuple items by referring to their index number.

Use the range() and len() functions to create a suitable iterable.

Example

Print all items by referring to their index number:

thistuple = ("apple", "banana", "cherry")  
for i in range(len(thistuple)):  
  print(thistuple[i])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_loop_index)

Using a While Loop

You can loop through the list items by using a while loop.

Use the len() function to determine the length of the tuple, then start at 0 and loop your way through the tuple items by refering to their indexes.

Remember to increase the index by 1 after each iteration.

Example

Print all items, using a while loop to go through all the index numbers:

thistuple = ("apple", "banana", "cherry")  
i = 0  
while i < len(thistuple):  
  print(thistuple[i])  
  i = i + 1

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_while)

Learn more about while loops in our [Python While Loops](https://www.w3schools.com/python/python_while_loops.asp) Chapter.

Python - Join Tuples

Join Two Tuples

To join two or more tuples you can use the + operator:

Example

Join two tuples:

tuple1 = ("a", "b" , "c")  
tuple2 = (1, 2, 3)  
  
tuple3 = tuple1 + tuple2  
print(tuple3)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_concat)

Multiply Tuples

If you want to multiply the content of a tuple a given number of times, you can use the \* operator:

Example

Multiply the fruits tuple by 2:

fruits = ("apple", "banana", "cherry")  
mytuple = fruits \* 2  
  
print(mytuple)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_multiply)

Python - Tuple Methods

Tuple Methods

Python has two built-in methods that you can use on tuples.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [count()](https://www.w3schools.com/python/ref_tuple_count.asp) | Returns the number of times a specified value occurs in a tuple |
| [index()](https://www.w3schools.com/python/ref_tuple_index.asp) | Searches the tuple for a specified value and returns the position of where it was found |

Python - Tuple Exercises

Test Yourself With Exercises

Now you have learned a lot about tuples, and how to use them in Python.

Are you ready for a test?

Try to insert the missing part to make the code work as expected:

Top of Form

Exercise:

Print the first item in the fruits tuple.

fruits = ("apple", "banana", "cherry")

print()

Submit Answer »

Bottom of Form

Go to the Exercise section and test all of our Python Tuple Exercises:

[Python Tuple Exercises](https://www.w3schools.com/python/exercise.asp?filename=exercise_tuples1)

Python Sets

myset = {"apple", "banana", "cherry"}

Set

Sets are used to store multiple items in a single variable.

Set is one of 4 built-in data types in Python used to store collections of data, the other 3 are [List](https://www.w3schools.com/python/python_lists.asp), [Tuple](https://www.w3schools.com/python/python_tuples.asp), and [Dictionary](https://www.w3schools.com/python/python_dictionaries.asp), all with different qualities and usage.

A set is a collection which is *unordered*, *unchangeable\**, and *unindexed*.

**\* Note:** Set *items* are unchangeable, but you can remove items and add new items.

Sets are written with curly brackets.

Example

Create a Set:

thisset = {"apple", "banana", "cherry"}  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set)

**Note:** Sets are unordered, so you cannot be sure in which order the items will appear.

Set Items

Set items are unordered, unchangeable, and do not allow duplicate values.

Unordered

Unordered means that the items in a set do not have a defined order.

Set items can appear in a different order every time you use them, and cannot be referred to by index or key.

Unchangeable

Set items are unchangeable, meaning that we cannot change the items after the set has been created.

Once a set is created, you cannot change its items, but you can remove items and add new items.

Duplicates Not Allowed

Sets cannot have two items with the same value.

Example

Duplicate values will be ignored:

thisset = {"apple", "banana", "cherry", "apple"}  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_sets_duplicate_error)

Get the Length of a Set

To determine how many items a set has, use the len() function.

Example

Get the number of items in a set:

thisset = {"apple", "banana", "cherry"}  
  
print(len(thisset))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_length)

Set Items - Data Types

Set items can be of any data type:

Example

String, int and boolean data types:

set1 = {"apple", "banana", "cherry"}  
set2 = {1, 5, 7, 9, 3}  
set3 = {True, False, False}

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_datatypes)

A set can contain different data types:

Example

A set with strings, integers and boolean values:

set1 = {"abc", 34, True, 40, "male"}

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_datatypes2)

type()

From Python's perspective, sets are defined as objects with the data type 'set':

<class 'set'>

Example

What is the data type of a set?

myset = {"apple", "banana", "cherry"}  
print(type(myset))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_datatypes3)

The set() Constructor

It is also possible to use the set() constructor to make a set.

Example

Using the set() constructor to make a set:

thisset = set(("apple", "banana", "cherry")) # note the double round-brackets  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set2)

Python Collections (Arrays)

There are four collection data types in the Python programming language:

* [**List**](https://www.w3schools.com/python/python_lists.asp) is a collection which is ordered and changeable. Allows duplicate members.
* [**Tuple**](https://www.w3schools.com/python/python_tuples.asp) is a collection which is ordered and unchangeable. Allows duplicate members.
* **Set** is a collection which is unordered, unchangeable\*, and unindexed. No duplicate members.
* [**Dictionary**](https://www.w3schools.com/python/python_dictionaries.asp) is a collection which is ordered\*\* and changeable. No duplicate members.

\*Set *items* are unchangeable, but you can remove items and add new items.

\*\*As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

When choosing a collection type, it is useful to understand the properties of that type. Choosing the right type for a particular data set could mean retention of meaning, and, it could mean an increase in efficiency or security.

Python - Access Set Items

Access Items

You cannot access items in a set by referring to an index or a key.

But you can loop through the set items using a for loop, or ask if a specified value is present in a set, by using the in keyword.

Example

Loop through the set, and print the values:

thisset = {"apple", "banana", "cherry"}  
  
for x in thisset:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_loop)

Example

Check if "banana" is present in the set:

thisset = {"apple", "banana", "cherry"}  
  
print("banana" in thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_in)

Change Items

Once a set is created, you cannot change its items, but you can add new items.

Python - Add Set Items

Add Items

Once a set is created, you cannot change its items, but you can add new items.

To add one item to a set use the add() method.

Example

Add an item to a set, using the add() method:

thisset = {"apple", "banana", "cherry"}  
  
thisset.add("orange")  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_add)

Add Sets

To add items from another set into the current set, use the update() method.

Example

Add elements from tropical into thisset:

thisset = {"apple", "banana", "cherry"}  
tropical = {"pineapple", "mango", "papaya"}  
  
thisset.update(tropical)  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_sets_update)

Add Any Iterable

The object in the update() method does not have to be a set, it can be any iterable object (tuples, lists, dictionaries etc.).

Example

Add elements of a list to at set:

thisset = {"apple", "banana", "cherry"}  
mylist = ["kiwi", "orange"]  
  
thisset.update(mylist)  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_sets_update2)

Python - Remove Set Items

Remove Item

To remove an item in a set, use the remove(), or the discard() method.

Example

Remove "banana" by using the remove() method:

thisset = {"apple", "banana", "cherry"}  
  
thisset.remove("banana")  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_remove)

**Note:** If the item to remove does not exist, remove() will raise an error.

Example

Remove "banana" by using the discard() method:

thisset = {"apple", "banana", "cherry"}  
  
thisset.discard("banana")  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_discard)

**Note:** If the item to remove does not exist, discard() will **NOT** raise an error.

You can also use the pop() method to remove an item, but this method will remove the *last* item. Remember that sets are unordered, so you will not know what item that gets removed.

The return value of the pop() method is the removed item.

Example

Remove the last item by using the pop() method:

thisset = {"apple", "banana", "cherry"}  
  
x = thisset.pop()  
  
print(x)  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_pop)

**Note:** Sets are *unordered*, so when using the pop() method, you do not know which item that gets removed.

Example

The clear() method empties the set:

thisset = {"apple", "banana", "cherry"}  
  
thisset.clear()  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_clear)

Example

The del keyword will delete the set completely:

thisset = {"apple", "banana", "cherry"}  
  
del thisset  
  
print(thisset)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_set_del)

Python - Set Methods

Set Methods

Python has a set of built-in methods that you can use on sets.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [add()](https://www.w3schools.com/python/ref_set_add.asp) | Adds an element to the set |
| [clear()](https://www.w3schools.com/python/ref_set_clear.asp) | Removes all the elements from the set |
| [copy()](https://www.w3schools.com/python/ref_set_copy.asp) | Returns a copy of the set |
| [difference()](https://www.w3schools.com/python/ref_set_difference.asp) | Returns a set containing the difference between two or more sets |
| [difference\_update()](https://www.w3schools.com/python/ref_set_difference_update.asp) | Removes the items in this set that are also included in another, specified set |
| [discard()](https://www.w3schools.com/python/ref_set_discard.asp) | Remove the specified item |
| [intersection()](https://www.w3schools.com/python/ref_set_intersection.asp) | Returns a set, that is the intersection of two other sets |
| [intersection\_update()](https://www.w3schools.com/python/ref_set_intersection_update.asp) | Removes the items in this set that are not present in other, specified set(s) |
| [isdisjoint()](https://www.w3schools.com/python/ref_set_isdisjoint.asp) | Returns whether two sets have a intersection or not |
| [issubset()](https://www.w3schools.com/python/ref_set_issubset.asp) | Returns whether another set contains this set or not |
| [issuperset()](https://www.w3schools.com/python/ref_set_issuperset.asp) | Returns whether this set contains another set or not |
| [pop()](https://www.w3schools.com/python/ref_set_pop.asp) | Removes an element from the set |
| [remove()](https://www.w3schools.com/python/ref_set_remove.asp) | Removes the specified element |
| [symmetric\_difference()](https://www.w3schools.com/python/ref_set_symmetric_difference.asp) | Returns a set with the symmetric differences of two sets |
| [symmetric\_difference\_update()](https://www.w3schools.com/python/ref_set_symmetric_difference_update.asp) | inserts the symmetric differences from this set and another |
| [union()](https://www.w3schools.com/python/ref_set_union.asp) | Return a set containing the union of sets |
| [update()](https://www.w3schools.com/python/ref_set_update.asp) | Update the set with the union of this set and others |

Python - Set Exercises

Test Yourself With Exercises

Now you have learned a lot about sets, and how to use them in Python.

Are you ready for a test?

Try to insert the missing part to make the code work as expected:

Top of Form

Exercise:

Check if "apple" is present in the fruits set.

fruits = {"apple", "banana", "cherry"}

if "apple"  fruits:

print("Yes, apple is a fruit!")

Submit Answer »

Bottom of Form

Go to the Exercise section and test all of our Python Set Exercises:

[Python Set Exercises](https://www.w3schools.com/python/exercise.asp?filename=exercise_sets1)

Python Dictionaries

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}

Dictionary

Dictionaries are used to store data values in key:value pairs.

A dictionary is a collection which is ordered\*, changeable and do not allow duplicates.

As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

Dictionaries are written with curly brackets, and have keys and values:

Example

Create and print a dictionary:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
print(thisdict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary)

Dictionary Items

Dictionary items are ordered, changeable, and does not allow duplicates.

Dictionary items are presented in key:value pairs, and can be referred to by using the key name.

Example

Print the "brand" value of the dictionary:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
print(thisdict["brand"])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_brand)

Ordered or Unordered?

As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

When we say that dictionaries are ordered, it means that the items have a defined order, and that order will not change.

Unordered means that the items does not have a defined order, you cannot refer to an item by using an index.

Changeable

Dictionaries are changeable, meaning that we can change, add or remove items after the dictionary has been created.

Duplicates Not Allowed

Dictionaries cannot have two items with the same key:

Example

Duplicate values will overwrite existing values:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964,  
  "year": 2020  
}  
print(thisdict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_duplicates)

Dictionary Length

To determine how many items a dictionary has, use the len() function:

Example

Print the number of items in the dictionary:

print(len(thisdict))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_len)

Dictionary Items - Data Types

The values in dictionary items can be of any data type:

Example

String, int, boolean, and list data types:

thisdict = {  
  "brand": "Ford",  
  "electric": False,  
  "year": 1964,  
  "colors": ["red", "white", "blue"]  
}

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_datatypes)

type()

From Python's perspective, dictionaries are defined as objects with the data type 'dict':

<class 'dict'>

Example

Print the data type of a dictionary:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
print(type(thisdict))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_datatypes2)

Python Collections (Arrays)

There are four collection data types in the Python programming language:

* [**List**](https://www.w3schools.com/python/python_lists.asp) is a collection which is ordered and changeable. Allows duplicate members.
* [**Tuple**](https://www.w3schools.com/python/python_tuples.asp) is a collection which is ordered and unchangeable. Allows duplicate members.
* [**Set**](https://www.w3schools.com/python/python_sets.asp) is a collection which is unordered, unchangeable\*, and unindexed. No duplicate members.
* **Dictionary** is a collection which is ordered\*\* and changeable. No duplicate members.

\*Set *items* are unchangeable, but you can remove and/or add items whenever you like.

\*\*As of Python version 3.7, dictionaries are *ordered*. In Python 3.6 and earlier, dictionaries are *unordered*.

When choosing a collection type, it is useful to understand the properties of that type. Choosing the right type for a particular data set could mean retention of meaning, and, it could mean an increase in efficiency or security.

Python - Access Dictionary Items

Accessing Items

You can access the items of a dictionary by referring to its key name, inside square brackets:

Example

Get the value of the "model" key:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
x = thisdict["model"]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_access)

There is also a method called get() that will give you the same result:

Example

Get the value of the "model" key:

x = thisdict.get("model")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_get)

Get Keys

The keys() method will return a list of all the keys in the dictionary.

Example

Get a list of the keys:

x = thisdict.keys()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_keys)

The list of the keys is a *view* of the dictionary, meaning that any changes done to the dictionary will be reflected in the keys list.

Example

Add a new item to the original dictionary, and see that the keys list gets updated as well:

car = {  
"brand": "Ford",  
"model": "Mustang",  
"year": 1964  
}  
  
x = car.keys()  
  
print(x) #before the change  
  
car["color"] = "white"  
  
print(x) #after the change

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_keys2)

Get Values

The values() method will return a list of all the values in the dictionary.

Example

Get a list of the values:

x = thisdict.values()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_values)

The list of the values is a *view* of the dictionary, meaning that any changes done to the dictionary will be reflected in the values list.

Example

Make a change in the original dictionary, and see that the values list gets updated as well:

car = {  
"brand": "Ford",  
"model": "Mustang",  
"year": 1964  
}  
  
x = car.values()  
  
print(x) #before the change  
  
car["year"] = 2020  
  
print(x) #after the change

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_values2)

Example

Add a new item to the original dictionary, and see that the values list gets updated as well:

car = {  
"brand": "Ford",  
"model": "Mustang",  
"year": 1964  
}  
  
x = car.values()  
  
print(x) #before the change  
  
car["color"] = "red"  
  
print(x) #after the change

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_values3)

Get Items

The items() method will return each item in a dictionary, as tuples in a list.

Example

Get a list of the key:value pairs

x = thisdict.items()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_items)

The returned list is a *view* of the items of the dictionary, meaning that any changes done to the dictionary will be reflected in the items list.

Example

Make a change in the original dictionary, and see that the items list gets updated as well:

car = {  
"brand": "Ford",  
"model": "Mustang",  
"year": 1964  
}  
  
x = car.items()  
  
print(x) #before the change  
  
car["year"] = 2020  
  
print(x) #after the change

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_items2)

Example

Add a new item to the original dictionary, and see that the items list gets updated as well:

car = {  
"brand": "Ford",  
"model": "Mustang",  
"year": 1964  
}  
  
x = car.items()  
  
print(x) #before the change  
  
car["color"] = "red"  
  
print(x) #after the change

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_items3)

Check if Key Exists

To determine if a specified key is present in a dictionary use the in keyword:

Example

Check if "model" is present in the dictionary:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
if "model" in thisdict:  
  print("Yes, 'model' is one of the keys in the thisdict dictionary")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_in)

Python - Change Dictionary Items

Change Values

You can change the value of a specific item by referring to its key name:

Example

Change the "year" to 2018:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict["year"] = 2018

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_change)

Update Dictionary

The update() method will update the dictionary with the items from the given argument.

The argument must be a dictionary, or an iterable object with key:value pairs.

Example

Update the "year" of the car by using the update() method:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict.update({"year": 2020})

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_update)

Python - Add Dictionary Items

Adding Items

Adding an item to the dictionary is done by using a new index key and assigning a value to it:

Example

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict["color"] = "red"  
print(thisdict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_add)

Update Dictionary

The update() method will update the dictionary with the items from a given argument. If the item does not exist, the item will be added.

The argument must be a dictionary, or an iterable object with key:value pairs.

Example

Add a color item to the dictionary by using the update() method:

thisdict = {  
  “brand”: “Ford”,  
  “model”: “Mustang”,  
  “year”: 1964  
}  
thisdict.update({“color”: “red”})

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dict_add2)

Python - Remove Dictionary Items

Removing Items

There are several methods to remove items from a dictionary:

Example

The pop() method removes the item with the specified key name:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict.pop("model")  
print(thisdict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_pop)

Example

The popitem() method removes the last inserted item (in versions before 3.7, a random item is removed instead):

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict.popitem()  
print(thisdict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_popitem)

Example

The del keyword removes the item with the specified key name:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
del thisdict["model"]  
print(thisdict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_del2)

Example

The del keyword can also delete the dictionary completely:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
del thisdict  
print(thisdict) #this will cause an error because "thisdict" no longer exists.

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_del3)

Example

The clear() method empties the dictionary:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
thisdict.clear()  
print(thisdict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_clear)

Python - Loop Dictionaries

Loop Through a Dictionary

You can loop through a dictionary by using a for loop.

When looping through a dictionary, the return value are the *keys* of the dictionary, but there are methods to return the *values* as well.

Example

Print all key names in the dictionary, one by one:

for x in thisdict:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop)

Example

Print all *values* in the dictionary, one by one:

for x in thisdict:  
  print(thisdict[x])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop2)

Example

You can also use the values() method to return values of a dictionary:

for x in thisdict.values():  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop_values)

Example

You can use the keys() method to return the keys of a dictionary:

for x in thisdict.keys():  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop_keys)

Example

Loop through both *keys* and *values*, by using the items() method:

for x, y in thisdict.items():  
  print(x, y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop_items)

Python - Copy Dictionaries

Copy a Dictionary

You cannot copy a dictionary simply by typing dict2 = dict1, because: dict2 will only be a *reference* to dict1, and changes made in dict1 will automatically also be made in dict2.

There are ways to make a copy, one way is to use the built-in Dictionary method copy().

Example

Make a copy of a dictionary with the copy() method:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
mydict = thisdict.copy()  
print(mydict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_copy)

Another way to make a copy is to use the built-in function dict().

Example

Make a copy of a dictionary with the dict() function:

thisdict = {  
  "brand": "Ford",  
  "model": "Mustang",  
  "year": 1964  
}  
mydict = dict(thisdict)  
print(mydict)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_copy2)

Python - Nested Dictionaries

Nested Dictionaries

A dictionary can contain dictionaries, this is called nested dictionaries.

Example

Create a dictionary that contain three dictionaries:

myfamily = {  
  "child1" : {  
    "name" : "Emil",  
    "year" : 2004  
  },  
  "child2" : {  
    "name" : "Tobias",  
    "year" : 2007  
  },  
  "child3" : {  
    "name" : "Linus",  
    "year" : 2011  
  }  
}

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_nested)

Or, if you want to add three dictionaries into a new dictionary:

Example

Create three dictionaries, then create one dictionary that will contain the other three dictionaries:

child1 = {  
  "name" : "Emil",  
  "year" : 2004  
}  
child2 = {  
  "name" : "Tobias",  
  "year" : 2007  
}  
child3 = {  
  "name" : "Linus",  
  "year" : 2011  
}  
  
myfamily = {  
  "child1" : child1,  
  "child2" : child2,  
  "child3" : child3  
}

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_nested2)

Python Dictionary Methods

Dictionary Methods

Python has a set of built-in methods that you can use on dictionaries.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [clear()](https://www.w3schools.com/python/ref_dictionary_clear.asp) | Removes all the elements from the dictionary |
| [copy()](https://www.w3schools.com/python/ref_dictionary_copy.asp) | Returns a copy of the dictionary |
| [fromkeys()](https://www.w3schools.com/python/ref_dictionary_fromkeys.asp) | Returns a dictionary with the specified keys and value |
| [get()](https://www.w3schools.com/python/ref_dictionary_get.asp) | Returns the value of the specified key |
| [items()](https://www.w3schools.com/python/ref_dictionary_items.asp) | Returns a list containing a tuple for each key value pair |
| [keys()](https://www.w3schools.com/python/ref_dictionary_keys.asp) | Returns a list containing the dictionary's keys |
| [pop()](https://www.w3schools.com/python/ref_dictionary_pop.asp) | Removes the element with the specified key |
| [popitem()](https://www.w3schools.com/python/ref_dictionary_popitem.asp) | Removes the last inserted key-value pair |
| [setdefault()](https://www.w3schools.com/python/ref_dictionary_setdefault.asp) | Returns the value of the specified key. If the key does not exist: insert the key, with the specified value |
| [update()](https://www.w3schools.com/python/ref_dictionary_update.asp) | Updates the dictionary with the specified key-value pairs |
| [values()](https://www.w3schools.com/python/ref_dictionary_values.asp) | Returns a list of all the values in the dictionary |

Python Dictionary Exercises

Test Yourself With Exercises

Now you have learned a lot about dictionaries, and how to use them in Python.

Are you ready for a test?

Try to insert the missing part to make the code work as expected:

Top of Form

Exercise:

Use the get method to print the value of the "model" key of the car dictionary.

car = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

print()

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_dictionaries1)

Bottom of Form

Go to the Exercise section and test all of our Python Dictionary Exercises:

[Python Dictionary Exercises](https://www.w3schools.com/python/exercise.asp?filename=exercise_dictionaries1)

Python If ... Else

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

If statement:

a = 33  
b = 200  
if b > a:  
  print("b is greater than a")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if2)

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):

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

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_error)

Elif

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

Example

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 »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_elif)

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.

Example

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 »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_else)

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

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

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_else2)

Short Hand If

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

Example

One line if statement:

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

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_short)

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

One line if else statement:

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

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_else_short)

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

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

Example

One line if else statement, with 3 conditions:

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

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_else_short2)

And

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

Example

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 »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_and)

Or

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

Example

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

a = 200  
b = 33  
c = 500  
if a > b or a > c:  
  print("At least one of the conditions is True")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_or)

Nested If

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

Example

x = 41  
  
if x > 10:  
  print("Above ten,")  
  if x > 20:  
    print("and also above 20!")  
  else:  
    print("but not above 20.")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_nested)

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

a = 33  
b = 200  
  
if b > a:  
  pass

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_if_pass)

Top of Form

Exercise:

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

a = 50

b = 10

 a  b

print("Hello World")

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_ifelse1)

Bottom of Form

Python While Loops

Python Loops

Python has two primitive loop commands:

* while loops
* for loops

The while Loop

With the while loop we can execute a set of statements as long as a condition is true.

Example

Print i as long as i is less than 6:

i = 1  
while i < 6:  
  print(i)  
  i += 1

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_while)

**Note:** remember to increment i, or else the loop will continue forever.

The while loop requires relevant variables to be ready, in this example we need to define an indexing variable, i, which we set to 1.

The break Statement

With the break statement we can stop the loop even if the while condition is true:

Example

Exit the loop when i is 3:

i = 1  
while i < 6:  
  print(i)  
  if i == 3:  
    break  
  i += 1

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_while_break)

The continue Statement

With the continue statement we can stop the current iteration, and continue with the next:

Example

Continue to the next iteration if i is 3:

i = 0  
while i < 6:  
  i += 1  
  if i == 3:  
    continue  
  print(i)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_while_continue)

The else Statement

With the else statement we can run a block of code once when the condition no longer is true:

Example

Print a message once the condition is false:

i = 1  
while i < 6:  
  print(i)  
  i += 1  
else:  
  print("i is no longer less than 6")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_while_else)

Top of Form

Exercise:

Print i as long as i is less than 6.

i = 1

 i < 6

print(i)

i += 1

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_while_loops1)

Python For Loops

Python For Loops

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).

This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.

With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.

Example

Print each fruit in a fruit list:

fruits = ["apple", "banana", "cherry"]  
for x in fruits:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for)

The for loop does not require an indexing variable to set beforehand.

Looping Through a String

Even strings are iterable objects, they contain a sequence of characters:

Example

Loop through the letters in the word "banana":

for x in "banana":  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_string)

The break Statement

With the break statement we can stop the loop before it has looped through all the items:

Example

Exit the loop when x is "banana":

fruits = ["apple", "banana", "cherry"]  
for x in fruits:  
  print(x)  
  if x == "banana":  
    break

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_break)

Example

Exit the loop when x is "banana", but this time the break comes before the print:

fruits = ["apple", "banana", "cherry"]  
for x in fruits:  
  if x == "banana":  
    break  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_break2)

The continue Statement

With the continue statement we can stop the current iteration of the loop, and continue with the next:

Example

Do not print banana:

fruits = ["apple", "banana", "cherry"]  
for x in fruits:  
  if x == "banana":  
    continue  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_continue)

The range() Function

To loop through a set of code a specified number of times, we can use the range() function,

The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

Example

Using the range() function:

for x in range(6):  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_range)

Note that range(6) is not the values of 0 to 6, but the values 0 to 5.

The range() function defaults to 0 as a starting value, however it is possible to specify the starting value by adding a parameter: range(2, 6), which means values from 2 to 6 (but not including 6):

Example

Using the start parameter:

for x in range(2, 6):  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_range2)

The range() function defaults to increment the sequence by 1, however it is possible to specify the increment value by adding a third parameter: range(2, 30, **3**):

Example

Increment the sequence with 3 (default is 1):

for x in range(2, 30, 3):  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_range3)

Else in For Loop

The else keyword in a for loop specifies a block of code to be executed when the loop is finished:

Example

Print all numbers from 0 to 5, and print a message when the loop has ended:

for x in range(6):  
  print(x)  
else:  
  print("Finally finished!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_else)

**Note:** The else block will NOT be executed if the loop is stopped by a break statement.

Example

Break the loop when x is 3, and see what happens with the else block:

for x in range(6):  
  if x == 3: break  
  print(x)  
else:  
  print("Finally finished!")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_else_break)

Nested Loops

A nested loop is a loop inside a loop.

The "inner loop" will be executed one time for each iteration of the "outer loop":

Example

Print each adjective for every fruit:

adj = ["red", "big", "tasty"]  
fruits = ["apple", "banana", "cherry"]  
  
for x in adj:  
  for y in fruits:  
    print(x, y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_nested)

The pass Statement

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

Example

for x in [0, 1, 2]:  
  pass

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_for_pass)

Top of Form

Exercise:

Loop through the items in the fruits list.

fruits = ["apple", "banana", "cherry"]

 x  fruits

print(x)

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_for_loops1)

Bottom of Form

Python Functions

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result.

Creating a Function

In Python a function is defined using the def keyword:

Example

def my\_function():  
  print("Hello from a function")

Calling a Function

To call a function, use the function name followed by parenthesis:

Example

def my\_function():  
  print("Hello from a function")  
  
**my\_function()**

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function)

Arguments

Information can be passed into functions as arguments.

Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

The following example has a function with one argument (fname). When the function is called, we pass along a first name, which is used inside the function to print the full name:

Example

def my\_function(**fname**):  
  print(fname + " Refsnes")  
  
my\_function(**"Emil"**)  
my\_function(**"Tobias"**)  
my\_function(**"Linus"**)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_param)

*Arguments* are often shortened to *args* in Python documentations.

Parameters or Arguments?

The terms *parameter* and *argument* can be used for the same thing: information that are passed into a function.

From a function's perspective:

A parameter is the variable listed inside the parentheses in the function definition.

An argument is the value that is sent to the function when it is called.

Number of Arguments

By default, a function must be called with the correct number of arguments. Meaning that if your function expects 2 arguments, you have to call the function with 2 arguments, not more, and not less.

Example

This function expects 2 arguments, and gets 2 arguments:

def my\_function(fname, lname):  
  print(fname + " " + lname)  
  
my\_function("Emil", "Refsnes")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_args_n)

If you try to call the function with 1 or 3 arguments, you will get an error:

Example

This function expects 2 arguments, but gets only 1:

def my\_function(fname, lname):  
  print(fname + " " + lname)  
  
my\_function("Emil")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_args_error)

Arbitrary Arguments, \*args

If you do not know how many arguments that will be passed into your function, add a \* before the parameter name in the function definition.

This way the function will receive a *tuple* of arguments, and can access the items accordingly:

Example

If the number of arguments is unknown, add a \* before the parameter name:

def my\_function(\*kids):  
  print("The youngest child is " + kids[2])  
  
my\_function("Emil", "Tobias", "Linus")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_args)

*Arbitrary Arguments* are often shortened to *\*args* in Python documentations.

Keyword Arguments

You can also send arguments with the *key* = *value* syntax.

This way the order of the arguments does not matter.

Example

def my\_function(child3, child2, child1):  
  print("The youngest child is " + child3)  
  
my\_function(child1 = "Emil", child2 = "Tobias", child3 = "Linus")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_kwargs)

The phrase *Keyword Arguments* are often shortened to *kwargs* in Python documentations.

Arbitrary Keyword Arguments, \*\*kwargs

If you do not know how many keyword arguments that will be passed into your function, add two asterisk: \*\* before the parameter name in the function definition.

This way the function will receive a *dictionary* of arguments, and can access the items accordingly:

Example

If the number of keyword arguments is unknown, add a double \*\* before the parameter name:

def my\_function(\*\*kid):  
  print("His last name is " + kid["lname"])  
  
my\_function(fname = "Tobias", lname = "Refsnes")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_kwargs_n)

*Arbitrary Kword Arguments* are often shortened to *\*\*kwargs* in Python documentations.

Default Parameter Value

The following example shows how to use a default parameter value.

If we call the function without argument, it uses the default value:

Example

def my\_function(**country = "Norway"**):  
  print("I am from " + country)  
  
my\_function("Sweden")  
my\_function("India")  
my\_function()  
my\_function("Brazil")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_param2)

Passing a List as an Argument

You can send any data types of argument to a function (string, number, list, dictionary etc.), and it will be treated as the same data type inside the function.

E.g. if you send a List as an argument, it will still be a List when it reaches the function:

Example

def my\_function(food):  
  for x in food:  
    print(x)  
  
fruits = ["apple", "banana", "cherry"]  
  
my\_function(fruits)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_param3)

Return Values

To let a function return a value, use the return statement:

Example

def my\_function(x):  
  **return 5 \* x**  
print(my\_function(3))  
print(my\_function(5))  
print(my\_function(9))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_return)

The pass Statement

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

Example

def myfunction():  
  pass

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_function_pass)

Recursion

Python also accepts function recursion, which means a defined function can call itself.

Recursion is a common mathematical and programming concept. It means that a function calls itself. This has the benefit of meaning that you can loop through data to reach a result.

The developer should be very careful with recursion as it can be quite easy to slip into writing a function which never terminates, or one that uses excess amounts of memory or processor power. However, when written correctly recursion can be a very efficient and mathematically-elegant approach to programming.

In this example, tri\_recursion() is a function that we have defined to call itself ("recurse"). We use the k variable as the data, which decrements (-1) every time we recurse. The recursion ends when the condition is not greater than 0 (i.e. when it is 0).

To a new developer it can take some time to work out how exactly this works, best way to find out is by testing and modifying it.

Example

Recursion Example

def tri\_recursion(k):  
  if(k > 0):  
    result = k + tri\_recursion(k - 1)  
    print(result)  
  else:  
    result = 0  
  return result  
  
print("\n\nRecursion Example Results")  
tri\_recursion(6)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_recursion)

Top of Form

Exercise:

Create a function named my\_function.

:

print("Hello from a function")

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_functions1)

Python Lambda

A lambda function is a small anonymous function.

A lambda function can take any number of arguments, but can only have one expression.

Syntax

lambda *arguments*: *expression*

The expression is executed and the result is returned:

Example

Add 10 to argument a, and return the result:

x = lambda a : a + 10  
print(x(5))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda)

Lambda functions can take any number of arguments:

Example

Multiply argument a with argument b and return the result:

x = lambda a, b : a \* b  
print(x(5, 6))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda2)

Example

Summarize argument a, b, and c and return the result:

x = lambda a, b, c : a + b + c  
print(x(5, 6, 2))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda3)

Why Use Lambda Functions?

The power of lambda is better shown when you use them as an anonymous function inside another function.

Say you have a function definition that takes one argument, and that argument will be multiplied with an unknown number:

def myfunc(n):  
  return lambda a : a \* n

Use that function definition to make a function that always doubles the number you send in:

Example

def myfunc(n):  
  return lambda a : a \* n  
  
mydoubler = myfunc(2)  
  
print(mydoubler(11))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda_double)

Or, use the same function definition to make a function that always *triples* the number you send in:

Example

def myfunc(n):  
  return lambda a : a \* n  
  
mytripler = myfunc(3)  
  
print(mytripler(11))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda_triple)

Or, use the same function definition to make both functions, in the same program:

Example

def myfunc(n):  
  return lambda a : a \* n  
  
mydoubler = myfunc(2)  
mytripler = myfunc(3)  
  
print(mydoubler(11))  
print(mytripler(11))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda_both)

Use lambda functions when an anonymous function is required for a short period of time.

Top of Form

Exercise:

Create a lambda function that takes one parameter (a) and returns it.

x =    

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_lambda1)

Bottom of Form

Python Arrays

**Note:** Python does not have built-in support for Arrays, but [Python Lists](https://www.w3schools.com/python/python_lists.asp) can be used instead.

Arrays

**Note:** This page shows you how to use LISTS as ARRAYS, however, to work with arrays in Python you will have to import a library, like the [NumPy library](https://www.w3schools.com/python/numpy/default.asp).

Arrays are used to store multiple values in one single variable:

Example

Create an array containing car names:

cars = ["Ford", "Volvo", "BMW"]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array1)

What is an Array?

An array is a special variable, which can hold more than one value at a time.

If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

car1 = "Ford"  
car2 = "Volvo"  
car3 = "BMW"

However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?

The solution is an array!

An array can hold many values under a single name, and you can access the values by referring to an index number.

Access the Elements of an Array

You refer to an array element by referring to the *index number*.

Example

Get the value of the first array item:

x = cars[0]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array2)

Example

Modify the value of the first array item:

cars[0] = "Toyota"

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array3)

The Length of an Array

Use the len() method to return the length of an array (the number of elements in an array).

Example

Return the number of elements in the cars array:

x = len(cars)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array4)

**Note:** The length of an array is always one more than the highest array index.

Looping Array Elements

You can use the for in loop to loop through all the elements of an array.

Example

Print each item in the cars array:

for x in cars:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array5)

Adding Array Elements

You can use the append() method to add an element to an array.

Example

Add one more element to the cars array:

cars.append("Honda")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array6)

Removing Array Elements

You can use the pop() method to remove an element from the array.

Example

Delete the second element of the cars array:

cars.pop(1)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array7)

You can also use the remove() method to remove an element from the array.

Example

Delete the element that has the value "Volvo":

cars.remove("Volvo")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_array8)

**Note:** The list's remove() method only removes the first occurrence of the specified value.

Array Methods

Python has a set of built-in methods that you can use on lists/arrays.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [append()](https://www.w3schools.com/python/ref_list_append.asp) | Adds an element at the end of the list |
| [clear()](https://www.w3schools.com/python/ref_list_clear.asp) | Removes all the elements from the list |
| [copy()](https://www.w3schools.com/python/ref_list_copy.asp) | Returns a copy of the list |
| [count()](https://www.w3schools.com/python/ref_list_count.asp) | Returns the number of elements with the specified value |
| [extend()](https://www.w3schools.com/python/ref_list_extend.asp) | Add the elements of a list (or any iterable), to the end of the current list |
| [index()](https://www.w3schools.com/python/ref_list_index.asp) | Returns the index of the first element with the specified value |
| [insert()](https://www.w3schools.com/python/ref_list_insert.asp) | Adds an element at the specified position |
| [pop()](https://www.w3schools.com/python/ref_list_pop.asp) | Removes the element at the specified position |
| [remove()](https://www.w3schools.com/python/ref_list_remove.asp) | Removes the first item with the specified value |
| [reverse()](https://www.w3schools.com/python/ref_list_reverse.asp) | Reverses the order of the list |
| [sort()](https://www.w3schools.com/python/ref_list_sort.asp) | Sorts the list |

**Note:** Python does not have built-in support for Arrays, but Python Lists can be used instead.

Python Classes and Objects

Python Classes/Objects

Python is an object-oriented programming language.

Almost everything in Python is an object, with its properties and methods.

A Class is like an object constructor, or a "blueprint" for creating objects.

Create a Class

To create a class, use the keyword class:

Example

Create a class named MyClass, with a property named x:

class MyClass:  
  x = 5

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class1)

Create Object

Now we can use the class named MyClass to create objects:

Example

Create an object named p1, and print the value of x:

p1 = MyClass()  
print(p1.x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class2)

The \_\_init\_\_() Function

The examples above are classes and objects in their simplest form, and are not really useful in real life applications.

To understand the meaning of classes we have to understand the built-in \_\_init\_\_() function.

All classes have a function called \_\_init\_\_(), which is always executed when the class is being initiated.

Use the \_\_init\_\_() function to assign values to object properties, or other operations that are necessary to do when the object is being created:

Example

Create a class named Person, use the \_\_init\_\_() function to assign values for name and age:

class Person:  
  def \_\_init\_\_(self, name, age):  
    self.name = name  
    self.age = age  
  
p1 = Person("John", 36)  
  
print(p1.name)  
print(p1.age)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class3)

**Note:** The \_\_init\_\_() function is called automatically every time the class is being used to create a new object.

The \_\_str\_\_() Function

The \_\_str\_\_() function controls what should be returned when the class object is represented as a string.

If the \_\_str\_\_() function is not set, the string representation of the object is returned:

Example

The string representation of an object WITHOUT the \_\_str\_\_() function:

class Person:  
  def \_\_init\_\_(self, name, age):  
    self.name = name  
    self.age = age  
  
p1 = Person("John", 36)  
  
print(p1)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class_str1)

Example

The string representation of an object WITH the \_\_str\_\_() function:

class Person:  
  def \_\_init\_\_(self, name, age):  
    self.name = name  
    self.age = age  
  
  def \_\_str\_\_(self):  
    return f"{self.name}({self.age})"  
  
p1 = Person("John", 36)  
  
print(p1)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class_str2)

Object Methods

Objects can also contain methods. Methods in objects are functions that belong to the object.

Let us create a method in the Person class:

Example

Insert a function that prints a greeting, and execute it on the p1 object:

class Person:  
  def \_\_init\_\_(self, name, age):  
    self.name = name  
    self.age = age  
  
  def myfunc(self):  
    print("Hello my name is " + self.name)  
  
p1 = Person("John", 36)  
p1.myfunc()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class4)

**Note:** The self parameter is a reference to the current instance of the class, and is used to access variables that belong to the class.

The self Parameter

The self parameter is a reference to the current instance of the class, and is used to access variables that belongs to the class.

It does not have to be named self , you can call it whatever you like, but it has to be the first parameter of any function in the class:

Example

Use the words *mysillyobject* and *abc* instead of *self*:

class Person:  
  def \_\_init\_\_(mysillyobject, name, age):  
    mysillyobject.name = name  
    mysillyobject.age = age  
  
  def myfunc(abc):  
    print("Hello my name is " + abc.name)  
  
p1 = Person("John", 36)  
p1.myfunc()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class5)

Modify Object Properties

You can modify properties on objects like this:

Example

Set the age of p1 to 40:

p1.age = 40

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class6)

Delete Object Properties

You can delete properties on objects by using the del keyword:

Example

Delete the age property from the p1 object:

del p1.age

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class7)

Delete Objects

You can delete objects by using the del keyword:

Example

Delete the p1 object:

del p1

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class8)

The pass Statement

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

Example

class Person:  
  pass

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_class_pass)

Top of Form

Exercise:

Create a class named MyClass:

 MyClass:

x = 5

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_classes1)

Python Inheritance

Python Inheritance

Inheritance allows us to define a class that inherits all the methods and properties from another class.

**Parent class** is the class being inherited from, also called base class.

**Child class** is the class that inherits from another class, also called derived class.

Create a Parent Class

Any class can be a parent class, so the syntax is the same as creating any other class:

Example

Create a class named Person, with firstname and lastname properties, and a printname method:

class Person:  
  def \_\_init\_\_(self, fname, lname):  
    self.firstname = fname  
    self.lastname = lname  
  
  def printname(self):  
    print(self.firstname, self.lastname)  
  
#Use the Person class to create an object, and then execute the printname method:  
  
x = Person("John", "Doe")  
x.printname()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_inheritance_parent)

Create a Child Class

To create a class that inherits the functionality from another class, send the parent class as a parameter when creating the child class:

Example

Create a class named Student, which will inherit the properties and methods from the Person class:

class Student(Person):  
  pass

**Note:** Use the pass keyword when you do not want to add any other properties or methods to the class.

Now the Student class has the same properties and methods as the Person class.

Example

Use the Student class to create an object, and then execute the printname method:

x = Student("Mike", "Olsen")  
x.printname()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_inheritance_child)

Add the \_\_init\_\_() Function

So far we have created a child class that inherits the properties and methods from its parent.

We want to add the \_\_init\_\_() function to the child class (instead of the pass keyword).

**Note:** The \_\_init\_\_() function is called automatically every time the class is being used to create a new object.

Example

Add the \_\_init\_\_() function to the Student class:

class Student(Person):  
  def \_\_init\_\_(self, fname, lname):  
    #add properties etc.

When you add the \_\_init\_\_() function, the child class will no longer inherit the parent's \_\_init\_\_() function.

**Note:** The child's \_\_init\_\_() function **overrides** the inheritance of the parent's \_\_init\_\_() function.

To keep the inheritance of the parent's \_\_init\_\_() function, add a call to the parent's \_\_init\_\_() function:

Example

class Student(Person):  
  def \_\_init\_\_(self, fname, lname):  
    Person.\_\_init\_\_(self, fname, lname)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_inheritance_init)

Now we have successfully added the \_\_init\_\_() function, and kept the inheritance of the parent class, and we are ready to add functionality in the \_\_init\_\_() function.

Use the super() Function

Python also has a super() function that will make the child class inherit all the methods and properties from its parent:

Example

class Student(Person):  
  def \_\_init\_\_(self, fname, lname):  
    super().\_\_init\_\_(fname, lname)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_inheritance_super)

By using the super() function, you do not have to use the name of the parent element, it will automatically inherit the methods and properties from its parent.

Add Properties

Example

Add a property called graduationyear to the Student class:

class Student(Person):  
  def \_\_init\_\_(self, fname, lname):  
    super().\_\_init\_\_(fname, lname)  
    self.graduationyear = 2019

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_inheritance_add_prop)

In the example below, the year 2019 should be a variable, and passed into the Student class when creating student objects. To do so, add another parameter in the \_\_init\_\_() function:

Example

Add a year parameter, and pass the correct year when creating objects:

class Student(Person):  
  def \_\_init\_\_(self, fname, lname, year):  
    super().\_\_init\_\_(fname, lname)  
    self.graduationyear = year  
  
x = Student("Mike", "Olsen", 2019)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_inheritance_add_prop2)

Add Methods

Example

Add a method called welcome to the Student class:

class Student(Person):  
  def \_\_init\_\_(self, fname, lname, year):  
    super().\_\_init\_\_(fname, lname)  
    self.graduationyear = year  
  
  def welcome(self):  
    print("Welcome", self.firstname, self.lastname, "to the class of", self.graduationyear)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_inheritance_add_method)

If you add a method in the child class with the same name as a function in the parent class, the inheritance of the parent method will be overridden.

Exercise:

What is the correct syntax to create a class named Student that will inherit properties and methods from a class named Person?

class :

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_inheritance1)

Bottom of Form

Python Iterators

Python Iterators

An iterator is an object that contains a countable number of values.

An iterator is an object that can be iterated upon, meaning that you can traverse through all the values.

Technically, in Python, an iterator is an object which implements the iterator protocol, which consist of the methods \_\_iter\_\_() and \_\_next\_\_().

Iterator vs Iterable

Lists, tuples, dictionaries, and sets are all iterable objects. They are iterable *containers* which you can get an iterator from.

All these objects have a iter() method which is used to get an iterator:

Example

Return an iterator from a tuple, and print each value:

mytuple = ("apple", "banana", "cherry")  
myit = iter(mytuple)  
  
print(next(myit))  
print(next(myit))  
print(next(myit))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator)

Even strings are iterable objects, and can return an iterator:

Example

Strings are also iterable objects, containing a sequence of characters:

mystr = "banana"  
myit = iter(mystr)  
  
print(next(myit))  
print(next(myit))  
print(next(myit))  
print(next(myit))  
print(next(myit))  
print(next(myit))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator2)

Looping Through an Iterator

We can also use a for loop to iterate through an iterable object:

Example

Iterate the values of a tuple:

mytuple = ("apple", "banana", "cherry")  
  
for x in mytuple:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator_loop)

Example

Iterate the characters of a string:

mystr = "banana"  
  
for x in mystr:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator_loop2)

The for loop actually creates an iterator object and executes the next() method for each loop.

Create an Iterator

To create an object/class as an iterator you have to implement the methods \_\_iter\_\_() and \_\_next\_\_() to your object.

As you have learned in the [Python Classes/Objects](https://www.w3schools.com/python/python_classes.asp) chapter, all classes have a function called \_\_init\_\_(), which allows you to do some initializing when the object is being created.

The \_\_iter\_\_() method acts similar, you can do operations (initializing etc.), but must always return the iterator object itself.

The \_\_next\_\_() method also allows you to do operations, and must return the next item in the sequence.

Example

Create an iterator that returns numbers, starting with 1, and each sequence will increase by one (returning 1,2,3,4,5 etc.):

class MyNumbers:  
  def \_\_iter\_\_(self):  
    self.a = 1  
    return self  
  
  def \_\_next\_\_(self):  
    x = self.a  
    self.a += 1  
    return x  
  
myclass = MyNumbers()  
myiter = iter(myclass)  
  
print(next(myiter))  
print(next(myiter))  
print(next(myiter))  
print(next(myiter))  
print(next(myiter))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator_create)

StopIteration

The example above would continue forever if you had enough next() statements, or if it was used in a for loop.

To prevent the iteration to go on forever, we can use the StopIteration statement.

In the \_\_next\_\_() method, we can add a terminating condition to raise an error if the iteration is done a specified number of times:

Example

Stop after 20 iterations:

class MyNumbers:  
  def \_\_iter\_\_(self):  
    self.a = 1  
    return self  
  
  def \_\_next\_\_(self):  
    if self.a <= 20:  
      x = self.a  
      self.a += 1  
      return x  
    else:  
      raise StopIteration  
  
myclass = MyNumbers()  
myiter = iter(myclass)  
  
for x in myiter:  
  print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator_create2)

Python Scope

A variable is only available from inside the region it is created. This is called **scope**.

Local Scope

A variable created inside a function belongs to the *local scope* of that function, and can only be used inside that function.

Example

A variable created inside a function is available inside that function:

def myfunc():  
  x = 300  
  print(x)  
  
myfunc()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_scope1)

Function Inside Function

As explained in the example above, the variable x is not available outside the function, but it is available for any function inside the function:

Example

The local variable can be accessed from a function within the function:

def myfunc():  
  x = 300  
  def myinnerfunc():  
    print(x)  
  myinnerfunc()  
  
myfunc()

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_scope2)

Global Scope

A variable created in the main body of the Python code is a global variable and belongs to the global scope.

Global variables are available from within any scope, global and local.

Example

A variable created outside of a function is global and can be used by anyone:

x = 300  
  
def myfunc():  
  print(x)  
  
myfunc()  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_scope3)

Naming Variables

If you operate with the same variable name inside and outside of a function, Python will treat them as two separate variables, one available in the global scope (outside the function) and one available in the local scope (inside the function):

Example

The function will print the local x, and then the code will print the global x:

x = 300  
  
def myfunc():  
  x = 200  
  print(x)  
  
myfunc()  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_scope4)

Global Keyword

If you need to create a global variable, but are stuck in the local scope, you can use the global keyword.

The global keyword makes the variable global.

Example

If you use the global keyword, the variable belongs to the global scope:

def myfunc():  
  global x  
  x = 300  
  
myfunc()  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_scope5)

Also, use the global keyword if you want to make a change to a global variable inside a function.

Example

To change the value of a global variable inside a function, refer to the variable by using the global keyword:

x = 300  
  
def myfunc():  
  global x  
  x = 200  
  
myfunc()  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_scope6)

Python Modules

What is a Module?

Consider a module to be the same as a code library.

A file containing a set of functions you want to include in your application.

Create a Module

To create a module just save the code you want in a file with the file extension .py:

Example

Save this code in a file named mymodule.py

def greeting(name):  
  print("Hello, " + name)

Use a Module

Now we can use the module we just created, by using the import statement:

Example

Import the module named mymodule, and call the greeting function:

import mymodule  
  
mymodule.greeting("Jonathan")

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_module1)

**Note:** When using a function from a module, use the syntax: *module\_name.function\_name*.

Variables in Module

The module can contain functions, as already described, but also variables of all types (arrays, dictionaries, objects etc):

Example

Save this code in the file mymodule.py

person1 = {  
  "name": "John",  
  "age": 36,  
  "country": "Norway"  
}

Example

Import the module named mymodule, and access the person1 dictionary:

import mymodule  
  
a = mymodule.person1["age"]  
print(a)

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_module2)

Naming a Module

You can name the module file whatever you like, but it must have the file extension .py

Re-naming a Module

You can create an alias when you import a module, by using the as keyword:

Example

Create an alias for mymodule called mx:

import mymodule as mx  
  
a = mx.person1["age"]  
print(a)

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_module3)

Built-in Modules

There are several built-in modules in Python, which you can import whenever you like.

Example

Import and use the platform module:

import platform  
  
x = platform.system()  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_module4)

Using the dir() Function

There is a built-in function to list all the function names (or variable names) in a module. The dir() function:

Example

List all the defined names belonging to the platform module:

import platform  
  
x = dir(platform)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_module5)

**Note:** The dir() function can be used on *all* modules, also the ones you create yourself.

Import From Module

You can choose to import only parts from a module, by using the from keyword.

Example

The module named mymodule has one function and one dictionary:

def greeting(name):  
  print("Hello, " + name)  
  
person1 = {  
  "name": "John",  
  "age": 36,  
  "country": "Norway"  
}

Example

Import only the person1 dictionary from the module:

from mymodule import person1  
  
print (person1["age"])

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_module6)

**Note:** When importing using the from keyword, do not use the module name when referring to elements in the module. Example: person1["age"], **not** ~~mymodule.person1["age"]~~

Top of Form

Exercise:

What is the correct syntax to import a module named "mymodule"?

 mymodule

Submit Answer »

[Start the Exercise](https://www.w3schools.com/python/exercise.asp?filename=exercise_modules1)

Python Datetime

Python Dates

A date in Python is not a data type of its own, but we can import a module named datetime to work with dates as date objects.

Example

Import the datetime module and display the current date:

import datetime  
  
x = datetime.datetime.now()  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime1)

Date Output

When we execute the code from the example above the result will be:

2022-10-01 12:20:39.877486

The date contains year, month, day, hour, minute, second, and microsecond.

The datetime module has many methods to return information about the date object.

Here are a few examples, you will learn more about them later in this chapter:

Example

Return the year and name of weekday:

import datetime  
  
x = datetime.datetime.now()  
  
print(x.year)  
print(x.strftime("%A"))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime2)

Creating Date Objects

To create a date, we can use the datetime() class (constructor) of the datetime module.

The datetime() class requires three parameters to create a date: year, month, day.

Example

Create a date object:

import datetime  
  
x = datetime.datetime(2020, 5, 17)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime3)

The datetime() class also takes parameters for time and timezone (hour, minute, second, microsecond, tzone), but they are optional, and has a default value of 0, (None for timezone).

The strftime() Method

The datetime object has a method for formatting date objects into readable strings.

The method is called strftime(), and takes one parameter, format, to specify the format of the returned string:

Example

Display the name of the month:

import datetime  
  
x = datetime.datetime(2018, 6, 1)  
  
print(x.strftime("%B"))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime)

A reference of all the legal format codes:

|  |  |  |  |
| --- | --- | --- | --- |
| **Directive** | **Description** | **Example** | **Try it** |
| %a | Weekday, short version | Wed | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_a) |
| %A | Weekday, full version | Wednesday | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_a2) |
| %w | Weekday as a number 0-6, 0 is Sunday | 3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_w) |
| %d | Day of month 01-31 | 31 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_d) |
| %b | Month name, short version | Dec | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_b) |
| %B | Month name, full version | December | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_b2) |
| %m | Month as a number 01-12 | 12 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_m) |
| %y | Year, short version, without century | 18 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_y) |
| %Y | Year, full version | 2018 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_y2) |
| %H | Hour 00-23 | 17 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_h2) |
| %I | Hour 00-12 | 05 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_i2) |
| %p | AM/PM | PM | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_p) |
| %M | Minute 00-59 | 41 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_m2) |
| %S | Second 00-59 | 08 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_s2) |
| %f | Microsecond 000000-999999 | 548513 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_f) |
| %z | UTC offset | +0100 |  |
| %Z | Timezone | CST |  |
| %j | Day number of year 001-366 | 365 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_j) |
| %U | Week number of year, Sunday as the first day of week, 00-53 | 52 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_u2) |
| %W | Week number of year, Monday as the first day of week, 00-53 | 52 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_w2) |
| %c | Local version of date and time | Mon Dec 31 17:41:00 2018 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_c) |
| %C | Century | 20 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_century) |
| %x | Local version of date | 12/31/18 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_x) |
| %X | Local version of time | 17:41:00 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_x2) |
| %% | A % character | % | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_percent) |
| %G | ISO 8601 year | 2018 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_g) |
| %u | ISO 8601 weekday (1-7) | 1 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_u) |
| %V | ISO 8601 weeknumber (01-53) | 01 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime_v) |

Python Math

Python has a set of built-in math functions, including an extensive math module, that allows you to perform mathematical tasks on numbers.

Built-in Math Functions

The min() and max() functions can be used to find the lowest or highest value in an iterable:

Example

x = min(5, 10, 25)  
y = max(5, 10, 25)  
  
print(x)  
print(y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_math_min_max)

The abs() function returns the absolute (positive) value of the specified number:

Example

x = abs(-7.25)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_math_abs)

The pow(*x*, *y*) function returns the value of x to the power of y (xy).

Example

Return the value of 4 to the power of 3 (same as 4 \* 4 \* 4):

x = pow(4, 3)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_math_pow)

The Math Module

Python has also a built-in module called math, which extends the list of mathematical functions.

To use it, you must import the math module:

import math

When you have imported the math module, you can start using methods and constants of the module.

The math.sqrt() method for example, returns the square root of a number:

Example

import math  
  
x = math.sqrt(64)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_math_sqrt)

The math.ceil() method rounds a number upwards to its nearest integer, and the math.floor() method rounds a number downwards to its nearest integer, and returns the result:

Example

import math  
  
x = math.ceil(1.4)  
y = math.floor(1.4)  
  
print(x) # returns 2  
print(y) # returns 1

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_math_ceil_floor)

The math.pi constant, returns the value of PI (3.14...):

Example

import math  
  
x = math.pi  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_math_pi)

Complete Math Module Reference

In our [Math Module Reference](https://www.w3schools.com/python/module_math.asp) you will find a complete reference of all methods and constants that belongs to the Math module.

Python JSON

JSON is a syntax for storing and exchanging data.

JSON is text, written with JavaScript object notation.

JSON in Python

Python has a built-in package called json, which can be used to work with JSON data.

Example

Import the json module:

import json

Parse JSON - Convert from JSON to Python

If you have a JSON string, you can parse it by using the json.loads() method.

The result will be a [Python dictionary](https://www.w3schools.com/python/python_dictionaries.asp).

Example

Convert from JSON to Python:

import json  
  
# some JSON:  
x =  '{ "name":"John", "age":30, "city":"New York"}'  
  
# parse x:  
y = json.loads(x)  
  
# the result is a Python dictionary:  
print(y["age"])

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_json)

Convert from Python to JSON

If you have a Python object, you can convert it into a JSON string by using the json.dumps() method.

Example

Convert from Python to JSON:

import json  
  
# a Python object (dict):  
x = {  
  "name": "John",  
  "age": 30,  
  "city": "New York"  
}  
  
# convert into JSON:  
y = json.dumps(x)  
  
# the result is a JSON string:  
print(y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python)

You can convert Python objects of the following types, into JSON strings:

* dict
* list
* tuple
* string
* int
* float
* True
* False
* None

Example

Convert Python objects into JSON strings, and print the values:

import json  
  
print(json.dumps({"name": "John", "age": 30}))  
print(json.dumps(["apple", "bananas"]))  
print(json.dumps(("apple", "bananas")))  
print(json.dumps("hello"))  
print(json.dumps(42))  
print(json.dumps(31.76))  
print(json.dumps(True))  
print(json.dumps(False))  
print(json.dumps(None))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_all)

When you convert from Python to JSON, Python objects are converted into the JSON (JavaScript) equivalent:

|  |  |
| --- | --- |
| **Python** | **JSON** |
| dict | Object |
| list | Array |
| tuple | Array |
| str | String |
| int | Number |
| float | Number |
| True | true |
| False | false |
| None | null |

Example

Convert a Python object containing all the legal data types:

import json  
  
x = {  
  "name": "John",  
  "age": 30,  
  "married": True,  
  "divorced": False,  
  "children": ("Ann","Billy"),  
  "pets": None,  
  "cars": [  
    {"model": "BMW 230", "mpg": 27.5},  
    {"model": "Ford Edge", "mpg": 24.1}  
  ]  
}  
  
print(json.dumps(x))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_all_in_one)

Format the Result

The example above prints a JSON string, but it is not very easy to read, with no indentations and line breaks.

The json.dumps() method has parameters to make it easier to read the result:

Example

Use the indent parameter to define the numbers of indents:

json.dumps(x, indent=4)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_indent)

You can also define the separators, default value is (", ", ": "), which means using a comma and a space to separate each object, and a colon and a space to separate keys from values:

Example

Use the separators parameter to change the default separator:

json.dumps(x, indent=4, separators=(". ", " = "))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_separators)

Order the Result

The json.dumps() method has parameters to order the keys in the result:

Example

Use the sort\_keys parameter to specify if the result should be sorted or not:

json.dumps(x, indent=4, sort\_keys=True)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_sort_keys)

Python RegEx

A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern.

RegEx can be used to check if a string contains the specified search pattern.

RegEx Module

Python has a built-in package called re, which can be used to work with Regular Expressions.

Import the re module:

import re

RegEx in Python

When you have imported the re module, you can start using regular expressions:

Example

Search the string to see if it starts with "The" and ends with "Spain":

import re  
  
txt = "The rain in Spain"  
x = re.search("^The.\*Spain$", txt)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex)

RegEx Functions

The re module offers a set of functions that allows us to search a string for a match:

|  |  |
| --- | --- |
| **Function** | **Description** |
| [findall](https://www.w3schools.com/python/python_regex.asp#findall) | Returns a list containing all matches |
| [search](https://www.w3schools.com/python/python_regex.asp#search) | Returns a [Match object](https://www.w3schools.com/python/python_regex.asp#matchobject) if there is a match anywhere in the string |
| [split](https://www.w3schools.com/python/python_regex.asp#split) | Returns a list where the string has been split at each match |
| [sub](https://www.w3schools.com/python/python_regex.asp#sub) | Replaces one or many matches with a string |

Metacharacters

Metacharacters are characters with a special meaning:

|  |  |  |  |
| --- | --- | --- | --- |
| **Character** | **Description** | **Example** | **Try it** |
| [] | A set of characters | "[a-m]" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta1) |
| \ | Signals a special sequence (can also be used to escape special characters) | "\d" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta2) |
| . | Any character (except newline character) | "he..o" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta3) |
| ^ | Starts with | "^hello" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta4) |
| $ | Ends with | "planet$" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta5) |
| \* | Zero or more occurrences | "he.\*o" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta6) |
| + | One or more occurrences | "he.+o" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta7) |
| ? | Zero or one occurrences | "he.?o" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta10) |
| {} | Exactly the specified number of occurrences | "he.{2}o" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta8) |
| | | Either or | "falls|stays" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_meta9) |
| () | Capture and group |  |  |

Special Sequences

A special sequence is a \ followed by one of the characters in the list below, and has a special meaning:

|  |  |  |  |
| --- | --- | --- | --- |
| **Character** | **Description** | **Example** | **Try it** |
| \A | Returns a match if the specified characters are at the beginning of the string | "\AThe" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq1) |
| \b | Returns a match where the specified characters are at the beginning or at the end of a word (the "r" in the beginning is making sure that the string is being treated as a "raw string") | r"\bain" r"ain\b" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq2) [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq2-2) |
| \B | Returns a match where the specified characters are present, but NOT at the beginning (or at the end) of a word (the "r" in the beginning is making sure that the string is being treated as a "raw string") | r"\Bain" r"ain\B" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq3) [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq3-2) |
| \d | Returns a match where the string contains digits (numbers from 0-9) | "\d" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq4) |
| \D | Returns a match where the string DOES NOT contain digits | "\D" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq5) |
| \s | Returns a match where the string contains a white space character | "\s" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq6) |
| \S | Returns a match where the string DOES NOT contain a white space character | "\S" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq7) |
| \w | Returns a match where the string contains any word characters (characters from a to Z, digits from 0-9, and the underscore \_ character) | "\w" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq8) |
| \W | Returns a match where the string DOES NOT contain any word characters | "\W" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq9) |
| \Z | Returns a match if the specified characters are at the end of the string | "Spain\Z" | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_seq10) |

Sets

A set is a set of characters inside a pair of square brackets [] with a special meaning:

|  |  |  |
| --- | --- | --- |
| **Set** | **Description** | **Try it** |
| [arn] | Returns a match where one of the specified characters (a, r, or n) is present | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set1) |
| [a-n] | Returns a match for any lower case character, alphabetically between a and n | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set2) |
| [^arn] | Returns a match for any character EXCEPT a, r, and n | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set3) |
| [0123] | Returns a match where any of the specified digits (0, 1, 2, or 3) are present | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set4) |
| [0-9] | Returns a match for any digit between 0 and 9 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set5) |
| [0-5][0-9] | Returns a match for any two-digit numbers from 00 and 59 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set6) |
| [a-zA-Z] | Returns a match for any character alphabetically between a and z, lower case OR upper case | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set7) |
| [+] | In sets, +, \*, ., |, (), $,{} has no special meaning, so [+] means: return a match for any + character in the string | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_set8) |

The findall() Function

The findall() function returns a list containing all matches.

Example

Print a list of all matches:

import re  
  
txt = "The rain in Spain"  
x = re.findall("ai", txt)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_findall)

The list contains the matches in the order they are found.

If no matches are found, an empty list is returned:

Example

Return an empty list if no match was found:

import re  
  
txt = "The rain in Spain"  
x = re.findall("Portugal", txt)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_findall2)

The search() Function

The search() function searches the string for a match, and returns a [Match object](https://www.w3schools.com/python/python_regex.asp#matchobject) if there is a match.

If there is more than one match, only the first occurrence of the match will be returned:

Example

Search for the first white-space character in the string:

import re  
  
txt = "The rain in Spain"  
x = re.search("\s", txt)  
  
print("The first white-space character is located in position:", x.start())

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_search)

If no matches are found, the value None is returned:

Example

Make a search that returns no match:

import re  
  
txt = "The rain in Spain"  
x = re.search("Portugal", txt)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_search2)

The split() Function

The split() function returns a list where the string has been split at each match:

Example

Split at each white-space character:

import re  
  
txt = "The rain in Spain"  
x = re.split("\s", txt)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_split)

You can control the number of occurrences by specifying the maxsplit parameter:

Example

Split the string only at the first occurrence:

import re  
  
txt = "The rain in Spain"  
x = re.split("\s", txt, 1)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_split2)

The sub() Function

The sub() function replaces the matches with the text of your choice:

Example

Replace every white-space character with the number 9:

import re  
  
txt = "The rain in Spain"  
x = re.sub("\s", "9", txt)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_sub)

You can control the number of replacements by specifying the count parameter:

Example

Replace the first 2 occurrences:

import re  
  
txt = "The rain in Spain"  
x = re.sub("\s", "9", txt, 2)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_sub2)

Match Object

A Match Object is an object containing information about the search and the result.

**Note:** If there is no match, the value None will be returned, instead of the Match Object.

Example

Do a search that will return a Match Object:

import re  
  
txt = "The rain in Spain"  
x = re.search("ai", txt)  
print(x) #this will print an object

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_match)

The Match object has properties and methods used to retrieve information about the search, and the result:

.span() returns a tuple containing the start-, and end positions of the match.  
.string returns the string passed into the function  
.group() returns the part of the string where there was a match

Example

Print the position (start- and end-position) of the first match occurrence.

The regular expression looks for any words that starts with an upper case "S":

import re  
  
txt = "The rain in Spain"  
x = re.search(r"\bS\w+", txt)  
print(**x.span()**)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_match_span)

Example

Print the string passed into the function:

import re  
  
txt = "The rain in Spain"  
x = re.search(r"\bS\w+", txt)  
print(**x.string**)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_match_string)

Example

Print the part of the string where there was a match.

The regular expression looks for any words that starts with an upper case "S":

import re  
  
txt = "The rain in Spain"  
x = re.search(r"\bS\w+", txt)  
print(**x.group()**)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_match_group)

**Note:** If there is no match, the value None will be returned, instead of the Match Object.

Python PIP

What is PIP?

PIP is a package manager for Python packages, or modules if you like.

**Note:** If you have Python version 3.4 or later, PIP is included by default.

What is a Package?

A package contains all the files you need for a module.

Modules are Python code libraries you can include in your project.

Check if PIP is Installed

Navigate your command line to the location of Python's script directory, and type the following:

Example

Check PIP version:

C:\Users\*Your Name*\AppData\Local\Programs\Python\Python36-32\Scripts>pip --version

Install PIP

If you do not have PIP installed, you can download and install it from this page: <https://pypi.org/project/pip/>

Download a Package

Downloading a package is very easy.

Open the command line interface and tell PIP to download the package you want.

Navigate your command line to the location of Python's script directory, and type the following:

Example

Download a package named "camelcase":

C:\Users\*Your Name*\AppData\Local\Programs\Python\Python36-32\Scripts>pip install camelcase

Now you have downloaded and installed your first package!

Using a Package

Once the package is installed, it is ready to use.

Import the "camelcase" package into your project.

Example

Import and use "camelcase":

import camelcase  
  
c = camelcase.CamelCase()  
  
txt = "hello world"  
  
print(c.hump(txt))

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_camelcase)

Find Packages

Find more packages at <https://pypi.org/>.

Remove a Package

Use the uninstall command to remove a package:

Example

Uninstall the package named "camelcase":

C:\Users\*Your Name*\AppData\Local\Programs\Python\Python36-32\Scripts>pip uninstall camelcase

The PIP Package Manager will ask you to confirm that you want to remove the camelcase package:

Uninstalling camelcase-02.1:  
  Would remove:  
    c:\users\*Your Name*\appdata\local\programs\python\python36-32\lib\site-packages\camecase-0.2-py3.6.egg-info  
    c:\users\*Your Name*\appdata\local\programs\python\python36-32\lib\site-packages\camecase\\*  
Proceed (y/n)?

Press y and the package will be removed.

List Packages

Use the list command to list all the packages installed on your system:

Example

List installed packages:

C:\Users\*Your Name*\AppData\Local\Programs\Python\Python36-32\Scripts>pip list

Result:

Package         Version  
-----------------------  
camelcase       0.2  
mysql-connector 2.1.6  
pip             18.1  
pymongo         3.6.1  
setuptools      39.0.1

Python Try Except

The try block lets you test a block of code for errors.

The except block lets you handle the error.

The else block lets you execute code when there is no error.

The finally block lets you execute code, regardless of the result of the try- and except blocks.

Exception Handling

When an error occurs, or exception as we call it, Python will normally stop and generate an error message.

These exceptions can be handled using the try statement:

Example

The try block will generate an exception, because x is not defined:

try:  
  print(x)  
except:  
  print("An exception occurred")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except)

Since the try block raises an error, the except block will be executed.

Without the try block, the program will crash and raise an error:

Example

This statement will raise an error, because x is not defined:

print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except_error)

Many Exceptions

You can define as many exception blocks as you want, e.g. if you want to execute a special block of code for a special kind of error:

Example

Print one message if the try block raises a NameError and another for other errors:

try:  
  print(x)  
except NameError:  
  print("Variable x is not defined")  
except:  
  print("Something else went wrong")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except2)

Else

You can use the else keyword to define a block of code to be executed if no errors were raised:

Example

In this example, the try block does not generate any error:

try:  
  print("Hello")  
except:  
  print("Something went wrong")  
else:  
  print("Nothing went wrong")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except3)

Finally

The finally block, if specified, will be executed regardless if the try block raises an error or not.

Example

try:  
  print(x)  
except:  
  print("Something went wrong")  
finally:  
  print("The 'try except' is finished")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except4)

This can be useful to close objects and clean up resources:

Example

Try to open and write to a file that is not writable:

try:  
  f = open("demofile.txt")  
  try:  
    f.write("Lorum Ipsum")  
  except:  
    print("Something went wrong when writing to the file")  
  finally:  
    f.close()  
except:  
  print("Something went wrong when opening the file")

[Try it Yourself »](https://www.w3schools.com/python/showpython.asp?filename=demo_try_except5)

The program can continue, without leaving the file object open.

Raise an exception

As a Python developer you can choose to throw an exception if a condition occurs.

To throw (or raise) an exception, use the raise keyword.

Example

Raise an error and stop the program if x is lower than 0:

x = -1  
  
if x < 0:  
  raise Exception("Sorry, no numbers below zero")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ref_keyword_raise)

The raise keyword is used to raise an exception.

You can define what kind of error to raise, and the text to print to the user.

Example

Raise a TypeError if x is not an integer:

x = "hello"  
  
if not type(x) is int:  
  raise TypeError("Only integers are allowed")

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ref_keyword_raise2)

Python User Input

User Input

Python allows for user input.

That means we are able to ask the user for input.

The method is a bit different in Python 3.6 than Python 2.7.

Python 3.6 uses the input() method.

Python 2.7 uses the raw\_input() method.

The following example asks for the username, and when you entered the username, it gets printed on the screen:

Python 3.6

username = input("Enter username:")  
print("Username is: " + username)

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_user_input3)

Python 2.7

username = raw\_input("Enter username:")  
print("Username is: " + username)

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_user_input2)

Python stops executing when it comes to the input() function, and continues when the user has given some input.

Python String Formatting

To make sure a string will display as expected, we can format the result with the format() method.

String format()

The format() method allows you to format selected parts of a string.

Sometimes there are parts of a text that you do not control, maybe they come from a database, or user input?

To control such values, add placeholders (curly brackets {}) in the text, and run the values through the format() method:

Example

Add a placeholder where you want to display the price:

price = 49  
txt = "The price is {} dollars"  
print(txt.format(price))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_formatting1)

You can add parameters inside the curly brackets to specify how to convert the value:

Example

Format the price to be displayed as a number with two decimals:

txt = "The price is {:.2f} dollars"

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_formatting2)

Check out all formatting types in our [String format() Reference](https://www.w3schools.com/python/ref_string_format.asp).

Multiple Values

If you want to use more values, just add more values to the format() method:

print(txt.format(price, itemno, count))

And add more placeholders:

Example

quantity = 3  
itemno = 567  
price = 49  
myorder = "I want {} pieces of item number {} for {:.2f} dollars."  
print(myorder.format(quantity, itemno, price))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_formatting3)

Index Numbers

You can use index numbers (a number inside the curly brackets {0}) to be sure the values are placed in the correct placeholders:

Example

quantity = 3  
itemno = 567  
price = 49  
myorder = "I want {0} pieces of item number {1} for {2:.2f} dollars."  
print(myorder.format(quantity, itemno, price))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_formatting4)

Also, if you want to refer to the same value more than once, use the index number:

Example

age = 36  
name = "John"  
txt = "His name is {1}. {1} is {0} years old."  
print(txt.format(age, name))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_formatting5)

Named Indexes

You can also use named indexes by entering a name inside the curly brackets {carname}, but then you must use names when you pass the parameter values txt.format(carname = "Ford"):

Example

myorder = "I have a {carname}, it is a {model}."  
print(myorder.format(carname = "Ford", model = "Mustang"))

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_string_formatting6)

Python File Open

File handling is an important part of any web application.

Python has several functions for creating, reading, updating, and deleting files.

File Handling

The key function for working with files in Python is the open() function.

The open() function takes two parameters; *filename*, and *mode*.

There are four different methods (modes) for opening a file:

"r" - Read - Default value. Opens a file for reading, error if the file does not exist

"a" - Append - Opens a file for appending, creates the file if it does not exist

"w" - Write - Opens a file for writing, creates the file if it does not exist

"x" - Create - Creates the specified file, returns an error if the file exists

In addition you can specify if the file should be handled as binary or text mode

"t" - Text - Default value. Text mode

"b" - Binary - Binary mode (e.g. images)

Syntax

To open a file for reading it is enough to specify the name of the file:

f = open("demofile.txt")

The code above is the same as:

f = open("demofile.txt", "rt")

Because "r" for read, and "t" for text are the default values, you do not need to specify them.

**Note:** Make sure the file exists, or else you will get an error.

Python File Open

Open a File on the Server

Assume we have the following file, located in the same folder as Python:

demofile.txt

Hello! Welcome to demofile.txt  
This file is for testing purposes.  
Good Luck!

To open the file, use the built-in open() function.

The open() function returns a file object, which has a read() method for reading the content of the file:

Example

f = open("demofile.txt", "r")  
print(f.read())

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_open)

If the file is located in a different location, you will have to specify the file path, like this:

Example

Open a file on a different location:

f = open("D:\\myfiles\welcome.txt", "r")  
print(f.read())

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_open_d)

Read Only Parts of the File

By default the read() method returns the whole text, but you can also specify how many characters you want to return:

Example

Return the 5 first characters of the file:

f = open("demofile.txt", "r")  
print(f.read(**5**))

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_open2)

Read Lines

You can return one line by using the readline() method:

Example

Read one line of the file:

f = open("demofile.txt", "r")  
print(f.readline())

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_readline)

By calling readline() two times, you can read the two first lines:

Example

Read two lines of the file:

f = open("demofile.txt", "r")  
print(f.readline())  
print(f.readline())

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_readline2)

By looping through the lines of the file, you can read the whole file, line by line:

Example

Loop through the file line by line:

f = open("demofile.txt", "r")  
for x in f:  
  print(x)

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_readline3)

Close Files

It is a good practice to always close the file when you are done with it.

Example

Close the file when you are finish with it:

f = open("demofile.txt", "r")  
print(f.readline())  
f.close()

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_close)

**Note:** You should always close your files, in some cases, due to buffering, changes made to a file may not show until you close the file.

Python File Write

Write to an Existing File

To write to an existing file, you must add a parameter to the open() function:

"a" - Append - will append to the end of the file

"w" - Write - will overwrite any existing content

Example

Open the file "demofile2.txt" and append content to the file:

f = open("demofile2.txt", "a")  
f.write("Now the file has more content!")  
f.close()  
  
#open and read the file after the appending:  
f = open("demofile2.txt", "r")  
print(f.read())

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_append)

Example

Open the file "demofile3.txt" and overwrite the content:

f = open("demofile3.txt", "w")  
f.write("Woops! I have deleted the content!")  
f.close()  
  
#open and read the file after the appending:  
f = open("demofile3.txt", "r")  
print(f.read())

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_file_write)

**Note:** the "w" method will overwrite the entire file.

Create a New File

To create a new file in Python, use the open() method, with one of the following parameters:

"x" - Create - will create a file, returns an error if the file exist

"a" - Append - will create a file if the specified file does not exist

"w" - Write - will create a file if the specified file does not exist

Example

Create a file called "myfile.txt":

f = open("myfile.txt", "x")

Result: a new empty file is created!

Example

Create a new file if it does not exist:

f = open("myfile.txt", "w")

Python Delete File

Delete a File

To delete a file, you must import the OS module, and run its os.remove() function:

Example

Remove the file "demofile.txt":

import os  
os.remove("demofile.txt")

Check if File exist:

To avoid getting an error, you might want to check if the file exists before you try to delete it:

Example

Check if file exists, *then* delete it:

import os  
if os.path.exists("demofile.txt"):  
  os.remove("demofile.txt")  
else:  
  print("The file does not exist")

Delete Folder

To delete an entire folder, use the os.rmdir() method:

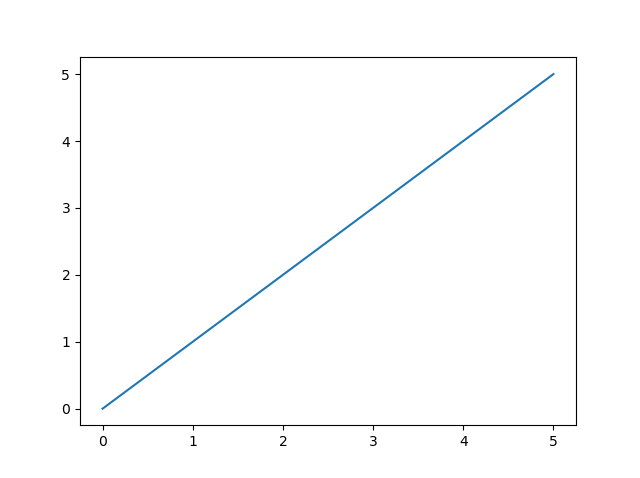
Example

Remove the folder "myfolder":

import os  
os.rmdir("myfolder")

**Note:** You can only remove *empty* folders.

Matplotlib Tutorial



What is Matplotlib?

Matplotlib is a low level graph plotting library in python that serves as a visualization utility.

Matplotlib was created by John D. Hunter.

Matplotlib is open source and we can use it freely.

Matplotlib is mostly written in python, a few segments are written in C, Objective-C and Javascript for Platform compatibility.

Where is the Matplotlib Codebase?

The source code for Matplotlib is located at this github repository <https://github.com/matplotlib/matplotlib>

Matplotlib Getting Started

Installation of Matplotlib

If you have [Python](https://www.w3schools.com/python/default.asp) and [PIP](https://www.w3schools.com/python/python_pip.asp) already installed on a system, then installation of Matplotlib is very easy.

Install it using this command:

C:\Users\*Your Name*>pip install matplotlib

If this command fails, then use a python distribution that already has Matplotlib installed,  like Anaconda, Spyder etc.

Import Matplotlib

Once Matplotlib is installed, import it in your applications by adding the import *module* statement:

import matplotlib

Now Matplotlib is imported and ready to use:

Checking Matplotlib Version

The version string is stored under \_\_version\_\_ attribute.

Example

import matplotlib  
  
print(matplotlib.\_\_version\_\_)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_version)

**Note:** two underscore characters are used in \_\_version\_\_.

Matplotlib Pyplot

Pyplot

Most of the Matplotlib utilities lies under the pyplot submodule, and are usually imported under the plt alias:

import matplotlib.pyplot as plt

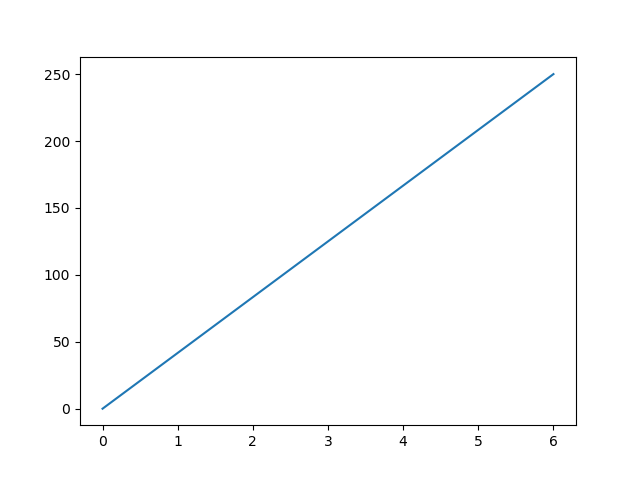
Now the Pyplot package can be referred to as plt.

Example

Draw a line in a diagram from position (0,0) to position (6,250):

import matplotlib.pyplot as plt  
import numpy as np  
  
xpoints = np.array([0, 6])  
ypoints = np.array([0, 250])  
  
plt.plot(xpoints, ypoints)  
plt.show()

Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pyplot)

You will learn more about drawing (plotting) in the next chapters.

Matplotlib Plotting

Plotting x and y points

The plot() function is used to draw points (markers) in a diagram.

By default, the plot() function draws a line from point to point.

The function takes parameters for specifying points in the diagram.

Parameter 1 is an array containing the points on the **x-axis**.

Parameter 2 is an array containing the points on the **y-axis**.

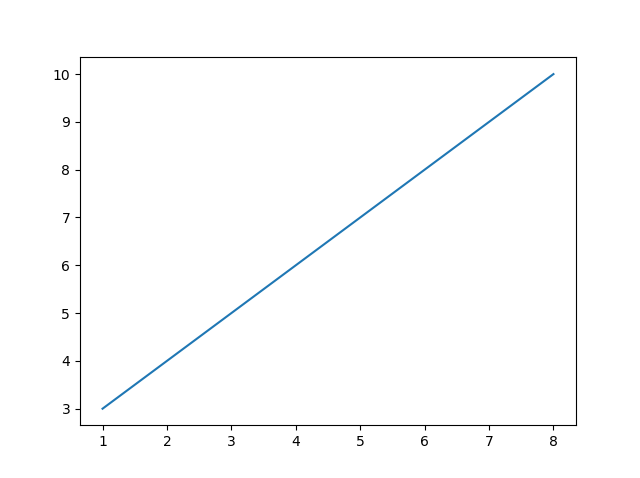
If we need to plot a line from (1, 3) to (8, 10), we have to pass two arrays [1, 8] and [3, 10] to the plot function.

Example

Draw a line in a diagram from position (1, 3) to position (8, 10):

import matplotlib.pyplot as plt  
import numpy as np  
  
xpoints = np.array([1, 8])  
ypoints = np.array([3, 10])  
  
plt.plot(xpoints, ypoints)  
plt.show()

Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_plotting1)

The **x-axis** is the horizontal axis.

The **y-axis** is the vertical axis.

Plotting Without Line

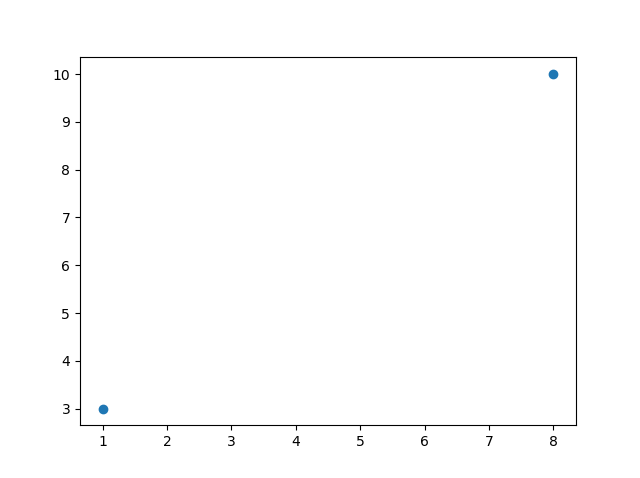
To plot only the markers, you can use *shortcut string notation* parameter 'o', which means 'rings'.

Example

Draw two points in the diagram, one at position (1, 3) and one in position (8, 10):

import matplotlib.pyplot as plt  
import numpy as np  
  
xpoints = np.array([1, 8])  
ypoints = np.array([3, 10])  
  
plt.plot(xpoints, ypoints, 'o')  
plt.show()

Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_plotting2)

You will learn more about markers in the next chapter.

Multiple Points

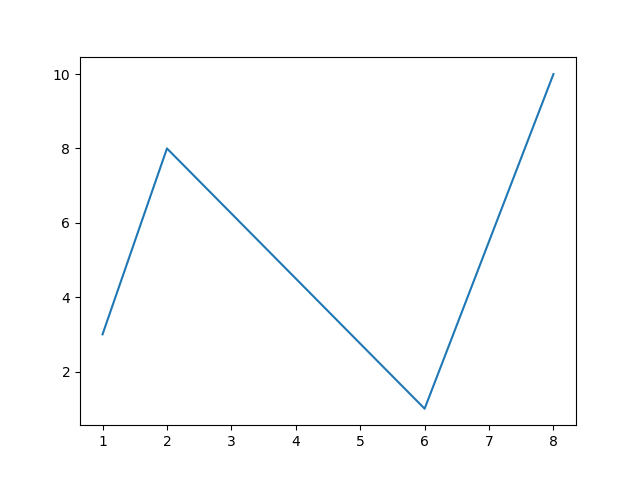
You can plot as many points as you like, just make sure you have the same number of points in both axis.

Example

Draw a line in a diagram from position (1, 3) to (2, 8) then to (6, 1) and finally to position (8, 10):

import matplotlib.pyplot as plt  
import numpy as np  
  
xpoints = np.array([1, 2, 6, 8])  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(xpoints, ypoints)  
plt.show()

Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_plotting3)

Default X-Points

If we do not specify the points in the x-axis, they will get the default values 0, 1, 2, 3, (etc. depending on the length of the y-points.

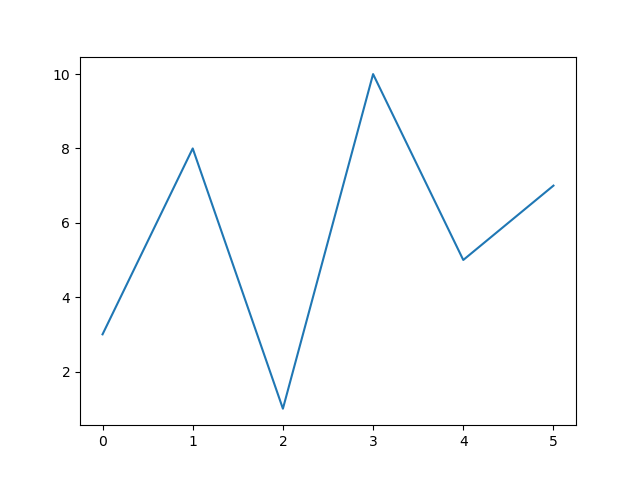
So, if we take the same example as above, and leave out the x-points, the diagram will look like this:

Example

Plotting without x-points:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10, 5, 7])  
  
plt.plot(ypoints)  
plt.show()

Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_plotting4)

The **x-points** in the example above is [0, 1, 2, 3, 4, 5].

# Matplotlib Markers

## Markers

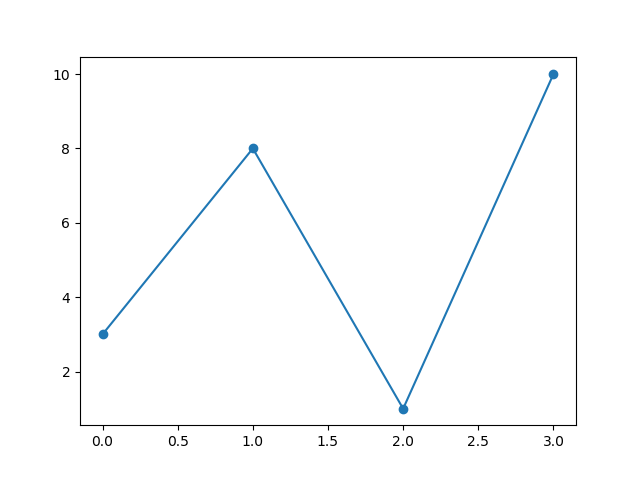
You can use the keyword argument marker to emphasize each point with a specified marker:

### Example

Mark each point with a circle:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, marker = 'o')  
plt.show()

### Result:



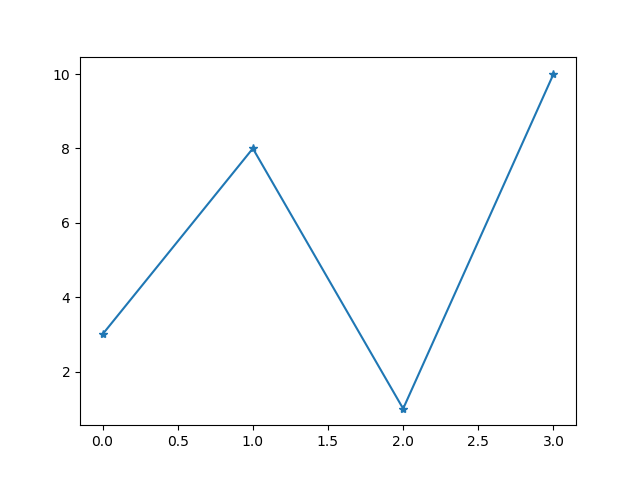
[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker1)

### Example

Mark each point with a star:

...  
plt.plot(ypoints, marker = '\*')  
...

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker2)

## Marker Reference

You can choose any of these markers:

|  |  |  |
| --- | --- | --- |
| **Marker** | **Description** | |
| 'o' | Circle | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_o) |
| '\*' | Star | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_star) |
| '.' | Point | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_point) |
| ',' | Pixel | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_pixel) |
| 'x' | X | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_x) |
| 'X' | X (filled) | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_x_filled) |
| '+' | Plus | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_plus) |
| 'P' | Plus (filled) | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_plus_filled) |
| 's' | Square | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_s) |
| 'D' | Diamond | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_D) |
| 'd' | Diamond (thin) | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_thin_d) |
| 'p' | Pentagon | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_p) |
| 'H' | Hexagon | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_H) |
| 'h' | Hexagon | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_hexagon) |
| 'v' | Triangle Down | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_v) |
| '^' | Triangle Up | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_triangle_up) |
| '<' | Triangle Left | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_triangle_left) |
| '>' | Triangle Right | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_triangle_right) |
| '1' | Tri Down | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_1) |
| '2' | Tri Up | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_2) |
| '3' | Tri Left | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_3) |
| '4' | Tri Right | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_4) |
| '|' | Vline | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_vline) |
| '\_' | Hline | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_hline) |

## Format Strings fmt

You can use also use the shortcut string notation parameter to specify the marker.

This parameter is also called fmt, and is written with this syntax:

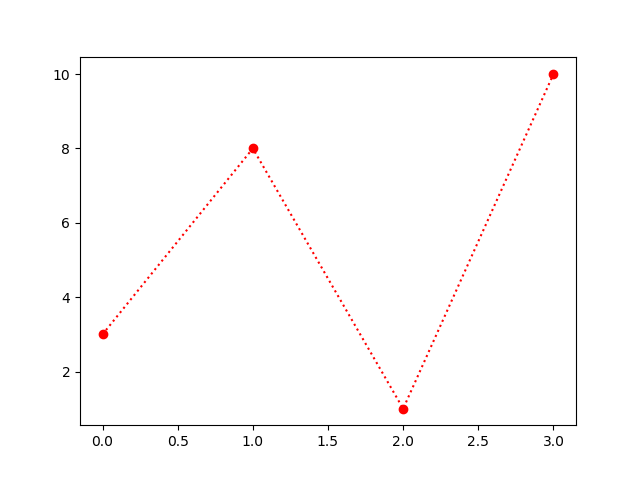
marker|line|color

### Example

Mark each point with a circle:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, 'o:r')  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker3)

The marker value can be anything from the Marker Reference above.

The line value can be one of the following:

## Line Reference

|  |  |  |
| --- | --- | --- |
| **Line Syntax** | **Description** | |
| '-' | Solid line | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_line_solid) |
| ':' | Dotted line | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_line_dot) |
| '--' | Dashed line | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_line_dash) |
| '-.' | Dashed/dotted line | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_line_dashdot) |

**Note:** If you leave out the line value in the fmt parameter, no line will be plotted.

The short color value can be one of the following:

## Color Reference

|  |  |  |
| --- | --- | --- |
| **Color Syntax** | **Description** | |
| 'r' | Red | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_r) |
| 'g' | Green | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_g) |
| 'b' | Blue | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_b) |
| 'c' | Cyan | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_c) |
| 'm' | Magenta | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_m) |
| 'y' | Yellow | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_y) |
| 'k' | Black | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_k) |
| 'w' | White | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_fmt_w) |

## Marker Size

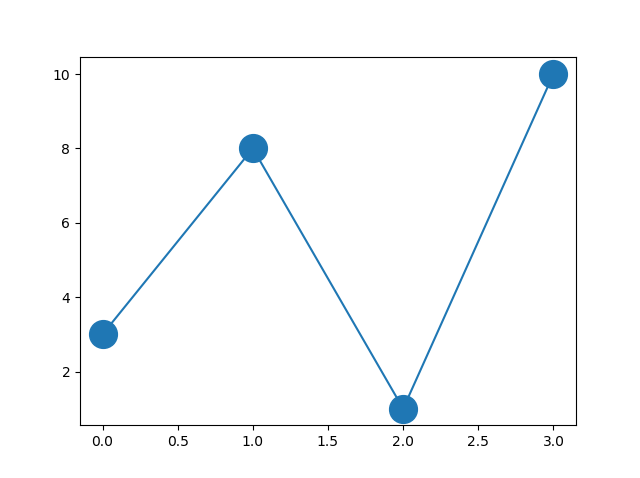
You can use the keyword argument markersize or the shorter version, ms to set the size of the markers:

### Example

Set the size of the markers to 20:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, marker = 'o', ms = 20)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_size)

## Marker Color

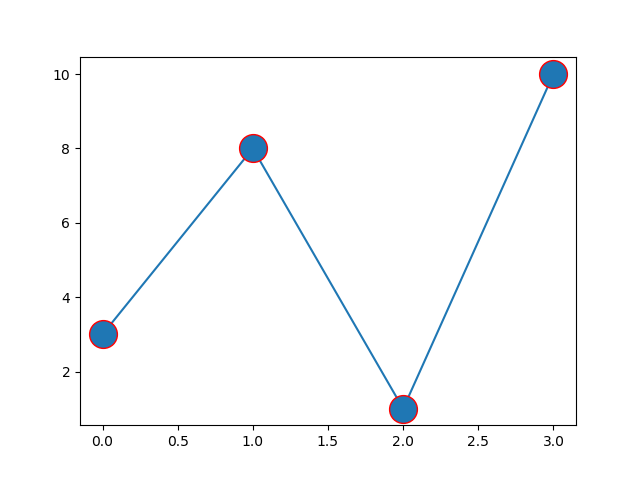
You can use the keyword argument markeredgecolor or the shorter mec to set the color of the edge of the markers:

### Example

Set the EDGE color to red:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, marker = 'o', ms = 20, mec = 'r')  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_color1)

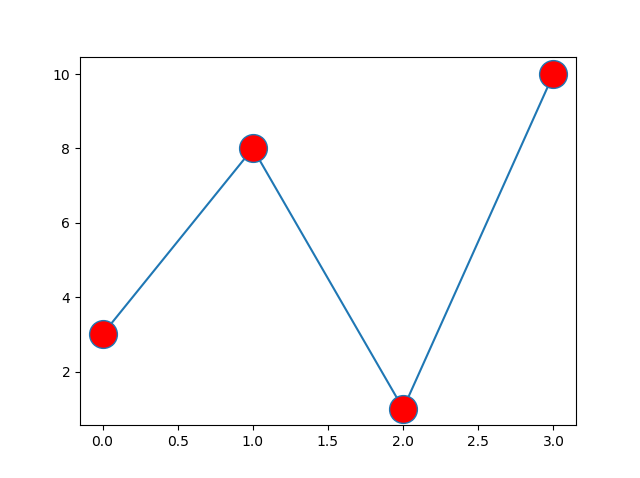
You can use the keyword argument markerfacecolor or the shorter mfc to set the color inside the edge of the markers:

### Example

Set the FACE color to red:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, marker = 'o', ms = 20, mfc = 'r')  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_color2)

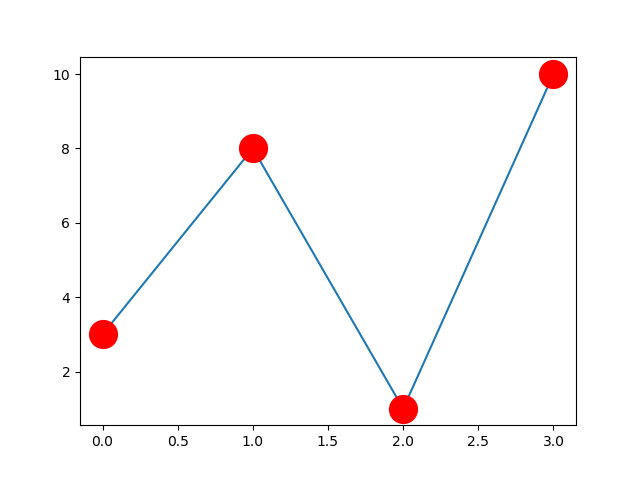
Use both the mec and mfc arguments to color of the entire marker:

### Example

Set the color of both the edge and the face to red:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, marker = 'o', ms = 20, mec = 'r', mfc = 'r')  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_color3)

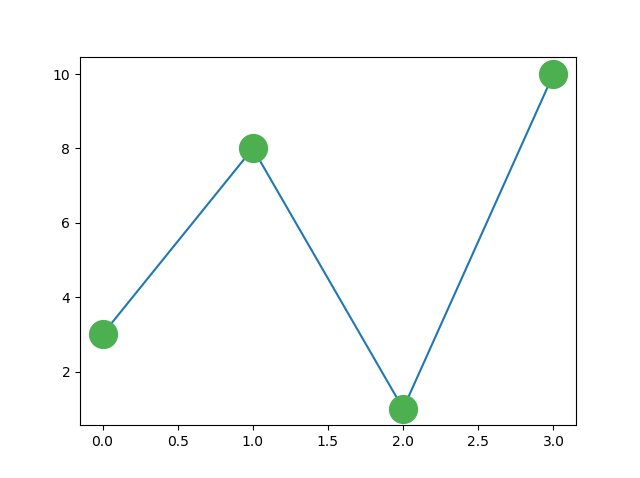
You can also use [Hexadecimal color values](https://www.w3schools.com/colors/colors_hexadecimal.asp):

### Example

Mark each point with a beautiful green color:

...  
plt.plot(ypoints, marker = 'o', ms = 20, mec = '#4CAF50', mfc = '#4CAF50')  
...

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_color_green)

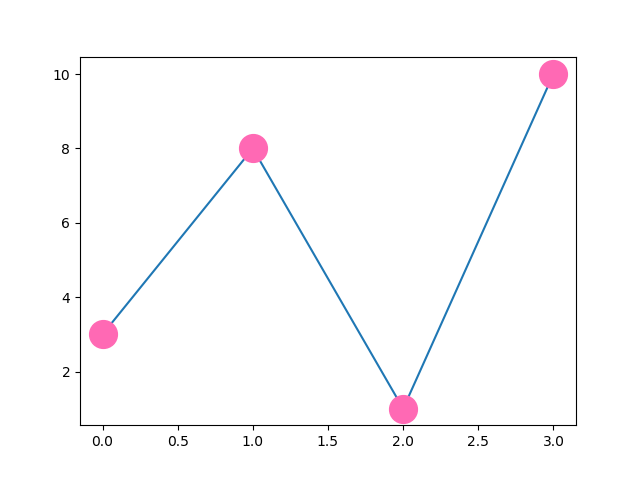
Or any of the [140 supported color names](https://www.w3schools.com/colors/colors_names.asp).

### Example

Mark each point with the color named "hotpink":

...  
plt.plot(ypoints, marker = 'o', ms = 20, mec = 'hotpink', mfc = 'hotpink')  
...

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_marker_color_hotpink)

# Matplotlib Line

## Linestyle

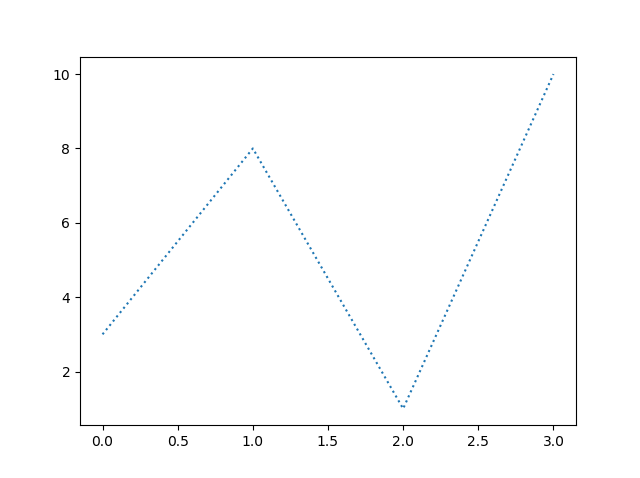
You can use the keyword argument linestyle, or shorter ls, to change the style of the plotted line:

### Example

Use a dotted line:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, linestyle = 'dotted')  
plt.show()

### Result:



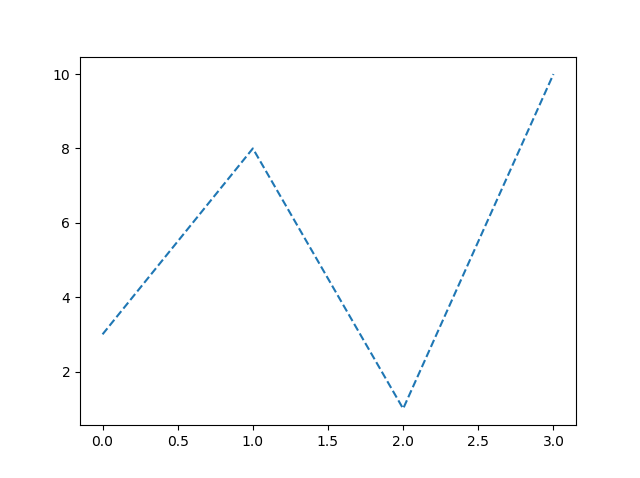
[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_line_dotted)

### Example

Use a dashed line:

plt.plot(ypoints, linestyle = 'dashed')

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_line_dashed)

## Shorter Syntax

The line style can be written in a shorter syntax:

linestyle can be written as ls.

dotted can be written as :.

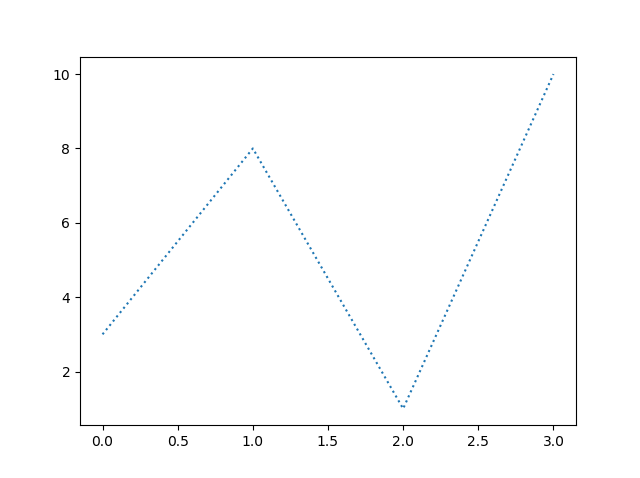
dashed can be written as --.

### Example

Shorter syntax:

plt.plot(ypoints, ls = ':')

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_line_short)

## Line Styles

You can choose any of these styles:

|  |  |  |
| --- | --- | --- |
| **Style** | **Or** | |
| 'solid' (default) | '-' | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_ls_solid) |
| 'dotted' | ':' | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_ls_dotted) |
| 'dashed' | '--' | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_ls_dashed) |
| 'dashdot' | '-.' | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_ls_dashdot) |
| 'None' | '' or ' ' | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_ls_none) |

## Line Color

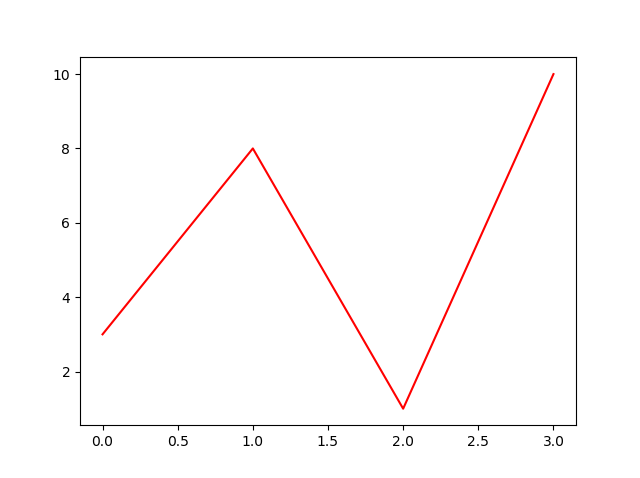
You can use the keyword argument color or the shorter c to set the color of the line:

### Example

Set the line color to red:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, color = 'r')  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_line_red)

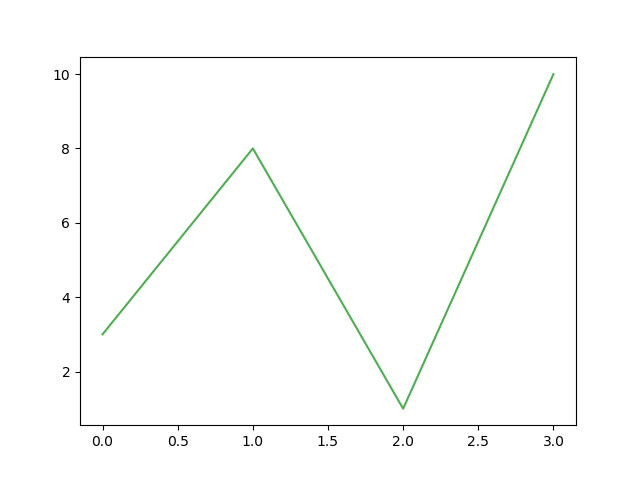
You can also use [Hexadecimal color values](https://www.w3schools.com/colors/colors_hexadecimal.asp):

### Example

Plot with a beautiful green line:

...  
plt.plot(ypoints, c = '#4CAF50')  
...

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_line_green)

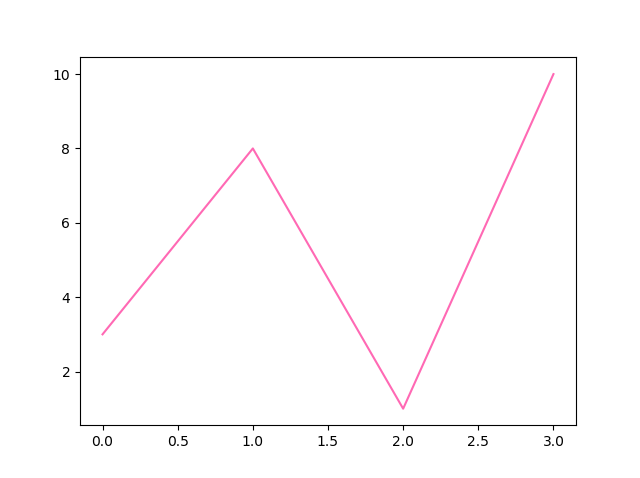
Or any of the [140 supported color names](https://www.w3schools.com/colors/colors_names.asp).

### Example

Plot with the color named "hotpink":

...  
plt.plot(ypoints, c = 'hotpink')  
...

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_line_hotpink)

## Line Width

You can use the keyword argument linewidth or the shorter lw to change the width of the line.

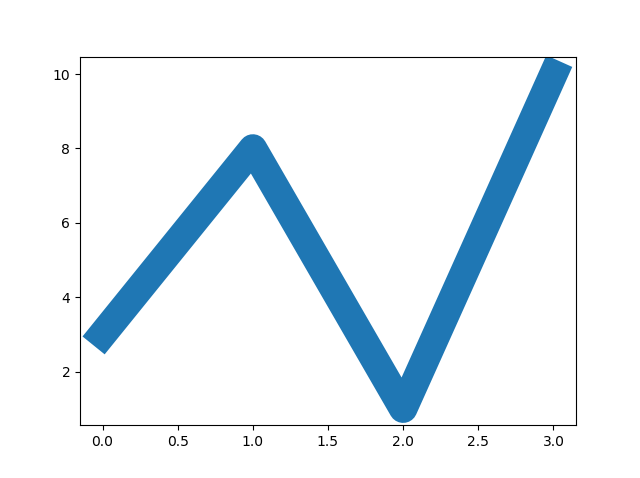
The value is a floating number, in points:

### Example

Plot with a 20.5pt wide line:

import matplotlib.pyplot as plt  
import numpy as np  
  
ypoints = np.array([3, 8, 1, 10])  
  
plt.plot(ypoints, linewidth = '20.5')  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_line_width)

## Multiple Lines

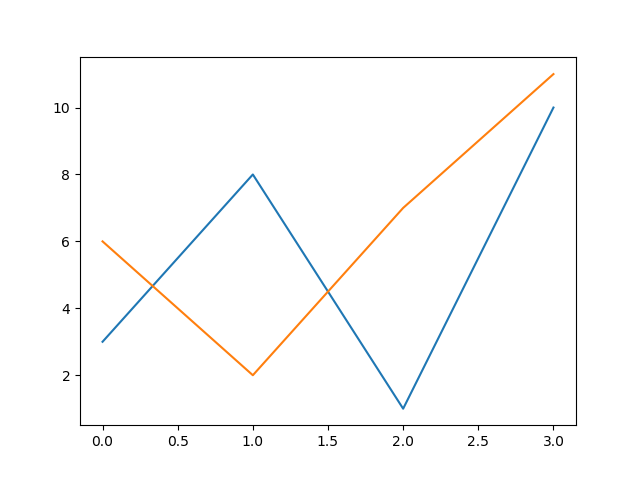
You can plot as many lines as you like by simply adding more plt.plot() functions:

### Example

Draw two lines by specifying a plt.plot() function for each line:

import matplotlib.pyplot as plt  
import numpy as np  
  
y1 = np.array([3, 8, 1, 10])  
y2 = np.array([6, 2, 7, 11])  
  
plt.plot(y1)  
plt.plot(y2)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_lines)

You can also plot many lines by adding the points for the x- and y-axis for each line in the same plt.plot() function.

(In the examples above we only specified the points on the y-axis, meaning that the points on the x-axis got the the default values (0, 1, 2, 3).)

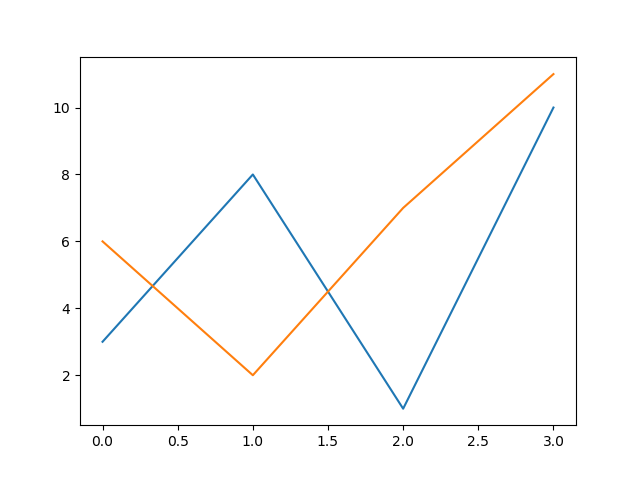
The x- and y- values come in pairs:

### Example

Draw two lines by specifiyng the x- and y-point values for both lines:

import matplotlib.pyplot as plt  
import numpy as np  
  
x1 = np.array([0, 1, 2, 3])  
y1 = np.array([3, 8, 1, 10])  
x2 = np.array([0, 1, 2, 3])  
y2 = np.array([6, 2, 7, 11])  
  
plt.plot(x1, y1, x2, y2)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_lines2)

# Matplotlib Labels and Title

## Create Labels for a Plot

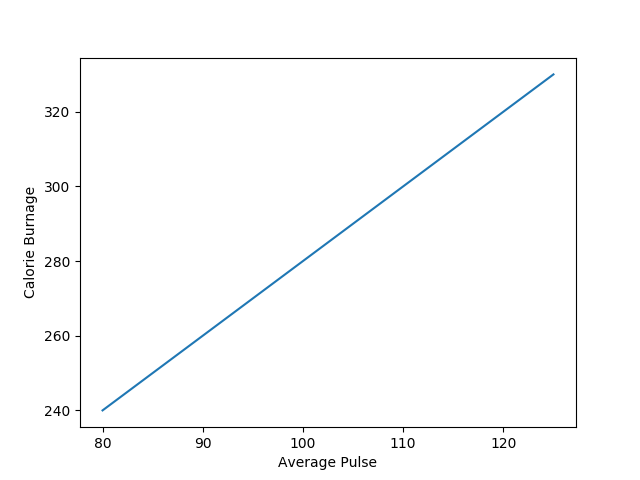
With Pyplot, you can use the xlabel() and ylabel() functions to set a label for the x- and y-axis.

### Example

Add labels to the x- and y-axis:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
plt.plot(x, y)  
  
plt.xlabel("Average Pulse")  
plt.ylabel("Calorie Burnage")  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_labels)

## Create a Title for a Plot

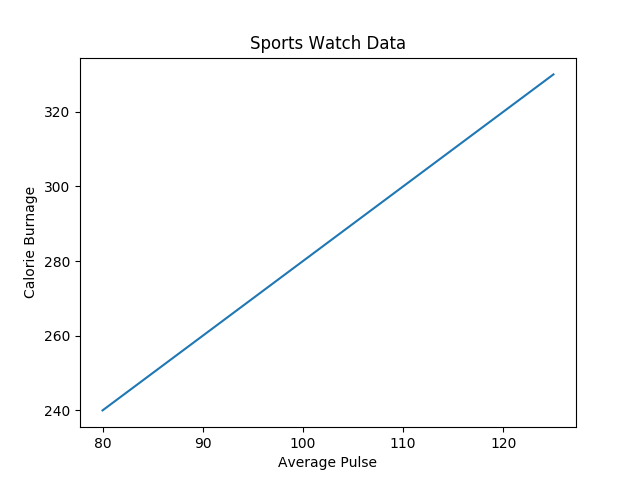
With Pyplot, you can use the title() function to set a title for the plot.

### Example

Add a plot title and labels for the x- and y-axis:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
plt.plot(x, y)  
  
plt.title("Sports Watch Data")  
plt.xlabel("Average Pulse")  
plt.ylabel("Calorie Burnage")  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_title)

## Set Font Properties for Title and Labels

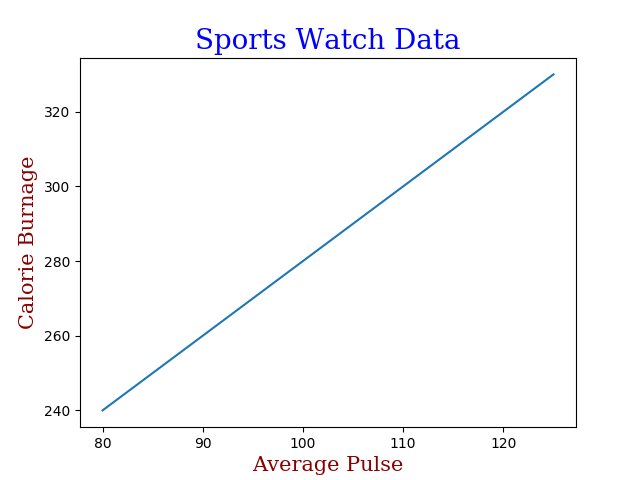
You can use the fontdict parameter in xlabel(), ylabel(), and title() to set font properties for the title and labels.

### Example

Set font properties for the title and labels:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
font1 = {'family':'serif','color':'blue','size':20}  
font2 = {'family':'serif','color':'darkred','size':15}  
  
plt.title("Sports Watch Data", fontdict = font1)  
plt.xlabel("Average Pulse", fontdict = font2)  
plt.ylabel("Calorie Burnage", fontdict = font2)  
  
plt.plot(x, y)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_title_fontdict)

## Position the Title

You can use the loc parameter in title() to position the title.

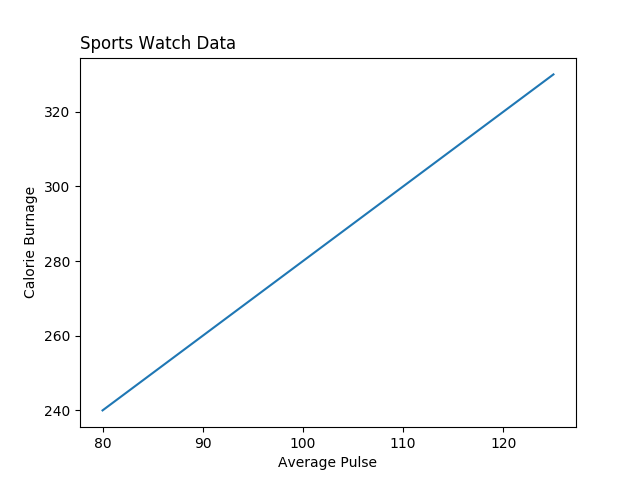
Legal values are: 'left', 'right', and 'center'. Default value is 'center'.

### Example

Position the title to the left:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
plt.title("Sports Watch Data", loc = 'left')  
plt.xlabel("Average Pulse")  
plt.ylabel("Calorie Burnage")  
  
plt.plot(x, y)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_title_loc)

# Matplotlib Adding Grid Lines

## Add Grid Lines to a Plot

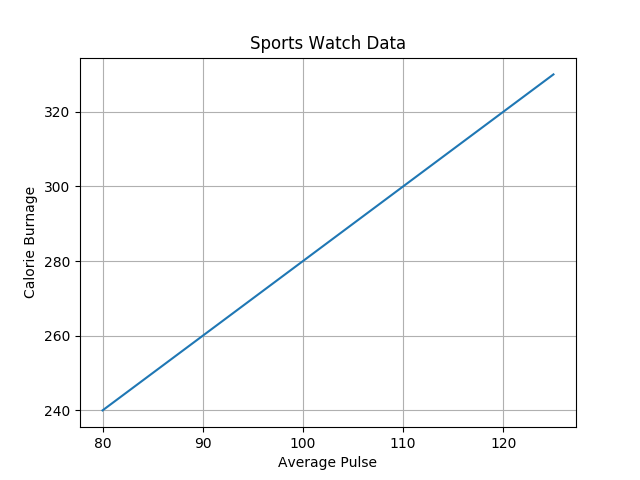
With Pyplot, you can use the grid() function to add grid lines to the plot.

### Example

Add grid lines to the plot:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
plt.title("Sports Watch Data")  
plt.xlabel("Average Pulse")  
plt.ylabel("Calorie Burnage")  
  
plt.plot(x, y)  
  
plt.grid()  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_grid)

## Specify Which Grid Lines to Display

You can use the axis parameter in the grid() function to specify which grid lines to display.

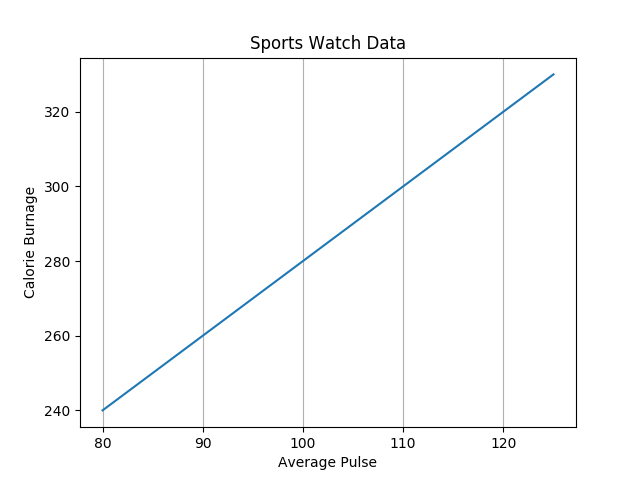
Legal values are: 'x', 'y', and 'both'. Default value is 'both'.

### Example

Display only grid lines for the x-axis:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
plt.title("Sports Watch Data")  
plt.xlabel("Average Pulse")  
plt.ylabel("Calorie Burnage")  
  
plt.plot(x, y)  
  
plt.grid(axis = 'x')  
  
plt.show()

### Result:



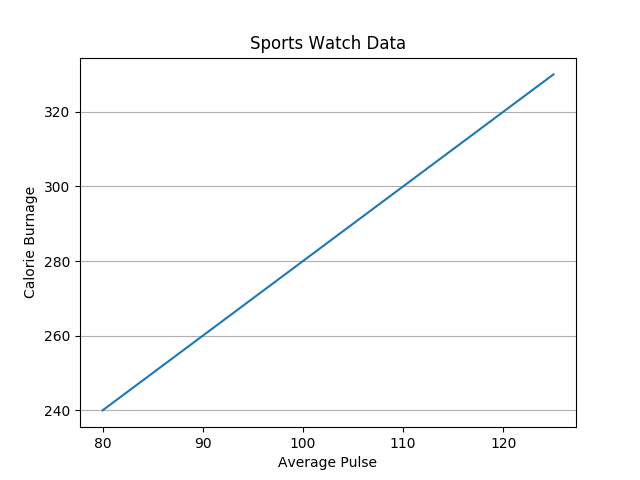
[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_grid_axisx)

### Example

Display only grid lines for the y-axis:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
plt.title("Sports Watch Data")  
plt.xlabel("Average Pulse")  
plt.ylabel("Calorie Burnage")  
  
plt.plot(x, y)  
  
plt.grid(axis = 'y')  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_grid_axisy)

## Set Line Properties for the Grid

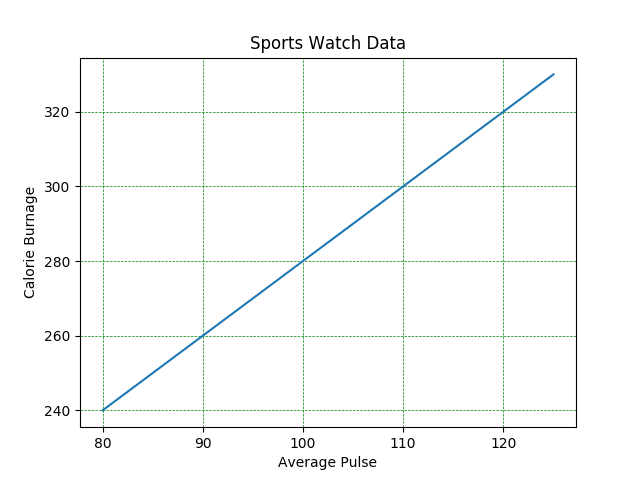
You can also set the line properties of the grid, like this: grid(color = 'color', linestyle = 'linestyle', linewidth = number).

### Example

Set the line properties of the grid:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])  
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])  
  
plt.title("Sports Watch Data")  
plt.xlabel("Average Pulse")  
plt.ylabel("Calorie Burnage")  
  
plt.plot(x, y)  
  
plt.grid(color = 'green', linestyle = '--', linewidth = 0.5)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_pyplot_grid_kwargs)

# Matplotlib Subplot

## Display Multiple Plots

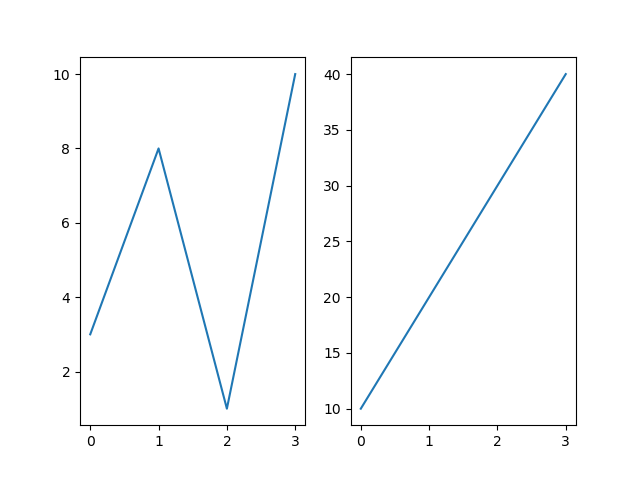
With the subplot() function you can draw multiple plots in one figure:

### Example

Draw 2 plots:

import matplotlib.pyplot as plt  
import numpy as np  
  
#plot 1:  
x = np.array([0, 1, 2, 3])  
y = np.array([3, 8, 1, 10])  
  
plt.subplot(1, 2, 1)  
plt.plot(x,y)  
  
#plot 2:  
x = np.array([0, 1, 2, 3])  
y = np.array([10, 20, 30, 40])  
  
plt.subplot(1, 2, 2)  
plt.plot(x,y)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_subplots1)

## The subplot() Function

The subplot() function takes three arguments that describes the layout of the figure.

The layout is organized in rows and columns, which are represented by the first and second argument.

The third argument represents the index of the current plot.

plt.subplot(1, 2, 1)  
#the figure has 1 row, 2 columns, and this plot is the first plot.

plt.subplot(1, 2, 2)  
#the figure has 1 row, 2 columns, and this plot is the second plot.

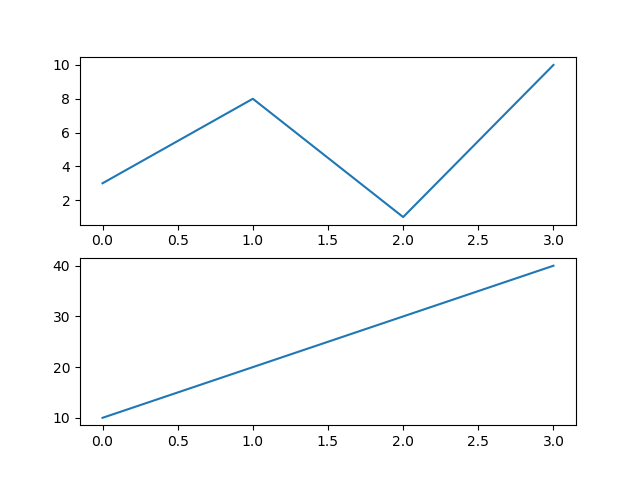
So, if we want a figure with 2 rows an 1 column (meaning that the two plots will be displayed on top of each other instead of side-by-side), we can write the syntax like this:

### Example

Draw 2 plots on top of each other:

import matplotlib.pyplot as plt  
import numpy as np  
  
#plot 1:  
x = np.array([0, 1, 2, 3])  
y = np.array([3, 8, 1, 10])  
  
plt.subplot(2, 1, 1)  
plt.plot(x,y)  
  
#plot 2:  
x = np.array([0, 1, 2, 3])  
y = np.array([10, 20, 30, 40])  
  
plt.subplot(2, 1, 2)  
plt.plot(x,y)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_subplots2)

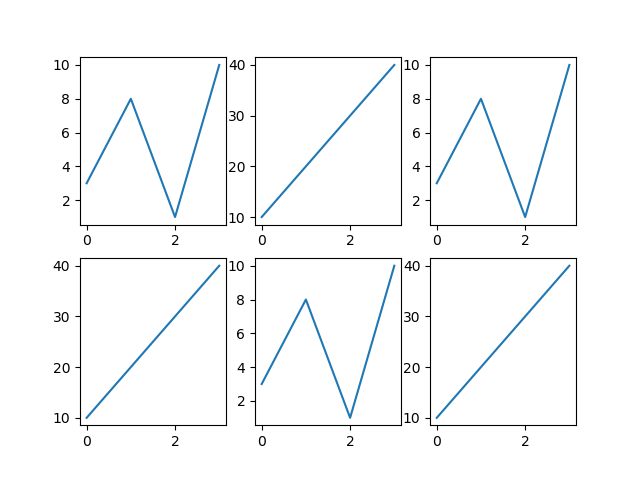
You can draw as many plots you like on one figure, just descibe the number of rows, columns, and the index of the plot.

### Example

Draw 6 plots:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([0, 1, 2, 3])  
y = np.array([3, 8, 1, 10])  
  
plt.subplot(2, 3, 1)  
plt.plot(x,y)  
  
x = np.array([0, 1, 2, 3])  
y = np.array([10, 20, 30, 40])  
  
plt.subplot(2, 3, 2)  
plt.plot(x,y)  
  
x = np.array([0, 1, 2, 3])  
y = np.array([3, 8, 1, 10])  
  
plt.subplot(2, 3, 3)  
plt.plot(x,y)  
  
x = np.array([0, 1, 2, 3])  
y = np.array([10, 20, 30, 40])  
  
plt.subplot(2, 3, 4)  
plt.plot(x,y)  
  
x = np.array([0, 1, 2, 3])  
y = np.array([3, 8, 1, 10])  
  
plt.subplot(2, 3, 5)  
plt.plot(x,y)  
  
x = np.array([0, 1, 2, 3])  
y = np.array([10, 20, 30, 40])  
  
plt.subplot(2, 3, 6)  
plt.plot(x,y)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_subplots3)

## Title

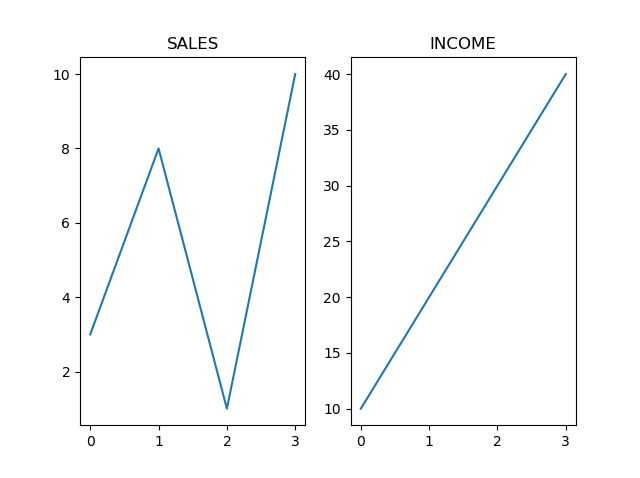
You can add a title to each plot with the title() function:

### Example

2 plots, with titles:

import matplotlib.pyplot as plt  
import numpy as np  
  
#plot 1:  
x = np.array([0, 1, 2, 3])  
y = np.array([3, 8, 1, 10])  
  
plt.subplot(1, 2, 1)  
plt.plot(x,y)  
plt.title("SALES")  
  
#plot 2:  
x = np.array([0, 1, 2, 3])  
y = np.array([10, 20, 30, 40])  
  
plt.subplot(1, 2, 2)  
plt.plot(x,y)  
plt.title("INCOME")  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_subplots4)

## Super Title

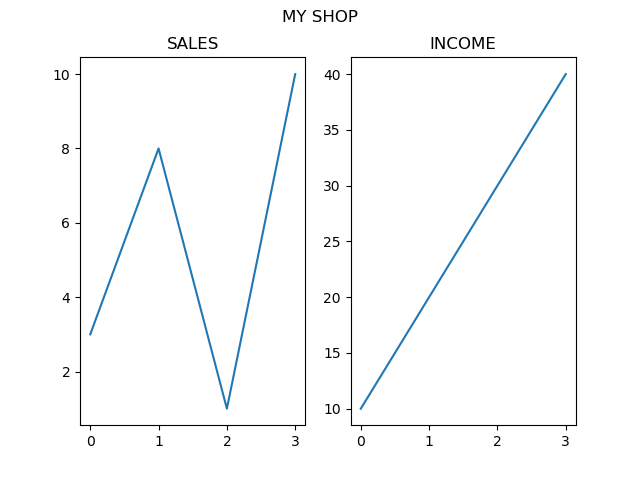
You can add a title to the entire figure with the suptitle() function:

### Example

Add a title for the entire figure:

import matplotlib.pyplot as plt  
import numpy as np  
  
#plot 1:  
x = np.array([0, 1, 2, 3])  
y = np.array([3, 8, 1, 10])  
  
plt.subplot(1, 2, 1)  
plt.plot(x,y)  
plt.title("SALES")  
  
#plot 2:  
x = np.array([0, 1, 2, 3])  
y = np.array([10, 20, 30, 40])  
  
plt.subplot(1, 2, 2)  
plt.plot(x,y)  
plt.title("INCOME")  
  
plt.suptitle("MY SHOP")  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_subplots5)

# Matplotlib Scatter

## Creating Scatter Plots

With Pyplot, you can use the scatter() function to draw a scatter plot.

The scatter() function plots one dot for each observation. It needs two arrays of the same length, one for the values of the x-axis, and one for values on the y-axis:

### Example

A simple scatter plot:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
  
plt.scatter(x, y)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter)

The observation in the example above is the result of 13 cars passing by.

The X-axis shows how old the car is.

The Y-axis shows the speed of the car when it passes.

Are there any relationships between the observations?

It seems that the newer the car, the faster it drives, but that could be a coincidence, after all we only registered 13 cars.

## Compare Plots

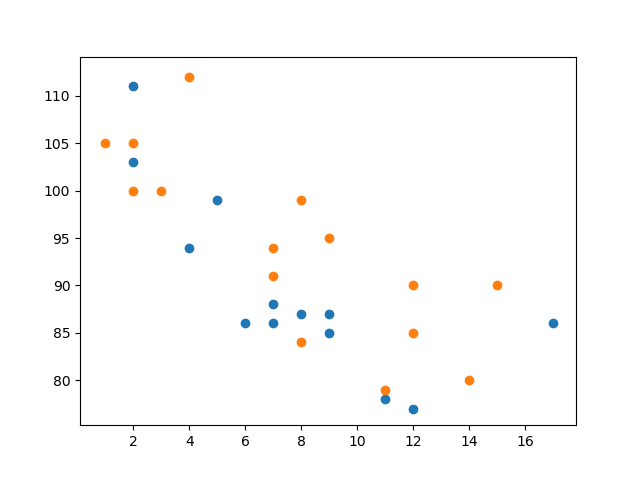
In the example above, there seems to be a relationship between speed and age, but what if we plot the observations from another day as well? Will the scatter plot tell us something else?

### Example

Draw two plots on the same figure:

import matplotlib.pyplot as plt  
import numpy as np  
  
#day one, the age and speed of 13 cars:  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
plt.scatter(x, y)  
  
#day two, the age and speed of 15 cars:  
x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])  
y = np.array([100,105,84,105,90,99,90,95,94,100,79,112,91,80,85])  
plt.scatter(x, y)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_compare)

**Note:** The two plots are plotted with two different colors, by default blue and orange, you will learn how to change colors later in this chapter.

By comparing the two plots, I think it is safe to say that they both gives us the same conclusion: the newer the car, the faster it drives.

## Colors

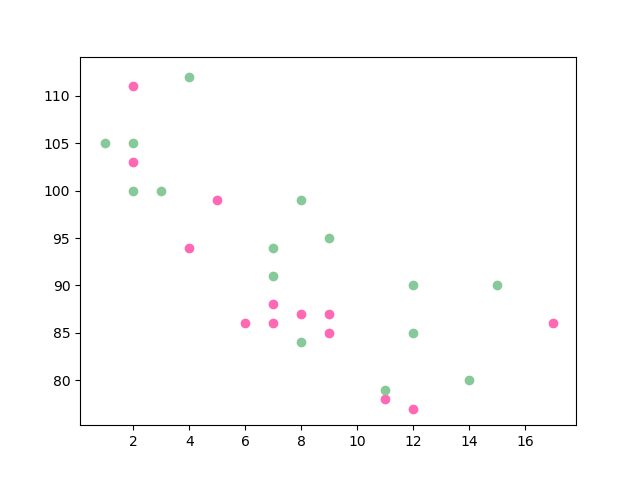
You can set your own color for each scatter plot with the color or the c argument:

### Example

Set your own color of the markers:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
plt.scatter(x, y, color = 'hotpink')  
  
x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])  
y = np.array([100,105,84,105,90,99,90,95,94,100,79,112,91,80,85])  
plt.scatter(x, y, color = '#88c999')  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_color)

## Color Each Dot

You can even set a specific color for each dot by using an array of colors as value for the c argument:

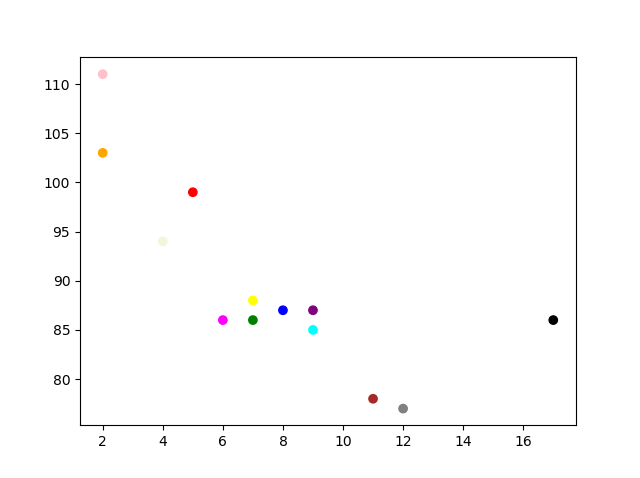
**Note:** You cannot use the color argument for this, only the c argument.

### Example

Set your own color of the markers:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
colors = np.array(["red","green","blue","yellow","pink","black","orange","purple","beige","brown","gray","cyan","magenta"])  
  
plt.scatter(x, y, c=colors)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_c)

## ColorMap

The Matplotlib module has a number of available colormaps.

A colormap is like a list of colors, where each color has a value that ranges from 0 to 100.

Here is an example of a colormap:



This colormap is called 'viridis' and as you can see it ranges from 0, which is a purple color, and up to 100, which is a yellow color.

### How to Use the ColorMap

You can specify the colormap with the keyword argument cmap with the value of the colormap, in this case 'viridis' which is one of the built-in colormaps available in Matplotlib.

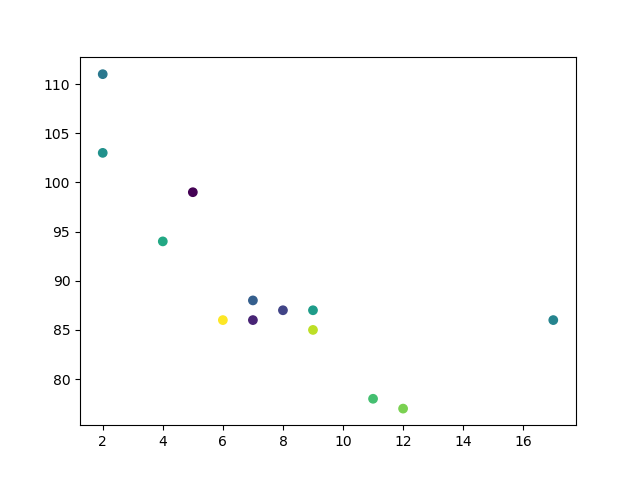
In addition you have to create an array with values (from 0 to 100), one value for each of the point in the scatter plot:

### Example

Create a color array, and specify a colormap in the scatter plot:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
colors = np.array([0, 10, 20, 30, 40, 45, 50, 55, 60, 70, 80, 90, 100])  
  
plt.scatter(x, y, c=colors, cmap='viridis')  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap)

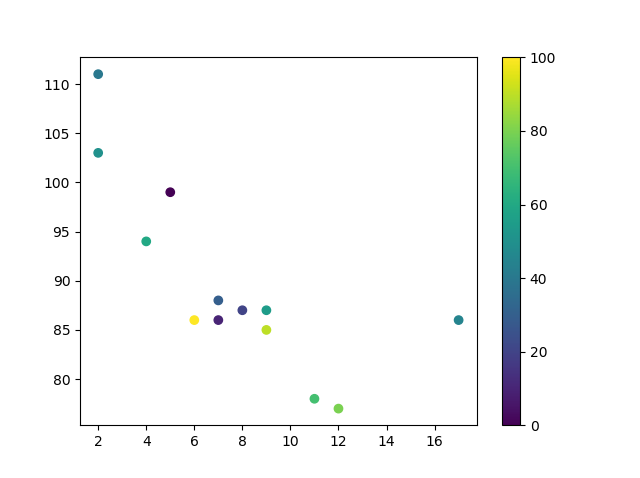
You can include the colormap in the drawing by including the plt.colorbar() statement:

### Example

Include the actual colormap:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
colors = np.array([0, 10, 20, 30, 40, 45, 50, 55, 60, 70, 80, 90, 100])  
  
plt.scatter(x, y, c=colors, cmap='viridis')  
  
plt.colorbar()  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap2)

### Available ColorMaps

You can choose any of the built-in colormaps:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** |  |  | **Reverse** |  |
| Accent | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Accent) |  | Accent\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Accent_r) |
| Blues | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Blues) |  | Blues\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Blues_r) |
| BrBG | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_BrBG) |  | BrBG\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_BrBG_r) |
| BuGn | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_BuGn) |  | BuGn\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_BuGn_r) |
| BuPu | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_BuPu) |  | BuPu\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_BuPu_r) |
| CMRmap | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_CMRmap) |  | CMRmap\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_CMRmap_r) |
| Dark2 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Dark2) |  | Dark2\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Dark2_r) |
| GnBu | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_GnBu) |  | GnBu\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_GnBu_r) |
| Greens | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Greens) |  | Greens\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Greens_r) |
| Greys | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Greys) |  | Greys\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Greys_r) |
| OrRd | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_OrRd) |  | OrRd\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_OrRd_r) |
| Oranges | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Oranges) |  | Oranges\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Oranges_r) |
| PRGn | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PRGn) |  | PRGn\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PRGn_r) |
| Paired | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Paired) |  | Paired\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Paired_r) |
| Pastel1 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Pastel1) |  | Pastel1\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Pastel1_r) |
| Pastel2 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Pastel2) |  | Pastel2\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Pastel2_r) |
| PiYG | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PiYG) |  | PiYG\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PiYG_r) |
| PuBu | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuBu) |  | PuBu\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuBu_r) |
| PuBuGn | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuBuGn) |  | PuBuGn\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuBuGn_r) |
| PuOr | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuOr) |  | PuOr\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuOr_r) |
| PuRd | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuRd) |  | PuRd\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_PuRd_r) |
| Purples | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Purples) |  | Purples\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Purples_r) |
| RdBu | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdBu) |  | RdBu\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdBu_r) |
| RdGy | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdGy) |  | RdGy\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdGy_r) |
| RdPu | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdPu) |  | RdPu\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdPu_r) |
| RdYlBu | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdYlBu) |  | RdYlBu\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdYlBu_r) |
| RdYlGn | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdYlGn) |  | RdYlGn\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_RdYlGn_r) |
| Reds | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Reds) |  | Reds\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Reds_r) |
| Set1 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Set1) |  | Set1\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Set1_r) |
| Set2 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Set2) |  | Set2\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Set2_r) |
| Set3 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Set3) |  | Set3\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Set3_r) |
| Spectral | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Spectral) |  | Spectral\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Spectral_r) |
| Wistia | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Wistia) |  | Wistia\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_Wistia_r) |
| YlGn | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlGn) |  | YlGn\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlGn_r) |
| YlGnBu | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlGnBu) |  | YlGnBu\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlGnBu_r) |
| YlOrBr | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlOrBr) |  | YlOrBr\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlOrBr_r) |
| YlOrRd | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlOrRd) |  | YlOrRd\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_YlOrRd_r) |
| afmhot | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_afmhot) |  | afmhot\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_afmhot_r) |
| autumn | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_autumn) |  | autumn\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_autumn_r) |
| binary | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_binary) |  | binary\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_binary_r) |
| bone | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_bone) |  | bone\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_bone_r) |
| brg | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_brg) |  | brg\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_brg_r) |
| bwr | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_bwr) |  | bwr\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_bwr_r) |
| cividis | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_cividis) |  | cividis\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_cividis_r) |
| cool | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_cool) |  | cool\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_cool_r) |
| coolwarm | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_coolwarm) |  | coolwarm\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_coolwarm_r) |
| copper | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_copper) |  | copper\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_copper_r) |
| cubehelix | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_cubehelix) |  | cubehelix\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_cubehelix_r) |
| flag | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_flag) |  | flag\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_flag_r) |
| gist\_earth | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_earth) |  | gist\_earth\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_earth_r) |
| gist\_gray | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_gray) |  | gist\_gray\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_gray_r) |
| gist\_heat | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_heat) |  | gist\_heat\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_heat_r) |
| gist\_ncar | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_ncar) |  | gist\_ncar\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_ncar_r) |
| gist\_rainbow | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_rainbow) |  | gist\_rainbow\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_rainbow_r) |
| gist\_stern | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_stern) |  | gist\_stern\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_stern_r) |
| gist\_yarg | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_yarg) |  | gist\_yarg\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gist_yarg_r) |
| gnuplot | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gnuplot) |  | gnuplot\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gnuplot_r) |
| gnuplot2 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gnuplot2) |  | gnuplot2\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gnuplot2_r) |
| gray | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gray) |  | gray\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_gray_r) |
| hot | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_hot) |  | hot\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_hot_r) |
| hsv | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_hsv) |  | hsv\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_hsv_r) |
| inferno | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_inferno) |  | inferno\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_inferno_r) |
| jet | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_jet) |  | jet\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_jet_r) |
| magma | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_magma) |  | magma\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_magma_r) |
| nipy\_spectral | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_nipy_spectral) |  | nipy\_spectral\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_nipy_spectral_r) |
| ocean | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_ocean) |  | ocean\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_ocean_r) |
| pink | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_pink) |  | pink\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_pink_r) |
| plasma | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_plasma) |  | plasma\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_plasma_r) |
| prism | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_prism) |  | prism\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_prism_r) |
| rainbow | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_rainbow) |  | rainbow\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_rainbow_r) |
| seismic | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_seismic) |  | seismic\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_seismic_r) |
| spring | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_spring) |  | spring\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_spring_r) |
| summer | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_summer) |  | summer\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_summer_r) |
| tab10 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab10) |  | tab10\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab10_r) |
| tab20 | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab20) |  | tab20\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab20_r) |
| tab20b | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab20b) |  | tab20b\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab20b_r) |
| tab20c | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab20c) |  | tab20c\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_tab20c_r) |
| terrain | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_terrain) |  | terrain\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_terrain_r) |
| twilight | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_twilight) |  | twilight\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_twilight_r) |
| twilight\_shifted | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_twilight_shifted) |  | twilight\_shifted\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_twilight_shifted_r) |
| viridis | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_viridis) |  | viridis\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_viridis_r) |
| winter | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_winter) |  | winter\_r | [Try it »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_cmap_winter_r) |

## Size

You can change the size of the dots with the s argument.

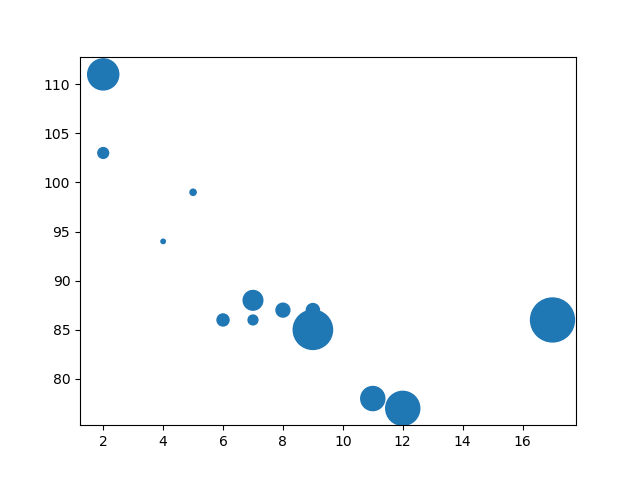
Just like colors, make sure the array for sizes has the same length as the arrays for the x- and y-axis:

### Example

Set your own size for the markers:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
sizes = np.array([20,50,100,200,500,1000,60,90,10,300,600,800,75])  
  
plt.scatter(x, y, s=sizes)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_size)

## Alpha

You can adjust the transparency of the dots with the alpha argument.

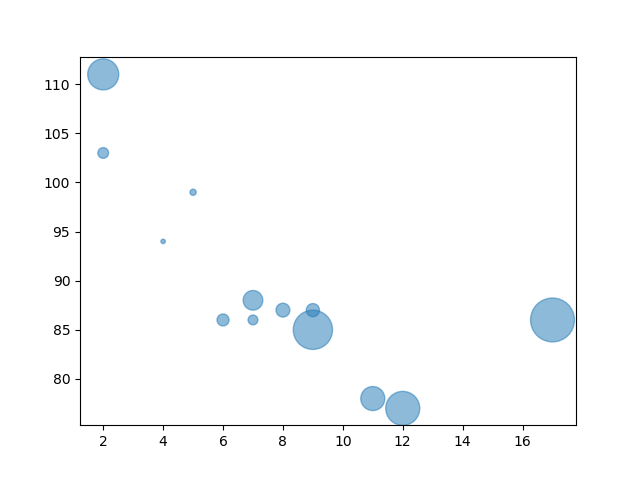
Just like colors, make sure the array for sizes has the same length as the arrays for the x- and y-axis:

### Example

Set your own size for the markers:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])  
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])  
sizes = np.array([20,50,100,200,500,1000,60,90,10,300,600,800,75])  
  
plt.scatter(x, y, s=sizes, alpha=0.5)  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_alpha)

## Combine Color Size and Alpha

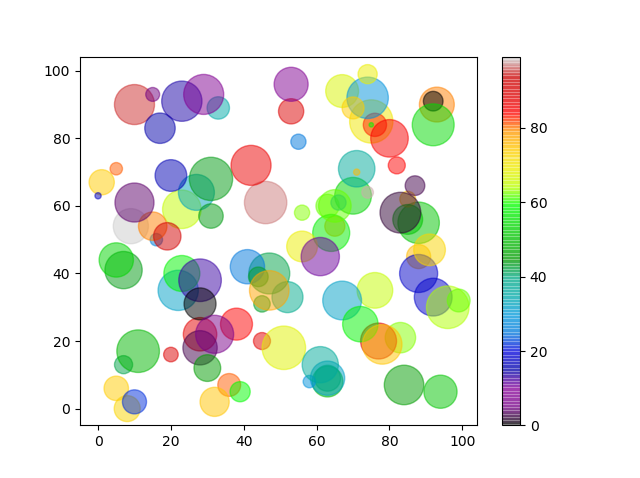
You can combine a colormap with different sizes on the dots. This is best visualized if the dots are transparent:

### Example

Create random arrays with 100 values for x-points, y-points, colors and sizes:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.random.randint(100, size=(100))  
y = np.random.randint(100, size=(100))  
colors = np.random.randint(100, size=(100))  
sizes = 10 \* np.random.randint(100, size=(100))  
  
plt.scatter(x, y, c=colors, s=sizes, alpha=0.5, cmap='nipy\_spectral')  
  
plt.colorbar()  
  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_scatter_combine)

# Matplotlib Bars

## Creating Bars

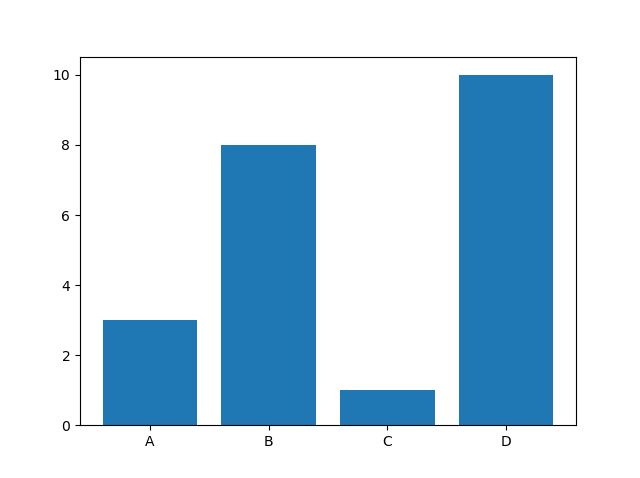
With Pyplot, you can use the bar() function to draw bar graphs:

### Example

Draw 4 bars:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array(["A", "B", "C", "D"])  
y = np.array([3, 8, 1, 10])  
  
plt.bar(x,y)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars1)

The bar() function takes arguments that describes the layout of the bars.

The categories and their values represented by the first and second argument as arrays.

### Example

x = ["APPLES", "BANANAS"]  
y = [400, 350]  
plt.bar(x, y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars2)

## Horizontal Bars

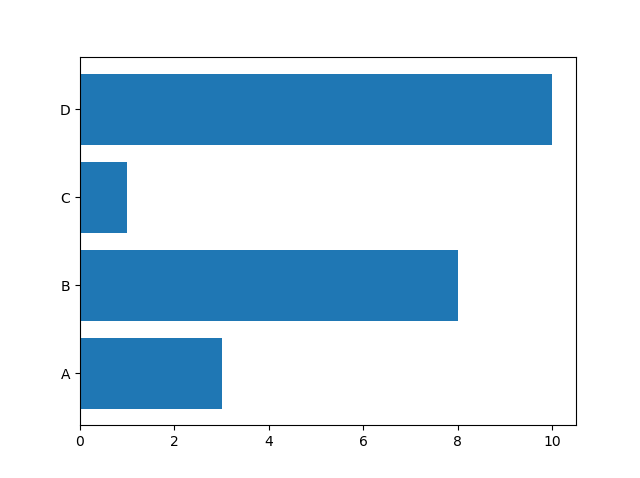
If you want the bars to be displayed horizontally instead of vertically, use the barh() function:

### Example

Draw 4 horizontal bars:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array(["A", "B", "C", "D"])  
y = np.array([3, 8, 1, 10])  
  
plt.barh(x, y)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars3)

## Bar Color

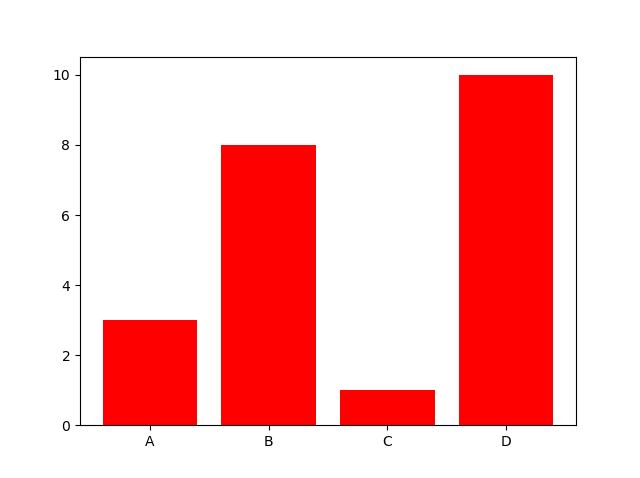
The bar() and barh() takes the keyword argument color to set the color of the bars:

### Example

Draw 4 red bars:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array(["A", "B", "C", "D"])  
y = np.array([3, 8, 1, 10])  
  
plt.bar(x, y, color = "red")  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars4)

### Color Names

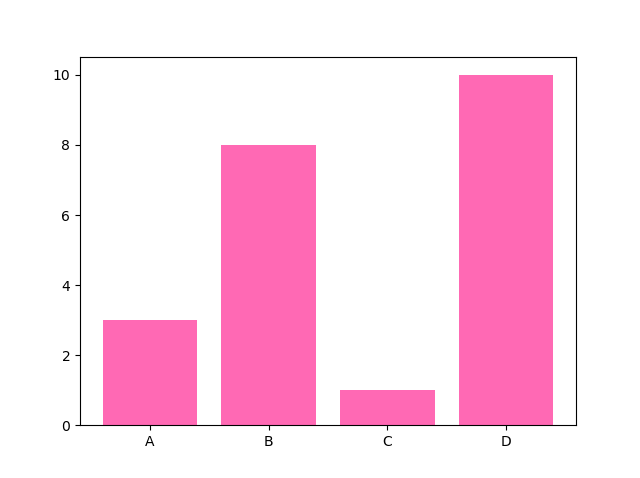
You can use any of the [140 supported color names](https://www.w3schools.com/colors/colors_names.asp).

### Example

Draw 4 "hot pink" bars:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array(["A", "B", "C", "D"])  
y = np.array([3, 8, 1, 10])  
  
plt.bar(x, y, color = "hotpink")  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars5)

### Color Hex

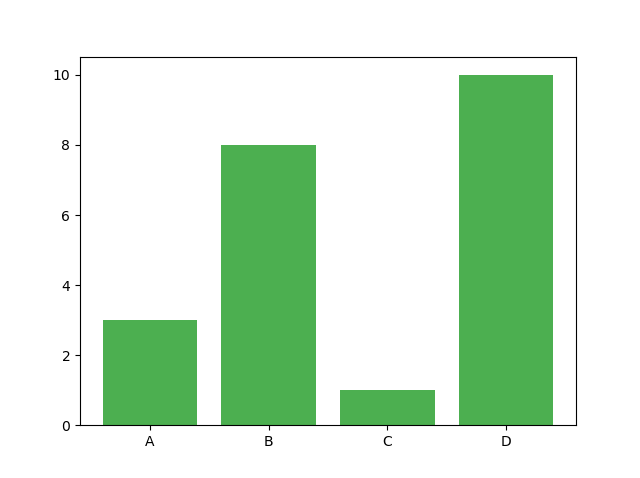
Or you can use [Hexadecimal color values](https://www.w3schools.com/colors/colors_hexadecimal.asp):

### Example

Draw 4 bars with a beautiful green color:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array(["A", "B", "C", "D"])  
y = np.array([3, 8, 1, 10])  
  
plt.bar(x, y, color = "#4CAF50")  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars6)

## Bar Width

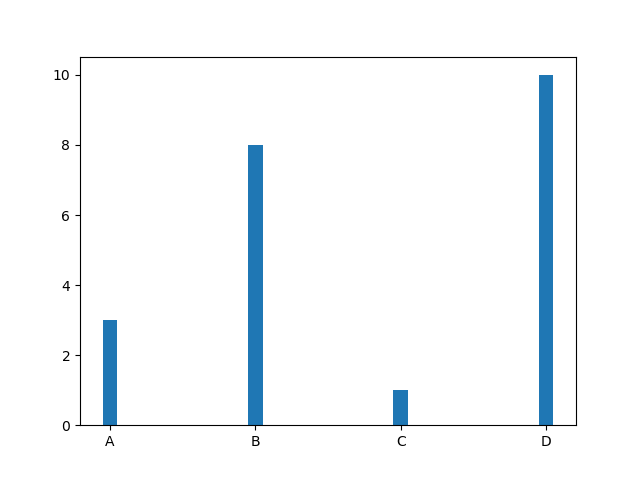
The bar() takes the keyword argument width to set the width of the bars:

### Example

Draw 4 very thin bars:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array(["A", "B", "C", "D"])  
y = np.array([3, 8, 1, 10])  
  
plt.bar(x, y, width = 0.1)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars7)

The default width value is 0.8

**Note:** For horizontal bars, use height instead of width.

## Bar Height

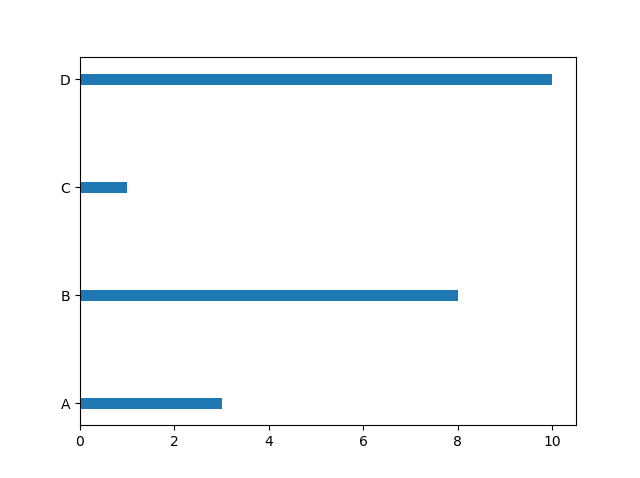
The barh() takes the keyword argument height to set the height of the bars:

### Example

Draw 4 very thin bars:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.array(["A", "B", "C", "D"])  
y = np.array([3, 8, 1, 10])  
  
plt.barh(x, y, height = 0.1)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_bars8)

The default height value is 0.8

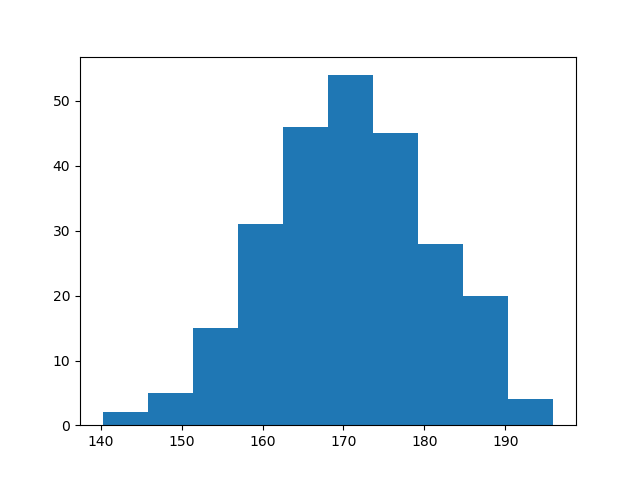
# Matplotlib Histograms

## Histogram

A histogram is a graph showing frequency distributions.

It is a graph showing the number of observations within each given interval.

Example: Say you ask for the height of 250 people, you might end up with a histogram like this:



You can read from the histogram that there are approximately:

2 people from 140 to 145cm  
5 people from 145 to 150cm  
15 people from 151 to 156cm  
31 people from 157 to 162cm  
46 people from 163 to 168cm  
53 people from 168 to 173cm  
45 people from 173 to 178cm  
28 people from 179 to 184cm  
21 people from 185 to 190cm  
4 people from 190 to 195cm

## Create Histogram

In Matplotlib, we use the hist() function to create histograms.

The hist() function will use an array of numbers to create a histogram, the array is sent into the function as an argument.

For simplicity we use NumPy to randomly generate an array with 250 values, where the values will concentrate around 170, and the standard deviation is 10. Learn more about [Normal Data Distribution](https://www.w3schools.com/python/python_ml_normal_data_distribution.asp) in our [Machine Learning Tutorial](https://www.w3schools.com/python/python_ml_getting_started.asp).

### Example

A Normal Data Distribution by NumPy:

import numpy as np  
  
x = np.random.normal(170, 10, 250)  
  
print(x)

### Result:

This will generate a random result, and could look like this:

[167.62255766 175.32495609 152.84661337 165.50264047 163.17457988

162.29867872 172.83638413 168.67303667 164.57361342 180.81120541

170.57782187 167.53075749 176.15356275 176.95378312 158.4125473

187.8842668 159.03730075 166.69284332 160.73882029 152.22378865

164.01255164 163.95288674 176.58146832 173.19849526 169.40206527

166.88861903 149.90348576 148.39039643 177.90349066 166.72462233

177.44776004 170.93335636 173.26312881 174.76534435 162.28791953

166.77301551 160.53785202 170.67972019 159.11594186 165.36992993

178.38979253 171.52158489 173.32636678 159.63894401 151.95735707

175.71274153 165.00458544 164.80607211 177.50988211 149.28106703

179.43586267 181.98365273 170.98196794 179.1093176 176.91855744

168.32092784 162.33939782 165.18364866 160.52300507 174.14316386

163.01947601 172.01767945 173.33491959 169.75842718 198.04834503

192.82490521 164.54557943 206.36247244 165.47748898 195.26377975

164.37569092 156.15175531 162.15564208 179.34100362 167.22138242

147.23667125 162.86940215 167.84986671 172.99302505 166.77279814

196.6137667 159.79012341 166.5840824 170.68645637 165.62204521

174.5559345 165.0079216 187.92545129 166.86186393 179.78383824

161.0973573 167.44890343 157.38075812 151.35412246 171.3107829

162.57149341 182.49985133 163.24700057 168.72639903 169.05309467

167.19232875 161.06405208 176.87667712 165.48750185 179.68799986

158.7913483 170.22465411 182.66432721 173.5675715 176.85646836

157.31299754 174.88959677 183.78323508 174.36814558 182.55474697

180.03359793 180.53094948 161.09560099 172.29179934 161.22665588

171.88382477 159.04626132 169.43886536 163.75793589 157.73710983

174.68921523 176.19843414 167.39315397 181.17128255 174.2674597

186.05053154 177.06516302 171.78523683 166.14875436 163.31607668

174.01429569 194.98819875 169.75129209 164.25748789 180.25773528

170.44784934 157.81966006 171.33315907 174.71390637 160.55423274

163.92896899 177.29159542 168.30674234 165.42853878 176.46256226

162.61719142 166.60810831 165.83648812 184.83238352 188.99833856

161.3054697 175.30396693 175.28109026 171.54765201 162.08762813

164.53011089 189.86213299 170.83784593 163.25869004 198.68079225

166.95154328 152.03381334 152.25444225 149.75522816 161.79200594

162.13535052 183.37298831 165.40405341 155.59224806 172.68678385

179.35359654 174.19668349 163.46176882 168.26621173 162.97527574

192.80170974 151.29673582 178.65251432 163.17266558 165.11172588

183.11107905 169.69556831 166.35149789 178.74419135 166.28562032

169.96465166 178.24368042 175.3035525 170.16496554 158.80682882

187.10006553 178.90542991 171.65790645 183.19289193 168.17446717

155.84544031 177.96091745 186.28887898 187.89867406 163.26716924

169.71242393 152.9410412 158.68101969 171.12655559 178.1482624

187.45272185 173.02872935 163.8047623 169.95676819 179.36887054

157.01955088 185.58143864 170.19037101 157.221245 168.90639755

178.7045601 168.64074373 172.37416382 165.61890535 163.40873027

168.98683006 149.48186389 172.20815568 172.82947206 173.71584064

189.42642762 172.79575803 177.00005573 169.24498561 171.55576698

161.36400372 176.47928342 163.02642822 165.09656415 186.70951892

153.27990317 165.59289527 180.34566865 189.19506385 183.10723435

173.48070474 170.28701875 157.24642079 157.9096498 176.4248199 ]

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_histogram1)

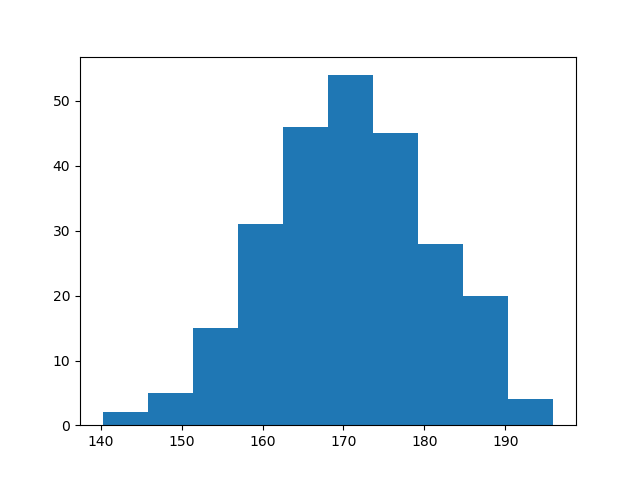
The hist() function will read the array and produce a histogram:

### Example

A simple histogram:

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.random.normal(170, 10, 250)  
  
plt.hist(x)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_histogram2)

# Matplotlib Pie Charts

## Creating Pie Charts

With Pyplot, you can use the pie() function to draw pie charts:

### Example

A simple pie chart:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
  
plt.pie(y)  
plt.show()

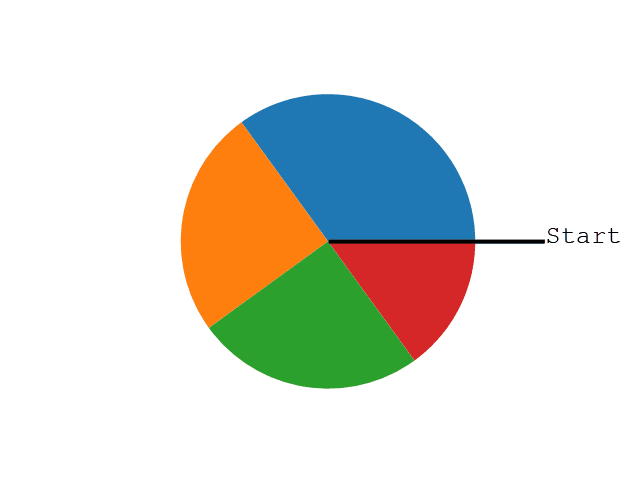
### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie1)

As you can see the pie chart draws one piece (called a wedge) for each value in the array (in this case [35, 25, 25, 15]).

By default the plotting of the first wedge starts from the x-axis and move counterclockwise:



**Note:** The size of each wedge is determined by comparing the value with all the other values, by using this formula:

The value divided by the sum of all values: x/sum(x)

## Labels

Add labels to the pie chart with the label parameter.

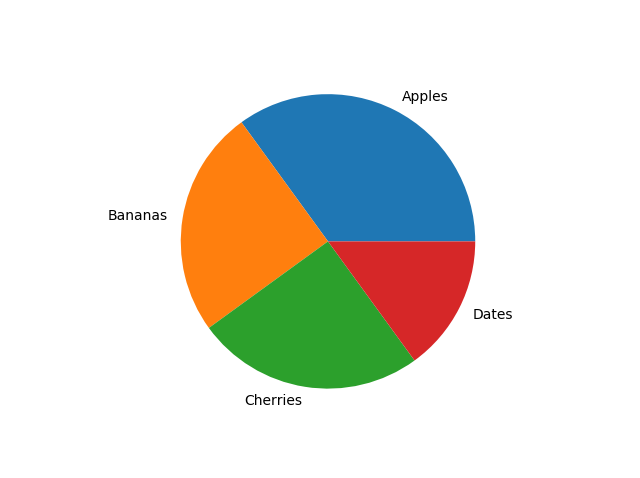
The label parameter must be an array with one label for each wedge:

### Example

A simple pie chart:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]  
  
plt.pie(y, labels = mylabels)  
plt.show()

### Result:

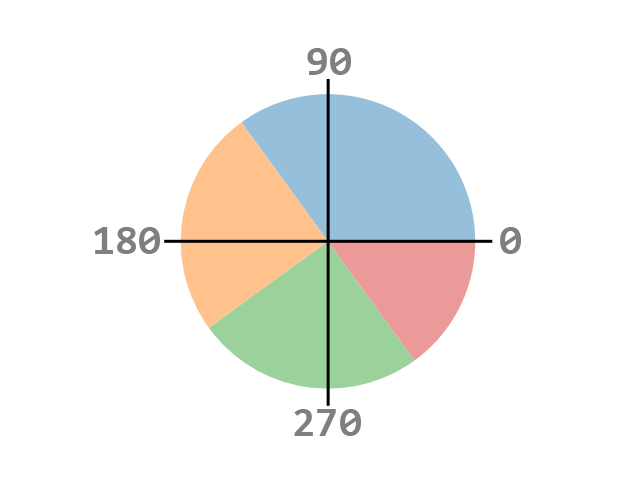


[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie2)

## Start Angle

As mentioned the default start angle is at the x-axis, but you can change the start angle by specifying a startangle parameter.

The startangle parameter is defined with an angle in degrees, default angle is 0:

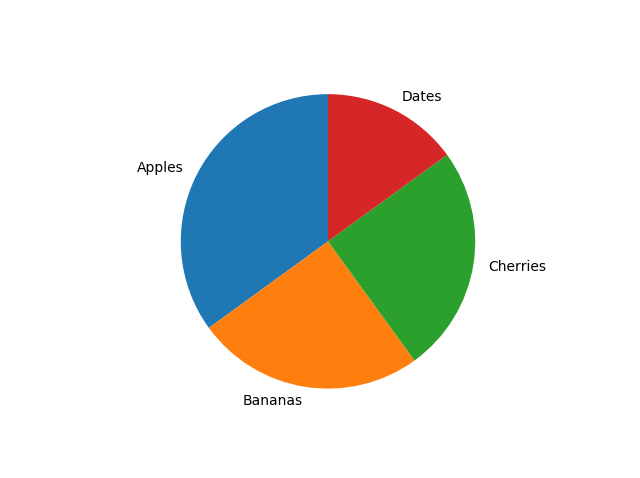


### Example

Start the first wedge at 90 degrees:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]  
  
plt.pie(y, labels = mylabels, startangle = 90)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie3)

## Explode

Maybe you want one of the wedges to stand out? The explode parameter allows you to do that.

The explode parameter, if specified, and not None, must be an array with one value for each wedge.

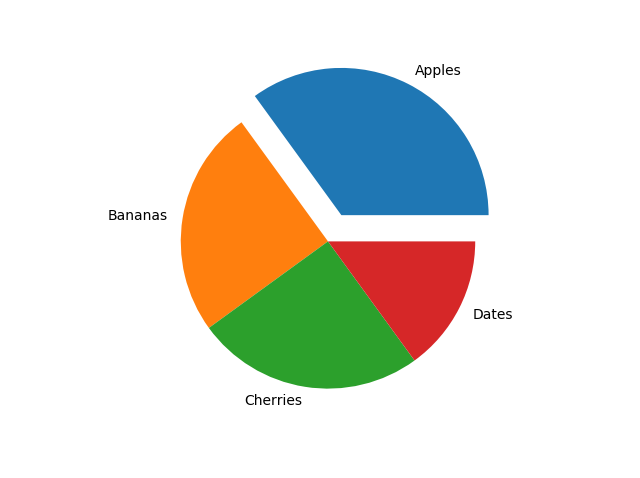
Each value represents how far from the center each wedge is displayed:

### Example

Pull the "Apples" wedge 0.2 from the center of the pie:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]  
myexplode = [0.2, 0, 0, 0]  
  
plt.pie(y, labels = mylabels, explode = myexplode)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie4)

## Shadow

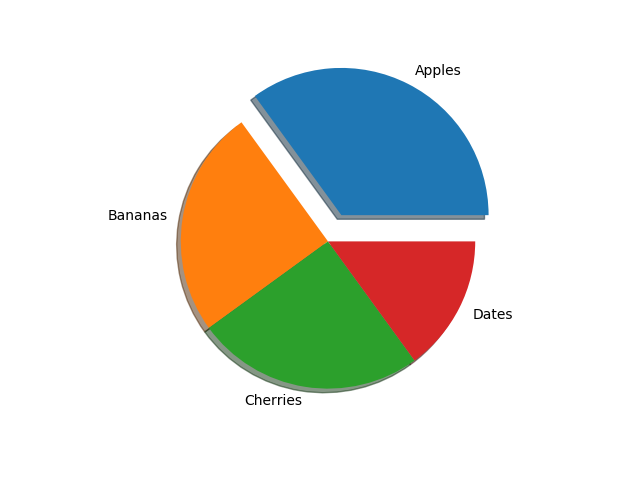
Add a shadow to the pie chart by setting the shadows parameter to True:

### Example

Add a shadow:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]  
myexplode = [0.2, 0, 0, 0]  
  
plt.pie(y, labels = mylabels, explode = myexplode, shadow = True)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie5)

## Colors

You can set the color of each wedge with the colors parameter.

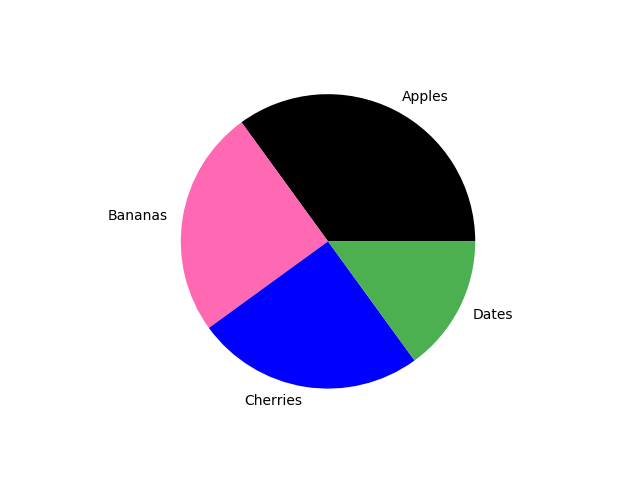
The colors parameter, if specified, must be an array with one value for each wedge:

### Example

Specify a new color for each wedge:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]  
mycolors = ["black", "hotpink", "b", "#4CAF50"]  
  
plt.pie(y, labels = mylabels, colors = mycolors)  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie6)

You can use [Hexadecimal color values](https://www.w3schools.com/colors/colors_hexadecimal.asp), any of the [140 supported color names](https://www.w3schools.com/colors/colors_names.asp), or one of these shortcuts:

'r' - Red  
'g' - Green  
'b' - Blue  
'c' - Cyan  
'm' - Magenta  
'y' - Yellow  
'k' - Black  
'w' - White

## Legend

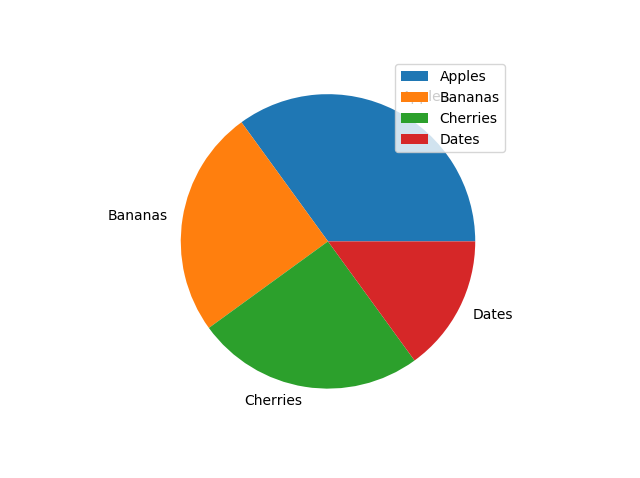
To add a list of explanation for each wedge, use the legend() function:

### Example

Add a legend:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]  
  
plt.pie(y, labels = mylabels)  
plt.legend()  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie7)

### Legend With Header

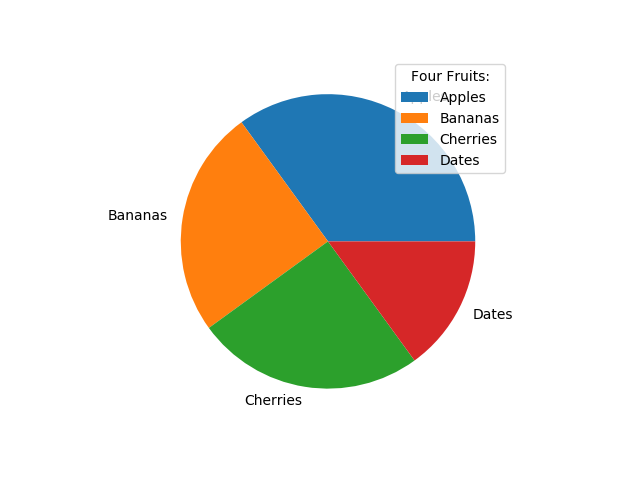
To add a header to the legend, add the title parameter to the legend function.

### Example

Add a legend with a header:

import matplotlib.pyplot as plt  
import numpy as np  
  
y = np.array([35, 25, 25, 15])  
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]  
  
plt.pie(y, labels = mylabels)  
plt.legend(title = "Four Fruits:")  
plt.show()

### Result:



[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_matplotlib_pie8)

# Machine Learning

Machine Learning is making the computer learn from studying data and statistics.

Machine Learning is a step into the direction of artificial intelligence (AI).

Machine Learning is a program that analyses data and learns to predict the outcome.

## Where To Start?

In this tutorial we will go back to mathematics and study statistics, and how to calculate important numbers based on data sets.

We will also learn how to use various Python modules to get the answers we need.

And we will learn how to make functions that are able to predict the outcome based on what we have learned.

## Data Set

In the mind of a computer, a data set is any collection of data. It can be anything from an array to a complete database.

Example of an array:

[99,86,87,88,111,86,103,87,94,78,77,85,86]

Example of a database:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Carname | Color | Age | Speed | AutoPass |
| BMW | red | 5 | 99 | Y |
| Volvo | black | 7 | 86 | Y |
| VW | gray | 8 | 87 | N |
| VW | white | 7 | 88 | Y |
| Ford | white | 2 | 111 | Y |
| VW | white | 17 | 86 | Y |
| Tesla | red | 2 | 103 | Y |
| BMW | black | 9 | 87 | Y |
| Volvo | gray | 4 | 94 | N |
| Ford | white | 11 | 78 | N |
| Toyota | gray | 12 | 77 | N |
| VW | white | 9 | 85 | N |
| Toyota | blue | 6 | 86 | Y |

By looking at the array, we can guess that the average value is probably around 80 or 90, and we are also able to determine the highest value and the lowest value, but what else can we do?

And by looking at the database we can see that the most popular color is white, and the oldest car is 17 years, but what if we could predict if a car had an AutoPass, just by looking at the other values?

That is what Machine Learning is for! Analyzing data and predicting the outcome!

In Machine Learning it is common to work with very large data sets. In this tutorial we will try to make it as easy as possible to understand the different concepts of machine learning, and we will work with small easy-to-understand data sets.

## Data Types

To analyze data, it is important to know what type of data we are dealing with.

We can split the data types into three main categories:

* **Numerical**
* **Categorical**
* **Ordinal**

**Numerical** data are numbers, and can be split into two numerical categories:

* Discrete Data  
  - numbers that are limited to integers. Example: The number of cars passing by.
* Continuous Data  
  - numbers that are of infinite value. Example: The price of an item, or the size of an item

**Categorical** data are values that cannot be measured up against each other. Example: a color value, or any yes/no values.

**Ordinal** data are like categorical data, but can be measured up against each other. Example: school grades where A is better than B and so on.

By knowing the data type of your data source, you will be able to know what technique to use when analyzing them.

You will learn more about statistics and analyzing data in the next chapters.

# Machine Learning - Mean Median Mode

## Mean, Median, and Mode

What can we learn from looking at a group of numbers?

In Machine Learning (and in mathematics) there are often three values that interests us:

* **Mean** - The average value
* **Median** - The mid point value
* **Mode** - The most common value

Example: We have registered the speed of 13 cars:

speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]

What is the average, the middle, or the most common speed value?

## Mean

The mean value is the average value.

To calculate the mean, find the sum of all values, and divide the sum by the number of values:

(99+86+87+88+111+86+103+87+94+78+77+85+86) / 13 = 89.77

The NumPy module has a method for this. Learn about the NumPy module in our [NumPy Tutorial](https://www.w3schools.com/python/numpy/default.asp).

### Example

Use the NumPy mean() method to find the average speed:

import numpy  
  
speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
x = numpy.mean(speed)  
  
print(x)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_mean)

## Median

The median value is the value in the middle, after you have sorted all the values:

77, 78, 85, 86, 86, 86, 87, 87, 88, 94, 99, 103, 111

It is important that the numbers are sorted before you can find the median.

The NumPy module has a method for this:

### Example

Use the NumPy median() method to find the middle value:

import numpy  
  
speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
x = numpy.median(speed)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_median)

If there are two numbers in the middle, divide the sum of those numbers by two.

77, 78, 85, 86, 86, 86, 87, 87, 94, 98, 99, 103  
  
(86 + 87) / 2 = 86.5

### Example

Using the NumPy module:

import numpy  
  
speed = [99,86,87,88,86,103,87,94,78,77,85,86]  
  
x = numpy.median(speed)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_median2)

## Mode

The Mode value is the value that appears the most number of times:

99, 86, 87, 88, 111, 86, 103, 87, 94, 78, 77, 85, 86 = 86

The SciPy module has a method for this. Learn about the SciPy module in our [SciPy Tutorial](https://www.w3schools.com/python/scipy_intro.asp).

### Example

Use the SciPy mode() method to find the number that appears the most:

from scipy import stats  
  
speed = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
x = stats.mode(speed)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_mode)

## Chapter Summary

The Mean, Median, and Mode are techniques that are often used in Machine Learning, so it is important to understand the concept behind them.

# Machine Learning - Standard Deviation

## What is Standard Deviation?

Standard deviation is a number that describes how spread out the values are.

A low standard deviation means that most of the numbers are close to the mean (average) value.

A high standard deviation means that the values are spread out over a wider range.

Example: This time we have registered the speed of 7 cars:

speed = [86,87,88,86,87,85,86]

The standard deviation is:

0.9

Meaning that most of the values are within the range of 0.9 from the mean value, which is 86.4.

Let us do the same with a selection of numbers with a wider range:

speed = [32,111,138,28,59,77,97]

The standard deviation is:

37.85

Meaning that most of the values are within the range of 37.85 from the mean value, which is 77.4.

As you can see, a higher standard deviation indicates that the values are spread out over a wider range.

The NumPy module has a method to calculate the standard deviation:

### Example

Use the NumPy std() method to find the standard deviation:

import numpy  
speed = [86,87,88,86,87,85,86]  
x = numpy.std(speed)  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_std)

### Example

import numpy  
  
speed = [32,111,138,28,59,77,97]  
  
x = numpy.std(speed)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_std2)

## Variance

Variance is another number that indicates how spread out the values are.

In fact, if you take the square root of the variance, you get the standard deviation!

Or the other way around, if you multiply the standard deviation by itself, you get the variance!

To calculate the variance you have to do as follows:

1. Find the mean:

(32+111+138+28+59+77+97) / 7 = 77.4

2. For each value: find the difference from the mean:

 32 - 77.4 = -45.4  
111 - 77.4 =  33.6  
138 - 77.4 =  60.6  
 28 - 77.4 = -49.4  
 59 - 77.4 = -18.4  
 77 - 77.4 = - 0.4  
 97 - 77.4 =  19.6

3. For each difference: find the square value:

(-45.4)2 = 2061.16  
 (33.6)2 = 1128.96  
 (60.6)2 = 3672.36  
(-49.4)2 = 2440.36  
(-18.4)2 =  338.56  
(- 0.4)2 =    0.16  
 (19.6)2 =  384.16

4. The variance is the average number of these squared differences:

(2061.16+1128.96+3672.36+2440.36+338.56+0.16+384.16) / 7 = 1432.2

Luckily, NumPy has a method to calculate the variance:

### Example

Use the NumPy var() method to find the variance:

import numpy  
  
speed = [32,111,138,28,59,77,97]  
  
x = numpy.var(speed)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_var)

## Standard Deviation

As we have learned, the formula to find the standard deviation is the square root of the variance:

√1432.25 = 37.85

Or, as in the example from before, use the NumPy to calculate the standard deviation:

### Example

Use the NumPy std() method to find the standard deviation:

import numpy  
  
speed = [32,111,138,28,59,77,97]  
  
x = numpy.std(speed)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_std2)

## Symbols

Standard Deviation is often represented by the symbol Sigma: σ

Variance is often represented by the symbol Sigma Square: σ2

## Chapter Summary

The Standard Deviation and Variance are terms that are often used in Machine Learning, so it is important to understand how to get them, and the concept behind them.

# Machine Learning - Percentiles

## What are Percentiles?

Percentiles are used in statistics to give you a number that describes the value that a given percent of the values are lower than.

Example: Let's say we have an array of the ages of all the people that lives in a street.

ages = [5,31,43,48,50,41,7,11,15,39,80,82,32,2,8,6,25,36,27,61,31]

What is the 75. percentile? The answer is 43, meaning that 75% of the people are 43 or younger.

The NumPy module has a method for finding the specified percentile:

### Example

Use the NumPy percentile() method to find the percentiles:

import numpy  
  
ages = [5,31,43,48,50,41,7,11,15,39,80,82,32,2,8,6,25,36,27,61,31]  
  
x = numpy.percentile(ages, 75)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_percentile1)

### Example

What is the age that 90% of the people are younger than?

import numpy  
  
ages = [5,31,43,48,50,41,7,11,15,39,80,82,32,2,8,6,25,36,27,61,31]  
  
x = numpy.percentile(ages, 90)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_percentile2)

# Machine Learning - Data Distribution

## Data Distribution

Earlier in this tutorial we have worked with very small amounts of data in our examples, just to understand the different concepts.

In the real world, the data sets are much bigger, but it can be difficult to gather real world data, at least at an early stage of a project.

### How Can we Get Big Data Sets?

To create big data sets for testing, we use the Python module NumPy, which comes with a number of methods to create random data sets, of any size.

### Example

Create an array containing 250 random floats between 0 and 5:

import numpy  
  
x = numpy.random.uniform(0.0, 5.0, 250)  
  
print(x)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_uniform)

## Histogram

To visualize the data set we can draw a histogram with the data we collected.

We will use the Python module Matplotlib to draw a histogram.

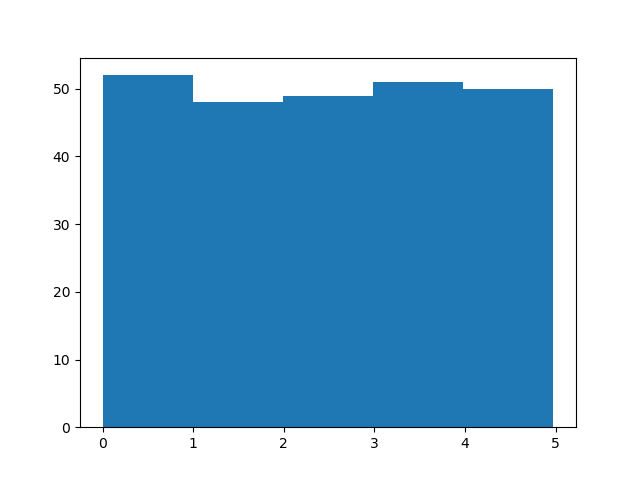
Learn about the Matplotlib module in our [Matplotlib Tutorial](https://www.w3schools.com/python/matplotlib_intro.asp).

### Example

Draw a histogram:

import numpy  
import matplotlib.pyplot as plt  
  
x = numpy.random.uniform(0.0, 5.0, 250)  
  
plt.hist(x, 5)  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_uniform_hist)

### Histogram Explained

We use the array from the example above to draw a histogram with 5 bars.

The first bar represents how many values in the array are between 0 and 1.

The second bar represents how many values are between 1 and 2.

Etc.

Which gives us this result:

* 52 values are between 0 and 1
* 48 values are between 1 and 2
* 49 values are between 2 and 3
* 51 values are between 3 and 4
* 50 values are between 4 and 5

**Note:** The array values are random numbers and will not show the exact same result on your computer.

## Big Data Distributions

An array containing 250 values is not considered very big, but now you know how to create a random set of values, and by changing the parameters, you can create the data set as big as you want.

### Example

Create an array with 100000 random numbers, and display them using a histogram with 100 bars:

import numpy  
import matplotlib.pyplot as plt  
  
x = numpy.random.uniform(0.0, 5.0, 100000)  
  
plt.hist(x, 100)  
plt.show()

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_uniform_big)

# Machine Learning - Normal Data Distribution

## Normal Data Distribution

In the previous chapter we learned how to create a completely random array, of a given size, and between two given values.

In this chapter we will learn how to create an array where the values are concentrated around a given value.

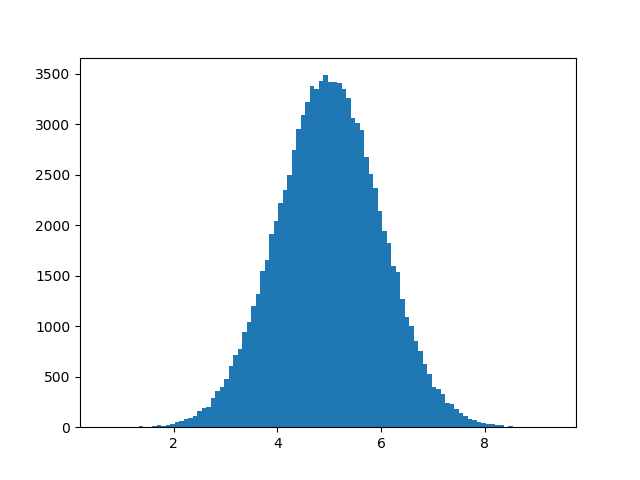
In probability theory this kind of data distribution is known as the normal data distribution, or the Gaussian data distribution, after the mathematician Carl Friedrich Gauss who came up with the formula of this data distribution.

### Example

A typical normal data distribution:

import numpy  
import matplotlib.pyplot as plt  
  
x = numpy.random.normal(5.0, 1.0, 100000)  
  
plt.hist(x, 100)  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_numpy_normal_hist)

**Note:** A normal distribution graph is also known as the bell curve because of it's characteristic shape of a bell.

### Histogram Explained

We use the array from the numpy.random.normal() method, with 100000 values,  to draw a histogram with 100 bars.

We specify that the mean value is 5.0, and the standard deviation is 1.0.

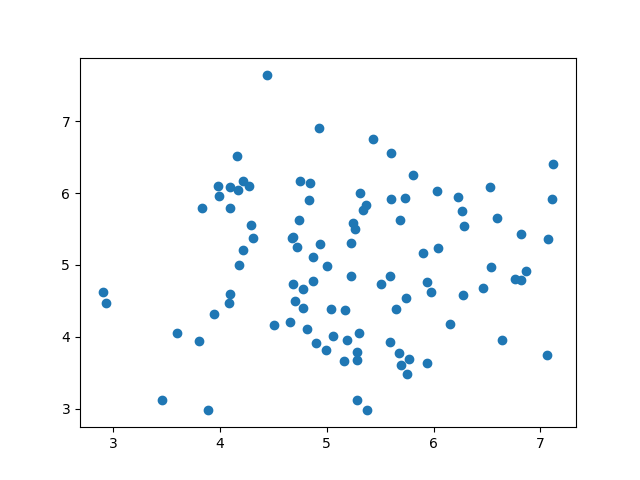
Meaning that the values should be concentrated around 5.0, and rarely further away than 1.0 from the mean.

And as you can see from the histogram, most values are between 4.0 and 6.0, with a top at approximately 5.0.

# Machine Learning - Scatter Plot

## Scatter Plot

A scatter plot is a diagram where each value in the data set is represented by a dot.



The Matplotlib module has a method for drawing scatter plots, it needs two arrays of the same length, one for the values of the x-axis, and one for the values of the y-axis:

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]

y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

The x array represents the age of each car.

The y array represents the speed of each car.

### Example

Use the scatter() method to draw a scatter plot diagram:

import matplotlib.pyplot as plt  
  
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]  
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
plt.scatter(x, y)  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_scatterplot)

### Scatter Plot Explained

The x-axis represents ages, and the y-axis represents speeds.

What we can read from the diagram is that the two fastest cars were both 2 years old, and the slowest car was 12 years old.

**Note:** It seems that the newer the car, the faster it drives, but that could be a coincidence, after all we only registered 13 cars.

## Random Data Distributions

In Machine Learning the data sets can contain thousands-, or even millions, of values.

You might not have real world data when you are testing an algorithm, you might have to use randomly generated values.

As we have learned in the previous chapter, the NumPy module can help us with that!

Let us create two arrays that are both filled with 1000 random numbers from a normal data distribution.

The first array will have the mean set to 5.0 with a standard deviation of 1.0.

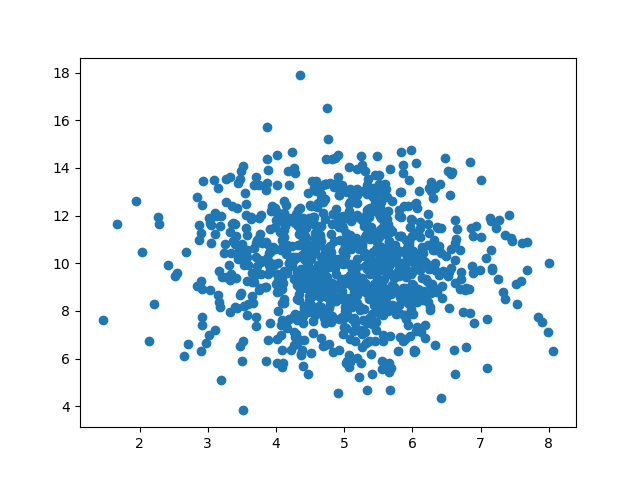
The second array will have the mean set to 10.0 with a standard deviation of 2.0:

### Example

A scatter plot with 1000 dots:

import numpy  
import matplotlib.pyplot as plt  
  
x = numpy.random.normal(5.0, 1.0, 1000)  
y = numpy.random.normal(10.0, 2.0, 1000)  
  
plt.scatter(x, y)  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_scatterplot2)

### Scatter Plot Explained

We can see that the dots are concentrated around the value 5 on the x-axis, and 10 on the y-axis.

We can also see that the spread is wider on the y-axis than on the x-axis.

# Machine Learning - Linear Regression

## Regression

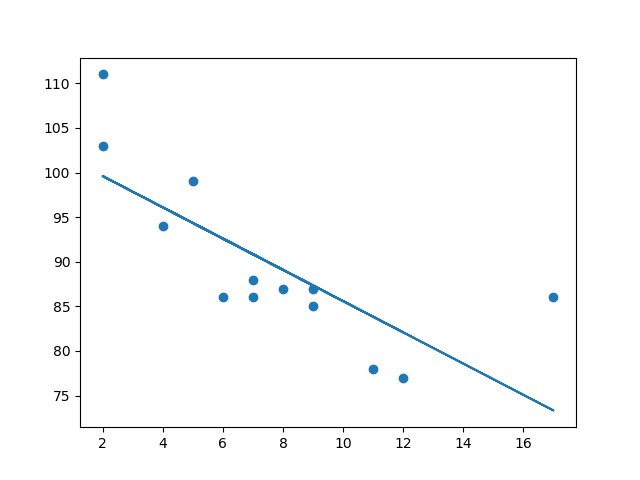
The term regression is used when you try to find the relationship between variables.

In Machine Learning, and in statistical modeling, that relationship is used to predict the outcome of future events.

## Linear Regression

Linear regression uses the relationship between the data-points to draw a straight line through all them.

This line can be used to predict future values.



In Machine Learning, predicting the future is very important.

## How Does it Work?

Python has methods for finding a relationship between data-points and to draw a line of linear regression. We will show you how to use these methods instead of going through the mathematic formula.

In the example below, the x-axis represents age, and the y-axis represents speed. We have registered the age and speed of 13 cars as they were passing a tollbooth. Let us see if the data we collected could be used in a linear regression:

### Example

Start by drawing a scatter plot:

import matplotlib.pyplot as plt  
  
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]  
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
plt.scatter(x, y)  
plt.show()

### Result:



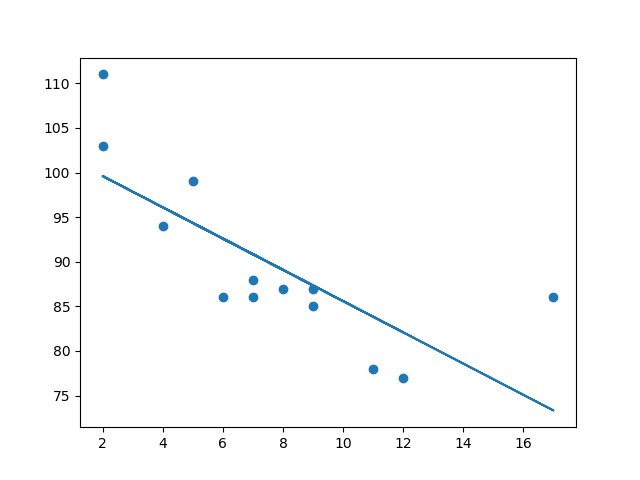
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_scatterplot)

### Example

Import scipy and draw the line of Linear Regression:

import matplotlib.pyplot as plt  
from scipy import stats  
  
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]  
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
slope, intercept, r, p, std\_err = stats.linregress(x, y)  
  
def myfunc(x):  
  return slope \* x + intercept  
  
mymodel = list(map(myfunc, x))  
  
plt.scatter(x, y)  
plt.plot(x, mymodel)  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_linear_regression)

### Example Explained

Import the modules you need.

You can learn about the Matplotlib module in our [Matplotlib Tutorial](https://www.w3schools.com/python/matplotlib_intro.asp).

You can learn about the SciPy module in our [SciPy Tutorial](https://www.w3schools.com/python/scipy_intro.asp).

import matplotlib.pyplot as plt  
from scipy import stats

Create the arrays that represent the values of the x and y axis:

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]  
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

Execute a method that returns some important key values of Linear Regression:

slope, intercept, r, p, std\_err = stats.linregress(x, y)

Create a function that uses the slope and intercept values to return a new value. This new value represents where on the y-axis the corresponding x value will be placed:

def myfunc(x):  
  return slope \* x + intercept

Run each value of the x array through the function. This will result in a new array with new values for the y-axis:

mymodel = list(map(myfunc, x))

Draw the original scatter plot:

plt.scatter(x, y)

Draw the line of linear regression:

plt.plot(x, mymodel)

Display the diagram:

plt.show()

## R for Relationship

It is important to know how the relationship between the values of the x-axis and the values of the y-axis is, if there are no relationship the linear regression can not be used to predict anything.

This relationship - the coefficient of correlation - is called r.

The r value ranges from -1 to 1, where 0 means no relationship, and 1 (and -1) means 100% related.

Python and the Scipy module will compute this value for you, all you have to do is feed it with the x and y values.

### Example

How well does my data fit in a linear regression?

from scipy import stats  
  
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]  
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
slope, intercept, r, p, std\_err = stats.linregress(x, y)  
  
print(r)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_r_squared)

**Note:** The result -0.76 shows that there is a relationship, not perfect, but it indicates that we could use linear regression in future predictions.

## Predict Future Values

Now we can use the information we have gathered to predict future values.

Example: Let us try to predict the speed of a 10 years old car.

To do so, we need the same myfunc() function from the example above:

def myfunc(x):  
  return slope \* x + intercept

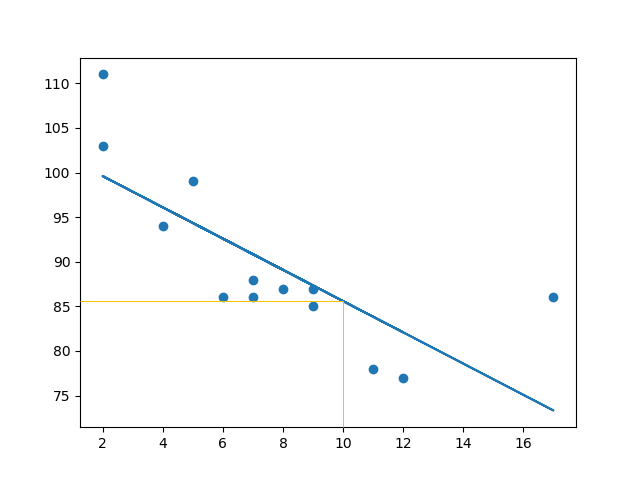
### Example

Predict the speed of a 10 years old car:

from scipy import stats  
  
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]  
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]  
  
slope, intercept, r, p, std\_err = stats.linregress(x, y)  
  
def myfunc(x):  
  return slope \* x + intercept  
  
speed = myfunc(10)  
  
print(speed)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_r_squared2)

The example predicted a speed at 85.6, which we also could read from the diagram:



## Bad Fit?

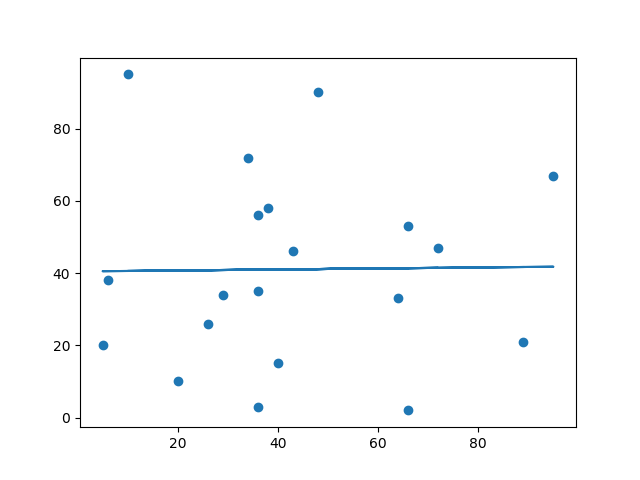
Let us create an example where linear regression would not be the best method to predict future values.

### Example

These values for the x- and y-axis should result in a very bad fit for linear regression:

import matplotlib.pyplot as plt  
from scipy import stats  
  
x = [89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]  
y = [21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]  
  
slope, intercept, r, p, std\_err = stats.linregress(x, y)  
  
def myfunc(x):  
  return slope \* x + intercept  
  
mymodel = list(map(myfunc, x))  
  
plt.scatter(x, y)  
plt.plot(x, mymodel)  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_bad_fit)

And the r for relationship?

### Example

You should get a very low r value.

import numpy  
from scipy import stats  
  
x = [89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]  
y = [21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]  
  
slope, intercept, r, p, std\_err = stats.linregress(x, y)  
  
print(r)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_r_squared_badfit)

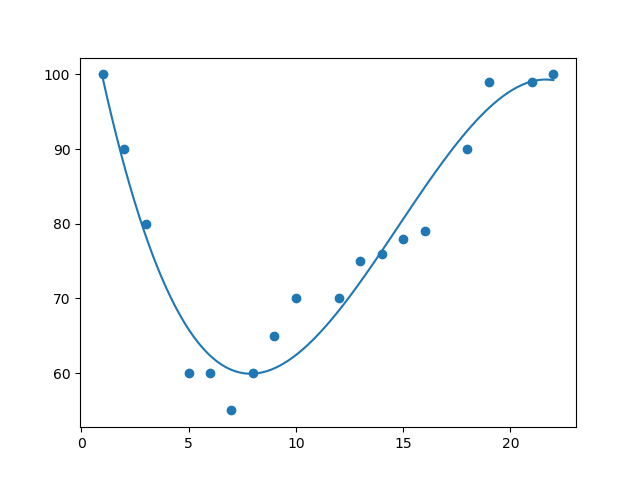
The result: 0.013 indicates a very bad relationship, and tells us that this data set is not suitable for linear regression.

# Machine Learning - Polynomial Regression

## Polynomial Regression

If your data points clearly will not fit a linear regression (a straight line through all data points), it might be ideal for polynomial regression.

Polynomial regression, like linear regression, uses the relationship between the variables x and y to find the best way to draw a line through the data points.



## How Does it Work?

Python has methods for finding a relationship between data-points and to draw a line of polynomial regression. We will show you how to use these methods instead of going through the mathematic formula.

In the example below, we have registered 18 cars as they were passing a certain tollbooth.

We have registered the car's speed, and the time of day (hour) the passing occurred.

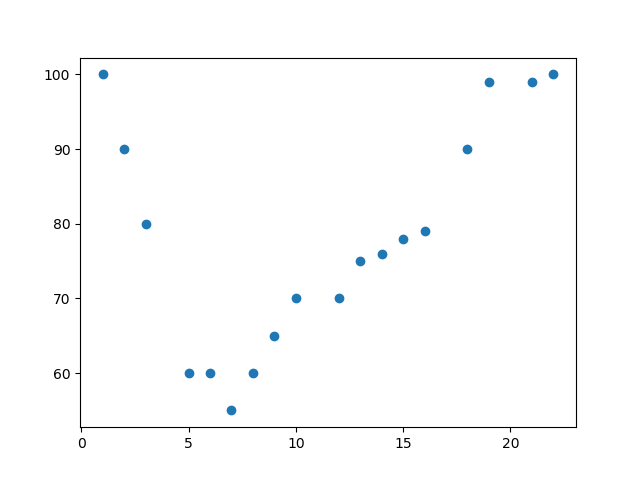
The x-axis represents the hours of the day and the y-axis represents the speed:

### Example

Start by drawing a scatter plot:

import matplotlib.pyplot as plt  
  
x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]  
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]  
  
plt.scatter(x, y)  
plt.show()

### Result:



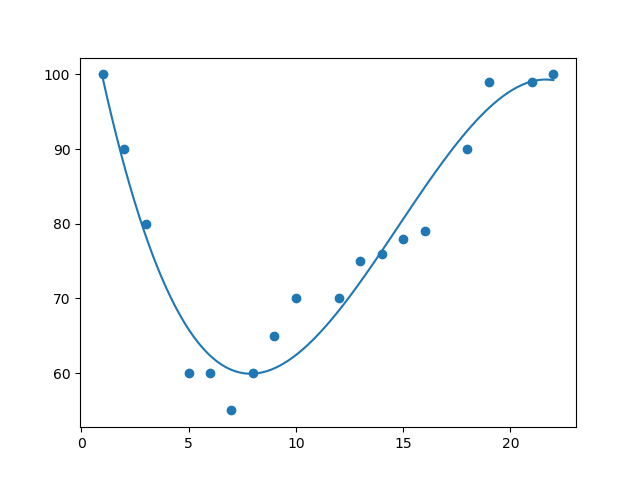
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_polynomial_scatterplot)

### Example

Import numpy and matplotlib then draw the line of Polynomial Regression:

import numpy  
import matplotlib.pyplot as plt  
  
x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]  
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]  
  
mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))  
  
myline = numpy.linspace(1, 22, 100)  
  
plt.scatter(x, y)  
plt.plot(myline, mymodel(myline))  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_polynomial_regr)

### Example Explained

Import the modules you need.

You can learn about the NumPy module in our [NumPy Tutorial](https://www.w3schools.com/python/numpy/default.asp).

You can learn about the SciPy module in our [SciPy Tutorial](https://www.w3schools.com/python/scipy_intro.asp).

import numpy  
import matplotlib.pyplot as plt

Create the arrays that represent the values of the x and y axis:

x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]  
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]

NumPy has a method that lets us make a polynomial model:

mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))

Then specify how the line will display, we start at position 1, and end at position 22:

myline = numpy.linspace(1, 22, 100)

Draw the original scatter plot:

plt.scatter(x, y)

Draw the line of polynomial regression:

plt.plot(myline, mymodel(myline))

Display the diagram:

plt.show()

## R-Squared

It is important to know how well the relationship between the values of the x- and y-axis is, if there are no relationship the polynomial regression can not be used to predict anything.

The relationship is measured with a value called the r-squared.

The r-squared value ranges from 0 to 1, where 0 means no relationship, and 1 means 100% related.

Python and the Sklearn module will compute this value for you, all you have to do is feed it with the x and y arrays:

### Example

How well does my data fit in a polynomial regression?

import numpy  
from sklearn.metrics import r2\_score  
  
x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]  
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]  
  
mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))  
  
print(r2\_score(y, mymodel(x)))

[Try if Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_polynomial_r)

**Note:** The result 0.94 shows that there is a very good relationship, and we can use polynomial regression in future predictions.

## Predict Future Values

Now we can use the information we have gathered to predict future values.

Example: Let us try to predict the speed of a car that passes the tollbooth at around the time 17:00:

To do so, we need the same mymodel array from the example above:

mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))

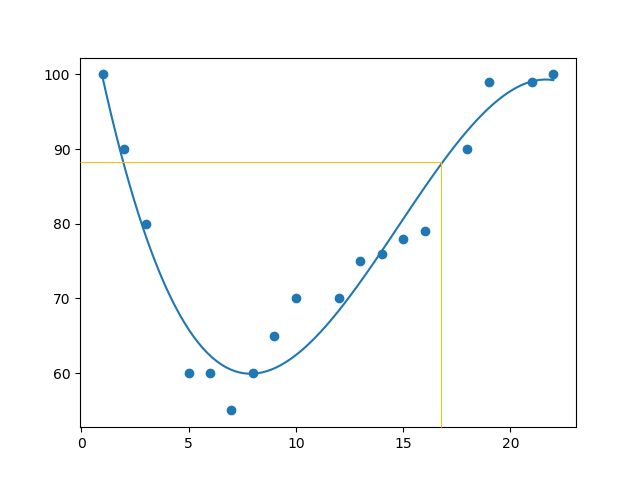
### Example

Predict the speed of a car passing at 17:00:

import numpy  
from sklearn.metrics import r2\_score  
  
x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]  
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]  
  
mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))  
  
speed = mymodel(17)  
print(speed)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_polynomial_predict)

The example predicted a speed to be 88.87, which we also could read from the diagram:



## Bad Fit?

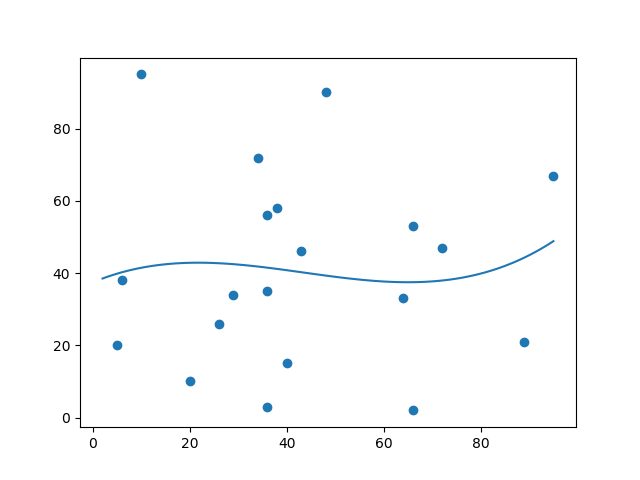
Let us create an example where polynomial regression would not be the best method to predict future values.

### Example

These values for the x- and y-axis should result in a very bad fit for polynomial regression:

import numpy  
import matplotlib.pyplot as plt  
  
x = [89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]  
y = [21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]  
  
mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))  
  
myline = numpy.linspace(2, 95, 100)  
  
plt.scatter(x, y)  
plt.plot(myline, mymodel(myline))  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_polynomial_badfit)

And the r-squared value?

### Example

You should get a very low r-squared value.

import numpy  
from sklearn.metrics import r2\_score  
  
x = [89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]  
y = [21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]  
  
mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))  
  
print(r2\_score(y, mymodel(x)))

[Try if Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_polynomial_badfit_r)

The result: 0.00995 indicates a very bad relationship, and tells us that this data set is not suitable for polynomial regression.

# Machine Learning - Multiple Regression

## Multiple Regression

Multiple regression is like [linear regression](https://www.w3schools.com/python/python_ml_linear_regression.asp), but with more than one independent value, meaning that we try to predict a value based on **two or more** variables.

Take a look at the data set below, it contains some information about cars.

| Car | Model | Volume | Weight | CO2 |  |
| --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Toyota | Aygo | 1000 | 790 | 99 |
| Mitsubishi | Space Star | 1200 | 1160 | 95 |
| Skoda | Citigo | 1000 | 929 | 95 |
| Fiat | 500 | 900 | 865 | 90 |
| Mini | Cooper | 1500 | 1140 | 105 |
| VW | Up! | 1000 | 929 | 105 |
| Skoda | Fabia | 1400 | 1109 | 90 |
| Mercedes | A-Class | 1500 | 1365 | 92 |
| Ford | Fiesta | 1500 | 1112 | 98 |
| Audi | A1 | 1600 | 1150 | 99 |
| Hyundai | I20 | 1100 | 980 | 99 |
| Suzuki | Swift | 1300 | 990 | 101 |
| Ford | Fiesta | 1000 | 1112 | 99 |
| Honda | Civic | 1600 | 1252 | 94 |
| Hundai | I30 | 1600 | 1326 | 97 |
| Opel | Astra | 1600 | 1330 | 97 |
| BMW | 1 | 1600 | 1365 | 99 |
| Mazda | 3 | 2200 | 1280 | 104 |
| Skoda | Rapid | 1600 | 1119 | 104 |
| Ford | Focus | 2000 | 1328 | 105 |
| Ford | Mondeo | 1600 | 1584 | 94 |
| Opel | Insignia | 2000 | 1428 | 99 |
| Mercedes | C-Class | 2100 | 1365 | 99 |
| Skoda | Octavia | 1600 | 1415 | 99 |
| Volvo | S60 | 2000 | 1415 | 99 |
| Mercedes | CLA | 1500 | 1465 | 102 |
| Audi | A4 | 2000 | 1490 | 104 |
| Audi | A6 | 2000 | 1725 | 114 |
| Volvo | V70 | 1600 | 1523 | 109 |
| BMW | 5 | 2000 | 1705 | 114 |
| Mercedes | E-Class | 2100 | 1605 | 115 |
| Volvo | XC70 | 2000 | 1746 | 117 |
| Ford | B-Max | 1600 | 1235 | 104 |
| BMW | 2 | 1600 | 1390 | 108 |
| Opel | Zafira | 1600 | 1405 | 109 |
| Mercedes | SLK | 2500 | 1395 | 120 |

We can predict the CO2 emission of a car based on the size of the engine, but with multiple regression we can throw in more variables, like the weight of the car, to make the prediction more accurate.

## How Does it Work?

In Python we have modules that will do the work for us. Start by importing the Pandas module.

import pandas

Learn about the Pandas module in our [Pandas Tutorial](https://www.w3schools.com/python/pandas_tutorial.asp).

The Pandas module allows us to read csv files and return a DataFrame object.

The file is meant for testing purposes only, you can download it here: [data.csv](https://www.w3schools.com/python/data.csv)

df = pandas.read\_csv("data.csv")

Then make a list of the independent values and call this variable X.

Put the dependent values in a variable called y.

X = df[['Weight', 'Volume']]  
y = df['CO2']

**Tip:** It is common to name the list of independent values with a upper case X, and the list of dependent values with a lower case y.

We will use some methods from the sklearn module, so we will have to import that module as well:

from sklearn import linear\_model

From the sklearn module we will use the LinearRegression() method to create a linear regression object.

This object has a method called fit() that takes the independent and dependent values as parameters and fills the regression object with data that describes the relationship:

regr = linear\_model.LinearRegression()  
regr.fit(X, y)

Now we have a regression object that are ready to predict CO2 values based on a car's weight and volume:

#predict the CO2 emission of a car where the weight is 2300kg, and the volume is 1300cm3:  
predictedCO2 = regr.predict([[2300, 1300]])

### Example

See the whole example in action:

import pandas  
from sklearn import linear\_model  
  
df = pandas.read\_csv("data.csv")  
  
X = df[['Weight', 'Volume']]  
y = df['CO2']  
  
regr = linear\_model.LinearRegression()  
regr.fit(X, y)  
  
#predict the CO2 emission of a car where the weight is 2300kg, and the volume is 1300cm3:  
predictedCO2 = regr.predict([[2300, 1300]])  
  
print(predictedCO2)

### Result:

[107.2087328]

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_multiple_regression)

We have predicted that a car with 1.3 liter engine, and a weight of 2300 kg, will release approximately 107 grams of CO2 for every kilometer it drives.

## Coefficient

The coefficient is a factor that describes the relationship with an unknown variable.

Example: if x is a variable, then 2x is x two times. x is the unknown variable, and the number 2 is the coefficient.

In this case, we can ask for the coefficient value of weight against CO2, and for volume against CO2. The answer(s) we get tells us what would happen if we increase, or decrease, one of the independent values.

### Example

Print the coefficient values of the regression object:

import pandas  
from sklearn import linear\_model  
  
df = pandas.read\_csv("data.csv")  
  
X = df[['Weight', 'Volume']]  
y = df['CO2']  
  
regr = linear\_model.LinearRegression()  
regr.fit(X, y)  
  
print(regr.coef\_)

### Result:

[0.00755095 0.00780526]

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_coef)

## Result Explained

The result array represents the coefficient values of weight and volume.

Weight: 0.00755095  
Volume: 0.00780526

These values tell us that if the weight increase by 1kg, the CO2 emission increases by 0.00755095g.

And if the engine size (Volume) increases by 1 cm3, the CO2 emission increases by 0.00780526 g.

I think that is a fair guess, but let test it!

We have already predicted that if a car with a 1300cm3 engine weighs 2300kg, the CO2 emission will be approximately 107g.

What if we increase the weight with 1000kg?

### Example

Copy the example from before, but change the weight from 2300 to 3300:

import pandas  
from sklearn import linear\_model  
  
df = pandas.read\_csv("data.csv")  
  
X = df[['Weight', 'Volume']]  
y = df['CO2']  
  
regr = linear\_model.LinearRegression()  
regr.fit(X, y)  
  
predictedCO2 = regr.predict([[3300, 1300]])  
  
print(predictedCO2)

### Result:

[114.75968007]

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_coef2)

We have predicted that a car with 1.3 liter engine, and a weight of 3300 kg, will release approximately 115 grams of CO2 for every kilometer it drives.

Which shows that the coefficient of 0.00755095 is correct:

107.2087328 + (1000 \* 0.00755095) = 114.75968

# Machine Learning - Scale

## Scale Features

When your data has different values, and even different measurement units, it can be difficult to compare them. What is kilograms compared to meters? Or altitude compared to time?

The answer to this problem is scaling. We can scale data into new values that are easier to compare.

Take a look at the table below, it is the same data set that we used in the [multiple regression chapter](https://www.w3schools.com/python/python_ml_multiple_regression.asp), but this time the **volume** column contains values in liters instead of cm3 (1.0 instead of 1000).

| Car | Model | Volume | Weight | CO2 |  |
| --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Toyota | Aygo | 1.0 | 790 | 99 |
| Mitsubishi | Space Star | 1.2 | 1160 | 95 |
| Skoda | Citigo | 1.0 | 929 | 95 |
| Fiat | 500 | 0.9 | 865 | 90 |
| Mini | Cooper | 1.5 | 1140 | 105 |
| VW | Up! | 1.0 | 929 | 105 |
| Skoda | Fabia | 1.4 | 1109 | 90 |
| Mercedes | A-Class | 1.5 | 1365 | 92 |
| Ford | Fiesta | 1.5 | 1112 | 98 |
| Audi | A1 | 1.6 | 1150 | 99 |
| Hyundai | I20 | 1.1 | 980 | 99 |
| Suzuki | Swift | 1.3 | 990 | 101 |
| Ford | Fiesta | 1.0 | 1112 | 99 |
| Honda | Civic | 1.6 | 1252 | 94 |
| Hundai | I30 | 1.6 | 1326 | 97 |
| Opel | Astra | 1.6 | 1330 | 97 |
| BMW | 1 | 1.6 | 1365 | 99 |
| Mazda | 3 | 2.2 | 1280 | 104 |
| Skoda | Rapid | 1.6 | 1119 | 104 |
| Ford | Focus | 2.0 | 1328 | 105 |
| Ford | Mondeo | 1.6 | 1584 | 94 |
| Opel | Insignia | 2.0 | 1428 | 99 |
| Mercedes | C-Class | 2.1 | 1365 | 99 |
| Skoda | Octavia | 1.6 | 1415 | 99 |
| Volvo | S60 | 2.0 | 1415 | 99 |
| Mercedes | CLA | 1.5 | 1465 | 102 |
| Audi | A4 | 2.0 | 1490 | 104 |
| Audi | A6 | 2.0 | 1725 | 114 |
| Volvo | V70 | 1.6 | 1523 | 109 |
| BMW | 5 | 2.0 | 1705 | 114 |
| Mercedes | E-Class | 2.1 | 1605 | 115 |
| Volvo | XC70 | 2.0 | 1746 | 117 |
| Ford | B-Max | 1.6 | 1235 | 104 |
| BMW | 2 | 1.6 | 1390 | 108 |
| Opel | Zafira | 1.6 | 1405 | 109 |
| Mercedes | SLK | 2.5 | 1395 | 120 |

It can be difficult to compare the volume 1.0 with the weight 790, but if we scale them both into comparable values, we can easily see how much one value is compared to the other.

There are different methods for scaling data, in this tutorial we will use a method called standardization.

The standardization method uses this formula:

z = (x - u) / s

Where z is the new value, x is the original value, u is the mean and s is the standard deviation.

If you take the **weight** column from the data set above, the first value is 790, and the scaled value will be:

(790 - [1292.23](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_scale_mean1)) / [238.74](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_scale_std1) = -2.1

If you take the **volume** column from the data set above, the first value is 1.0, and the scaled value will be:

(1.0 - [1.61](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_scale_mean2)) / [0.38](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_scale_std2) = -1.59

Now you can compare -2.1 with -1.59 instead of comparing 790 with 1.0.

You do not have to do this manually, the Python sklearn module has a method called StandardScaler() which returns a Scaler object with methods for transforming data sets.

### Example

Scale all values in the Weight and Volume columns:

import pandas  
from sklearn import linear\_model  
from sklearn.preprocessing import StandardScaler  
scale = StandardScaler()  
  
df = pandas.read\_csv("data.csv")  
  
X = df[['Weight', 'Volume']]  
  
scaledX = scale.fit\_transform(X)  
  
print(scaledX)

### Result:

Note that the first two values are -2.1 and -1.59, which corresponds to our calculations:

[[-2.10389253 -1.59336644]

[-0.55407235 -1.07190106]

[-1.52166278 -1.59336644]

[-1.78973979 -1.85409913]

[-0.63784641 -0.28970299]

[-1.52166278 -1.59336644]

[-0.76769621 -0.55043568]

[ 0.3046118 -0.28970299]

[-0.7551301 -0.28970299]

[-0.59595938 -0.0289703 ]

[-1.30803892 -1.33263375]

[-1.26615189 -0.81116837]

[-0.7551301 -1.59336644]

[-0.16871166 -0.0289703 ]

[ 0.14125238 -0.0289703 ]

[ 0.15800719 -0.0289703 ]

[ 0.3046118 -0.0289703 ]

[-0.05142797 1.53542584]

[-0.72580918 -0.0289703 ]

[ 0.14962979 1.01396046]

[ 1.2219378 -0.0289703 ]

[ 0.5685001 1.01396046]

[ 0.3046118 1.27469315]

[ 0.51404696 -0.0289703 ]

[ 0.51404696 1.01396046]

[ 0.72348212 -0.28970299]

[ 0.8281997 1.01396046]

[ 1.81254495 1.01396046]

[ 0.96642691 -0.0289703 ]

[ 1.72877089 1.01396046]

[ 1.30990057 1.27469315]

[ 1.90050772 1.01396046]

[-0.23991961 -0.0289703 ]

[ 0.40932938 -0.0289703 ]

[ 0.47215993 -0.0289703 ]

[ 0.4302729 2.31762392]]

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_scale1)

## Predict CO2 Values

The task in the [Multiple Regression chapter](https://www.w3schools.com/python/python_ml_multiple_regression.asp) was to predict the CO2 emission from a car when you only knew its weight and volume.

When the data set is scaled, you will have to use the scale when you predict values:

### Example

Predict the CO2 emission from a 1.3 liter car that weighs 2300 kilograms:

import pandas  
from sklearn import linear\_model  
from sklearn.preprocessing import StandardScaler  
scale = StandardScaler()  
  
df = pandas.read\_csv("data.csv")  
  
X = df[['Weight', 'Volume']]  
y = df['CO2']  
  
scaledX = scale.fit\_transform(X)  
  
regr = linear\_model.LinearRegression()  
regr.fit(scaledX, y)  
  
scaled = scale.transform([[2300, 1.3]])  
  
predictedCO2 = regr.predict([scaled[0]])  
print(predictedCO2)

### Result:

[107.2087328]

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_scale_predict)

# Machine Learning - Train/Test

## Evaluate Your Model

In Machine Learning we create models to predict the outcome of certain events, like in the previous chapter where we predicted the CO2 emission of a car when we knew the weight and engine size.

To measure if the model is good enough, we can use a method called Train/Test.

## What is Train/Test

Train/Test is a method to measure the accuracy of your model.

It is called Train/Test because you split the the data set into two sets: a training set and a testing set.

80% for training, and 20% for testing.

You train the model using the training set.

You test the model using the testing set.

Train the model means create the model.

Test the model means test the accuracy of the model.

## Start With a Data Set

Start with a data set you want to test.

Our data set illustrates 100 customers in a shop, and their shopping habits.

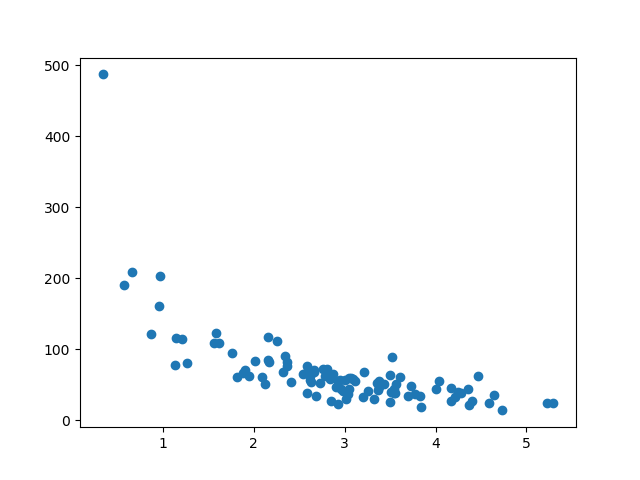
### Example

import numpy  
import matplotlib.pyplot as plt  
numpy.random.seed(2)  
  
x = numpy.random.normal(3, 1, 100)  
y = numpy.random.normal(150, 40, 100) / x  
  
plt.scatter(x, y)  
plt.show()

### Result:

The x axis represents the number of minutes before making a purchase.

The y axis represents the amount of money spent on the purchase.



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_traintest1)

## Split Into Train/Test

The training set should be a random selection of 80% of the original data.

The testing set should be the remaining 20%.

train\_x = x[:80]  
train\_y = y[:80]  
  
test\_x = x[80:]  
test\_y = y[80:]

## Display the Training Set

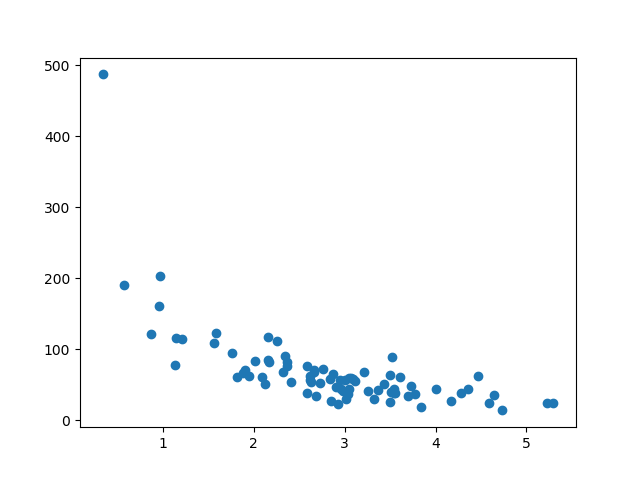
Display the same scatter plot with the training set:

### Example

plt.scatter(train\_x, train\_y)  
plt.show()

### Result:

It looks like the original data set, so it seems to be a fair selection:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_traintest2)

## Display the Testing Set

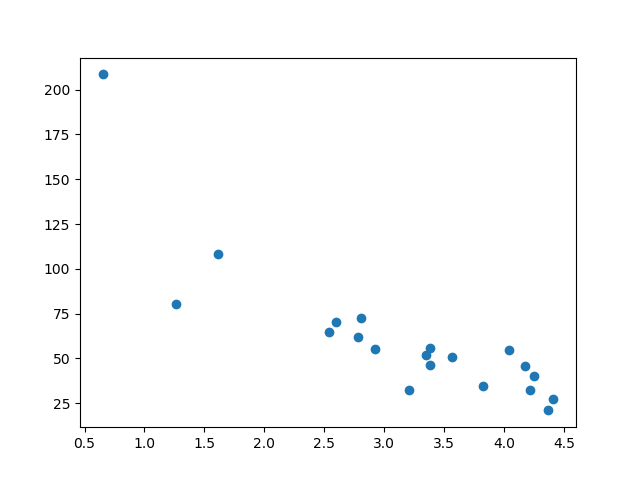
To make sure the testing set is not completely different, we will take a look at the testing set as well.

### Example

plt.scatter(test\_x, test\_y)  
plt.show()

### Result:

The testing set also looks like the original data set:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_traintest3)

## Fit the Data Set

What does the data set look like? In my opinion I think the best fit would be a [polynomial regression](https://www.w3schools.com/python/python_ml_polynomial_regression.asp), so let us draw a line of polynomial regression.

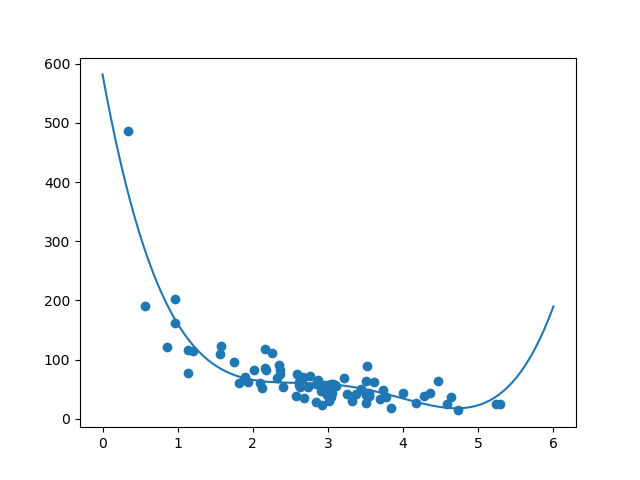
To draw a line through the data points, we use the plot() method of the matplotlib module:

### Example

Draw a polynomial regression line through the data points:

import numpy  
import matplotlib.pyplot as plt  
numpy.random.seed(2)  
  
x = numpy.random.normal(3, 1, 100)  
y = numpy.random.normal(150, 40, 100) / x  
  
train\_x = x[:80]  
train\_y = y[:80]  
  
test\_x = x[80:]  
test\_y = y[80:]  
  
mymodel = numpy.poly1d(numpy.polyfit(train\_x, train\_y, 4))  
  
myline = numpy.linspace(0, 6, 100)  
  
plt.scatter(train\_x, train\_y)  
plt.plot(myline, mymodel(myline))  
plt.show()

### Result:



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_traintest3_2)

The result can back my suggestion of the data set fitting a polynomial regression, even though it would give us some weird results if we try to predict values outside of the data set. Example: the line indicates that a customer spending 6 minutes in the shop would make a purchase worth 200. That is probably a sign of overfitting.

But what about the R-squared score? The R-squared score is a good indicator of how well my data set is fitting the model.

## R2

Remember R2, also known as R-squared?

It measures the relationship between the x axis and the y axis, and the value ranges from 0 to 1, where 0 means no relationship, and 1 means totally related.

The sklearn module has a method called r2\_score() that will help us find this relationship.

In this case we would like to measure the relationship between the minutes a customer stays in the shop and how much money they spend.

### Example

How well does my training data fit in a polynomial regression?

import numpy  
from sklearn.metrics import r2\_score  
numpy.random.seed(2)  
  
x = numpy.random.normal(3, 1, 100)  
y = numpy.random.normal(150, 40, 100) / x  
  
train\_x = x[:80]  
train\_y = y[:80]  
  
test\_x = x[80:]  
test\_y = y[80:]  
  
mymodel = numpy.poly1d(numpy.polyfit(train\_x, train\_y, 4))  
  
r2 = r2\_score(train\_y, mymodel(train\_x))  
  
print(r2)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_traintest4)

**Note:** The result 0.799 shows that there is a OK relationship.

## Bring in the Testing Set

Now we have made a model that is OK, at least when it comes to training data.

Now we want to test the model with the testing data as well, to see if gives us the same result.

### Example

Let us find the R2 score when using testing data:

import numpy  
from sklearn.metrics import r2\_score  
numpy.random.seed(2)  
  
x = numpy.random.normal(3, 1, 100)  
y = numpy.random.normal(150, 40, 100) / x  
  
train\_x = x[:80]  
train\_y = y[:80]  
  
test\_x = x[80:]  
test\_y = y[80:]  
  
mymodel = numpy.poly1d(numpy.polyfit(train\_x, train\_y, 4))  
  
r2 = r2\_score(test\_y, mymodel(test\_x))  
  
print(r2)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_traintest5)

**Note:** The result 0.809 shows that the model fits the testing set as well, and we are confident that we can use the model to predict future values.

## Predict Values

Now that we have established that our model is OK, we can start predicting new values.

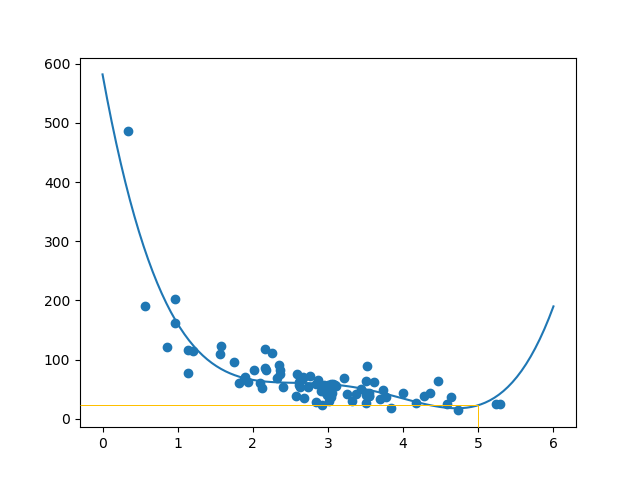
### Example

How much money will a buying customer spend, if she or he stays in the shop for 5 minutes?

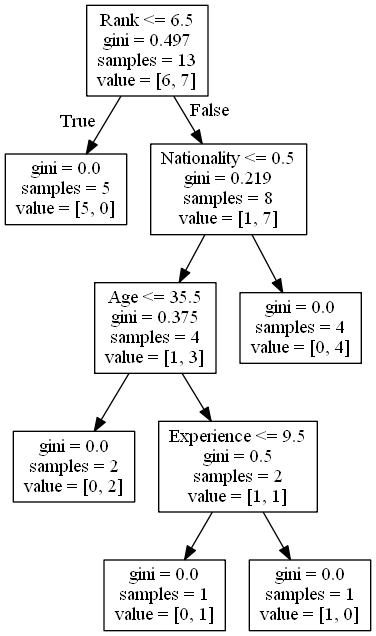
print(mymodel(5))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_traintest6)

The example predicted the customer to spend 22.88 dollars, as seems to correspond to the diagram:



# Machine Learning - Decision Tree



## Decision Tree

In this chapter we will show you how to make a "Decision Tree". A Decision Tree is a Flow Chart, and can help you make decisions based on previous experience.

In the example, a person will try to decide if he/she should go to a comedy show or not.

Luckily our example person has registered every time there was a comedy show in town, and registered some information about the comedian, and also registered if he/she went or not.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Age | Experience | Rank | Nationality | Go |
| 36 | 10 | 9 | UK | NO |
| 42 | 12 | 4 | USA | NO |
| 23 | 4 | 6 | N | NO |
| 52 | 4 | 4 | USA | NO |
| 43 | 21 | 8 | USA | YES |
| 44 | 14 | 5 | UK | NO |
| 66 | 3 | 7 | N | YES |
| 35 | 14 | 9 | UK | YES |
| 52 | 13 | 7 | N | YES |
| 35 | 5 | 9 | N | YES |
| 24 | 3 | 5 | USA | NO |
| 18 | 3 | 7 | UK | YES |
| 45 | 9 | 9 | UK | YES |

Now, based on this data set, Python can create a decision tree that can be used to decide if any new shows are worth attending to.

## How Does it Work?

First, read the dataset with pandas:

### Example

Read and print the data set:

import pandas  
  
df = pandas.read\_csv("data.csv")  
  
print(df)

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_dtree1)

To make a decision tree, all data has to be numerical.

We have to convert the non numerical columns 'Nationality' and 'Go' into numerical values.

Pandas has a map() method that takes a dictionary with information on how to convert the values.

{'UK': 0, 'USA': 1, 'N': 2}

Means convert the values 'UK' to 0, 'USA' to 1, and 'N' to 2.

### Example

Change string values into numerical values:

d = {'UK': 0, 'USA': 1, 'N': 2}  
df['Nationality'] = df['Nationality'].map(d)  
d = {'YES': 1, 'NO': 0}  
df['Go'] = df['Go'].map(d)  
  
print(df)

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_dtree2)

Then we have to separate the feature columns from the target column.

The feature columns are the columns that we try to predict from, and the target column is the column with the values we try to predict.

### Example

X is the feature columns, y is the target column:

features = ['Age', 'Experience', 'Rank', 'Nationality']  
  
X = df[features]  
y = df['Go']  
  
print(X)  
print(y)

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_dtree3)

Now we can create the actual decision tree, fit it with our details. Start by importing the modules we need:

### Example

Create and display a Decision Tree:

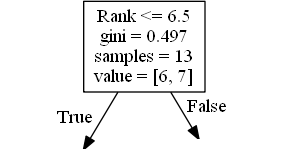
import pandas  
from sklearn import tree  
from sklearn.tree import DecisionTreeClassifier  
import matplotlib.pyplot as plt  
  
df = pandas.read\_csv("data.csv")  
  
d = {'UK': 0, 'USA': 1, 'N': 2}  
df['Nationality'] = df['Nationality'].map(d)  
d = {'YES': 1, 'NO': 0}  
df['Go'] = df['Go'].map(d)  
  
features = ['Age', 'Experience', 'Rank', 'Nationality']  
  
X = df[features]  
y = df['Go']  
  
dtree = DecisionTreeClassifier()  
dtree = dtree.fit(X, y)  
  
tree.plot\_tree(dtree, feature\_names=features)

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_dtree4)

## Result Explained

The decision tree uses your earlier decisions to calculate the odds for you to wanting to go see a comedian or not.

Let us read the different aspects of the decision tree:



### Rank

Rank <= 6.5 means that every comedian with a rank of 6.5 or lower will follow the True arrow (to the left), and the rest will follow the False arrow (to the right).

gini = 0.497 refers to the quality of the split, and is always a number between 0.0 and 0.5, where 0.0 would mean all of the samples got the same result, and 0.5 would mean that the split is done exactly in the middle.

samples = 13 means that there are 13 comedians left at this point in the decision, which is all of them since this is the first step.

value = [6, 7] means that of these 13 comedians, 6 will get a "NO", and 7 will get a "GO".

### Gini

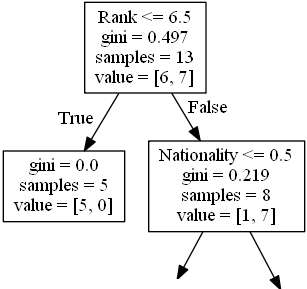
There are many ways to split the samples, we use the GINI method in this tutorial.

The Gini method uses this formula:

Gini = 1 - (x/n)2 - (y/n)2

Where x is the number of positive answers("GO"), n is the number of samples, and y is the number of negative answers ("NO"), which gives us this calculation:

1 - (7 / 13)2 - (6 / 13)2 = 0.497



The next step contains two boxes, one box for the comedians with a 'Rank' of 6.5 or lower, and one box with the rest.

### True - 5 Comedians End Here:

gini = 0.0 means all of the samples got the same result.

samples = 5 means that there are 5 comedians left in this branch (5 comedian with a Rank of 6.5 or lower).

value = [5, 0] means that 5 will get a "NO" and 0 will get a "GO".

### False - 8 Comedians Continue:

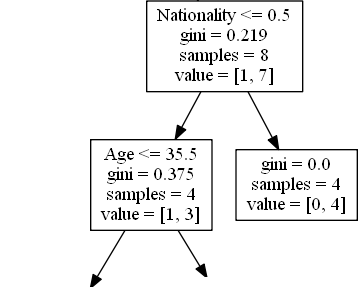
### Nationality

Nationality <= 0.5 means that the comedians with a nationality value of less than 0.5 will follow the arrow to the left (which means everyone from the UK, ), and the rest will follow the arrow to the right.

gini = 0.219 means that about 22% of the samples would go in one direction.

samples = 8 means that there are 8 comedians left in this branch (8 comedian with a Rank higher than 6.5).

value = [1, 7] means that of these 8 comedians, 1 will get a "NO" and 7 will get a "GO".



### True - 4 Comedians Continue:

### Age

Age <= 35.5 means that comedians at the age of 35.5 or younger will follow the arrow to the left, and the rest will follow the arrow to the right.

gini = 0.375 means that about 37,5% of the samples would go in one direction.

samples = 4 means that there are 4 comedians left in this branch (4 comedians from the UK).

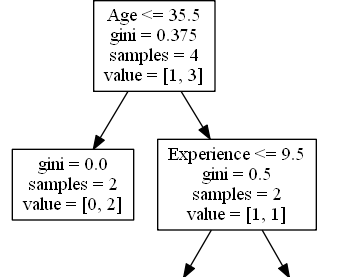
value = [1, 3] means that of these 4 comedians, 1 will get a "NO" and 3 will get a "GO".

### False - 4 Comedians End Here:

gini = 0.0 means all of the samples got the same result.

samples = 4 means that there are 4 comedians left in this branch (4 comedians not from the UK).

value = [0, 4] means that of these 4 comedians, 0 will get a "NO" and 4 will get a "GO".



### True - 2 Comedians End Here:

gini = 0.0 means all of the samples got the same result.

samples = 2 means that there are 2 comedians left in this branch (2 comedians at the age 35.5 or younger).

value = [0, 2] means that of these 2 comedians, 0 will get a "NO" and 2 will get a "GO".

### False - 2 Comedians Continue:

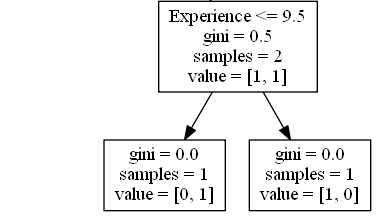
### Experience

Experience <= 9.5 means that comedians with 9.5 years of experience, or less, will follow the arrow to the left, and the rest will follow the arrow to the right.

gini = 0.5 means that 50% of the samples would go in one direction.

samples = 2 means that there are 2 comedians left in this branch (2 comedians older than 35.5).

value = [1, 1] means that of these 2 comedians, 1 will get a "NO" and 1 will get a "GO".



### True - 1 Comedian Ends Here:

gini = 0.0 means all of the samples got the same result.

samples = 1 means that there is 1 comedian left in this branch (1 comedian with 9.5 years of experience or less).

value = [0, 1] means that 0 will get a "NO" and 1 will get a "GO".

### False - 1 Comedian Ends Here:

gini = 0.0 means all of the samples got the same result.

samples = 1 means that there is 1 comedians left in this branch (1 comedian with more than 9.5 years of experience).

value = [1, 0] means that 1 will get a "NO" and 0 will get a "GO".

## Predict Values

We can use the Decision Tree to predict new values.

Example: Should I go see a show starring a 40 years old American comedian, with 10 years of experience, and a comedy ranking of 7?

### Example

Use predict() method to predict new values:

print(dtree.predict([[40, 10, 7, 1]]))

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_dtree_predict1)

### Example

What would the answer be if the comedy rank was 6?

print(dtree.predict([[40, 10, 6, 1]]))

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_dtree_predict2)

### Different Results

You will see that the Decision Tree gives you different results if you run it enough times, even if you feed it with the same data.

That is because the Decision Tree does not give us a 100% certain answer. It is based on the probability of an outcome, and the answer will vary.

# Machine Learning - Confusion Matrix

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## What is a confusion matrix?

It is a table that is used in classification problems to assess where errors in the model were made.

The rows represent the actual classes the outcomes should have been. While the columns represent the predictions we have made. Using this table it is easy to see which predictions are wrong.

## Creating a Confusion Matrix

Confusion matrixes can be created by predictions made from a logistic regression.

For now we will generate actual and predicted values by utilizing NumPy:

import numpy

Next we will need to generate the numbers for "actual" and "predicted" values.

actual = numpy.random.binomial(1, 0.9, size = 1000)  
predicted = numpy.random.binomial(1, 0.9, size = 1000)

In order to create the confusion matrix we need to import metrics from the sklearn module.

from sklearn import metrics

Once metrics is imported we can use the confusion matrix function on our actual and predicted values.

confusion\_matrix = metrics.confusion\_matrix(actual, predicted)

To create a more interpretable visual display we need to convert the table into a confusion matrix display.

cm\_display = metrics.ConfusionMatrixDisplay(confusion\_matrix = confusion\_matrix, display\_labels = [False, True])

Vizualizing the display requires that we import pyplot from matplotlib.

import matplotlib.pyplot as plt

Finally to display the plot we can use the functions plot() and show() from pyplot.

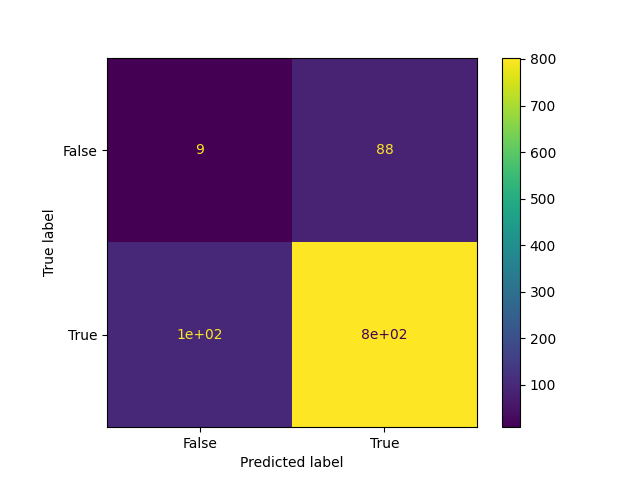
cm\_display.plot()  
plt.show()

See the whole example in action:

### Example

import matplotlib.pyplot as plt  
import numpy  
from sklearn import metrics  
  
actual = numpy.random.binomial(1,.9,size = 1000)  
predicted = numpy.random.binomial(1,.9,size = 1000)  
  
confusion\_matrix = metrics.confusion\_matrix(actual, predicted)  
  
cm\_display = metrics.ConfusionMatrixDisplay(confusion\_matrix = confusion\_matrix, display\_labels = [False, True])  
  
cm\_display.plot()  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_confusion_matrix)

## Results Explained

The Confusion Matrix created has four different quadrants:

True Negative (Top-Left Quadrant)  
False Positive (Top-Right Quadrant)  
False Negative (Bottom-Left Quadrant)  
True Positive (Bottom-Right Quadrant)

True means that the values were accurately predicted, False means that there was an error or wrong prediction.

Now that we have made a Confusion Matrix, we can calculate different measures to quantify the quality of the model. First, lets look at Accuracy.

ADVERTISEMENT

[](https://community.nycdatascience.edu/courses/intro-to-data-science?utm_campaign=w3school&utm_source=W3School&utm_medium=display%20ads&utm_term=PPC&utm_content=Free%20course)

## Created Metrics

The matrix provides us with many useful metrics that help us to evaluate out classification model.

The different measures include: Accuracy, Precision, Sensitivity (Recall), Specificity, and the F-score, explained below.

## Accuracy

Accuracy measures how often the model is correct.

### How to Calculate

(True Positive + True Negative) / Total Predictions

### Example

Accuracy = metrics.accuracy\_score(actual, predicted)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_confusion_matrix_accuracy)

## Precision

Of the positives predicted, what percentage is truly positive?

### How to Calculate

True Positive / (True Positive + False Positive)

Precision does not evaluate the correctly predicted negative cases:

### Example

Precision = metrics.precision\_score(actual, predicted)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_confusion_matrix_precision)

## Sensitivity (Recall)

Of all the positive cases, what percentage are predicted positive?

Sensitivity (sometimes called Recall) measures how good the model is at predicting positives.

This means it looks at true positives and false negatives (which are positives that have been incorrectly predicted as negative).

### How to Calculate

True Positive / (True Positive + False Negative)

Sensitivity is good at understanding how well the model predicts something is positive:

### Example

Sensitivity\_recall = metrics.recall\_score(actual, predicted)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_confusion_matrix_sensivity)

## Specificity

How well the model is at prediciting negative results?

Specificity is similar to sensitivity, but looks at it from the persepctive of negative results.

### How to Calculate

True Negative / (True Negative + False Positive)

Since it is just the opposite of Recall, we use the recall\_score function, taking the opposite position label:

### Example

Specificity = metrics.recall\_score(actual, predicted, pos\_label=0)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_confusion_matrix_specificity)

## F-score

F-score is the "harmonic mean" of precision and sensitivity.

It considers both false positive and false negative cases and is good for imbalanced datasets.

### How to Calculate

2 \* ((Precision \* Sensitivity) / (Precision + Sensitivity))

This score does not take into consideration the True Negative values:

### Example

F1\_score = metrics.f1\_score(actual, predicted)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_confusion_matrix_f-score)

All calulations in one:

### Example

#metrics  
print({"Accuracy":Accuracy,"Precision":Precision,"Sensitivity\_recall":Sensitivity\_recall,"Specificity":Specificity,"F1\_score":F1\_score})

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_confusion_matrix_all)

# Machine Learning - Hierarchical Clustering

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## Hierarchical Clustering

Hierarchical clustering is an unsupervised learning method for clustering data points. The algorithm builds clusters by measuring the dissimilarities between data. Unsupervised learning means that a model does not have to be trained, and we do not need a "target" variable. This method can be used on any data to visualize and interpret the relationship between individual data points.

Here we will use hierarchical clustering to group data points and visualize the clusters using both a dendrogram and scatter plot.

## How does it work?

We will use Agglomerative Clustering, a type of hierarchical clustering that follows a bottom up approach. We begin by treating each data point as its own cluster. Then, we join clusters together that have the shortest distance between them to create larger clusters. This step is repeated until one large cluster is formed containing all of the data points.

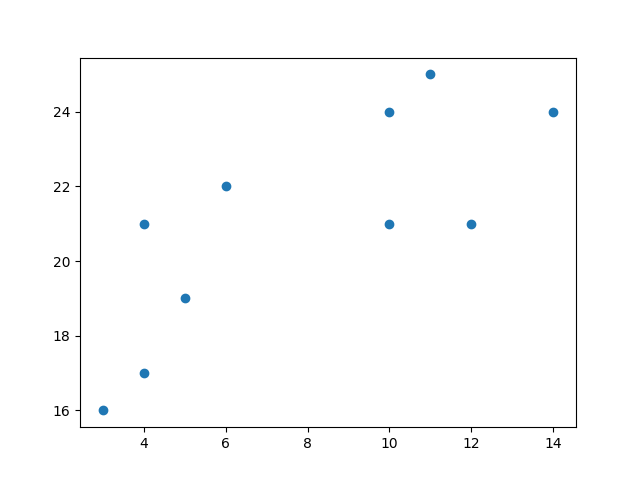
Hierarchical clustering requires us to decide on both a distance and linkage method. We will use euclidean distance and the Ward linkage method, which attempts to minimize the variance between clusters.

### Example

Start by visualizing some data points:

import numpy as np  
import matplotlib.pyplot as plt  
  
x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]  
  
plt.scatter(x, y)  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_hierarchial_clustering)

ADVERTISEMENT

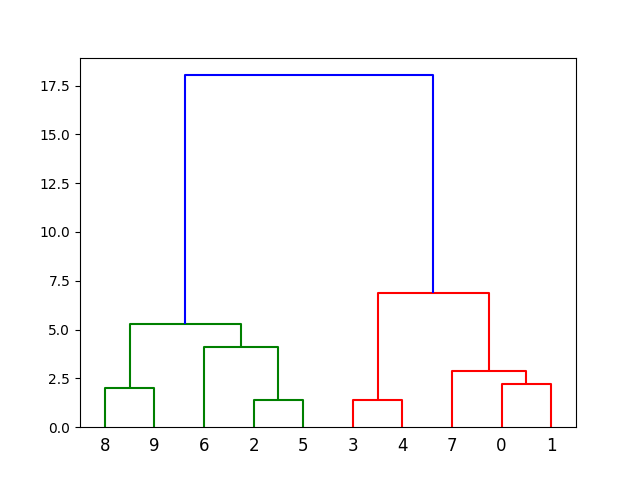
[](https://info.nycdatascience.edu/become-a-data-scientist-in-12-week-nyc-data-science-academy-brand?utm_campaign=w3school&utm_source=W3school&utm_medium=display%20ads&utm_term=PPC&utm_content=Testimonial-Download%20the%20syllabus)

Now we compute the ward linkage using euclidean distance, and visualize it using a dendrogram:

### Example

import numpy as np  
import matplotlib.pyplot as plt  
from scipy.cluster.hierarchy import dendrogram, linkage  
  
x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]  
  
data = list(zip(x, y))  
  
linkage\_data = linkage(data, method='ward', metric='euclidean')  
dendrogram(linkage\_data)  
  
plt.show()

### Result



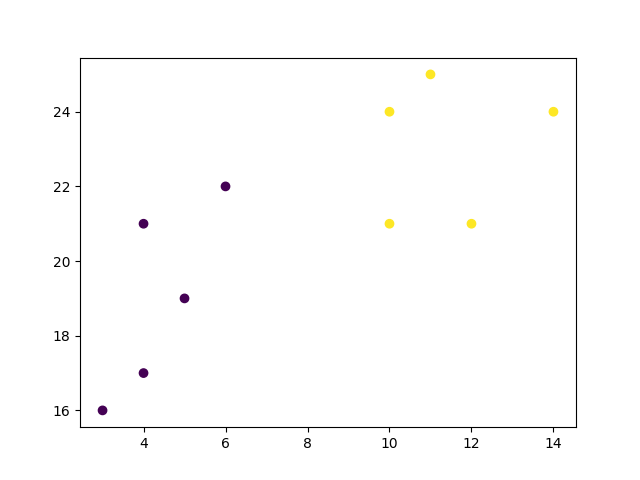
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_hierarchial_clustering2)

Here, we do the same thing with Python's scikit-learn library. Then, visualize on a 2-dimensional plot:

### Example

import numpy as np  
import matplotlib.pyplot as plt  
from sklearn.cluster import AgglomerativeClustering  
  
x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]  
  
data = list(zip(x, y))  
  
hierarchical\_cluster = AgglomerativeClustering(n\_clusters=2, affinity='euclidean', linkage='ward')  
labels = hierarchical\_cluster.fit\_predict(data)  
  
plt.scatter(x, y, c=labels)  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_hierarchial_clustering3)

## Example Explained

Import the modules you need.

import numpy as np  
import matplotlib.pyplot as plt  
from scipy.cluster.hierarchy import dendrogram, linkage  
from sklearn.cluster import AgglomerativeClustering

You can learn about the Matplotlib module in our ["Matplotlib Tutorial](https://www.w3schools.com/python/matplotlib_intro.asp).

You can learn about the SciPy module in our [SciPy Tutorial](https://www.w3schools.com/python/scipy/index.php).

NumPy is a library for working with arrays and matricies in Python, you can learn about the NumPy module in our [NumPy Tutorial](https://www.w3schools.com/python/numpy/default.asp).

scikit-learn is a popular library for machine learning.

Create arrays that resemble two variables in a dataset. Note that while we only two variables here, this method will work with any number of variables:

x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]

Turn the data into a set of points:

data = list(zip(x, y))  
print(data)

Result:

[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24), (6, 22), (10, 21), (12, 21)]

Compute the linkage between all of the different points. Here we use a simple euclidean distance measure and Ward's linkage, which seeks to minimize the variance between clusters.

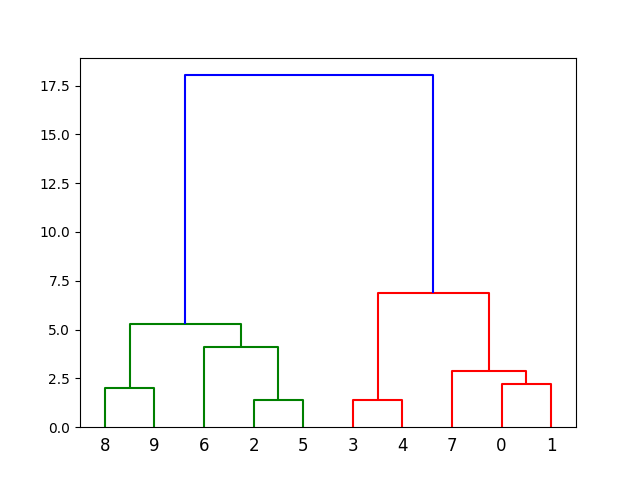
linkage\_data = linkage(data, method='ward', metric='euclidean')

Finally, plot the results in a dendrogram. This plot will show us the hierarchy of clusters from the bottom (individual points) to the top (a single cluster consisting of all data points).

plt.show() lets us visualize the dendrogram instead of just the raw linkage data.

dendrogram(linkage\_data)  
plt.show()

Result:



The scikit-learn library allows us to use hierarchichal clustering in a different manner. First, we initialize the AgglomerativeClustering class with 2 clusters, using the same euclidean distance and Ward linkage.

hierarchical\_cluster = AgglomerativeClustering(n\_clusters=2, affinity='euclidean', linkage='ward')

The .fit\_predict method can be called on our data to compute the clusters using the defined parameters across our chosen number of clusters.

labels = hierarchical\_cluster.fit\_predict(data) print(labels)

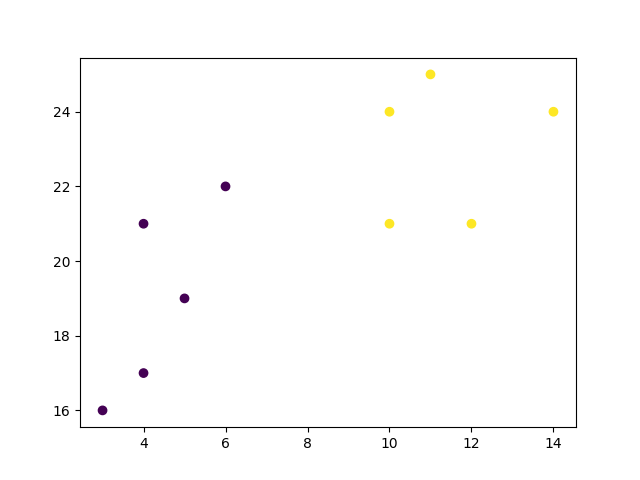
Result:

[0 0 1 0 0 1 1 0 1 1]

Finally, if we plot the same data and color the points using the labels assigned to each index by the hierarchical clustering method, we can see the cluster each point was assigned to:

plt.scatter(x, y, c=labels)  
plt.show()

Result:



# Machine Learning - Logistic Regression

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## Logistic Regression

Logistic regression aims to solve classification problems. It does this by predicting categorical outcomes, unlike linear regression that predicts a continuous outcome.

In the simplest case there are two outcomes, which is called binomial, an example of which is predicting if a tumor is malignant or benign. Other cases have more than two outcomes to classify, in this case it is called multinomial. A common example for multinomial logistic regression would be predicting the class of an iris flower between 3 different species.

Here we will be using basic logistic regression to predict a binomial variable. This means it has only two possible outcomes.

## How does it work?

In Python we have modules that will do the work for us. Start by importing the NumPy module.

import numpy

Store the independent variables in X.

Store the dependent variable in y.

Below is a sample dataset:

#X represents the size of a tumor in centimeters.  
X = numpy.array([3.78, 2.44, 2.09, 0.14, 1.72, 1.65, 4.92, 4.37, 4.96, 4.52, 3.69, 5.88]).reshape(-1,1)  
  
#Note: X has to be reshaped into a column from a row for the LogisticRegression() function to work.  
#y represents whether or not the tumor is cancerous (0 for "No", 1 for "Yes").  
y = numpy.array([0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1])

We will use a method from the sklearn module, so we will have to import that module as well:

from sklearn import linear\_model

From the sklearn module we will use the LogisticRegression() method to create a logistic regression object.

This object has a method called fit() that takes the independent and dependent values as parameters and fills the regression object with data that describes the relationship:

logr = linear\_model.LogisticRegression()  
logr.fit(X,y)

Now we have a logistic regression object that is ready to whether a tumor is cancerous based on the tumor size:

#predict if tumor is cancerous where the size is 3.46mm:  
predicted = logr.predict(numpy.array([3.46]).reshape(-1,1))

### Example

See the whole example in action:

import numpy  
from sklearn import linear\_model  
  
#Reshaped for Logistic function.  
X = numpy.array([3.78, 2.44, 2.09, 0.14, 1.72, 1.65, 4.92, 4.37, 4.96, 4.52, 3.69, 5.88]).reshape(-1,1)  
y = numpy.array([0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1])  
  
logr = linear\_model.LogisticRegression()  
logr.fit(X,y)  
  
#predict if tumor is cancerous where the size is 3.46mm:  
predicted = logr.predict(numpy.array([3.46]).reshape(-1,1))  
print(predicted)

### Result

[0]

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_logistic_regression)

We have predicted that a tumor with a size of 3.46mm will not be cancerous.

ADVERTISEMENT

[](https://community.nycdatascience.edu/courses/intro-to-data-science?utm_campaign=w3school&utm_source=W3School&utm_medium=display%20ads&utm_term=PPC&utm_content=Free%20course)

## Coefficient

In logistic regression the coefficient is the expected change in log-odds of having the outcome per unit change in X.

This does not have the most intuitive understanding so let's use it to create something that makes more sense, odds.

### Example

See the whole example in action:

import numpy  
from sklearn import linear\_model  
  
#Reshaped for Logistic function.  
X = numpy.array([3.78, 2.44, 2.09, 0.14, 1.72, 1.65, 4.92, 4.37, 4.96, 4.52, 3.69, 5.88]).reshape(-1,1)  
y = numpy.array([0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1])  
  
logr = linear\_model.LogisticRegression()  
logr.fit(X,y)  
  
log\_odds = logr.coef\_  
odds = numpy.exp(log\_odds)  
  
print(odds)

### Result

[4.03541657]

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_logistic_regression_odds)

This tells us that as the size of a tumor increases by 1mm the odds of it being a tumor increases by 4x.

## Probability

The coefficient and intercept values can be used to find the probability that each tumor is cancerous.

Create a function that uses the model's coefficient and intercept values to return a new value. This new value represents probability that the given observation is a tumor:

def logit2prob(logr,x):  
  log\_odds = logr.coef\_ \* x + logr.intercept\_  
  odds = numpy.exp(log\_odds)  
  probability = odds / (1 + odds)  
  return(probability)

## Function Explained

To find the log-odds for each observation, we must first create a formula that looks similar to the one from linear regression, extracting the coefficient and the intercept.

log\_odds = logr.coef\_ \* x + logr.intercept\_

To then convert the log-odds to odds we must exponentiate the log-odds.

odds = numpy.exp(log\_odds)

Now that we have the odds, we can convert it to probability by dividing it by 1 plus the odds.

probability = odds / (1 + odds)

Let us now use the function with what we have learned to find out the probability that each tumor is cancerous.

### Example

See the whole example in action:

import numpy  
from sklearn import linear\_model  
  
X = numpy.array([3.78, 2.44, 2.09, 0.14, 1.72, 1.65, 4.92, 4.37, 4.96, 4.52, 3.69, 5.88]).reshape(-1,1)  
y = numpy.array([0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1])  
  
logr = linear\_model.LogisticRegression()  
logr.fit(X,y)  
  
def logit2prob(logr, X):  
  log\_odds = logr.coef\_ \* X + logr.intercept\_  
  odds = numpy.exp(log\_odds)  
  probability = odds / (1 + odds)  
  return(probability)  
  
print(logit2prob(logr, X))

### Result

[[0.60749955]

[0.19268876]

[0.12775886]

[0.00955221]

[0.08038616]

[0.07345637]

[0.88362743]

[0.77901378]

[0.88924409]

[0.81293497]

[0.57719129]

[0.96664243]]

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_logistic_regression2)

## Results Explained

3.78 0.61 The probability that a tumor with the size 3.78cm is cancerous is 61%.

2.44 0.19 The probability that a tumor with the size 2.44cm is cancerous is 19%.

2.09 0.13 The probability that a tumor with the size 2.09cm is cancerous is 13%.

# Machine Learning - Grid Search

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## Grid Search

The majority of machine learning models contain parameters that can be adjusted to vary how the model learns. For example, the logistic regression model, from sklearn, has a parameter C that controls regularization,which affects the complexity of the model.

How do we pick the best value for C? The best value is dependent on the data used to train the model.

## How does it work?

One method is to try out different values and then pick the value that gives the best score. This technique is known as a **grid search**. If we had to select the values for two or more parameters, we would evaluate all combinations of the sets of values thus forming a grid of values.

Before we get into the example it is good to know what the parameter we are changing does. Higher values of C tell the model, the training data resembles real world information, place a greater weight on the training data. While lower values of C do the opposite.

## Using Default Parameters

First let's see what kind of results we can generate without a grid search using only the base parameters.

To get started we must first load in the dataset we will be working with.

from sklearn import datasets  
iris = datasets.load\_iris()

Next in order to create the model we must have a set of independent variables X and a dependant variable y.

X = iris['data']  
y = iris['target']

Now we will load the logistic model for classifying the iris flowers.

from sklearn.linear\_model import LogisticRegression

Creating the model, setting max\_iter to a higher value to ensure that the model finds a result.

Keep in mind the default value for C in a logistic regression model is 1, we will compare this later.

In the example below, we look at the iris data set and try to train a model with varying values for C in logistic regression.

logit = LogisticRegression(max\_iter = 10000)

After we create the model, we must fit the model to the data.

print(logit.fit(X,y))

To evaluate the model we run the score method.

print(logit.score(X,y))

### Example

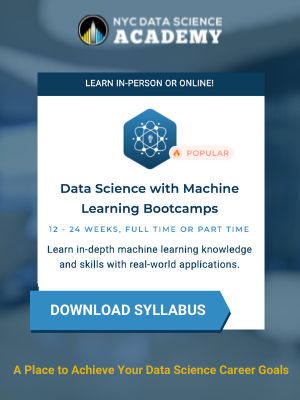
from sklearn import datasets  
from sklearn.linear\_model import LogisticRegression  
  
iris = datasets.load\_iris()  
  
X = iris['data']  
y = iris['target']  
  
logit = LogisticRegression(max\_iter = 10000)  
  
print(logit.fit(X,y))  
  
print(logit.score(X,y))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_grid_search)

With the default setting of C = 1, we achieved a score of 0.973.

Let's see if we can do any better by implementing a grid search with difference values of 0.973.

ADVERTISEMENT

[](https://info.nycdatascience.edu/become-a-data-scientist-in-12-week-nyc-data-science-academy-brand?utm_campaign=w3school&utm_source=W3school&utm_medium=display%20ads&utm_term=PPC&utm_content=Download%20the%20syllabus)

## Implementing Grid Search

We will follow the same steps of before except this time we will set a range of values for C.

Knowing which values to set for the searched parameters will take a combination of domain knowledge and practice.

Since the default value for C is 1, we will set a range of values surrounding it.

C = [0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2]

Next we will create a for loop to change out the values of C and evaluate the model with each change.

First we will create an empty list to store the score within.

scores = []

To change the values of C we must loop over the range of values and update the parameter each time.

for choice in C:  
  logit.set\_params(C=choice)  
  logit.fit(X, y)  
  scores.append(logit.score(X, y))

With the scores stored in a list, we can evaluate what the best choice of C is.

print(scores)

### Example

from sklearn import datasets  
from sklearn.linear\_model import LogisticRegression  
  
iris = datasets.load\_iris()  
  
X = iris['data']  
y = iris['target']  
  
logit = LogisticRegression(max\_iter = 10000)  
  
C = [0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2]  
  
scores = []  
  
for choice in C:  
  logit.set\_params(C=choice)  
  logit.fit(X, y)  
  scores.append(logit.score(X, y))  
  
print(scores)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_grid_search2)

## Results Explained

We can see that the lower values of C performed worse than the base parameter of 1. However, as we increased the value of C to 1.75 the model experienced increased accuracy.

It seems that increasing C beyond this amount does not help increase model accuracy.

## Note on Best Practices

We scored our logistic regression model by using the same data that was used to train it. If the model corresponds too closely to that data, it may not be great at predicting unseen data. This statistical error is known as **over fitting**.

To avoid being misled by the scores on the training data, we can put aside a portion of our data and use it specifically for the purpose of testing the model. Refer to the lecture on train/test splitting to avoid being misled and overfitting.

# Preprocessing - Categorical Data

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## Categorical Data

When your data has categories represented by strings, it will be difficult to use them to train machine learning models which often only accepts numeric data.

Instead of ignoring the categorical data and excluding the information from our model, you can tranform the data so it can be used in your models.

Take a look at the table below, it is the same data set that we used in the [multiple regression](https://www.w3schools.com/python/python_ml_multiple_regression.asp) chapter.

### Example

import pandas as pd  
  
cars = pd.read\_csv('data.csv')  
print(cars.to\_string())

### Result

Car Model Volume Weight CO2

0 Toyoty Aygo 1000 790 99

1 Mitsubishi Space Star 1200 1160 95

2 Skoda Citigo 1000 929 95

3 Fiat 500 900 865 90

4 Mini Cooper 1500 1140 105

5 VW Up! 1000 929 105

6 Skoda Fabia 1400 1109 90

7 Mercedes A-Class 1500 1365 92

8 Ford Fiesta 1500 1112 98

9 Audi A1 1600 1150 99

10 Hyundai I20 1100 980 99

11 Suzuki Swift 1300 990 101

12 Ford Fiesta 1000 1112 99

13 Honda Civic 1600 1252 94

14 Hundai I30 1600 1326 97

15 Opel Astra 1600 1330 97

16 BMW 1 1600 1365 99

17 Mazda 3 2200 1280 104

18 Skoda Rapid 1600 1119 104

19 Ford Focus 2000 1328 105

20 Ford Mondeo 1600 1584 94

21 Opel Insignia 2000 1428 99

22 Mercedes C-Class 2100 1365 99

23 Skoda Octavia 1600 1415 99

24 Volvo S60 2000 1415 99

25 Mercedes CLA 1500 1465 102

26 Audi A4 2000 1490 104

27 Audi A6 2000 1725 114

28 Volvo V70 1600 1523 109

29 BMW 5 2000 1705 114

30 Mercedes E-Class 2100 1605 115

31 Volvo XC70 2000 1746 117

32 Ford B-Max 1600 1235 104

33 BMW 216 1600 1390 108

34 Opel Zafira 1600 1405 109

35 Mercedes SLK 2500 1395 120

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_preprocessing)

In the multiple regression chapter, we tried to predict the CO2 emitted based on the volume of the engine and the weight of the car but we excluded information about the car brand and model.

The information about the car brand or the car model might help us make a better prediction of the CO2 emitted.

ADVERTISEMENT

[](https://info.nycdatascience.edu/become-a-data-scientist-in-12-week-nyc-data-science-academy-brand?utm_campaign=w3school&utm_source=W3school&utm_medium=display%20ads&utm_term=PPC&utm_content=Testimonial-Download%20the%20syllabus)

## One Hot Encoding

We cannot make use of the Car or Model column in our data since they are not numeric. A linear relationship between a categorical variable, Car or Model, and a numeric variable, CO2, cannot be determined.

To fix this issue, we must have a numeric representation of the categorical variable. One way to do this is to have a column representing each group in the category.

For each column, the values will be 1 or 0 where 1 represents the inclusion of the group and 0 represents the exclusion. This transformation is called one hot encoding.

You do not have to do this manually, the Python Pandas module has a function that called get\_dummies() which does one hot encoding.

Learn about the Pandas module in our [Pandas Tutorial](https://www.w3schools.com/python/pandas/default.asp).

### Example

One Hot Encode the Car column:

import pandas as pd  
  
cars = pd.read\_csv('data.csv')  
ohe\_cars = pd.get\_dummies(cars[['Car']])  
  
print(ohe\_cars.to\_string())

### Result

Car\_Audi Car\_BMW Car\_Fiat Car\_Ford Car\_Honda Car\_Hundai Car\_Hyundai Car\_Mazda Car\_Mercedes Car\_Mini Car\_Mitsubishi Car\_Opel Car\_Skoda Car\_Suzuki Car\_Toyoty Car\_VW Car\_Volvo

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0

1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0

2 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

3 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

4 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0

5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0

6 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

7 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

8 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

9 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

10 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0

11 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0

12 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

13 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0

14 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0

15 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0

16 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

17 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0

18 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

19 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

20 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

21 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0

22 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

23 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

24 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

25 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

26 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

27 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

28 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

29 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

30 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

31 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

32 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

33 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

34 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0

35 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_preprocessing2)

## Results

A column was created for every car brand in the Car column.

## Predict CO2

We can use this additional information alongside the volume and weight to predict CO2

To combine the information, we can use the concat() function from pandas.

First we will need to import a couple modules.

We will start with importing the Pandas.

import pandas

The pandas module allows us to read csv files and manipulate DataFrame objects:

cars = pandas.read\_csv("data.csv")

It also allows us to create the dummy variables:

ohe\_cars = pandas.get\_dummies(cars[['Car']])

Then we must select the independent variables (X) and add the dummy variables columnwise.

Also store the dependent variable in y.

X = pandas.concat([cars[['Volume', 'Weight']], ohe\_cars], axis=1)  
y = cars['CO2']

We also need to import a method from sklearn to create a linear model

Learn about [linear regression](https://www.w3schools.com/python/python_ml_linear_regression.asp).

from sklearn import linear\_model

Now we can fit the data to a linear regression:

regr = linear\_model.LinearRegression()  
regr.fit(X,y)

Finally we can predict the CO2 emissions based on the car's weight, volume, and manufacturer.

##predict the CO2 emission of a Volvo where the weight is 2300kg, and the volume is 1300cm3:  
predictedCO2 = regr.predict([[2300, 1300,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0]])

### Example

import pandas  
from sklearn import linear\_model  
  
cars = pandas.read\_csv("data.csv")  
ohe\_cars = pandas.get\_dummies(cars[['Car']])  
  
X = pandas.concat([cars[['Volume', 'Weight']], ohe\_cars], axis=1)  
y = cars['CO2']  
  
regr = linear\_model.LinearRegression()  
regr.fit(X,y)  
  
##predict the CO2 emission of a Volvo where the weight is 2300kg, and the volume is 1300cm3:  
predictedCO2 = regr.predict([[2300, 1300,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0]])  
  
print(predictedCO2)

### Result

[122.45153299]

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_preprocessing5)

We now have a coefficient for the volume, the weight, and each car brand in the data set

## Dummifying

It is not necessary to create one column for each group in your category. The information can be retained using 1 column less than the number of groups you have.

For example, you have a column representing colors and in that column, you have two colors, red and blue.

### Example

import pandas as pd  
  
colors = pd.DataFrame({'color': ['blue', 'red']})  
  
print(colors)

### Result

color

0 blue

1 red

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_preprocessing6)

You can create 1 column called red where 1 represents red and 0 represents not red, which means it is blue.

To do this, we can use the same function that we used for one hot encoding, get\_dummies, and then drop one of the columns. There is an argument, drop\_first, which allows us to exclude the first column from the resulting table.

### Example

import pandas as pd  
  
colors = pd.DataFrame({'color': ['blue', 'red']})  
dummies = pd.get\_dummies(colors, drop\_first=True)  
  
print(dummies)

### Result

color\_red

0 0

1 1

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_preprocessing7)

What if you have more than 2 groups? How can the multiple groups be represented by 1 less column?

Let's say we have three colors this time, red, blue and green. When we get\_dummies while dropping the first column, we get the following table.

### Example

import pandas as pd  
  
colors = pd.DataFrame({'color': ['blue', 'red', 'green']})  
dummies = pd.get\_dummies(colors, drop\_first=True)  
dummies['color'] = colors['color']  
  
print(dummies)

### Result

color\_green color\_red color

0 0 0 blue

1 0 1 red

2 1 0 green

[Run example »](https://www.w3schools.com/python/trypandas.asp?filename=demo_ml_preprocessing8)

# Machine Learning - K-means

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## K-means

K-means is an unsupervised learning method for clustering data points. The algorithm iteratively divides data points into K clusters by minimizing the variance in each cluster.

Here, we will show you how to estimate the best value for K using the elbow method, then use K-means clustering to group the data points into clusters.

## How does it work?

First, each data point is randomly assigned to one of the K clusters. Then, we compute the centroid (functionally the center) of each cluster, and reassign each data point to the cluster with the closest centroid. We repeat this process until the cluster assignments for each data point are no longer changing.

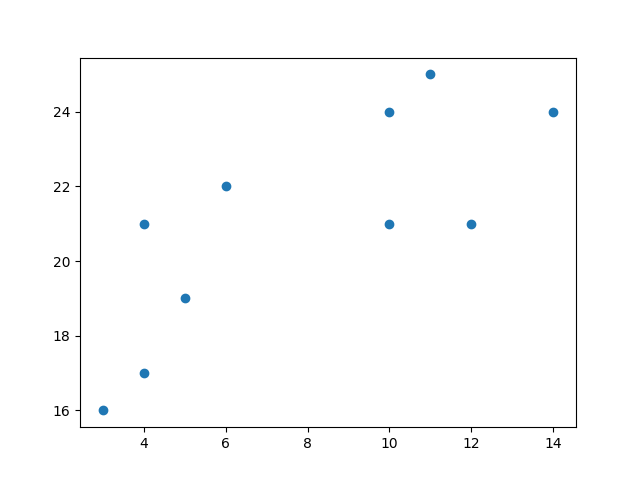
K-means clustering requires us to select K, the number of clusters we want to group the data into. The elbow method lets us graph the inertia (a distance-based metric) and visualize the point at which it starts decreasing linearly. This point is referred to as the "eblow" and is a good estimate for the best value for K based on our data.

### Example

Start by visualizing some data points:

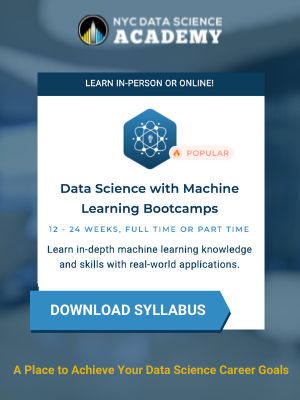
import matplotlib.pyplot as plt  
  
x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]  
  
plt.scatter(x, y)  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_k-means)

ADVERTISEMENT

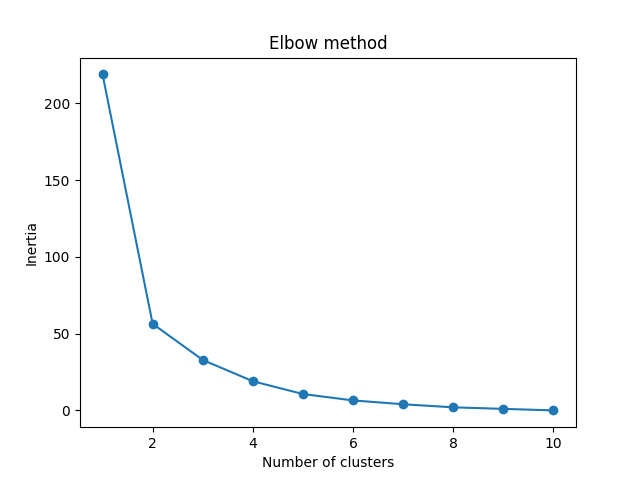
[](https://info.nycdatascience.edu/become-a-data-scientist-in-12-week-nyc-data-science-academy-brand?utm_campaign=w3school&utm_source=W3school&utm_medium=display%20ads&utm_term=PPC&utm_content=Download%20the%20syllabus)

Now we utilize the elbow method to visualize the intertia for different values of K:

### Example

from sklearn.cluster import KMeans  
  
data = list(zip(x, y))  
inertias = []  
  
for i in range(1,11):  
    kmeans = KMeans(n\_clusters=i)  
    kmeans.fit(data)  
    inertias.append(kmeans.inertia\_)  
  
plt.plot(range(1,11), inertias, marker='o')  
plt.title('Elbow method')  
plt.xlabel('Number of clusters')  
plt.ylabel('Inertia')  
plt.show()

### Result



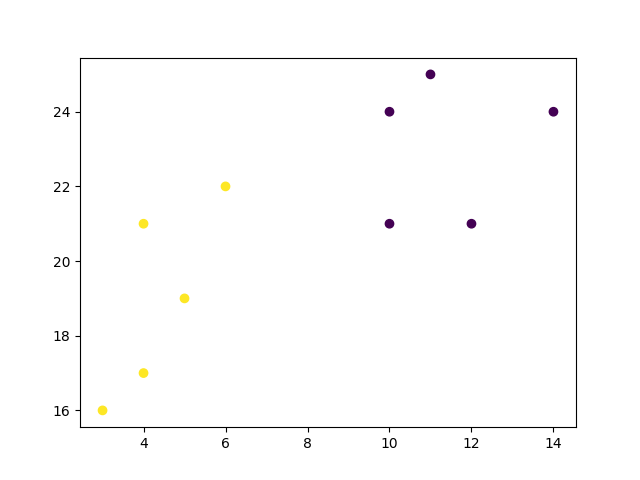
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_k-means2)

The elbow method shows that 2 is a good value for K, so we retrain and visualize the result:

### Example

kmeans = KMeans(n\_clusters=2)  
kmeans.fit(data)  
  
plt.scatter(x, y, c=kmeans.labels\_)  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_k-means3)

## Example Explained

Import the modules you need.

import matplotlib.pyplot as plt  
from sklearn.cluster import KMeans

You can learn about the Matplotlib module in our ["Matplotlib Tutorial](https://www.w3schools.com/python/matplotlib_intro.asp).

scikit-learn is a popular library for machine learning.

Create arrays that resemble two variables in a dataset. Note that while we only use two variables here, this method will work with any number of variables:

x = [4, 5, 10, 4, 3, 11, 14 , 6, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]

Turn the data into a set of points:

data = list(zip(x, y))  
print(data)

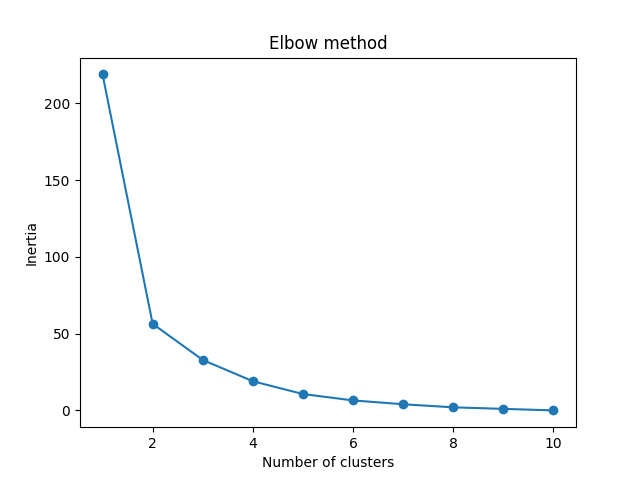
Result:

[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24), (6, 22), (10, 21), (12, 21)]

In order to find the best value for K, we need to run K-means across our data for a range of possible values. We only have 10 data points, so the maximum number of clusters is 10. So for each value K in range(1,11), we train a K-means model and plot the intertia at that number of clusters:

inertias = []  
  
for i in range(1,11):  
    kmeans = KMeans(n\_clusters=i)  
    kmeans.fit(data)  
    inertias.append(kmeans.inertia\_)  
  
plt.plot(range(1,11), inertias, marker='o')  
plt.title('Elbow method')  
plt.xlabel('Number of clusters')  
plt.ylabel('Inertia')  
plt.show()

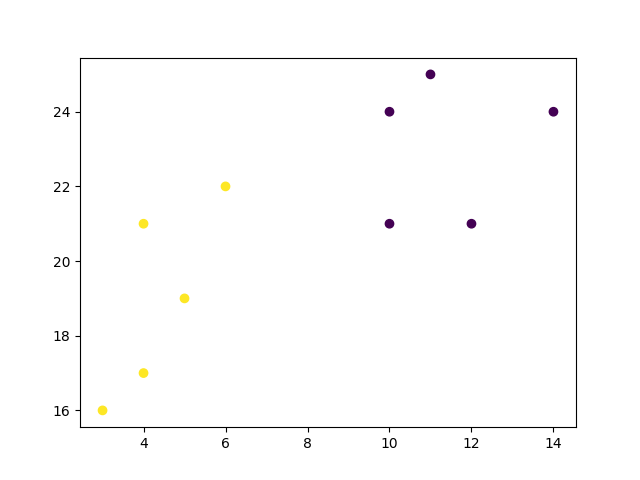
Result:



We can see that the "elbow" on the graph above (where the interia becomes more linear) is at K=2. We can then fit our K-means algorithm one more time and plot the different clusters assigned to the data:

kmeans = KMeans(n\_clusters=2)  
kmeans.fit(data)  
  
plt.scatter(x, y, c=kmeans.labels\_)  
plt.show()

Result:



# Machine Learning - Bootstrap Aggregation (Bagging)

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## Bagging

Methods such as Decision Trees, can be prone to overfitting on the training set which can lead to wrong predictions on new data.

Bootstrap Aggregation (bagging) is a ensembling method that attempts to resolve overfitting for classification or regression problems. Bagging aims to improve the accuracy and performance of machine learning algorithms. It does this by taking random subsets of an original dataset, with replacement, and fits either a classifier (for classification) or regressor (for regression) to each subset. The predictions for each subset are then aggregated through majority vote for classification or averaging for regression, increasing prediction accuracy.

## Evaluating a Base Classifier

To see how bagging can improve model performance, we must start by evaluating how the base classifier performs on the dataset. If you do not know what decision trees are review the lesson on decision trees before moving forward, as bagging is an continuation of the concept.

We will be looking to identify different classes of wines found in Sklearn's wine dataset.

Let's start by importing the necessary modules.

from sklearn import datasets  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import accuracy\_score  
from sklearn.tree import DecisionTreeClassifier

Next we need to load in the data and store it into X (input features) and y (target). The parameter as\_frame is set equal to True so we do not lose the feature names when loading the data. (sklearn version older than 0.23 must skip the as\_frame argument as it is not supported)

data = datasets.load\_wine(as\_frame = True)  
  
X = data.data  
y = data.target

In order to properly evaluate our model on unseen data, we need to split X and y into train and test sets. For information on splitting data, see the Train/Test lesson.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 22)

With our data prepared, we can now instantiate a base classifier and fit it to the training data.

dtree = DecisionTreeClassifier(random\_state = 22)  
dtree.fit(X\_train,y\_train)

Result:

DecisionTreeClassifier(random\_state=22)

We can now predict the class of wine the unseen test set and evaluate the model performance.

y\_pred = dtree.predict(X\_test)  
  
print("Train data accuracy:",accuracy\_score(y\_true = y\_train, y\_pred = dtree.predict(X\_train)))  
print("Test data accuracy:",accuracy\_score(y\_true = y\_test, y\_pred = y\_pred))

Result:

Train data accuracy: 1.0  
Test data accuracy: 0.8222222222222222

### Example

Import the necessary data and evaluate base classifier performance.

from sklearn import datasets  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import accuracy\_score  
from sklearn.tree import DecisionTreeClassifier  
  
data = datasets.load\_wine(as\_frame = True)  
  
X = data.data  
y = data.target  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 22)  
  
dtree = DecisionTreeClassifier(random\_state = 22)  
dtree.fit(X\_train,y\_train)  
  
y\_pred = dtree.predict(X\_test)  
  
print("Train data accuracy:",accuracy\_score(y\_true = y\_train, y\_pred = dtree.predict(X\_train)))  
print("Test data accuracy:",accuracy\_score(y\_true = y\_test, y\_pred = y\_pred))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_bagging)

The base classifier performs reasonably well on the dataset achieving 82% accuracy on the test dataset with the current parameters (Different results may occur if you do not have the random\_state parameter set).

Now that we have a baseline accuracy for the test dataset, we can see how the Bagging Classifier out performs a single Decision Tree Classifier.

ADVERTISEMENT

[](https://info.nycdatascience.edu/become-a-data-scientist-in-12-week-nyc-data-science-academy-brand?utm_campaign=w3school&utm_source=W3school&utm_medium=display%20ads&utm_term=PPC&utm_content=Testimonial-Download%20the%20syllabus)

## Creating a Bagging Classifier

For bagging we need to set the parameter n\_estimators, this is the number of base classifiers that our model is going to aggregate together.

For this sample dataset the number of estimators is relatively low, it is often the case that much larger ranges are explored. Hyperparameter tuning is usually done with a [grid search](https://www.w3schools.com/python/python_ml_grid_search.asp), but for now we will use a select set of values for the number of estimators.

We start by importing the necessary model.

from sklearn.ensemble import BaggingClassifier

Now lets create a range of values that represent the number of estimators we want to use in each ensemble.

estimator\_range = [2,4,6,8,10,12,14,16]

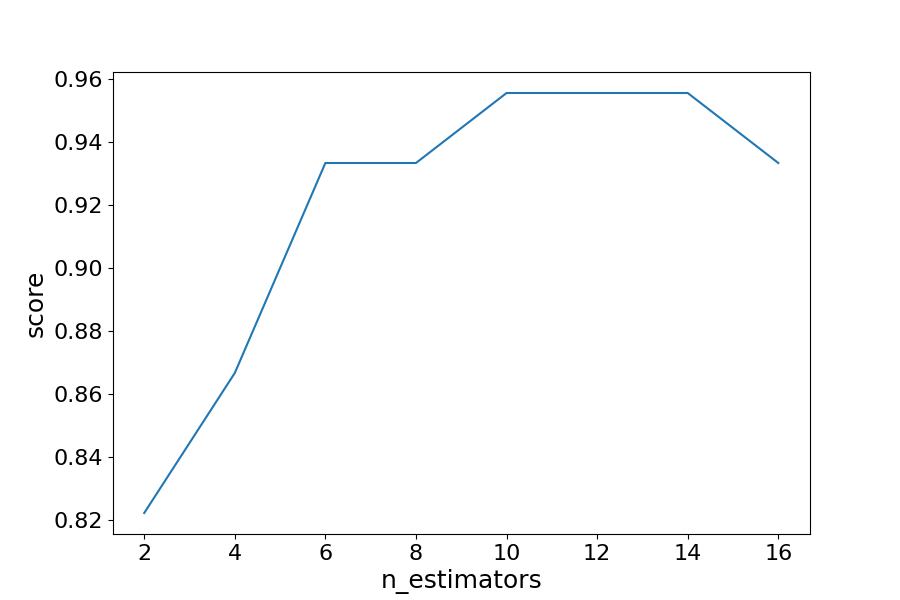
To see how the Bagging Classifier performs with differing values of n\_estimators we need a way to iterate over the range of values and store the results from each ensemble. To do this we will create a for loop, storing the models and scores in separate lists for later vizualizations.

Note: The default parameter for the base classifier in BaggingClassifier is the DicisionTreeClassifier therefore we do not need to set it when instantiating the bagging model.

models = []  
scores = []  
  
for n\_estimators in estimator\_range:  
  
    # Create bagging classifier  
    clf = BaggingClassifier(n\_estimators = n\_estimators, random\_state = 22)  
  
    # Fit the model  
    clf.fit(X\_train, y\_train)  
  
    # Append the model and score to their respective list  
    models.append(clf)  
    scores.append(accuracy\_score(y\_true = y\_test, y\_pred = clf.predict(X\_test)))

With the models and scores stored, we can now visualize the improvement in model performance.

import matplotlib.pyplot as plt  
  
# Generate the plot of scores against number of estimators  
plt.figure(figsize=(9,6))  
plt.plot(estimator\_range, scores)  
  
# Adjust labels and font (to make visable)  
plt.xlabel("n\_estimators", fontsize = 18)  
plt.ylabel("score", fontsize = 18)  
plt.tick\_params(labelsize = 16)  
  
# Visualize plot  
plt.show()

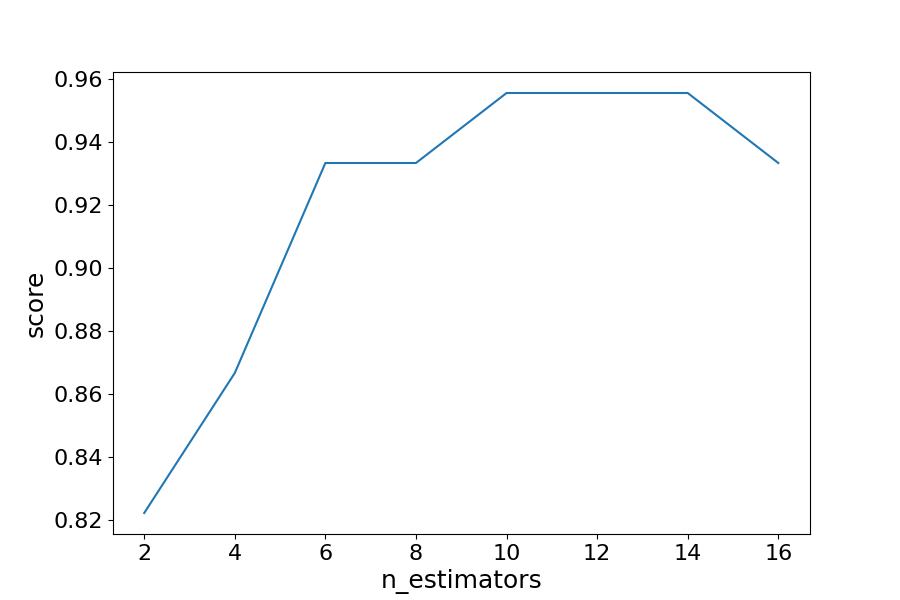


### Example

Import the necessary data and evaluate the BaggingClassifier performance.

import matplotlib.pyplot as plt  
from sklearn import datasets  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import accuracy\_score  
from sklearn.ensemble import BaggingClassifier  
  
data = datasets.load\_wine(as\_frame = True)  
  
X = data.data  
y = data.target  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 22)  
  
estimator\_range = [2,4,6,8,10,12,14,16]  
  
models = []  
scores = []  
  
for n\_estimators in estimator\_range:  
  
    # Create bagging classifier  
    clf = BaggingClassifier(n\_estimators = n\_estimators, random\_state = 22)  
  
    # Fit the model  
    clf.fit(X\_train, y\_train)  
  
    # Append the model and score to their respective list  
    models.append(clf)  
    scores.append(accuracy\_score(y\_true = y\_test, y\_pred = clf.predict(X\_test)))  
  
# Generate the plot of scores against number of estimators  
plt.figure(figsize=(9,6))  
plt.plot(estimator\_range, scores)  
  
# Adjust labels and font (to make visable)  
plt.xlabel("n\_estimators", fontsize = 18)  
plt.ylabel("score", fontsize = 18)  
plt.tick\_params(labelsize = 16)  
  
# Visualize plot  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_bagging2)

## Results Explained

By iterating through different values for the number of estimators we can see an increase in model performance from 82.2% to 95.5%. After 14 estimators the accuracy begins to drop, again if you set a different random\_state the values you see will vary. That is why it is best practice to use [cross validation](https://www.w3schools.com/python/python_ml_cross_validation.asp) to ensure stable results.

In this case, we see a 13.3% increase in accuracy when it comes to identifying the type of the wine.

## Another Form of Evaluation

As bootstrapping chooses random subsets of observations to create classifiers, there are observations that are left out in the selection process. These "out-of-bag" observations can then be used to evaluate the model, similarly to that of a test set. Keep in mind, that out-of-bag estimation can overestimate error in binary classification problems and should only be used as a compliment to other metrics.

We saw in the last exercise that 12 estimators yielded the highest accuracy, so we will use that to create our model. This time setting the parameter oob\_score to true to evaluate the model with out-of-bag score.

### Example

Create a model with out-of-bag metric.

from sklearn import datasets  
from sklearn.model\_selection import train\_test\_split  
from sklearn.ensemble import BaggingClassifier  
  
data = datasets.load\_wine(as\_frame = True)  
  
X = data.data  
y = data.target  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 22)  
  
oob\_model = BaggingClassifier(n\_estimators = 12, oob\_score = True,random\_state = 22)  
  
oob\_model.fit(X\_train, y\_train)  
  
print(oob\_model.oob\_score\_)

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_bagging3)

Since the samples used in OOB and the test set are different, and the dataset is relatively small, there is a difference in the accuracy. It is rare that they would be exactly the same, again OOB should be used quick means for estimating error, but is not the only evaluation metric.

## Generating Decision Trees from Bagging Classifier

As was seen in the [Decision Tree](https://www.w3schools.com/python/python_ml_decision_tree.asp) lesson, it is possible to graph the decision tree the model created. It is also possible to see the individual decision trees that went into the aggregated classifier. This helps us to gain a more intuitive understanding on how the bagging model arrives at its predictions.

Note: This is only functional with smaller datasets, where the trees are relatively shallow and narrow making them easy to visualize.

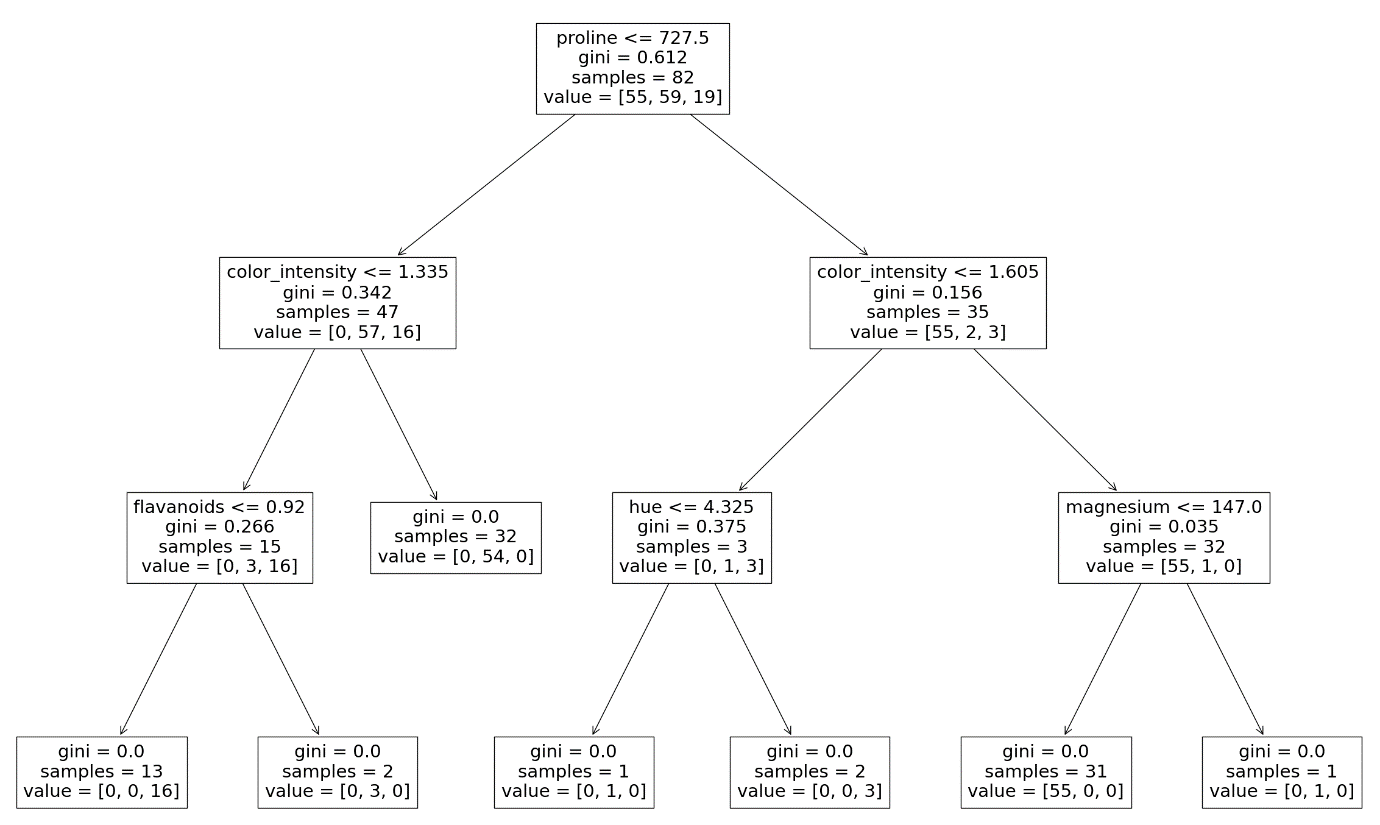
We will need to import plot\_tree function from sklearn.tree. The different trees can be graphed by changing the estimator you wish to visualize.

### Example

Generate Decision Trees from Bagging Classifier

from sklearn import datasets  
from sklearn.model\_selection import train\_test\_split  
from sklearn.ensemble import BaggingClassifier  
from sklearn.tree import plot\_tree  
  
X = data.data  
y = data.target  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 22)  
  
clf = BaggingClassifier(n\_estimators = 12, oob\_score = True,random\_state = 22)  
  
clf.fit(X\_train, y\_train)  
  
plt.figure(figsize=(30, 20))  
  
plot\_tree(clf.estimators\_[0], feature\_names = X.columns)

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_bagging4)

Here we can see just the first decision tree that was used to vote on the final prediction. Again, by changing the index of the classifier you can see each of the trees that have been aggregated.

# Machine Learning - Cross Validation

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## Cross Validation

When adjusting models we are aiming to increase overall model performance on unseen data. Hyperparameter tuning can lead to much better performance on test sets. However, optimizing parameters to the test set can lead information leakage causing the model to preform worse on unseen data. To correct for this we can perform cross validation.

To better understand CV, we will be performing different methods on the iris dataset. Let us first load in and separate the data.

from sklearn import datasets  
  
X, y = datasets.load\_iris(return\_X\_y=True)

There are many methods to cross validation, we will start by looking at k-fold cross validation.

## K-Fold

The training data used in the model is split, into k number of smaller sets, to be used to validate the model. The model is then trained on k-1 folds of training set. The remaining fold is then used as a validation set to evaluate the model.

As we will be trying to classify different species of iris flowers we will need to import a classifier model, for this exercise we will be using a DecisionTreeClassifier. We will also need to import CV modules from sklearn.

from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import KFold, cross\_val\_score

With the data loaded we can now create and fit a model for evaluation.

clf = DecisionTreeClassifier(random\_state=42)

Now let's evaluate our model and see how it performs on each k-fold.

k\_folds = KFold(n\_splits = 5)  
  
scores = cross\_val\_score(clf, X, y, cv = k\_folds)

It is also good pratice to see how CV performed overall by averaging the scores for all folds.

### Example

Run k-fold CV:

from sklearn import datasets  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import KFold, cross\_val\_score  
  
X, y = datasets.load\_iris(return\_X\_y=True)  
  
clf = DecisionTreeClassifier(random\_state=42)  
  
k\_folds = KFold(n\_splits = 5)  
  
scores = cross\_val\_score(clf, X, y, cv = k\_folds)  
  
print("Cross Validation Scores: ", scores)  
print("Average CV Score: ", scores.mean())  
print("Number of CV Scores used in Average: ", len(scores))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_k-fold)

ADVERTISEMENT

[](https://community.nycdatascience.edu/courses/intro-to-data-science?utm_campaign=w3school&utm_source=W3School&utm_medium=display%20ads&utm_term=PPC&utm_content=Free%20course)

## Stratified K-Fold

In cases where classes are imbalanced we need a way to account for the imbalance in both the train and validation sets. To do so we can stratify the target classes, meaning that both sets will have an equal proportion of all classes.

### Example

from sklearn import datasets  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import StratifiedKFold, cross\_val\_score  
  
X, y = datasets.load\_iris(return\_X\_y=True)  
  
clf = DecisionTreeClassifier(random\_state=42)  
  
sk\_folds = StratifiedKFold(n\_splits = 5)  
  
scores = cross\_val\_score(clf, X, y, cv = sk\_folds)  
  
print("Cross Validation Scores: ", scores)  
print("Average CV Score: ", scores.mean())  
print("Number of CV Scores used in Average: ", len(scores))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_stratified_k-fold)

While the number of folds is the same, the average CV increases from the basic k-fold when making sure there is stratified classes.

## Leave-One-Out (LOO)

Instead of selecting the number of splits in the training data set like k-fold LeaveOneOut, utilize 1 observation to validate and n-1 observations to train. This method is an exaustive technique.

### Example

Run LOO CV:

from sklearn import datasets  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import LeaveOneOut, cross\_val\_score  
  
X, y = datasets.load\_iris(return\_X\_y=True)  
  
clf = DecisionTreeClassifier(random\_state=42)  
  
loo = LeaveOneOut()  
  
scores = cross\_val\_score(clf, X, y, cv = loo)  
  
print("Cross Validation Scores: ", scores)  
print("Average CV Score: ", scores.mean())  
print("Number of CV Scores used in Average: ", len(scores))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_loo)

We can observe that the number of cross validation scores performed is equal to the number of observations in the dataset. In this case there are 150 observations in the iris dataset.

The average CV score is 94%.

## Leave-P-Out (LPO)

Leave-P-Out is simply a nuanced diffence to the Leave-One-Out idea, in that we can select the number of p to use in our validation set.

### Example

Run LPO CV:

from sklearn import datasets  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import LeavePOut, cross\_val\_score  
  
X, y = datasets.load\_iris(return\_X\_y=True)  
  
clf = DecisionTreeClassifier(random\_state=42)  
  
lpo = LeavePOut(p=2)  
  
scores = cross\_val\_score(clf, X, y, cv = lpo)  
  
print("Cross Validation Scores: ", scores)  
print("Average CV Score: ", scores.mean())  
print("Number of CV Scores used in Average: ", len(scores))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_lpo)

As we can see this is an exhaustive method we many more scores being calculated than Leave-One-Out, even with a p = 2, yet it achieves roughly the same average CV score.

## Shuffle Split

Unlike KFold, ShuffleSplit leaves out a percentage of the data, not to be used in the train or validation sets. To do so we must decide what the train and test sizes are, as well as the number of splits.

### Example

Run Shuffle Split CV:

from sklearn import datasets  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import ShuffleSplit, cross\_val\_score  
  
X, y = datasets.load\_iris(return\_X\_y=True)  
  
clf = DecisionTreeClassifier(random\_state=42)  
  
ss = ShuffleSplit(train\_size=0.6, test\_size=0.3, n\_splits = 5)  
  
scores = cross\_val\_score(clf, X, y, cv = ss)  
  
print("Cross Validation Scores: ", scores)  
print("Average CV Score: ", scores.mean())  
print("Number of CV Scores used in Average: ", len(scores))

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_shuffle_split)

## Ending Notes

These are just a few of the CV methods that can be applied to models. There are many more cross validation classes, with most models having their own class. Check out sklearns cross validation for more CV options.

# Machine Learning - AUC - ROC Curve

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## AUC - ROC Curve

In classification, there are many different evaluation metrics. The most popular is **accuracy**, which measures how often the model is correct. This is a great metric because it is easy to understand and getting the most correct guesses is often desired. There are some cases where you might consider using another evaluation metric.

Another common metric is **AUC**, area under the receiver operating characteristic (**ROC**) curve. The Reciever operating characteristic curve plots the true positive (**TP**) rate versus the false positive (**FP**) rate at different classification thresholds. The thresholds are different probability cutoffs that separate the two classes in binary classification. It uses probability to tell us how well a model separates the classes.

## Imbalanced Data

Suppose we have an imbalanced data set where the majority of our data is of one value. We can obtain high accuracy for the model by predicting the majority class.

### Example

import numpy as np  
from sklearn.metrics import accuracy\_score, confusion\_matrix, roc\_auc\_score, roc\_curve  
  
n = 10000  
ratio = .95  
n\_0 = int((1-ratio) \* n)  
n\_1 = int(ratio \* n)  
  
y = np.array([0] \* n\_0 + [1] \* n\_1)  
# below are the probabilities obtained from a hypothetical model that always predicts the majority class  
# probability of predicting class 1 is going to be 100%  
y\_proba = np.array([1]\*n)  
y\_pred = y\_proba > .5  
  
print(f'accuracy score: {accuracy\_score(y, y\_pred)}')  
cf\_mat = confusion\_matrix(y, y\_pred)  
print('Confusion matrix')  
print(cf\_mat)  
print(f'class 0 accuracy: {cf\_mat[0][0]/n\_0}')  
print(f'class 1 accuracy: {cf\_mat[1][1]/n\_1}')

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_auc_roc1)

ADVERTISEMENT

[](https://info.nycdatascience.edu/become-a-data-scientist-in-12-week-nyc-data-science-academy-brand?utm_campaign=w3school&utm_source=W3school&utm_medium=display%20ads&utm_term=PPC&utm_content=Testimonial-Download%20the%20syllabus)

Although we obtain a very high accuracy, the model provided no information about the data so it's not useful. We accurately predict class 1 100% of the time while inaccurately predict class 0 0% of the time. At the expense of accuracy, it might be better to have a model that can somewhat separate the two classes.

### Example

# below are the probabilities obtained from a hypothetical model that doesn't always predict the mode  
y\_proba\_2 = np.array(  
    np.random.uniform(0, .7, n\_0).tolist() +  
    np.random.uniform(.3, 1, n\_1).tolist()  
)  
y\_pred\_2 = y\_proba\_2 > .5  
  
print(f'accuracy score: {accuracy\_score(y, y\_pred\_2)}')  
cf\_mat = confusion\_matrix(y, y\_pred\_2)  
print('Confusion matrix')  
print(cf\_mat)  
print(f'class 0 accuracy: {cf\_mat[0][0]/n\_0}')  
print(f'class 1 accuracy: {cf\_mat[1][1]/n\_1}')

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_auc_roc2)

For the second set of predictions, we do not have as high of an accuracy score as the first but the accuracy for each class is more balanced. Using accuracy as an evaluation metric we would rate the first model higher than the second even though it doesn't tell us anything about the data.

In cases like this, using another evaluation metric like AUC would be preferred.

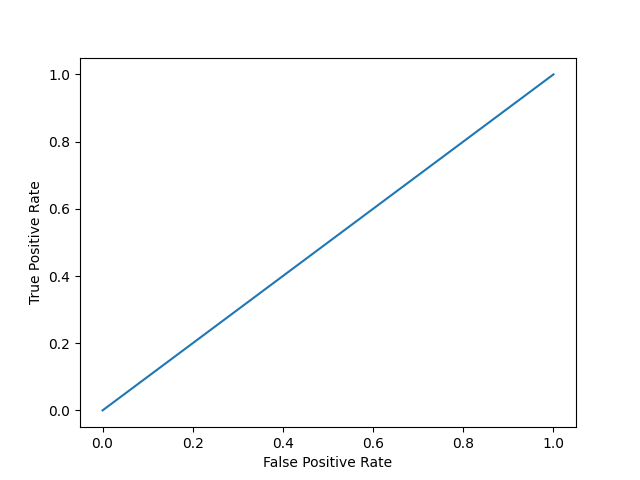
import matplotlib.pyplot as plt  
  
def plot\_roc\_curve(true\_y, y\_prob):  
    """  
    plots the roc curve based of the probabilities  
    """  
  
    fpr, tpr, thresholds = roc\_curve(true\_y, y\_prob)  
    plt.plot(fpr, tpr)  
    plt.xlabel('False Positive Rate')  
    plt.ylabel('True Positive Rate')

### Example

Model 1:

plot\_roc\_curve(y, y\_proba)  
print(f'model 1 AUC score: {roc\_auc\_score(y, y\_proba)}')

### Result

  
model 1 AUC score: 0.5

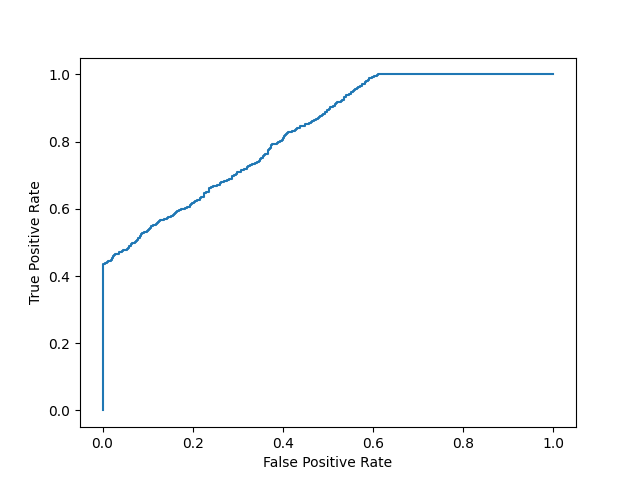
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_auc_roc3)

### Example

Model 2:

plot\_roc\_curve(y, y\_proba\_2)  
print(f'model 2 AUC score: {roc\_auc\_score(y, y\_proba\_2)}')

### Result

  
model 2 AUC score: 0.8270551578947367

[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_auc_roc4)

An AUC score of around .5 would mean that the model is unable to make a distinction between the two classes and the curve would look like a line with a slope of 1. An AUC score closer to 1 means that the model has the ability to separate the two classes and the curve would come closer to the top left corner of the graph.

## Probabilities

Because AUC is a metric that utilizes probabilities of the class predictions, we can be more confident in a model that has a higher AUC score than one with a lower score even if they have similar accuracies.

In the data below, we have two sets of probabilites from hypothetical models. The first has probabilities that are not as "confident" when predicting the two classes (the probabilities are close to .5). The second has probabilities that are more "confident" when predicting the two classes (the probabilities are close to the extremes of 0 or 1).

### Example

import numpy as np  
  
n = 10000  
y = np.array([0] \* n + [1] \* n)  
#  
y\_prob\_1 = np.array(  
    np.random.uniform(.25, .5, n//2).tolist() +  
    np.random.uniform(.3, .7, n).tolist() +  
    np.random.uniform(.5, .75, n//2).tolist()  
)  
y\_prob\_2 = np.array(  
    np.random.uniform(0, .4, n//2).tolist() +  
    np.random.uniform(.3, .7, n).tolist() +  
    np.random.uniform(.6, 1, n//2).tolist()  
)  
  
print(f'model 1 accuracy score: {accuracy\_score(y, y\_prob\_1>.5)}')  
print(f'model 2 accuracy score: {accuracy\_score(y, y\_prob\_2>.5)}')  
  
print(f'model 1 AUC score: {roc\_auc\_score(y, y\_prob\_1)}')  
print(f'model 2 AUC score: {roc\_auc\_score(y, y\_prob\_2)}')

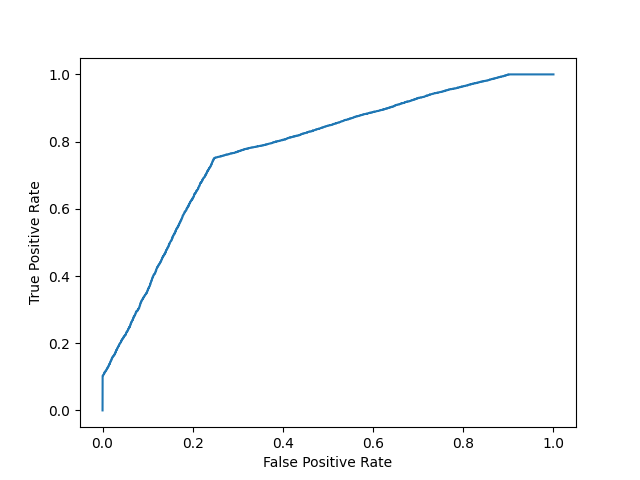
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_auc_roc5)

### Example

Plot model 1:

plot\_roc\_curve(y, y\_prob\_1)

### Result



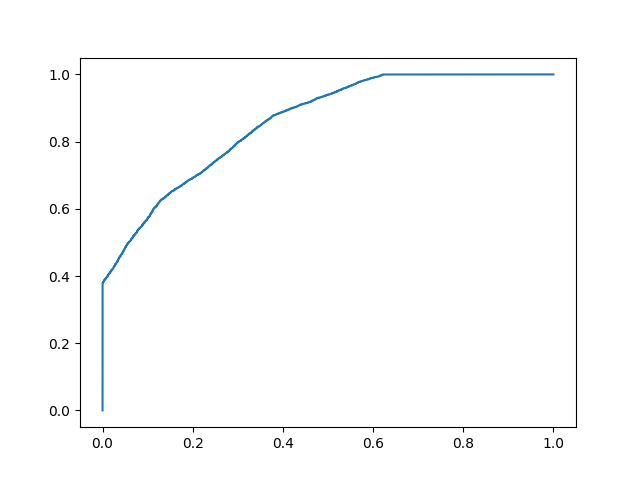
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_auc_roc6)

### Example

Plot model 2:

fpr, tpr, thresholds = roc\_curve(y, y\_prob\_2)  
plt.plot(fpr, tpr)

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_auc_roc7)

Even though the accuracies for the two models are similar, the model with the higher AUC score will be more reliable because it takes into account the predicted probability. It is more likely to give you higher accuracy when predicting future data.

# Machine Learning - K-nearest neighbors (KNN)

On this page, W3schools.com collaborates with [NYC Data Science Academy](https://t.sidekickopen01.com/s3t/c/5/f18dQhb0S7kv8c7RP2W1z75qk59hl3kW7_k2847tBZxCVvfv0f1GTV9PW2RxlDT2bzNMYdZ8s8G01?te=W3R5hFj4cm2zwW41-DxM3zhrr_W3F7ZBj3F6bSSW43T4N94hMnzcW3F4Fvd3zbTMqW2fgYK73F7y_5W3Zp0KM3_Qh5fW4hLxLG2f1Dw_W3F7y_53Zp0KMW3_QgzD3H6xvkW2dTzCR3SYMmGW1mp5CL3ZWTYrW4cbSf23_Qh9QW49NMw73HdlCPW3Hbzjf4fKWwkW43TDjD41YtR1W3zbVlf1S1tRf38WL2&si=6321147182055424&pi=566c78b2-d5c2-45ad-ad0c-74291c07630b), to deliver digital training content to our students.

## KNN

KNN is a simple, supervised machine learning (ML) algorithm that can be used for classification or regression tasks - and is also frequently used in missing value imputation. It is based on the idea that the observations closest to a given data point are the most "similar" observations in a data set, and we can therefore classify unforeseen points based on the values of the closest existing points. By choosing K, the user can select the number of nearby observations to use in the algorithm.

Here, we will show you how to implement the KNN algorithm for classification, and show how different values of K affect the results.

## How does it work?

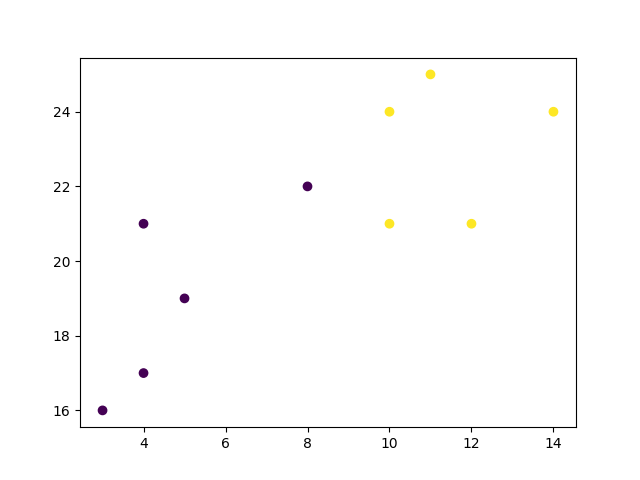
K is the number of nearest neighbors to use. For classification, a majority vote is used to determined which class a new observation should fall into. Larger values of K are often more robust to outliers and produce more stable decision boundaries than very small values (K=3 would be better than K=1, which might produce undesirable results.

### Example

Start by visualizing some data points:

import matplotlib.pyplot as plt  
  
x = [4, 5, 10, 4, 3, 11, 14 , 8, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]  
classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1]  
  
plt.scatter(x, y, c=classes)  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_knn1)

ADVERTISEMENT

Learn more about [NYCDSA](https://nycdatascience.edu/about-us/?utm_campaign=w3school&utm_source=W3school&utm_medium=display%20Ads&utm_term=PPC&utm_content=About%20Us%20video)

Now we fit the KNN algorithm with K=1:

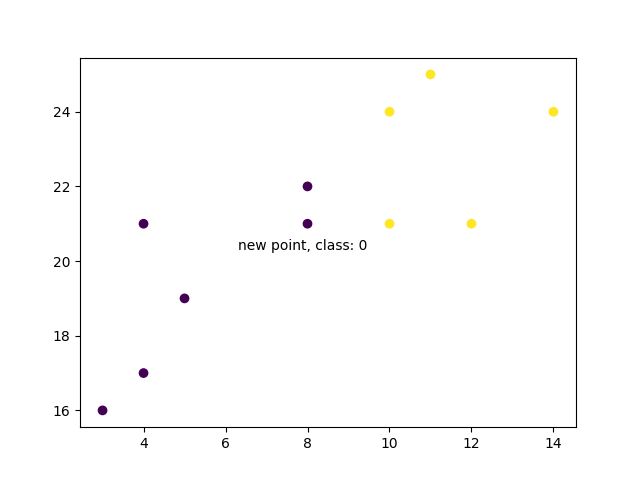
from sklearn.neighbors import KNeighborsClassifier  
  
data = list(zip(x, y))  
knn = KNeighborsClassifier(n\_neighbors=1)  
  
knn.fit(data, classes)

And use it to classify a new data point:

### Example

new\_x = 8  
new\_y = 21  
new\_point = [(new\_x, new\_y)]  
  
prediction = knn.predict(new\_point)  
  
plt.scatter(x + [new\_x], y + [new\_y], c=classes + [prediction[0]])  
plt.text(x=new\_x-1.7, y=new\_y-0.7, s=f"new point, class: {prediction[0]}")  
plt.show()

### Result



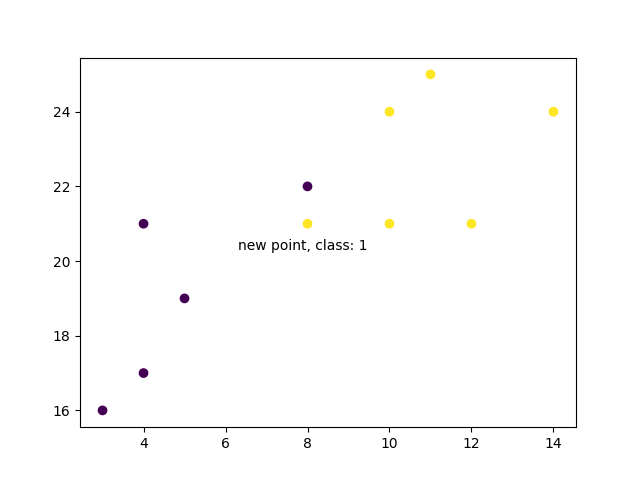
[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_knn2)

Now we do the same thing, but with a higher K value which changes the prediction:

### Example

knn = KNeighborsClassifier(n\_neighbors=5)  
  
knn.fit(data, classes)  
prediction = knn.predict(new\_point)  
  
plt.scatter(x + [new\_x], y + [new\_y], c=classes + [prediction[0]])  
plt.text(x=new\_x-1.7, y=new\_y-0.7, s=f"new point, class: {prediction[0]}")  
plt.show()

### Result



[Run example »](https://www.w3schools.com/python/trypython.asp?filename=demo_ml_knn3)

## Example Explained

Import the modules you need.

You can learn about the Matplotlib module in our ["Matplotlib Tutorial](https://www.w3schools.com/python/matplotlib_intro.asp).

scikit-learn is a popular library for machine learning in Python.

import matplotlib.pyplot as plt  
from sklearn.neighbors import KNeighborsClassifier

Create arrays that resemble variables in a dataset. We have two input features (x and y) and then a target class (class). The input features that are pre-labeled with our target class will be used to predict the class of new data. Note that while we only use two input features here, this method will work with any number of variables:

x = [4, 5, 10, 4, 3, 11, 14 , 8, 10, 12]  
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]  
classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1]

Turn the input features into a set of points:

data = list(zip(x, y))  
print(data)

### Result:

[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24), (8, 22), (10, 21), (12, 21)]

Using the input features and target class, we fit a KNN model on the model using 1 nearest neighbor:

knn = KNeighborsClassifier(n\_neighbors=1)  
knn.fit(data, classes)

Then, we can use the same KNN object to predict the class of new, unforeseen data points. First we create new x and y features, and then call knn.predict() on the new data point to get a class of 0 or 1:

new\_x = 8  
new\_y = 21  
new\_point = [(new\_x, new\_y)]  
prediction = knn.predict(new\_point)  
print(prediction)

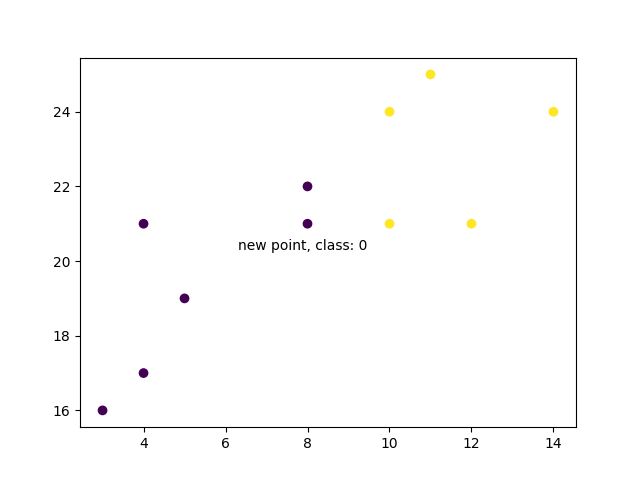
### Result:

[0]

When we plot all the data along with the new point and class, we can see it's been labeled blue with the 1 class. The text annotation is just to highlight the location of the new point:

plt.scatter(x + [new\_x], y + [new\_y], c=classes + [prediction[0]])  
plt.text(x=new\_x-1.7, y=new\_y-0.7, s=f"new point, class: {prediction[0]}")  
plt.show()

### Result:



However, when we changes the number of neighbors to 5, the number of points used to classify our new point changes. As a result, so does the classification of the new point:

knn = KNeighborsClassifier(n\_neighbors=5)  
knn.fit(data, classes)  
prediction = knn.predict(new\_point)  
print(prediction)

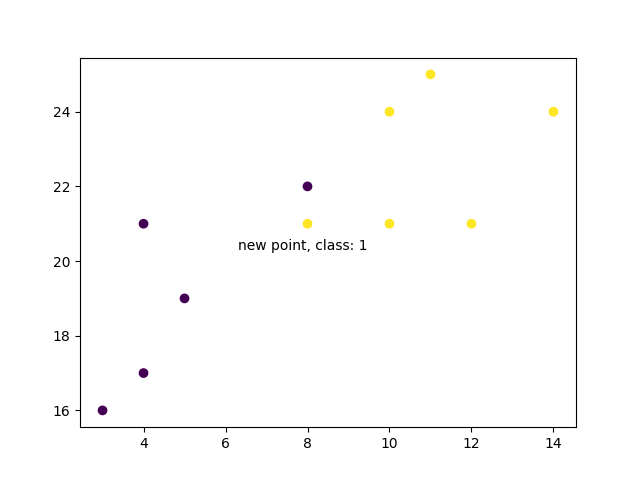
### Result:

[1]

When we plot the class of the new point along with the older points, we note that the color has changed based on the associated class label:

plt.scatter(x + [new\_x], y + [new\_y], c=classes + [prediction[0]])  
plt.text(x=new\_x-1.7, y=new\_y-0.7, s=f"new point, class: {prediction[0]}")  
plt.show()

### Result:



# Python MySQL

Python can be used in database applications.

One of the most popular databases is MySQL.

## MySQL Database

To be able to experiment with the code examples in this tutorial, you should have MySQL installed on your computer.

You can download a free MySQL database at <https://www.mysql.com/downloads/>.

## Install MySQL Driver

Python needs a MySQL driver to access the MySQL database.

In this tutorial we will use the driver "MySQL Connector".

We recommend that you use PIP to install "MySQL Connector".

PIP is most likely already installed in your Python environment.

Navigate your command line to the location of PIP, and type the following:

Download and install "MySQL Connector":

C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>python -m pip install mysql-connector-python

Now you have downloaded and installed a MySQL driver.

## Test MySQL Connector

To test if the installation was successful, or if you already have "MySQL Connector" installed, create a Python page with the following content:

demo\_mysql\_test.py:

import mysql.connector

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_test)

If the above code was executed with no errors, "MySQL Connector" is installed and ready to be used.

## Create Connection

Start by creating a connection to the database.

Use the username and password from your MySQL database:

demo\_mysql\_connection.py:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword"  
)  
  
print(mydb)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_connection)

Now you can start querying the database using SQL statements.

# Python MySQL Create Database

## Creating a Database

To create a database in MySQL, use the "CREATE DATABASE" statement:

### Example

create a database named "mydatabase":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword"  
)  
  
mycursor = mydb.cursor()  
mycursor.execute("CREATE DATABASE mydatabase")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_create_db)

If the above code was executed with no errors, you have successfully created a database.

## Check if Database Exists

You can check if a database exist by listing all databases in your system by using the "SHOW DATABASES" statement:

### Example

Return a list of your system's databases:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword"  
)  
  
mycursor = mydb.cursor()  
mycursor.execute("SHOW DATABASES")  
  
for x in mycursor:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_show_databases)

Or you can try to access the database when making the connection:

### Example

Try connecting to the database "mydatabase":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
**database="mydatabase"**  
)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_db_exist)

If the database does not exist, you will get an error.

# Python MySQL Create Table

## Creating a Table

To create a table in MySQL, use the "CREATE TABLE" statement.

Make sure you define the name of the database when you create the connection

### Example

Create a table named "customers":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
mycursor = mydb.cursor()  
mycursor.execute("CREATE TABLE customers (name VARCHAR(255), address VARCHAR(255))")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_create_table)

If the above code was executed with no errors, you have now successfully created a table.

## Check if Table Exists

You can check if a table exist by listing all tables in your database with the "SHOW TABLES" statement:

### Example

Return a list of your system's databases:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
mycursor.execute("SHOW TABLES")  
  
for x in mycursor:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_show_tables)

## Primary Key

When creating a table, you should also create a column with a unique key for each record.

This can be done by defining a PRIMARY KEY.

We use the statement "INT AUTO\_INCREMENT PRIMARY KEY" which will insert a unique number for each record. Starting at 1, and increased by one for each record.

### Example

Create primary key when creating the table:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
mycursor.execute("CREATE TABLE customers (id INT AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(255), address VARCHAR(255))")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_primary_key)

If the table already exists, use the ALTER TABLE keyword:

### Example

Create primary key on an existing table:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
mycursor.execute("ALTER TABLE customers ADD COLUMN id INT AUTO\_INCREMENT PRIMARY KEY")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_alter_primary_key)

# Python MySQL Insert Into Table

## Insert Into Table

To fill a table in MySQL, use the "INSERT INTO" statement.

### Example

Insert a record in the "customers" table:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"  
val = ("John", "Highway 21")  
mycursor.execute(sql, val)  
 **mydb.commit()**  
print(mycursor.rowcount, "record inserted.")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_insert)

**Important!:** Notice the statement: mydb.commit(). It is required to make the changes, otherwise no changes are made to the table.

## Insert Multiple Rows

To insert multiple rows into a table, use the executemany() method.

The second parameter of the executemany() method is a list of tuples, containing the data you want to insert:

### Example

Fill the "customers" table with data:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"  
val = [  
  ('Peter', 'Lowstreet 4'),  
  ('Amy', 'Apple st 652'),  
  ('Hannah', 'Mountain 21'),  
  ('Michael', 'Valley 345'),  
  ('Sandy', 'Ocean blvd 2'),  
  ('Betty', 'Green Grass 1'),  
  ('Richard', 'Sky st 331'),  
  ('Susan', 'One way 98'),  
  ('Vicky', 'Yellow Garden 2'),  
  ('Ben', 'Park Lane 38'),  
  ('William', 'Central st 954'),  
  ('Chuck', 'Main Road 989'),  
  ('Viola', 'Sideway 1633')  
]  
  
mycursor.executemany(sql, val)  
  
mydb.commit()  
  
print(mycursor.rowcount, "was inserted.")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_insert_many)

## Get Inserted ID

You can get the id of the row you just inserted by asking the cursor object.

**Note:** If you insert more than one row, the id of the last inserted row is returned.

### Example

Insert one row, and return the ID:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"  
val = ("Michelle", "Blue Village")  
mycursor.execute(sql, val)  
  
mydb.commit()  
  
print("1 record inserted, ID:", mycursor.lastrowid)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_insert_id)

# Python MySQL Select From

## Select From a Table

To select from a table in MySQL, use the "SELECT" statement:

### Example

Select all records from the "customers" table, and display the result:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
mycursor.execute("SELECT \* FROM customers")  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_select)

**Note:** We use the fetchall() method, which fetches all rows from the last executed statement.

## Selecting Columns

To select only some of the columns in a table, use the "SELECT" statement followed by the column name(s):

### Example

Select only the name and address columns:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
mycursor.execute("SELECT name, address FROM customers")  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_select_columns)

## Using the fetchone() Method

If you are only interested in one row, you can use the fetchone() method.

The fetchone() method will return the first row of the result:

### Example

Fetch only one row:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
mycursor.execute("SELECT \* FROM customers")  
myresult = mycursor.fetchone()  
  
print(myresult)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_select_fetchone)

# Python MySQL Where

## Select With a Filter

When selecting records from a table, you can filter the selection by using the "WHERE" statement:

### Example

Select record(s) where the address is "Park Lane 38": result:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "SELECT \* FROM customers WHERE address ='Park Lane 38'"  
mycursor.execute(sql)  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_where)

## Wildcard Characters

You can also select the records that starts, includes, or ends with a given letter or phrase.

Use the %  to represent wildcard characters:

### Example

Select records where the address contains the word "way":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "SELECT \* FROM customers WHERE address LIKE '%way%'"  
mycursor.execute(sql)  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_where_wildcard)

## Prevent SQL Injection

When query values are provided by the user, you should escape the values.

This is to prevent SQL injections, which is a common web hacking technique to destroy or misuse your database.

The mysql.connector module has methods to escape query values:

### Example

Escape query values by using the placholder %s method:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "SELECT \* FROM customers WHERE address = %s"  
adr = ("Yellow Garden 2", )  
mycursor.execute(sql, adr)  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_where_escape)

# Python MySQL Order By

## Sort the Result

Use the ORDER BY statement to sort the result in ascending or descending order.

The ORDER BY keyword sorts the result ascending by default. To sort the result in descending order, use the DESC keyword.

### Example

Sort the result alphabetically by name: result:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "SELECT \* FROM customers ORDER BY name"  
mycursor.execute(sql)  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_orderby)

## ORDER BY DESC

Use the DESC keyword to sort the result in a descending order.

### Example

Sort the result reverse alphabetically by name:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "SELECT \* FROM customers ORDER BY name DESC"  
mycursor.execute(sql)  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_orderby_desc)

# Python MySQL Delete From By

## Delete Record

You can delete records from an existing table by using the "DELETE FROM" statement:

### Example

Delete any record where the address is "Mountain 21":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "DELETE FROM customers WHERE address = 'Mountain 21'"  
mycursor.execute(sql)  
mydb.commit()  
print(mycursor.rowcount, "record(s) deleted")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_delete)

**Important!:** Notice the statement: mydb.commit(). It is required to make the changes, otherwise no changes are made to the table.

**Notice the WHERE clause in the DELETE syntax:** The WHERE clause specifies which record(s) that should be deleted. If you omit the WHERE clause, all records will be deleted!

## Prevent SQL Injection

It is considered a good practice to escape the values of any query, also in delete statements.

This is to prevent SQL injections, which is a common web hacking technique to destroy or misuse your database.

The mysql.connector module uses the placeholder %s to escape values in the delete statement:

### Example

Escape values by using the placeholder %s method:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "DELETE FROM customers WHERE address = %s"  
adr = ("Yellow Garden 2", )  
mycursor.execute(sql, adr)  
mydb.commit()  
  
print(mycursor.rowcount, "record(s) deleted")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_delete_escape)

# Python MySQL Drop Table

## Delete a Table

You can delete an existing table by using the "DROP TABLE" statement:

### Example

Delete the table "customers":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "DROP TABLE customers"  
mycursor.execute(sql)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_drop_table)

## Drop Only if Exist

If the table you want to delete is already deleted, or for any other reason does not exist, you can use the IF EXISTS keyword to avoid getting an error.

### Example

Delete the table "customers" if it exists:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "DROP TABLE IF EXISTS customers"  
mycursor.execute(sql)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_drop_table2)

# Python MySQL Update Table

## Update Table

You can update existing records in a table by using the "UPDATE" statement:

### Example

Overwrite the address column from "Valley 345" to "Canyon 123":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "UPDATE customers SET address = 'Canyon 123' WHERE address = 'Valley 345'"  
mycursor.execute(sql)  
mydb.commit()  
print(mycursor.rowcount, "record(s) affected")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_update)

**Important!:** Notice the statement: mydb.commit(). It is required to make the changes, otherwise no changes are made to the table.

**Notice the WHERE clause in the UPDATE syntax:** The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

## Prevent SQL Injection

It is considered a good practice to escape the values of any query, also in update statements.

This is to prevent SQL injections, which is a common web hacking technique to destroy or misuse your database.

The mysql.connector module uses the placeholder %s to escape values in the delete statement:

### Example

Escape values by using the placeholder %s method:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
sql = "UPDATE customers SET address = %s WHERE address = %s"  
val = ("Valley 345", "Canyon 123")  
mycursor.execute(sql, val)  
mydb.commit()  
print(mycursor.rowcount, "record(s) affected")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_update_escape)

# Python MySQL Limit

## Limit the Result

You can limit the number of records returned from the query, by using the "LIMIT" statement:

### Example

Select the 5 first records in the "customers" table:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
mycursor.execute("SELECT \* FROM customers LIMIT 5")  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_limit)

## Start From Another Position

If you want to return five records, starting from the third record, you can use the "OFFSET" keyword:

### Example

Start from position 3, and return 5 records:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
mycursor.execute("SELECT \* FROM customers LIMIT 5 OFFSET 2")  
myresult = mycursor.fetchall()

for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_limit_offset)

# Python MySQL Join

## Join Two or More Tables

You can combine rows from two or more tables, based on a related column between them, by using a JOIN statement.

Consider you have a "users" table and a "products" table:

### users

{ id: 1, name: 'John', fav: 154},  
{ id: 2, name: 'Peter', fav: 154},  
{ id: 3, name: 'Amy', fav: 155},  
{ id: 4, name: 'Hannah', fav:},  
{ id: 5, name: 'Michael', fav:}

### products

{ id: 154, name: 'Chocolate Heaven' },  
{ id: 155, name: 'Tasty Lemons' },  
{ id: 156, name: 'Vanilla Dreams' }

These two tables can be combined by using users' fav field and products' id field.

### Example

Join users and products to see the name of the users favorite product:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
sql = "SELECT \  
  users.name AS user, \  
  products.name AS favorite \  
  FROM users \  
  INNER JOIN products ON users.fav = products.id"  
mycursor.execute(sql)  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_inner_join)

**Note:** You can use JOIN instead of INNER JOIN. They will both give you the same result.

## LEFT JOIN

In the example above, Hannah, and Michael were excluded from the result, that is because INNER JOIN only shows the records where there is a match.

If you want to show all users, even if they do not have a favorite product, use the LEFT JOIN statement:

### Example

Select all users and their favorite product:

sql = "SELECT \  
  users.name AS user, \  
  products.name AS favorite \  
  FROM users \  
  LEFT JOIN products ON users.fav = products.id"

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_left_join)

## RIGHT JOIN

If you want to return all products, and the users who have them as their favorite, even if no user have them as their favorite, use the RIGHT JOIN statement:

### Example

Select all products, and the user(s) who have them as their favorite:

sql = "SELECT \  
  users.name AS user, \  
  products.name AS favorite \  
  FROM users \  
  RIGHT JOIN products ON users.fav = products.id"

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mysql_right_join)

**Note:** Hannah and Michael, who have no favorite product, are not included in the result.

# Python MongoDB

Python can be used in database applications.

One of the most popular NoSQL database is MongoDB.

## MongoDB

MongoDB stores data in JSON-like documents, which makes the database very flexible and scalable.

To be able to experiment with the code examples in this tutorial, you will need access to a MongoDB database.

You can download a free MongoDB database at [https://www.mongodb.com](https://www.mongodb.com/).

Or get started right away with a MongoDB cloud service at <https://www.mongodb.com/cloud/atlas>.

## PyMongo

Python needs a MongoDB driver to access the MongoDB database.

In this tutorial we will use the MongoDB driver "PyMongo".

We recommend that you use PIP to install "PyMongo".

PIP is most likely already installed in your Python environment.

Navigate your command line to the location of PIP, and type the following:

Download and install "PyMongo":

C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>python -m pip install pymongo

Now you have downloaded and installed a mongoDB driver.

## Test PyMongo

To test if the installation was successful, or if you already have "pymongo" installed, create a Python page with the following content:

demo\_mongodb\_test.py:

import pymongo

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_test)

If the above code was executed with no errors, "pymongo" is installed and ready to be used.

# Python MongoDB Create Database

## Creating a Database

To create a database in MongoDB, start by creating a MongoClient object, then specify a connection URL with the correct ip address and the name of the database you want to create.

MongoDB will create the database if it does not exist, and make a connection to it.

### Example

Create a database called "mydatabase":

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_create_db)

**Important:** In MongoDB, a database is not created until it gets content!

MongoDB waits until you have created a collection (table), with at least one document (record) before it actually creates the database (and collection).

## Check if Database Exists

**Remember:** In MongoDB, a database is not created until it gets content, so if this is your first time creating a database, you should complete the next two chapters (create collection and create document) before you check if the database exists!

You can check if a database exist by listing all databases in you system:

### Example

Return a list of your system's databases:

print(myclient.list\_database\_names())

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_check_db)

Or you can check a specific database by name:

### Example

Check if "mydatabase" exists:

dblist = myclient.list\_database\_names()  
if "mydatabase" in dblist:  
  print("The database exists.")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_check_db2)

# Python MongoDB Create Collection

A **collection** in MongoDB is the same as a **table** in SQL databases.

## Creating a Collection

To create a collection in MongoDB, use database object and specify the name of the collection you want to create.

MongoDB will create the collection if it does not exist.

### Example

Create a collection called "customers":

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
  
mycol = mydb["customers"]

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_create_collection)

**Important:** In MongoDB, a collection is not created until it gets content!

MongoDB waits until you have inserted a document before it actually creates the collection.

## Check if Collection Exists

**Remember:** In MongoDB, a collection is not created until it gets content, so if this is your first time creating a collection, you should complete the next chapter (create document) before you check if the collection exists!

You can check if a collection exist in a database by listing all collections:

### Example

Return a list of all collections in your database:

print(mydb.list\_collection\_names())

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_check_collection)

Or you can check a specific collection by name:

### Example

Check if the "customers" collection exists:

collist = mydb.list\_collection\_names()  
if "customers" in collist:  
  print("The collection exists.")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_check_collection2)

# Python MongoDB Insert Document

A **document** in MongoDB is the same as a **record** in SQL databases.

## Insert Into Collection

To insert a record, or document as it is called in MongoDB, into a collection, we use the insert\_one() method.

The first parameter of the insert\_one() method is a dictionary containing the name(s) and value(s) of each field in the document you want to insert.

### Example

Insert a record in the "customers" collection:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
mydict = { "name": "John", "address": "Highway 37" }  
  
x = mycol.insert\_one(mydict)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_insert)

## Return the \_id Field

The insert\_one() method returns a InsertOneResult object, which has a property, inserted\_id, that holds the id of the inserted document.

### Example

Insert another record in the "customers" collection, and return the value of the \_id field:

mydict = { "name": "Peter", "address": "Lowstreet 27" }  
  
x = mycol.insert\_one(mydict)  
  
print(x.inserted\_id)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_insert_id)

If you do not specify an \_id field, then MongoDB will add one for you and assign a unique id for each document.

In the example above no \_id field was specified, so MongoDB assigned a unique \_id for the record (document).

## Insert Multiple Documents

To insert multiple documents into a collection in MongoDB, we use the insert\_many() method.

The first parameter of the insert\_many() method is a list containing dictionaries with the data you want to insert:

### Example

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
mylist = [  
  { "name": "Amy", "address": "Apple st 652"},  
  { "name": "Hannah", "address": "Mountain 21"},  
  { "name": "Michael", "address": "Valley 345"},  
  { "name": "Sandy", "address": "Ocean blvd 2"},  
  { "name": "Betty", "address": "Green Grass 1"},  
  { "name": "Richard", "address": "Sky st 331"},  
  { "name": "Susan", "address": "One way 98"},  
  { "name": "Vicky", "address": "Yellow Garden 2"},  
  { "name": "Ben", "address": "Park Lane 38"},  
  { "name": "William", "address": "Central st 954"},  
  { "name": "Chuck", "address": "Main Road 989"},  
  { "name": "Viola", "address": "Sideway 1633"}  
]  
x = mycol.insert\_many(mylist)  
#print list of the \_id values of the inserted documents:  
print(x.inserted\_ids)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_insert_many)

The insert\_many() method returns a InsertManyResult object, which has a property, inserted\_ids, that holds the ids of the inserted documents.

## Insert Multiple Documents, with Specified IDs

If you do not want MongoDB to assign unique ids for you document, you can specify the \_id field when you insert the document(s).

Remember that the values has to be unique. Two documents cannot have the same \_id.

### Example

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
mylist = [  
  { "\_id": 1, "name": "John", "address": "Highway 37"},  
  { "\_id": 2, "name": "Peter", "address": "Lowstreet 27"},  
  { "\_id": 3, "name": "Amy", "address": "Apple st 652"},  
  { "\_id": 4, "name": "Hannah", "address": "Mountain 21"},  
  { "\_id": 5, "name": "Michael", "address": "Valley 345"},  
  { "\_id": 6, "name": "Sandy", "address": "Ocean blvd 2"},  
  { "\_id": 7, "name": "Betty", "address": "Green Grass 1"},  
  { "\_id": 8, "name": "Richard", "address": "Sky st 331"},  
  { "\_id": 9, "name": "Susan", "address": "One way 98"},  
  { "\_id": 10, "name": "Vicky", "address": "Yellow Garden 2"},  
  { "\_id": 11, "name": "Ben", "address": "Park Lane 38"},  
  { "\_id": 12, "name": "William", "address": "Central st 954"},  
  { "\_id": 13, "name": "Chuck", "address": "Main Road 989"},  
  { "\_id": 14, "name": "Viola", "address": "Sideway 1633"}  
]  
  
x = mycol.insert\_many(mylist)  
  
#print list of the \_id values of the inserted documents:  
print(x.inserted\_ids)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_insert_many2)

# Python MongoDB Find

In MongoDB we use the find() and find\_one() methods to find data in a collection.

Just like the **SELECT** statement is used to find data in a table in a MySQL database.

## Find One

To select data from a collection in MongoDB, we can use the find\_one() method.

The find\_one() method returns the first occurrence in the selection.

### Example

Find the first document in the customers collection:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
x = mycol.find\_one()  
  
print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_find_one)

## Find All

To select data from a table in MongoDB, we can also use the find() method.

The find() method returns all occurrences in the selection.

The first parameter of the find() method is a query object. In this example we use an empty query object, which selects all documents in the collection.

No parameters in the find() method gives you the same result as **SELECT \*** in MySQL.

### Example

Return all documents in the "customers" collection, and print each document:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
for x in mycol.find():  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_find)

## Return Only Some Fields

The second parameter of the find() method is an object describing which fields to include in the result.

This parameter is optional, and if omitted, all fields will be included in the result.

### Example

Return only the names and addresses, not the \_ids:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
for x in mycol.find({},{ "\_id": 0, "name": 1, "address": 1 }):  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_find_some)

You are not allowed to specify both 0 and 1 values in the same object (except if one of the fields is the \_id field). If you specify a field with the value 0, all other fields get the value 1, and vice versa:

### Example

This example will exclude "address" from the result:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
for x in mycol.find({},{ "address": 0 }):  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_find_some2)

### Example

You get an error if you specify both 0 and 1 values in the same object (except if one of the fields is the \_id field):

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
for x in mycol.find({},{ "name": 1, "address": 0 }):  
  print(x)

# Python MongoDB Query

## Filter the Result

When finding documents in a collection, you can filter the result by using a query object.

The first argument of the find() method is a query object, and is used to limit the search.

### Example

Find document(s) with the address "Park Lane 38":

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myquery = { "address": "Park Lane 38" }  
mydoc = mycol.find(myquery)  
  
for x in mydoc:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_query)

## Advanced Query

To make advanced queries you can use modifiers as values in the query object.

E.g. to find the documents where the "address" field starts with the letter "S" or higher (alphabetically), use the greater than modifier: {"$gt": "S"}:

### Example

Find documents where the address starts with the letter "S" or higher:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myquery = { "address": { "$gt": "S" } }  
mydoc = mycol.find(myquery)  
  
for x in mydoc:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_query_modifier)

## Filter With Regular Expressions

You can also use regular expressions as a modifier.

**Regular expressions can only be used to query strings.**

To find only the documents where the "address" field starts with the letter "S", use the regular expression {"$regex": "^S"}:

### Example

Find documents where the address starts with the letter "S":

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myquery = { "address": { "$regex": "^S" } }  
mydoc = mycol.find(myquery)

for x in mydoc:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_query_regex)

# Python MongoDB Sort

## Sort the Result

Use the sort() method to sort the result in ascending or descending order.

The sort() method takes one parameter for "fieldname" and one parameter for "direction" (ascending is the default direction).

### Example

Sort the result alphabetically by name:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
mydoc = mycol.find().sort("name")  
  
for x in mydoc:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_sort)

## Sort Descending

Use the value -1 as the second parameter to sort descending.

sort("name", 1) #ascending  
sort("name", -1) #descending

### Example

Sort the result reverse alphabetically by name:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
mydoc = mycol.find().sort("name", -1)  
  
for x in mydoc:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_sort2)

# Python MongoDB Delete Document

## Delete Document

To delete one document, we use the delete\_one() method.

The first parameter of the delete\_one() method is a query object defining which document to delete.

**Note:** If the query finds more than one document, only the first occurrence is deleted.

### Example

Delete the document with the address "Mountain 21":

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myquery = { "address": "Mountain 21" }  
mycol.delete\_one(myquery)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_delete_one)

## Delete Many Documents

To delete more than one document, use the delete\_many() method.

The first parameter of the delete\_many() method is a query object defining which documents to delete.

### Example

Delete all documents were the address starts with the letter S:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myquery = { "address": {"$regex": "^S"} }  
x = mycol.delete\_many(myquery)  
print(x.deleted\_count, " documents deleted.")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_delete_many)

## Delete All Documents in a Collection

To delete all documents in a collection, pass an empty query object to the delete\_many() method:

### Example

Delete all documents in the "customers" collection:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
x = mycol.delete\_many({})  
  
print(x.deleted\_count, " documents deleted.")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_delete_all)

# Python MongoDB Drop Collection

## Delete Collection

You can delete a table, or collection as it is called in MongoDB, by using the drop() method.

### Example

Delete the "customers" collection:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
mycol.drop()

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_drop_collection)

The drop() method returns true if the collection was dropped successfully, and false if the collection does not exist.

# Python MongoDB Update

## Update Collection

You can update a record, or document as it is called in MongoDB, by using the update\_one() method.

The first parameter of the update\_one() method is a query object defining which document to update.

**Note:** If the query finds more than one record, only the first occurrence is updated.

The second parameter is an object defining the new values of the document.

### Example

Change the address from "Valley 345" to "Canyon 123":

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myquery = { "address": "Valley 345" }  
newvalues = { "$set": { "address": "Canyon 123" } }  
  
mycol.update\_one(myquery, newvalues)  
  
#print "customers" after the update:  
for x in mycol.find():  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_update_one)

## Update Many

To update all documents that meets the criteria of the query, use the update\_many() method.

### Example

Update all documents where the address starts with the letter "S":

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myquery = { "address": { "$regex": "^S" } }  
newvalues = { "$set": { "name": "Minnie" } }  
  
x = mycol.update\_many(myquery, newvalues)  
  
print(x.modified\_count, "documents updated.")

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_update_many)

# Python MongoDB Limit

## Limit the Result

To limit the result in MongoDB, we use the limit() method.

The limit() method takes one parameter, a number defining how many documents to return.

Consider you have a "customers" collection:

### Customers

{'\_id': 1, 'name': 'John', 'address': 'Highway37'}  
{'\_id': 2, 'name': 'Peter', 'address': 'Lowstreet 27'}  
{'\_id': 3, 'name': 'Amy', 'address': 'Apple st 652'}  
{'\_id': 4, 'name': 'Hannah', 'address': 'Mountain 21'}  
{'\_id': 5, 'name': 'Michael', 'address': 'Valley 345'}  
{'\_id': 6, 'name': 'Sandy', 'address': 'Ocean blvd 2'}  
{'\_id': 7, 'name': 'Betty', 'address': 'Green Grass 1'}  
{'\_id': 8, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 9, 'name': 'Susan', 'address': 'One way 98'}  
{'\_id': 10, 'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'\_id': 11, 'name': 'Ben', 'address': 'Park Lane 38'}  
{'\_id': 12, 'name': 'William', 'address': 'Central st 954'}  
{'\_id': 13, 'name': 'Chuck', 'address': 'Main Road 989'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}

### Example

Limit the result to only return 5 documents:

import pymongo  
  
myclient = pymongo.MongoClient("mongodb://localhost:27017/")  
mydb = myclient["mydatabase"]  
mycol = mydb["customers"]  
  
myresult = mycol.find().limit(5)  
  
#print the result:  
for x in myresult:  
  print(x)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_mongodb_limit)

# Python Reference

This section contains a Python reference documentation.

## Python Reference

[Built-in Functions](https://www.w3schools.com/python/python_ref_functions.asp) [String Methods](https://www.w3schools.com/python/python_ref_string.asp) [List Methods](https://www.w3schools.com/python/python_ref_list.asp) [Dictionary Methods](https://www.w3schools.com/python/python_ref_dictionary.asp) [Tuple Methods](https://www.w3schools.com/python/python_ref_tuple.asp) [Set Methods](https://www.w3schools.com/python/python_ref_set.asp) [File Methods](https://www.w3schools.com/python/python_ref_file.asp) [Keywords](https://www.w3schools.com/python/python_ref_keywords.asp) [Exceptions](https://www.w3schools.com/python/python_ref_exceptions.asp) [Glossary](https://www.w3schools.com/python/python_ref_glossary.asp)

**Module Reference**

[Random Module](https://www.w3schools.com/python/module_random.asp) [Requests Module](https://www.w3schools.com/python/module_requests.asp) [Math Module](https://www.w3schools.com/python/module_math.asp) [CMath Module](https://www.w3schools.com/python/module_cmath.asp)

# Python Built in Functions

Python has a set of built-in functions.

|  |  |
| --- | --- |
| **Function** | **Description** |
| [abs()](https://www.w3schools.com/python/ref_func_abs.asp) | Returns the absolute value of a number |
| [all()](https://www.w3schools.com/python/ref_func_all.asp) | Returns True if all items in an iterable object are true |
| [any()](https://www.w3schools.com/python/ref_func_any.asp) | Returns True if any item in an iterable object is true |
| [ascii()](https://www.w3schools.com/python/ref_func_ascii.asp) | Returns a readable version of an object. Replaces none-ascii characters with escape character |
| [bin()](https://www.w3schools.com/python/ref_func_bin.asp) | Returns the binary version of a number |
| [bool()](https://www.w3schools.com/python/ref_func_bool.asp) | Returns the boolean value of the specified object |
| [bytearray()](https://www.w3schools.com/python/ref_func_bytearray.asp) | Returns an array of bytes |
| [bytes()](https://www.w3schools.com/python/ref_func_bytes.asp) | Returns a bytes object |
| [callable()](https://www.w3schools.com/python/ref_func_callable.asp) | Returns True if the specified object is callable, otherwise False |
| [chr()](https://www.w3schools.com/python/ref_func_chr.asp) | Returns a character from the specified Unicode code. |
| classmethod() | Converts a method into a class method |
| [compile()](https://www.w3schools.com/python/ref_func_compile.asp) | Returns the specified source as an object, ready to be executed |
| [complex()](https://www.w3schools.com/python/ref_func_complex.asp) | Returns a complex number |
| [delattr()](https://www.w3schools.com/python/ref_func_delattr.asp) | Deletes the specified attribute (property or method) from the specified object |
| [dict()](https://www.w3schools.com/python/ref_func_dict.asp) | Returns a dictionary (Array) |
| [dir()](https://www.w3schools.com/python/ref_func_dir.asp) | Returns a list of the specified object's properties and methods |
| [divmod()](https://www.w3schools.com/python/ref_func_divmod.asp) | Returns the quotient and the remainder when argument1 is divided by argument2 |
| [enumerate()](https://www.w3schools.com/python/ref_func_enumerate.asp) | Takes a collection (e.g. a tuple) and returns it as an enumerate object |
| [eval()](https://www.w3schools.com/python/ref_func_eval.asp) | Evaluates and executes an expression |
| [exec()](https://www.w3schools.com/python/ref_func_exec.asp) | Executes the specified code (or object) |
| [filter()](https://www.w3schools.com/python/ref_func_filter.asp) | Use a filter function to exclude items in an iterable object |
| [float()](https://www.w3schools.com/python/ref_func_float.asp) | Returns a floating point number |
| [format()](https://www.w3schools.com/python/ref_func_format.asp) | Formats a specified value |
| [frozenset()](https://www.w3schools.com/python/ref_func_frozenset.asp) | Returns a frozenset object |
| [getattr()](https://www.w3schools.com/python/ref_func_getattr.asp) | Returns the value of the specified attribute (property or method) |
| [globals()](https://www.w3schools.com/python/ref_func_globals.asp) | Returns the current global symbol table as a dictionary |
| [hasattr()](https://www.w3schools.com/python/ref_func_hasattr.asp) | Returns True if the specified object has the specified attribute (property/method) |
| hash() | Returns the hash value of a specified object |
| help() | Executes the built-in help system |
| [hex()](https://www.w3schools.com/python/ref_func_hex.asp) | Converts a number into a hexadecimal value |
| [id()](https://www.w3schools.com/python/ref_func_id.asp) | Returns the id of an object |
| [input()](https://www.w3schools.com/python/ref_func_input.asp) | Allowing user input |
| [int()](https://www.w3schools.com/python/ref_func_int.asp) | Returns an integer number |
| [isinstance()](https://www.w3schools.com/python/ref_func_isinstance.asp) | Returns True if a specified object is an instance of a specified object |
| [issubclass()](https://www.w3schools.com/python/ref_func_issubclass.asp) | Returns True if a specified class is a subclass of a specified object |
| [iter()](https://www.w3schools.com/python/ref_func_iter.asp) | Returns an iterator object |
| [len()](https://www.w3schools.com/python/ref_func_len.asp) | Returns the length of an object |
| [list()](https://www.w3schools.com/python/ref_func_list.asp) | Returns a list |
| [locals()](https://www.w3schools.com/python/ref_func_locals.asp) | Returns an updated dictionary of the current local symbol table |
| [map()](https://www.w3schools.com/python/ref_func_map.asp) | Returns the specified iterator with the specified function applied to each item |
| [max()](https://www.w3schools.com/python/ref_func_max.asp) | Returns the largest item in an iterable |
| [memoryview()](https://www.w3schools.com/python/ref_func_memoryview.asp) | Returns a memory view object |
| [min()](https://www.w3schools.com/python/ref_func_min.asp) | Returns the smallest item in an iterable |
| [next()](https://www.w3schools.com/python/ref_func_next.asp) | Returns the next item in an iterable |
| [object()](https://www.w3schools.com/python/ref_func_object.asp) | Returns a new object |
| [oct()](https://www.w3schools.com/python/ref_func_oct.asp) | Converts a number into an octal |
| [open()](https://www.w3schools.com/python/ref_func_open.asp) | Opens a file and returns a file object |
| [ord()](https://www.w3schools.com/python/ref_func_ord.asp) | Convert an integer representing the Unicode of the specified character |
| [pow()](https://www.w3schools.com/python/ref_func_pow.asp) | Returns the value of x to the power of y |
| [print()](https://www.w3schools.com/python/ref_func_print.asp) | Prints to the standard output device |
| property() | Gets, sets, deletes a property |
| [range()](https://www.w3schools.com/python/ref_func_range.asp) | Returns a sequence of numbers, starting from 0 and increments by 1 (by default) |
| repr() | Returns a readable version of an object |
| [reversed()](https://www.w3schools.com/python/ref_func_reversed.asp) | Returns a reversed iterator |
| [round()](https://www.w3schools.com/python/ref_func_round.asp) | Rounds a numbers |
| [set()](https://www.w3schools.com/python/ref_func_set.asp) | Returns a new set object |
| [setattr()](https://www.w3schools.com/python/ref_func_setattr.asp) | Sets an attribute (property/method) of an object |
| [slice()](https://www.w3schools.com/python/ref_func_slice.asp) | Returns a slice object |
| [sorted()](https://www.w3schools.com/python/ref_func_sorted.asp) | Returns a sorted list |
| staticmethod() | Converts a method into a static method |
| [str()](https://www.w3schools.com/python/ref_func_str.asp) | Returns a string object |
| [sum()](https://www.w3schools.com/python/ref_func_sum.asp) | Sums the items of an iterator |
| [super()](https://www.w3schools.com/python/ref_func_super.asp) | Returns an object that represents the parent class |
| [tuple()](https://www.w3schools.com/python/ref_func_tuple.asp) | Returns a tuple |
| [type()](https://www.w3schools.com/python/ref_func_type.asp) | Returns the type of an object |
| [vars()](https://www.w3schools.com/python/ref_func_vars.asp) | Returns the \_\_dict\_\_ property of an object |
| [zip()](https://www.w3schools.com/python/ref_func_zip.asp) | Returns an iterator, from two or more iterators |

# Python String Methods

Python has a set of built-in methods that you can use on strings.

**Note:** All string methods returns new values. They do not change the original string.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [capitalize()](https://www.w3schools.com/python/ref_string_capitalize.asp) | Converts the first character to upper case |
| [casefold()](https://www.w3schools.com/python/ref_string_casefold.asp) | Converts string into lower case |
| [center()](https://www.w3schools.com/python/ref_string_center.asp) | Returns a centered string |
| [count()](https://www.w3schools.com/python/ref_string_count.asp) | Returns the number of times a specified value occurs in a string |
| [encode()](https://www.w3schools.com/python/ref_string_encode.asp) | Returns an encoded version of the string |
| [endswith()](https://www.w3schools.com/python/ref_string_endswith.asp) | Returns true if the string ends with the specified value |
| [expandtabs()](https://www.w3schools.com/python/ref_string_expandtabs.asp) | Sets the tab size of the string |
| [find()](https://www.w3schools.com/python/ref_string_find.asp) | Searches the string for a specified value and returns the position of where it was found |
| [format()](https://www.w3schools.com/python/ref_string_format.asp) | Formats specified values in a string |
| format\_map() | Formats specified values in a string |
| [index()](https://www.w3schools.com/python/ref_string_index.asp) | Searches the string for a specified value and returns the position of where it was found |
| [isalnum()](https://www.w3schools.com/python/ref_string_isalnum.asp) | Returns True if all characters in the string are alphanumeric |
| [isalpha()](https://www.w3schools.com/python/ref_string_isalpha.asp) | Returns True if all characters in the string are in the alphabet |
| [isascii()](https://www.w3schools.com/python/ref_string_isascii.asp) | Returns True if all characters in the string are ascii characters |
| [isdecimal()](https://www.w3schools.com/python/ref_string_isdecimal.asp) | Returns True if all characters in the string are decimals |
| [isdigit()](https://www.w3schools.com/python/ref_string_isdigit.asp) | Returns True if all characters in the string are digits |
| [isidentifier()](https://www.w3schools.com/python/ref_string_isidentifier.asp) | Returns True if the string is an identifier |
| [islower()](https://www.w3schools.com/python/ref_string_islower.asp) | Returns True if all characters in the string are lower case |
| [isnumeric()](https://www.w3schools.com/python/ref_string_isnumeric.asp) | Returns True if all characters in the string are numeric |
| [isprintable()](https://www.w3schools.com/python/ref_string_isprintable.asp) | Returns True if all characters in the string are printable |
| [isspace()](https://www.w3schools.com/python/ref_string_isspace.asp) | Returns True if all characters in the string are whitespaces |
| [istitle()](https://www.w3schools.com/python/ref_string_istitle.asp) | Returns True if the string follows the rules of a title |
| [isupper()](https://www.w3schools.com/python/ref_string_isupper.asp) | Returns True if all characters in the string are upper case |
| [join()](https://www.w3schools.com/python/ref_string_join.asp) | Converts the elements of an iterable into a string |
| [ljust()](https://www.w3schools.com/python/ref_string_ljust.asp) | Returns a left justified version of the string |
| [lower()](https://www.w3schools.com/python/ref_string_lower.asp) | Converts a string into lower case |
| [lstrip()](https://www.w3schools.com/python/ref_string_lstrip.asp) | Returns a left trim version of the string |
| [maketrans()](https://www.w3schools.com/python/ref_string_maketrans.asp) | Returns a translation table to be used in translations |
| [partition()](https://www.w3schools.com/python/ref_string_partition.asp) | Returns a tuple where the string is parted into three parts |
| [replace()](https://www.w3schools.com/python/ref_string_replace.asp) | Returns a string where a specified value is replaced with a specified value |
| [rfind()](https://www.w3schools.com/python/ref_string_rfind.asp) | Searches the string for a specified value and returns the last position of where it was found |
| [rindex()](https://www.w3schools.com/python/ref_string_rindex.asp) | Searches the string for a specified value and returns the last position of where it was found |
| [rjust()](https://www.w3schools.com/python/ref_string_rjust.asp) | Returns a right justified version of the string |
| [rpartition()](https://www.w3schools.com/python/ref_string_rpartition.asp) | Returns a tuple where the string is parted into three parts |
| [rsplit()](https://www.w3schools.com/python/ref_string_rsplit.asp) | Splits the string at the specified separator, and returns a list |
| [rstrip()](https://www.w3schools.com/python/ref_string_rstrip.asp) | Returns a right trim version of the string |
| [split()](https://www.w3schools.com/python/ref_string_split.asp) | Splits the string at the specified separator, and returns a list |
| [splitlines()](https://www.w3schools.com/python/ref_string_splitlines.asp) | Splits the string at line breaks and returns a list |
| [startswith()](https://www.w3schools.com/python/ref_string_startswith.asp) | Returns true if the string starts with the specified value |
| [strip()](https://www.w3schools.com/python/ref_string_strip.asp) | Returns a trimmed version of the string |
| [swapcase()](https://www.w3schools.com/python/ref_string_swapcase.asp) | Swaps cases, lower case becomes upper case and vice versa |
| [title()](https://www.w3schools.com/python/ref_string_title.asp) | Converts the first character of each word to upper case |
| [translate()](https://www.w3schools.com/python/ref_string_translate.asp) | Returns a translated string |
| [upper()](https://www.w3schools.com/python/ref_string_upper.asp) | Converts a string into upper case |
| [zfill()](https://www.w3schools.com/python/ref_string_zfill.asp) | Fills the string with a specified number of 0 values at the beginning |

**Note:** All string methods returns new values. They do not change the original string.

Learn more about strings in our [Python Strings Tutorial](https://www.w3schools.com/python/python_strings.asp).

# Python List/Array Methods

Python has a set of built-in methods that you can use on lists/arrays.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [append()](https://www.w3schools.com/python/ref_list_append.asp) | Adds an element at the end of the list |
| [clear()](https://www.w3schools.com/python/ref_list_clear.asp) | Removes all the elements from the list |
| [copy()](https://www.w3schools.com/python/ref_list_copy.asp) | Returns a copy of the list |
| [count()](https://www.w3schools.com/python/ref_list_count.asp) | Returns the number of elements with the specified value |
| [extend()](https://www.w3schools.com/python/ref_list_extend.asp) | Add the elements of a list (or any iterable), to the end of the current list |
| [index()](https://www.w3schools.com/python/ref_list_index.asp) | Returns the index of the first element with the specified value |
| [insert()](https://www.w3schools.com/python/ref_list_insert.asp) | Adds an element at the specified position |
| [pop()](https://www.w3schools.com/python/ref_list_pop.asp) | Removes the element at the specified position |
| [remove()](https://www.w3schools.com/python/ref_list_remove.asp) | Removes the first item with the specified value |
| [reverse()](https://www.w3schools.com/python/ref_list_reverse.asp) | Reverses the order of the list |
| [sort()](https://www.w3schools.com/python/ref_list_sort.asp) | Sorts the list |

**Note:** Python does not have built-in support for Arrays, but Python Lists can be used instead.

Learn more about lists in our [Python Lists Tutorial](https://www.w3schools.com/python/python_lists.asp).

Learn more about arrays in our [Python Arrays Tutorial](https://www.w3schools.com/python/python_arrays.asp).

# Python Dictionary Methods

Python has a set of built-in methods that you can use on dictionaries.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [clear()](https://www.w3schools.com/python/ref_dictionary_clear.asp) | Removes all the elements from the dictionary |
| [copy()](https://www.w3schools.com/python/ref_dictionary_copy.asp) | Returns a copy of the dictionary |
| [fromkeys()](https://www.w3schools.com/python/ref_dictionary_fromkeys.asp) | Returns a dictionary with the specified keys and value |
| [get()](https://www.w3schools.com/python/ref_dictionary_get.asp) | Returns the value of the specified key |
| [items()](https://www.w3schools.com/python/ref_dictionary_items.asp) | Returns a list containing a tuple for each key value pair |
| [keys()](https://www.w3schools.com/python/ref_dictionary_keys.asp) | Returns a list containing the dictionary's keys |
| [pop()](https://www.w3schools.com/python/ref_dictionary_pop.asp) | Removes the element with the specified key |
| [popitem()](https://www.w3schools.com/python/ref_dictionary_popitem.asp) | Removes the last inserted key-value pair |
| [setdefault()](https://www.w3schools.com/python/ref_dictionary_setdefault.asp) | Returns the value of the specified key. If the key does not exist: insert the key, with the specified value |
| [update()](https://www.w3schools.com/python/ref_dictionary_update.asp) | Updates the dictionary with the specified key-value pairs |
| [values()](https://www.w3schools.com/python/ref_dictionary_values.asp) | Returns a list of all the values in the dictionary |

Learn more about dictionaries in our [Python Dictionaries Tutorial](https://www.w3schools.com/python/python_dictionaries.asp).

# Python Tuple Methods

Python has two built-in methods that you can use on tuples.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [count()](https://www.w3schools.com/python/ref_tuple_count.asp) | Returns the number of times a specified value occurs in a tuple |
| [index()](https://www.w3schools.com/python/ref_tuple_index.asp) | Searches the tuple for a specified value and returns the position of where it was found |

Learn more about tuples in our [Python Tuples Tutorial](https://www.w3schools.com/python/python_tuples.asp).

# Python Set Methods

Python has a set of built-in methods that you can use on sets.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [add()](https://www.w3schools.com/python/ref_set_add.asp) | Adds an element to the set |
| [clear()](https://www.w3schools.com/python/ref_set_clear.asp) | Removes all the elements from the set |
| [copy()](https://www.w3schools.com/python/ref_set_copy.asp) | Returns a copy of the set |
| [difference()](https://www.w3schools.com/python/ref_set_difference.asp) | Returns a set containing the difference between two or more sets |
| [difference\_update()](https://www.w3schools.com/python/ref_set_difference_update.asp) | Removes the items in this set that are also included in another, specified set |
| [discard()](https://www.w3schools.com/python/ref_set_discard.asp) | Remove the specified item |
| [intersection()](https://www.w3schools.com/python/ref_set_intersection.asp) | Returns a set, that is the intersection of two or more sets |
| [intersection\_update()](https://www.w3schools.com/python/ref_set_intersection_update.asp) | Removes the items in this set that are not present in other, specified set(s) |
| [isdisjoint()](https://www.w3schools.com/python/ref_set_isdisjoint.asp) | Returns whether two sets have a intersection or not |
| [issubset()](https://www.w3schools.com/python/ref_set_issubset.asp) | Returns whether another set contains this set or not |
| [issuperset()](https://www.w3schools.com/python/ref_set_issuperset.asp) | Returns whether this set contains another set or not |
| [pop()](https://www.w3schools.com/python/ref_set_pop.asp) | Removes an element from the set |
| [remove()](https://www.w3schools.com/python/ref_set_remove.asp) | Removes the specified element |
| [symmetric\_difference()](https://www.w3schools.com/python/ref_set_symmetric_difference.asp) | Returns a set with the symmetric differences of two sets |
| [symmetric\_difference\_update()](https://www.w3schools.com/python/ref_set_symmetric_difference_update.asp) | inserts the symmetric differences from this set and another |
| [union()](https://www.w3schools.com/python/ref_set_union.asp) | Return a set containing the union of sets |
| [update()](https://www.w3schools.com/python/ref_set_update.asp) | Update the set with another set, or any other iterable |

Learn more about sets in our [Python Sets Tutorial](https://www.w3schools.com/python/python_sets.asp).

# Python File Methods

Python has a set of methods available for the file object.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [close()](https://www.w3schools.com/python/ref_file_close.asp) | Closes the file |
| detach() | Returns the separated raw stream from the buffer |
| [fileno()](https://www.w3schools.com/python/ref_file_fileno.asp) | Returns a number that represents the stream, from the operating system's perspective |
| [flush()](https://www.w3schools.com/python/ref_file_flush.asp) | Flushes the internal buffer |
| [isatty()](https://www.w3schools.com/python/ref_file_isatty.asp) | Returns whether the file stream is interactive or not |
| [read()](https://www.w3schools.com/python/ref_file_read.asp) | Returns the file content |
| [readable()](https://www.w3schools.com/python/ref_file_readable.asp) | Returns whether the file stream can be read or not |
| [readline()](https://www.w3schools.com/python/ref_file_readline.asp) | Returns one line from the file |
| [readlines()](https://www.w3schools.com/python/ref_file_readlines.asp) | Returns a list of lines from the file |
| [seek()](https://www.w3schools.com/python/ref_file_seek.asp) | Change the file position |
| [seekable()](https://www.w3schools.com/python/ref_file_seekable.asp) | Returns whether the file allows us to change the file position |
| [tell()](https://www.w3schools.com/python/ref_file_tell.asp) | Returns the current file position |
| [truncate()](https://www.w3schools.com/python/ref_file_truncate.asp) | Resizes the file to a specified size |
| [writable()](https://www.w3schools.com/python/ref_file_writable.asp) | Returns whether the file can be written to or not |
| [write()](https://www.w3schools.com/python/ref_file_write.asp) | Writes the specified string to the file |
| [writelines()](https://www.w3schools.com/python/ref_file_writelines.asp) | Writes a list of strings to the file |

Learn more about the file object in our [Python File Handling Tutorial](https://www.w3schools.com/python/python_file_handling.asp).

# Python Keywords

Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers:

|  |  |
| --- | --- |
| **Keyword** | **Description** |
| [and](https://www.w3schools.com/python/ref_keyword_and.asp) | A logical operator |
| [as](https://www.w3schools.com/python/ref_keyword_as.asp) | To create an alias |
| [assert](https://www.w3schools.com/python/ref_keyword_assert.asp) | For debugging |
| [break](https://www.w3schools.com/python/ref_keyword_break.asp) | To break out of a loop |
| [class](https://www.w3schools.com/python/ref_keyword_class.asp) | To define a class |
| [continue](https://www.w3schools.com/python/ref_keyword_continue.asp) | To continue to the next iteration of a loop |
| [def](https://www.w3schools.com/python/ref_keyword_def.asp) | To define a function |
| [del](https://www.w3schools.com/python/ref_keyword_del.asp) | To delete an object |
| [elif](https://www.w3schools.com/python/ref_keyword_elif.asp) | Used in conditional statements, same as else if |
| [else](https://www.w3schools.com/python/ref_keyword_else.asp) | Used in conditional statements |
| [except](https://www.w3schools.com/python/ref_keyword_except.asp) | Used with exceptions, what to do when an exception occurs |
| [False](https://www.w3schools.com/python/ref_keyword_false.asp) | Boolean value, result of comparison operations |
| [finally](https://www.w3schools.com/python/ref_keyword_finally.asp) | Used with exceptions, a block of code that will be executed no matter if there is an exception or not |
| [for](https://www.w3schools.com/python/ref_keyword_for.asp) | To create a for loop |
| [from](https://www.w3schools.com/python/ref_keyword_from.asp) | To import specific parts of a module |
| [global](https://www.w3schools.com/python/ref_keyword_global.asp) | To declare a global variable |
| [if](https://www.w3schools.com/python/ref_keyword_if.asp) | To make a conditional statement |
| [import](https://www.w3schools.com/python/ref_keyword_import.asp) | To import a module |
| [in](https://www.w3schools.com/python/ref_keyword_in.asp) | To check if a value is present in a list, tuple, etc. |
| [is](https://www.w3schools.com/python/ref_keyword_is.asp) | To test if two variables are equal |
| [lambda](https://www.w3schools.com/python/ref_keyword_lambda.asp) | To create an anonymous function |
| [None](https://www.w3schools.com/python/ref_keyword_none.asp) | Represents a null value |
| [nonlocal](https://www.w3schools.com/python/ref_keyword_nonlocal.asp) | To declare a non-local variable |
| [not](https://www.w3schools.com/python/ref_keyword_not.asp) | A logical operator |
| [or](https://www.w3schools.com/python/ref_keyword_or.asp) | A logical operator |
| [pass](https://www.w3schools.com/python/ref_keyword_pass.asp) | A null statement, a statement that will do nothing |
| [raise](https://www.w3schools.com/python/ref_keyword_raise.asp) | To raise an exception |
| [return](https://www.w3schools.com/python/ref_keyword_return.asp) | To exit a function and return a value |
| [True](https://www.w3schools.com/python/ref_keyword_true.asp) | Boolean value, result of comparison operations |
| [try](https://www.w3schools.com/python/ref_keyword_try.asp) | To make a try...except statement |
| [while](https://www.w3schools.com/python/ref_keyword_while.asp) | To create a while loop |
| with | Used to simplify exception handling |
| yield | To end a function, returns a generator |

# Python Built-in Exceptions

## Built-in Exceptions

The table below shows built-in exceptions that are usually raised in Python:

|  |  |
| --- | --- |
| **Exception** | **Description** |
| ArithmeticError | Raised when an error occurs in numeric calculations |
| AssertionError | Raised when an assert statement fails |
| AttributeError | Raised when attribute reference or assignment fails |
| Exception | Base class for all exceptions |
| EOFError | Raised when the input() method hits an "end of file" condition (EOF) |
| FloatingPointError | Raised when a floating point calculation fails |
| GeneratorExit | Raised when a generator is closed (with the close() method) |
| ImportError | Raised when an imported module does not exist |
| IndentationError | Raised when indendation is not correct |
| IndexError | Raised when an index of a sequence does not exist |
| KeyError | Raised when a key does not exist in a dictionary |
| KeyboardInterrupt | Raised when the user presses Ctrl+c, Ctrl+z or Delete |
| LookupError | Raised when errors raised cant be found |
| MemoryError | Raised when a program runs out of memory |
| NameError | Raised when a variable does not exist |
| NotImplementedError | Raised when an abstract method requires an inherited class to override the method |
| OSError | Raised when a system related operation causes an error |
| OverflowError | Raised when the result of a numeric calculation is too large |
| ReferenceError | Raised when a weak reference object does not exist |
| RuntimeError | Raised when an error occurs that do not belong to any specific expections |
| StopIteration | Raised when the next() method of an iterator has no further values |
| SyntaxError | Raised when a syntax error occurs |
| TabError | Raised when indentation consists of tabs or spaces |
| SystemError | Raised when a system error occurs |
| SystemExit | Raised when the sys.exit() function is called |
| TypeError | Raised when two different types are combined |
| UnboundLocalError | Raised when a local variable is referenced before assignment |
| UnicodeError | Raised when a unicode problem occurs |
| UnicodeEncodeError | Raised when a unicode encoding problem occurs |
| UnicodeDecodeError | Raised when a unicode decoding problem occurs |
| UnicodeTranslateError | Raised when a unicode translation problem occurs |
| ValueError | Raised when there is a wrong value in a specified data type |
| ZeroDivisionError | Raised when the second operator in a division is zero |

# Python Glossary

This is a list of all the features explained in the Python Tutorial.



|  |  |
| --- | --- |
| **Feature** | **Description** |
| [Indentation](https://www.w3schools.com/python/gloss_python_indentation.asp) | Indentation refers to the spaces at the beginning of a code line |
| [Comments](https://www.w3schools.com/python/gloss_python_comments.asp) | Comments are code lines that will not be executed |
| [Multi Line Comments](https://www.w3schools.com/python/gloss_python_multi_line_comments.asp) | How to insert comments on multiple lines |
| [Creating Variables](https://www.w3schools.com/python/gloss_python_creating_variables.asp) | Variables are containers for storing data values |
| [Variable Names](https://www.w3schools.com/python/gloss_python_variable_names.asp) | How to name your variables |
| [Assign Values to Multiple Variables](https://www.w3schools.com/python/gloss_python_assign_value_to_multiple_variables.asp) | How to assign values to multiple variables |
| [Output Variables](https://www.w3schools.com/python/gloss_python_variable_output.asp) | Use the print statement to output variables |
| [String Concatenation](https://www.w3schools.com/python/gloss_python_string_concatenation.asp) | How to combine strings |
| [Global Variables](https://www.w3schools.com/python/gloss_python_global_variables.asp) | Global variables are variables that belongs to the global scope |
| [Built-In Data Types](https://www.w3schools.com/python/gloss_python_built-in_data_types.asp) | Python has a set of built-in data types |
| [Getting Data Type](https://www.w3schools.com/python/gloss_python_getting_data_type.asp) | How to get the data type of an object |
| [Setting Data Type](https://www.w3schools.com/python/gloss_python_setting_data_type.asp) | How to set the data type of an object |
| [Numbers](https://www.w3schools.com/python/gloss_python_numbers.asp) | There are three numeric types in Python |
| [Int](https://www.w3schools.com/python/gloss_python_int.asp) | The integer number type |
| [Float](https://www.w3schools.com/python/gloss_python_float.asp) | The floating number type |
| [Complex](https://www.w3schools.com/python/gloss_python_complex.asp) | The complex number type |
| [Type Conversion](https://www.w3schools.com/python/gloss_python_type_conversion.asp) | How to convert from one number type to another |
| [Random Number](https://www.w3schools.com/python/gloss_python_random_number.asp) | How to create a random number |
| [Specify a Variable Type](https://www.w3schools.com/python/gloss_python_specify_variable_type.asp) | How to specify a certain data type for a variable |
| [String Literals](https://www.w3schools.com/python/gloss_python_string_literals.asp) | How to create string literals |
| [Assigning a String to a Variable](https://www.w3schools.com/python/gloss_python_assign_string_variable.asp) | How to assign a string value to a variable |
| [Multiline Strings](https://www.w3schools.com/python/gloss_python_multi_line_strings.asp) | How to create a multi line string |
| [Strings are Arrays](https://www.w3schools.com/python/gloss_python_strings_are_arrays.asp) | Strings in Python are arrays of bytes representing Unicode characters |
| [Slicing a String](https://www.w3schools.com/python/gloss_python_string_slice.asp) | How to slice a string |
| [Negative Indexing on a String](https://www.w3schools.com/python/gloss_python_string_negative_indexing.asp) | How to use negative indexing when accessing a string |
| [String Length](https://www.w3schools.com/python/gloss_python_string_length.asp) | How to get the length of a string |
| [Check In String](https://www.w3schools.com/python/gloss_python_check_string.asp) | How to check if a string contains a specified phrase |
| [Format String](https://www.w3schools.com/python/gloss_python_string_format.asp) | How to combine two strings |
| [Escape Characters](https://www.w3schools.com/python/gloss_python_escape_characters.asp) | How to use escape characters |
| [Boolean Values](https://www.w3schools.com/python/gloss_python_boolean_values.asp) | True or False |
| [Evaluate Booleans](https://www.w3schools.com/python/gloss_python_evaluate_boolean_values.asp) | Evaluate a value or statement and return either True or False |
| [Return Boolean Value](https://www.w3schools.com/python/gloss_python_return_boolean.asp) | Functions that return a Boolean value |
| [Operators](https://www.w3schools.com/python/gloss_python_operators.asp) | Use operator to perform operations in Python |
| [Arithmetic Operators](https://www.w3schools.com/python/gloss_python_arithmetic_operators.asp) | Arithmetic operator are used to perform common mathematical operations |
| [Assignment Operators](https://www.w3schools.com/python/gloss_python_assignment_operators.asp) | Assignment operators are use to assign values to variables |
| [Comparison Operators](https://www.w3schools.com/python/gloss_python_comparison_operators.asp) | Comparison operators are used to compare two values |
| [Logical Operators](https://www.w3schools.com/python/gloss_python_logical_operators.asp) | Logical operators are used to combine conditional statements |
| [Identity Operators](https://www.w3schools.com/python/gloss_python_identity_operators.asp) | Identity operators are used to see if two objects are in fact the same object |
| [Membership Operators](https://www.w3schools.com/python/gloss_python_membership_operators.asp) | Membership operators are used to test is a sequence is present in an object |
| [Bitwise Operators](https://www.w3schools.com/python/gloss_python_bitwise_operators.asp) | Bitwise operators are used to compare (binary) numbers |
| [Lists](https://www.w3schools.com/python/gloss_python_lists.asp) | A list is an ordered, and changeable, collection |
| [Access List Items](https://www.w3schools.com/python/gloss_python_access_list_items.asp) | How to access items in a list |
| [Change List Item](https://www.w3schools.com/python/gloss_python_change_list_item.asp) | How to change the value of a list item |
| [Loop Through List Items](https://www.w3schools.com/python/gloss_python_loop_list_items.asp) | How to loop through the items in a list |
| [List Comprehension](https://www.w3schools.com/python/gloss_python_list_comprehension.asp) | How use a list comprehensive |
| [Check if List Item Exists](https://www.w3schools.com/python/gloss_python_check_if_list_item_exists.asp) | How to check if a specified item is present in a list |
| [List Length](https://www.w3schools.com/python/gloss_python_list_length.asp) | How to determine the length of a list |
| [Add List Items](https://www.w3schools.com/python/gloss_python_add_list_items.asp) | How to add items to a list |
| [Remove List Items](https://www.w3schools.com/python/gloss_python_remove_list_items.asp) | How to remove list items |
| [Copy a List](https://www.w3schools.com/python/gloss_python_copy_list.asp) | How to copy a list |
| [Join Two Lists](https://www.w3schools.com/python/gloss_python_join_lists.asp) | How to join two lists |
| [Tuple](https://www.w3schools.com/python/gloss_python_tuple.asp) | A tuple is an ordered, and unchangeable, collection |
| [Access Tuple Items](https://www.w3schools.com/python/gloss_python_access_tuple_items.asp) | How to access items in a tuple |
| [Change Tuple Item](https://www.w3schools.com/python/gloss_python_change_tuple_item.asp) | How to change the value of a tuple item |
| [Loop List Items](https://www.w3schools.com/python/gloss_python_loop_tuple_items.asp) | How to loop through the items in a tuple |
| [Check if Tuple Item Exists](https://www.w3schools.com/python/gloss_python_check_if_tuple_item_exists.asp) | How to check if a specified item is present in a tuple |
| [Tuple Length](https://www.w3schools.com/python/gloss_python_tuple_length.asp) | How to determine the length of a tuple |
| [Tuple With One Item](https://www.w3schools.com/python/gloss_python_tuple_one_item.asp) | How to create a tuple with only one item |
| [Remove Tuple Items](https://www.w3schools.com/python/gloss_python_remove_tuple_items.asp) | How to remove tuple items |
| [Join Two Tuples](https://www.w3schools.com/python/gloss_python_join_tuple.asp) | How to join two tuples |
| [Set](https://www.w3schools.com/python/gloss_python_set.asp) | A set is an unordered, and unchangeable, collection |
| [Access Set Items](https://www.w3schools.com/python/gloss_python_access_set_items.asp) | How to access items in a set |
| [Add Set Items](https://www.w3schools.com/python/gloss_python_add_set_items.asp) | How to add items to a set |
| [Loop Set Items](https://www.w3schools.com/python/gloss_python_loop_set_items.asp) | How to loop through the items in a set |
| [Check if Set Item Exists](https://www.w3schools.com/python/gloss_python_check_if_set_item_exists.asp) | How to check if a item exists |
| [Set Length](https://www.w3schools.com/python/gloss_python_set_length.asp) | How to determine the length of a set |
| [Remove Set Items](https://www.w3schools.com/python/gloss_python_remove_set_items.asp) | How to remove set items |
| [Join Two Sets](https://www.w3schools.com/python/gloss_python_join_sets.asp) | How to join two sets |
| [Dictionary](https://www.w3schools.com/python/gloss_python_dictionary.asp) | A dictionary is an unordered, and changeable, collection |
| [Access Dictionary Items](https://www.w3schools.com/python/gloss_python_access_dictionary_items.asp) | How to access items in a dictionary |
| [Change Dictionary Item](https://www.w3schools.com/python/gloss_python_change_dictionary_item.asp) | How to change the value of a dictionary item |
| [Loop Dictionary Items](https://www.w3schools.com/python/gloss_python_loop_dictionary_items.asp) | How to loop through the items in a tuple |
| [Check if Dictionary Item Exists](https://www.w3schools.com/python/gloss_python_check_if_dictionary_item_exists.asp) | How to check if a specified item is present in a dictionary |
| [Dictionary Length](https://www.w3schools.com/python/gloss_python_dictionary_length.asp) | How to determine the length of a dictionary |
| [Add Dictionary Item](https://www.w3schools.com/python/gloss_python_dictionary_add_item.asp) | How to add an item to a dictionary |
| [Remove Dictionary Items](https://www.w3schools.com/python/gloss_python_remove_dictionary_items.asp) | How to remove dictionary items |
| [Copy Dictionary](https://www.w3schools.com/python/gloss_python_copy_dictionary.asp) | How to copy a dictionary |
| [Nested Dictionaries](https://www.w3schools.com/python/gloss_python_nested_dictionaries.asp) | A dictionary within a dictionary |
| [If Statement](https://www.w3schools.com/python/gloss_python_if_statement.asp) | How to write an if statement |
| [If Indentation](https://www.w3schools.com/python/gloss_python_if_indentation.asp) | If statemnts in Python relies on indentation (whitespace at the beginning of a line) |
| [Elif](https://www.w3schools.com/python/gloss_python_elif.asp) | elif is the same as "else if" in other programming languages |
| [Else](https://www.w3schools.com/python/gloss_python_else.asp) | How to write an if...else statement |
| [Shorthand If](https://www.w3schools.com/python/gloss_python_if_shorthand.asp) | How to write an if statement in one line |
| [Shorthand If Else](https://www.w3schools.com/python/gloss_python_if_else_shorthand.asp) | How to write an if...else statement in one line |
| [If AND](https://www.w3schools.com/python/gloss_python_if_and.asp) | Use the and keyword to combine if statements |
| [If OR](https://www.w3schools.com/python/gloss_python_if_or.asp) | Use the or keyword to combine if statements |
| [Nested If](https://www.w3schools.com/python/gloss_python_if_nested.asp) | How to write an if statement inside an if statement |
| [The pass Keyword in If](https://www.w3schools.com/python/gloss_python_if_pass.asp) | Use the pass keyword inside empty if statements |
| [While](https://www.w3schools.com/python/gloss_python_while.asp) | How to write a while loop |
| [While Break](https://www.w3schools.com/python/gloss_python_while_break.asp) | How to break a while loop |
| [While Continue](https://www.w3schools.com/python/gloss_python_while_continue.asp) | How to stop the current iteration and continue wit the next |
| [While Else](https://www.w3schools.com/python/gloss_python_while_else.asp) | How to use an else statement in a while loop |
| [For](https://www.w3schools.com/python/gloss_python_for.asp) | How to write a for loop |
| [Loop Through a String](https://www.w3schools.com/python/gloss_python_for_string.asp) | How to loop through a string |
| [For Break](https://www.w3schools.com/python/gloss_python_for_break.asp) | How to break a for loop |
| [For Continue](https://www.w3schools.com/python/gloss_python_for_continue.asp) | How to stop the current iteration and continue wit the next |
| [Looping Through a rangee](https://www.w3schools.com/python/gloss_python_for_range.asp) | How to loop through a range of values |
| [For Else](https://www.w3schools.com/python/gloss_python_for_else.asp) | How to use an else statement in a for loop |
| [Nested Loops](https://www.w3schools.com/python/gloss_python_for_nested.asp) | How to write a loop inside a loop |
| [For pass](https://www.w3schools.com/python/gloss_python_for_pass.asp) | Use the pass keyword inside empty for loops |
| [Function](https://www.w3schools.com/python/gloss_python_function.asp) | How to create a function in Python |
| [Call a Function](https://www.w3schools.com/python/gloss_python_function_call.asp) | How to call a function in Python |
| [Function Arguments](https://www.w3schools.com/python/gloss_python_function_arguments.asp) | How to use arguments in a function |
| [\*args](https://www.w3schools.com/python/gloss_python_function_arbitrary_arguments.asp) | To deal with an unknown number of arguments in a function, use the \* symbol before the parameter name |
| [Keyword Arguments](https://www.w3schools.com/python/gloss_python_function_keyword_arguments.asp) | How to use keyword arguments in a function |
| [\*\*kwargs](https://www.w3schools.com/python/gloss_python_function_arbitrary_keyword_arguments.asp) | To deal with an unknown number of keyword arguments in a function, use the \* symbol before the parameter name |
| [Default Parameter Value](https://www.w3schools.com/python/gloss_python_function_default_parameter.asp) | How to use a default parameter value |
| [Passing a List as an Argument](https://www.w3schools.com/python/gloss_python_function_passing_list.asp) | How to pass a list as an argument |
| [Function Return Value](https://www.w3schools.com/python/gloss_python_function_return_value.asp) | How to return a value from a function |
| [The pass Statement i Functions](https://www.w3schools.com/python/gloss_python_function_pass.asp) | Use the pass statement in empty functions |
| [Function Recursion](https://www.w3schools.com/python/gloss_python_function_recursion.asp) | Functions that can call itself is called recursive functions |
| [Lambda Function](https://www.w3schools.com/python/gloss_python_lambda.asp) | How to create anonymous functions in Python |
| [Why Use Lambda Functions](https://www.w3schools.com/python/gloss_python_lambda_why.asp) | Learn when to use a lambda function or not |
| [Array](https://www.w3schools.com/python/gloss_python_array.asp) | Lists can be used as Arrays |
| [What is an Array](https://www.w3schools.com/python/gloss_python_arrray_what_is.asp) | Arrays are variables that can hold more than one value |
| [Access Arrays](https://www.w3schools.com/python/gloss_python_array_access.asp) | How to access array items |
| [Array Length](https://www.w3schools.com/python/gloss_python_array_length.asp) | How to get the length of an array |
| [Looping Array Elements](https://www.w3schools.com/python/gloss_python_array_loop.asp) | How to loop through array elements |
| [Add Array Element](https://www.w3schools.com/python/gloss_python_array_add.asp) | How to add elements from an array |
| [Remove Array Element](https://www.w3schools.com/python/gloss_python_array_remove.asp) | How to remove elements from an array |
| [Array Methods](https://www.w3schools.com/python/gloss_python_array_methods.asp) | Python has a set of Array/Lists methods |
| [Class](https://www.w3schools.com/python/gloss_python_class.asp) | A class is like an object constructor |
| [Create Class](https://www.w3schools.com/python/gloss_python_class_create.asp) | How to create a class |
| [The Class \_\_init\_\_() Function](https://www.w3schools.com/python/gloss_python_class_init.asp) | The \_\_init\_\_() function is executed when the class is initiated |
| [Object Methods](https://www.w3schools.com/python/gloss_python_object_methods.asp) | Methods in objects are functions that belongs to the object |
| [self](https://www.w3schools.com/python/gloss_python_self.asp) | The self parameter refers to the current instance of the class |
| [Modify Object Properties](https://www.w3schools.com/python/gloss_python_object_modify_properties.asp) | How to modify properties of an object |
| [Delete Object Properties](https://www.w3schools.com/python/gloss_python_object_delete_properties.asp) | How to modify properties of an object |
| [Delete Object](https://www.w3schools.com/python/gloss_python_object_delete.asp) | How to delete an object |
| [Class pass Statement](https://www.w3schools.com/python/gloss_python_class_pass.asp) | Use the pass statement in empty classes |
| [Create Parent Class](https://www.w3schools.com/python/gloss_python_parent_class.asp) | How to create a parent class |
| [Create Child Class](https://www.w3schools.com/python/gloss_python_child_class.asp) | How to create a child class |
| [Create the \_\_init\_\_() Function](https://www.w3schools.com/python/gloss_python_create_init.asp) | How to create the \_\_init\_\_() function |
| [super Function](https://www.w3schools.com/python/gloss_python_super.asp) | The super() function make the child class inherit the parent class |
| [Add Class Properties](https://www.w3schools.com/python/gloss_python_class_add_properties.asp) | How to add a property to a class |
| [Add Class Methods](https://www.w3schools.com/python/gloss_python_class_add_methods.asp) | How to add a method to a class |
| [Iterators](https://www.w3schools.com/python/gloss_python_iterators.asp) | An iterator is an object that contains a countable number of values |
| [Iterator vs Iterable](https://www.w3schools.com/python/gloss_python_iterator_vs_iterable.asp) | What is the difference between an iterator and an iterable |
| [Loop Through an Iterator](https://www.w3schools.com/python/gloss_python_iterator_loop.asp) | How to loop through the elements of an iterator |
| [Create an Iterator](https://www.w3schools.com/python/gloss_python_iterator_create.asp) | How to create an iterator |
| [StopIteration](https://www.w3schools.com/python/gloss_python_iterator_stop.asp) | How to stop an iterator |
| [Global Scope](https://www.w3schools.com/python/gloss_python_global_scope.asp) | When does a variable belong to the global scope? |
| [Global Keyword](https://www.w3schools.com/python/gloss_python_global_keyword.asp) | The global keyword makes the variable global |
| [Create a Module](https://www.w3schools.com/python/gloss_python_module_create.asp) | How to create a module |
| [Variables in Modules](https://www.w3schools.com/python/gloss_python_module_variables.asp) | How to use variables in a module |
| [Renaming a Module](https://www.w3schools.com/python/gloss_python_module_rename.asp) | How to rename a module |
| [Built-in Modules](https://www.w3schools.com/python/gloss_python_module_built-in.asp) | How to import built-in modules |
| [Using the dir() Function](https://www.w3schools.com/python/gloss_python_module_dir.asp) | List all variable names and function names in a module |
| [Import From Module](https://www.w3schools.com/python/gloss_python_module_import_from.asp) | How to import only parts from a module |
| [Datetime Module](https://www.w3schools.com/python/gloss_python_date.asp) | How to work with dates in Python |
| [Date Output](https://www.w3schools.com/python/gloss_python_date_output.asp) | How to output a date |
| [Create a Date Object](https://www.w3schools.com/python/gloss_python_date_create.asp) | How to create a date object |
| [The strftime Method](https://www.w3schools.com/python/gloss_python_date_strftime.asp) | How to format a date object into a readable string |
| [Date Format Codes](https://www.w3schools.com/python/gloss_python_date_format_codes.asp) | The datetime module has a set of legal format codes |
| [JSON](https://www.w3schools.com/python/gloss_python_json.asp) | How to work with JSON in Python |
| [Parse JSON](https://www.w3schools.com/python/gloss_python_json_parse.asp) | How to parse JSON code in Python |
| [Convert into JSON](https://www.w3schools.com/python/gloss_python_convert_into_JSON.asp) | How to convert a Python object in to JSON |
| [Format JSON](https://www.w3schools.com/python/gloss_python_format_json.asp) | How to format JSON output with indentations and line breaks |
| [Sort JSON](https://www.w3schools.com/python/gloss_python_json_sort.asp) | How to sort JSON |
| [RegEx Module](https://www.w3schools.com/python/gloss_python_regex.asp) | How to import the regex module |
| [RegEx Functions](https://www.w3schools.com/python/gloss_python_regex_functions.asp) | The re module has a set of functions |
| [Metacharacters in RegEx](https://www.w3schools.com/python/gloss_python_regex_metacharacters.asp) | Metacharacters are characters with a special meaning |
| [RegEx Special Sequences](https://www.w3schools.com/python/gloss_python_regex_sequences.asp) | A backslash followed by a a character has a special meaning |
| [RegEx Sets](https://www.w3schools.com/python/gloss_python_regex_sets.asp) | A set is a set of characters inside a pair of square brackets with a special meaning |
| [RegEx Match Object](https://www.w3schools.com/python/gloss_python_regex_match.asp) | The Match Object is an object containing information about the search and the result |
| [Install PIP](https://www.w3schools.com/python/gloss_python_pip_install.asp) | How to install PIP |
| [PIP Packages](https://www.w3schools.com/python/gloss_python_pip_packages.asp) | How to download and install a package with PIP |
| [PIP Remove Package](https://www.w3schools.com/python/gloss_python_pip_packages_remove.asp) | How to remove a package with PIP |
| [Error Handling](https://www.w3schools.com/python/gloss_python_error_handling.asp) | How to handle errors in Python |
| [Handle Many Exceptions](https://www.w3schools.com/python/gloss_python_try_except.asp) | How to handle more than one exception |
| [Try Else](https://www.w3schools.com/python/gloss_python_try_else.asp) | How to use the else keyword in a try statement |
| [Try Finally](https://www.w3schools.com/python/gloss_python_try_finally.asp) | How to use the finally keyword in a try statement |
| [raise](https://www.w3schools.com/python/gloss_python_raise.asp) | How to raise an exception in Python |

# Python Random Module

Python has a built-in module that you can use to make random numbers.

The random module has a set of methods:

|  |  |
| --- | --- |
| **Method** | **Description** |
| [seed()](https://www.w3schools.com/python/ref_random_seed.asp) | Initialize the random number generator |
| [getstate()](https://www.w3schools.com/python/ref_random_getstate.asp) | Returns the current internal state of the random number generator |
| [setstate()](https://www.w3schools.com/python/ref_random_setstate.asp) | Restores the internal state of the random number generator |
| [getrandbits()](https://www.w3schools.com/python/ref_random_getrandbits.asp) | Returns a number representing the random bits |
| [randrange()](https://www.w3schools.com/python/ref_random_randrange.asp) | Returns a random number between the given range |
| [randint()](https://www.w3schools.com/python/ref_random_randint.asp) | Returns a random number between the given range |
| [choice()](https://www.w3schools.com/python/ref_random_choice.asp) | Returns a random element from the given sequence |
| [choices()](https://www.w3schools.com/python/ref_random_choices.asp) | Returns a list with a random selection from the given sequence |
| [shuffle()](https://www.w3schools.com/python/ref_random_shuffle.asp) | Takes a sequence and returns the sequence in a random order |
| [sample()](https://www.w3schools.com/python/ref_random_sample.asp) | Returns a given sample of a sequence |
| [random()](https://www.w3schools.com/python/ref_random_random.asp) | Returns a random float number between 0 and 1 |
| [uniform()](https://www.w3schools.com/python/ref_random_uniform.asp) | Returns a random float number between two given parameters |
| [triangular()](https://www.w3schools.com/python/ref_random_triangular.asp) | Returns a random float number between two given parameters, you can also set a mode parameter to specify the midpoint between the two other parameters |
| betavariate() | Returns a random float number between 0 and 1 based on the Beta distribution (used in statistics) |
| expovariate() | Returns a random float number based on the Exponential distribution (used in statistics) |
| gammavariate() | Returns a random float number based on the Gamma distribution (used in statistics) |
| gauss() | Returns a random float number based on the Gaussian distribution (used in probability theories) |
| lognormvariate() | Returns a random float number based on a log-normal distribution (used in probability theories) |
| normalvariate() | Returns a random float number based on the normal distribution (used in probability theories) |
| vonmisesvariate() | Returns a random float number based on the von Mises distribution (used in directional statistics) |
| paretovariate() | Returns a random float number based on the Pareto distribution (used in probability theories) |
| weibullvariate() | Returns a random float number based on the Weibull distribution (used in statistics) |

# Python Requests Module

### Example

Make a request to a web page, and print the response text:

import requests  
  
x = requests.get('https://w3schools.com/python/demopage.htm')  
  
print(x.text)

[Run Example »](https://www.w3schools.com/python/showpython.asp?filename=demo_requests_get_url)

## Definition and Usage

The requests module allows you to send HTTP requests using Python.

The HTTP request returns a [Response Object](https://www.w3schools.com/python/ref_requests_response.asp) with all the response data (content, encoding, status, etc).

## Download and Install the Requests Module

Navigate your command line to the location of PIP, and type the following:

C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>pip install requests

Syntax

requests.methodname(params)

## Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| [delete(url, args)](https://www.w3schools.com/python/ref_requests_delete.asp) | Sends a DELETE request to the specified url |
| [get(url, params, args)](https://www.w3schools.com/python/ref_requests_get.asp) | Sends a GET request to the specified url |
| [head(url, args)](https://www.w3schools.com/python/ref_requests_head.asp) | Sends a HEAD request to the specified url |
| patch(url, data, args) | Sends a PATCH request to the specified url |
| [post(url, data, json, args)](https://www.w3schools.com/python/ref_requests_post.asp) | Sends a POST request to the specified url |
| put(url, data, args) | Sends a PUT request to the specified url |
| request(method, url, args) | Sends a request of the specified method to the specified url |

# Python statistics Module

## Python statistics Module

Python has a built-in module that you can use to calculate mathematical statistics of numeric data.

The statistics module was new in Python 3.4.

## Statistics Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| [statistics.harmonic\_mean()](https://www.w3schools.com/python/ref_stat_harmonic_mean.asp) | Calculates the harmonic mean (central location) of the given data |
| [statistics.mean()](https://www.w3schools.com/python/ref_stat_mean.asp) | Calculates the mean (average) of the given data |
| [statistics.median()](https://www.w3schools.com/python/ref_stat_median.asp) | Calculates the median (middle value) of the given data |
| [statistics.median\_grouped()](https://www.w3schools.com/python/ref_stat_median_grouped.asp) | Calculates the median of grouped continuous data |
| [statistics.median\_high()](https://www.w3schools.com/python/ref_stat_median_high.asp) | Calculates the high median of the given data |
| [statistics.median\_low()](https://www.w3schools.com/python/ref_stat_median_low.asp) | Calculates the low median of the given data |
| [statistics.mode()](https://www.w3schools.com/python/ref_stat_mode.asp) | Calculates the mode (central tendency) of the given numeric or nominal data |
| [statistics.pstdev()](https://www.w3schools.com/python/ref_stat_pstdev.asp) | Calculates the standard deviation from an entire population |
| [statistics.stdev()](https://www.w3schools.com/python/ref_stat_stdev.asp) | Calculates the standard deviation from a sample of data |
| [statistics.pvariance()](https://www.w3schools.com/python/ref_stat_pvariance.asp) | Calculates the variance of an entire population |
| [statistics.variance()](https://www.w3schools.com/python/ref_stat_variance.asp) | Calculates the variance from a sample of data |

# Python math Module

## Python math Module

Python has a built-in module that you can use for mathematical tasks.

The math module has a set of methods and constants.

## Math Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| [math.acos()](https://www.w3schools.com/python/ref_math_acos.asp) | Returns the arc cosine of a number |
| [math.acosh()](https://www.w3schools.com/python/ref_math_acosh.asp) | Returns the inverse hyperbolic cosine of a number |
| [math.asin()](https://www.w3schools.com/python/ref_math_asin.asp) | Returns the arc sine of a number |
| [math.asinh()](https://www.w3schools.com/python/ref_math_asinh.asp) | Returns the inverse hyperbolic sine of a number |
| [math.atan()](https://www.w3schools.com/python/ref_math_atan.asp) | Returns the arc tangent of a number in radians |
| [math.atan2()](https://www.w3schools.com/python/ref_math_atan2.asp) | Returns the arc tangent of y/x in radians |
| [math.atanh()](https://www.w3schools.com/python/ref_math_atanh.asp) | Returns the inverse hyperbolic tangent of a number |
| [math.ceil()](https://www.w3schools.com/python/ref_math_ceil.asp) | Rounds a number up to the nearest integer |
| [math.comb()](https://www.w3schools.com/python/ref_math_comb.asp) | Returns the number of ways to choose k items from n items without repetition and order |
| [math.copysign()](https://www.w3schools.com/python/ref_math_copysign.asp) | Returns a float consisting of the value of the first parameter and the sign of the second parameter |
| [math.cos()](https://www.w3schools.com/python/ref_math_cos.asp) | Returns the cosine of a number |
| [math.cosh()](https://www.w3schools.com/python/ref_math_cosh.asp) | Returns the hyperbolic cosine of a number |
| [math.degrees()](https://www.w3schools.com/python/ref_math_degrees.asp) | Converts an angle from radians to degrees |
| [math.dist()](https://www.w3schools.com/python/ref_math_dist.asp) | Returns the Euclidean distance between two points (p and q), where p and q are the coordinates of that point |
| [math.erf()](https://www.w3schools.com/python/ref_math_erf.asp) | Returns the error function of a number |
| [math.erfc()](https://www.w3schools.com/python/ref_math_erfc.asp) | Returns the complementary error function of a number |
| [math.exp()](https://www.w3schools.com/python/ref_math_exp.asp) | Returns E raised to the power of x |
| [math.expm1()](https://www.w3schools.com/python/ref_math_expm1.asp) | Returns Ex - 1 |
| [math.fabs()](https://www.w3schools.com/python/ref_math_fabs.asp) | Returns the absolute value of a number |
| [math.factorial()](https://www.w3schools.com/python/ref_math_factorial.asp) | Returns the factorial of a number |
| [math.floor()](https://www.w3schools.com/python/ref_math_floor.asp) | Rounds a number down to the nearest integer |
| [math.fmod()](https://www.w3schools.com/python/ref_math_fmod.asp) | Returns the remainder of x/y |
| [math.frexp()](https://www.w3schools.com/python/ref_math_frexp.asp) | Returns the mantissa and the exponent, of a specified number |
| [math.fsum()](https://www.w3schools.com/python/ref_math_fsum.asp) | Returns the sum of all items in any iterable (tuples, arrays, lists, etc.) |
| [math.gamma()](https://www.w3schools.com/python/ref_math_gamma.asp) | Returns the gamma function at x |
| [math.gcd()](https://www.w3schools.com/python/ref_math_gcd.asp) | Returns the greatest common divisor of two integers |
| [math.hypot()](https://www.w3schools.com/python/ref_math_hypot.asp) | Returns the Euclidean norm |
| [math.isclose()](https://www.w3schools.com/python/ref_math_isclose.asp) | Checks whether two values are close to each other, or not |
| [math.isfinite()](https://www.w3schools.com/python/ref_math_isfinite.asp) | Checks whether a number is finite or not |
| [math.isinf()](https://www.w3schools.com/python/ref_math_isinf.asp) | Checks whether a number is infinite or not |
| [math.isnan()](https://www.w3schools.com/python/ref_math_isnan.asp) | Checks whether a value is NaN (not a number) or not |
| [math.isqrt()](https://www.w3schools.com/python/ref_math_isqrt.asp) | Rounds a square root number downwards to the nearest integer |
| [math.ldexp()](https://www.w3schools.com/python/ref_math_ldexp.asp) | Returns the inverse of [math.frexp()](https://www.w3schools.com/python/ref_math_frexp.asp) which is x \* (2\*\*i) of the given numbers x and i |
| [math.lgamma()](https://www.w3schools.com/python/ref_math_lgamma.asp) | Returns the log gamma value of x |
| [math.log()](https://www.w3schools.com/python/ref_math_log.asp) | Returns the natural logarithm of a number, or the logarithm of number to base |
| [math.log10()](https://www.w3schools.com/python/ref_math_log10.asp) | Returns the base-10 logarithm of x |
| [math.log1p()](https://www.w3schools.com/python/ref_math_log1p.asp) | Returns the natural logarithm of 1+x |
| [math.log2()](https://www.w3schools.com/python/ref_math_log2.asp) | Returns the base-2 logarithm of x |
| [math.perm()](https://www.w3schools.com/python/ref_math_perm.asp) | Returns the number of ways to choose k items from n items with order and without repetition |
| [math.pow()](https://www.w3schools.com/python/ref_math_pow.asp) | Returns the value of x to the power of y |
| [math.prod()](https://www.w3schools.com/python/ref_math_prod.asp) | Returns the product of all the elements in an iterable |
| [math.radians()](https://www.w3schools.com/python/ref_math_radians.asp) | Converts a degree value into radians |
| [math.remainder()](https://www.w3schools.com/python/ref_math_remainder.asp) | Returns the closest value that can make numerator completely divisible by the denominator |
| [math.sin()](https://www.w3schools.com/python/ref_math_sin.asp) | Returns the sine of a number |
| [math.sinh()](https://www.w3schools.com/python/ref_math_sinh.asp) | Returns the hyperbolic sine of a number |
| [math.sqrt()](https://www.w3schools.com/python/ref_math_sqrt.asp) | Returns the square root of a number |
| [math.tan()](https://www.w3schools.com/python/ref_math_tan.asp) | Returns the tangent of a number |
| [math.tanh()](https://www.w3schools.com/python/ref_math_tanh.asp) | Returns the hyperbolic tangent of a number |
| [math.trunc()](https://www.w3schools.com/python/ref_math_trunc.asp) | Returns the truncated integer parts of a number |

**Math Constants**

|  |  |
| --- | --- |
| **Constant** | **Description** |
| [math.e](https://www.w3schools.com/python/ref_math_e.asp) | Returns Euler's number (2.7182...) |
| [math.inf](https://www.w3schools.com/python/ref_math_inf.asp) | Returns a floating-point positive infinity |
| [math.nan](https://www.w3schools.com/python/ref_math_nan.asp) | Returns a floating-point NaN (Not a Number) value |
| [math.pi](https://www.w3schools.com/python/ref_math_pi.asp) | Returns PI (3.1415...) |
| [math.tau](https://www.w3schools.com/python/ref_math_tau.asp) | Returns tau (6.2831...) |

# Python cmath Module

## Python cmath Module

Python has a built-in module that you can use for mathematical tasks for complex numbers.

The methods in this module accepts int, float, and complex numbers. It even accepts Python objects that has a \_\_complex\_\_() or \_\_float\_\_() method.

The methods in this module almost always return a complex number. If the return value can be expressed as a real number, the return value has an imaginary part of 0.

The cmath module has a set of methods and constants.

## cMath Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| [cmath.acos(x)](https://www.w3schools.com/python/ref_cmath_acos.asp) | Returns the arc cosine value of x |
| [cmath.acosh(x)](https://www.w3schools.com/python/ref_cmath_acosh.asp) | Returns the hyperbolic arc cosine of x |
| [cmath.asin(x)](https://www.w3schools.com/python/ref_cmath_asin.asp) | Returns the arc sine of x |
| [cmath.asinh(x)](https://www.w3schools.com/python/ref_cmath_asinh.asp) | Returns the hyperbolic arc sine of x |
| [cmath.atan(x)](https://www.w3schools.com/python/ref_cmath_atan.asp) | Returns the arc tangent value of x |
| [cmath.atanh(x)](https://www.w3schools.com/python/ref_cmath_atanh.asp) | Returns the hyperbolic arctangent value of x |
| [cmath.cos(x)](https://www.w3schools.com/python/ref_cmath_cos.asp) | Returns the cosine of x |
| [cmath.cosh(x)](https://www.w3schools.com/python/ref_cmath_cosh.asp) | Returns the hyperbolic cosine of x |
| [cmath.exp(x)](https://www.w3schools.com/python/ref_cmath_exp.asp) | Returns the value of Ex, where E is Euler's number (approximately 2.718281...), and x is the number passed to it |
| [cmath.isclose()](https://www.w3schools.com/python/ref_cmath_isclose.asp) | Checks whether two values are close, or not |
| [cmath.isfinite(x)](https://www.w3schools.com/python/ref_cmath_isfinite.asp) | Checks whether x is a finite number |
| [cmath.isinf(x)](https://www.w3schools.com/python/ref_cmath_isinf.asp) | Check whether x is a positive or negative infinty |
| [cmath.isnan(x)](https://www.w3schools.com/python/ref_cmath_isnan.asp) | Checks whether x is NaN (not a number) |
| [cmath.log(x[, base])](https://www.w3schools.com/python/ref_cmath_log.asp) | Returns the logarithm of x to the base |
| [cmath.log10(x)](https://www.w3schools.com/python/ref_cmath_log10.asp) | Returns the base-10 logarithm of x |
| [cmath.phase()](https://www.w3schools.com/python/ref_cmath_phase.asp) | Return the phase of a complex number |
| [cmath.polar()](https://www.w3schools.com/python/ref_cmath_polar.asp) | Convert a complex number to polar coordinates |
| [cmath.rect()](https://www.w3schools.com/python/ref_cmath_rect.asp) | Convert polar coordinates to rectangular form |
| [cmath.sin(x)](https://www.w3schools.com/python/ref_cmath_sin.asp) | Returns the sine of x |
| [cmath.sinh(x)](https://www.w3schools.com/python/ref_cmath_sinh.asp) | Returns the hyperbolic sine of x |
| [cmath.sqrt(x)](https://www.w3schools.com/python/ref_cmath_sqrt.asp) | Returns the square root of x |
| [cmath.tan(x)](https://www.w3schools.com/python/ref_cmath_tan.asp) | Returns the tangent of x |
| [cmath.tanh(x)](https://www.w3schools.com/python/ref_cmath_tanh.asp) | Returns the hyperbolic tangent of x |

**cMath Constants**

|  |  |
| --- | --- |
| **Constant** | **Description** |
| [cmath.e](https://www.w3schools.com/python/ref_cmath_e.asp) | Returns Euler's number (2.7182...) |
| [cmath.inf](https://www.w3schools.com/python/ref_cmath_inf.asp) | Returns a floating-point positive infinity value |
| [cmath.infj](https://www.w3schools.com/python/ref_cmath_infj.asp) | Returns a complex infinity value |
| [cmath.nan](https://www.w3schools.com/python/ref_cmath_nan.asp) | Returns floating-point NaN (Not a Number) value |
| [cmath.nanj](https://www.w3schools.com/python/ref_cmath_nanj.asp) | Returns coplext NaN (Not a Number) value |
| [cmath.pi](https://www.w3schools.com/python/ref_cmath_pi.asp) | Returns PI (3.1415...) |
| [cmath.tau](https://www.w3schools.com/python/ref_cmath_tau.asp) | Returns tau (6.2831...) |

# How to Remove Duplicates From a Python List

Learn how to remove duplicates from a List in Python.

### Example

Remove any duplicates from a List:

mylist = ["a", "b", "a", "c", "c"]  
mylist = list(dict.fromkeys(mylist))  
print(mylist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_howto_remove_duplicates)

## Example Explained

First we have a List that contains duplicates:

### A List with Duplicates

**mylist = ["a", "b", "a", "c", "c"]**mylist = list(dict.fromkeys(mylist))  
print(mylist)

Create a dictionary, using the List items as keys. This will automatically remove any duplicates because dictionaries cannot have duplicate keys.

### Create a Dictionary

mylist = ["a", "b", "a", "c", "c"]  
mylist = list( **dict.fromkeys(mylist)**)  
print(mylist)

Then, convert the dictionary back into a list:

### Convert Into a List

mylist = ["a", "b", "a", "c", "c"]  
**mylist = list(**dict.fromkeys(mylist) **)**  
print(mylist)

Now we have a List without any duplicates, and it has the same order as the original List.

Print the List to demonstrate the result

### Print the List

mylist = ["a", "b", "a", "c", "c"]  
mylist = list(dict.fromkeys(mylist))  
**print(mylist)**

## Create a Function

If you like to have a function where you can send your lists, and get them back without duplicates, you can create a function and insert the code from the example above.

### Example

def my\_function(x):  
  return list(dict.fromkeys(x))  
  
mylist = my\_function(["a", "b", "a", "c", "c"])  
print(mylist)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_howto_remove_duplicates2)

## Example Explained

Create a function that takes a List as an argument.

### Create a Function

**def my\_function(x):**  
  return list(dict.fromkeys(x))  
  
mylist = my\_function(["a", "b", "a", "c", "c"])  
print(mylist)

Create a dictionary, using this List items as keys.

### Create a Dictionary

def my\_function(x):  
  return list( **dict.fromkeys(x)**)  
  
mylist = my\_function(["a", "b", "a", "c", "c"])  
print(mylist)

Convert the dictionary into a list.

### Convert Into a List

def my\_function(x):  
  return **list(**dict.fromkeys(x) **)**  
  
mylist = my\_function(["a", "b", "a", "c", "c"])  
print(mylist)

Return the list

### Return List

def my\_function(x):  
  **return**list(dict.fromkeys(x))  
  
mylist = my\_function(["a", "b", "a", "c", "c"])  
print(mylist)

Call the function, with a list as a parameter:

### Call the Function

def my\_function(x):  
  return list(dict.fromkeys(x))  
  
**mylist = my\_function(["a", "b", "a", "c", "c"])**print(mylist)

Print the result:

### Print the Result

def my\_function(x):  
  return list(dict.fromkeys(x))  
  
mylist = my\_function(["a", "b", "a", "c", "c"])  
**print(mylist)**

# How to Reverse a String in Python

Learn how to reverse a String in Python.

There is no built-in function to reverse a String in Python.

The fastest (and easiest?) way is to use a slice that steps backwards, -1.

### Example

Reverse the string "Hello World":

txt = "Hello World"[::-1]  
print(txt)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_howto_reverse_string)

## Example Explained

We have a string, "Hello World", which we want to reverse:

### The String to Reverse

txt = **"Hello World"**[::-1]  
print(txt)

Create a slice that starts at the end of the string, and moves backwards.

In this particular example, the slice statement [::-1] means start at the end of the string and end at position 0, move with the step -1, negative one, which means one step backwards.

### Slice the String

txt = "Hello World" **[::-1]**  
print(txt)

Now we have a string txt that reads "Hello World" backwards.

Print the String to demonstrate the result

### Print the List

txt = "Hello World"[::-1]  
**print(txt)**

## Create a Function

If you like to have a function where you can send your strings, and return them backwards, you can create a function and insert the code from the example above.

### Example

def my\_function(x):  
  return x[::-1]  
  
mytxt = my\_function("I wonder how this text looks like backwards")  
print(mytxt)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_howto_reverse_string2)

## Example Explained

Create a function that takes a String as an argument.

### Create a Function

**def my\_function(x):**  
  return x[::-1]  
  
mytxt = my\_function("I wonder how this text looks like backwards")  
print(mytxt)

Slice the string starting at the end of the string and move backwards.

### Slice the String

def my\_function(x):  
  return x **[::-1]**  
  
mytxt = my\_function("I wonder how this text looks like backwards")  
print(mytxt)

Return the backward String

### Return the String

def my\_function(x):  
  **return**x[::-1]  
  
mytxt = my\_function("I wonder how this text looks like backwards")  
print(mytxt )

Call the function, with a string as a parameter:

### Call the Function

def my\_function(x):  
  return x[::-1]  
  
**mytxt = my\_function("I wonder how this text looks like backwards")**print(mytxt)

Print the result:

### Print the Result

def my\_function(x):  
  return x[::-1]  
  
mytxt = my\_function("I wonder how this text looks like backwards")  
**print(mytxt)**

# How to Add Two Numbers in Python

Learn how to add two numbers in Python.

Use the + operator to add two numbers:

### Example

x = 5  
y = 10  
print(x + y)

[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_howto_add_two_numbers)

## Add Two Numbers with User Input

In this example, the user must input two numbers. Then we print the sum by calculating (adding) the two numbers:

### Example

x = input("Type a number: ")  
y = input("Type another number: ")  
  
sum = int(x) + int(y)  
  
print("The sum is: ", sum)

[Try it Yourself »](https://www.w3schools.com/python/showpython.asp?filename=demo_howto_add_two_numbers2)

# Python Examples

## Python Syntax

[Print "Hello World"](https://www.w3schools.com/python/trypython.asp?filename=demo_default)

print("Hello, World!")

Output:

Hello, World!

[Comments in Python](https://www.w3schools.com/python/trypython.asp?filename=demo_comment)

#This is a comment.

print("Hello, World!")

Output:

Hello, World!

[Docstrings](https://www.w3schools.com/python/trypython.asp?filename=demo_docstring)

"""This is a

multiline docstring."""

print("Hello, World!")

Output:

Hello, World!

[Syntax Explained](https://www.w3schools.com/python/python_syntax.asp)

## Python Variables

[Create a variable](https://www.w3schools.com/python/trypython.asp?filename=demo_variables1)

x = 5

y = "John"

print(x)

print(y)

Output:

5  
John

[Output both text and a variable](https://www.w3schools.com/python/trypython.asp?filename=demo_variables3)

x = "Python"

y = "is"

z = "awesome"

print(x, y, z)

Output:

Python is awesome

[Add a variable to another variable](https://www.w3schools.com/python/trypython.asp?filename=demo_variables4)

x = "Python "

y = "is "

z = "awesome"

print(x + y + z)

Output:

Python is awesome

[Variables Explained](https://www.w3schools.com/python/python_variables.asp)

## Python Numbers

[Verify the type of an object](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers)

x = 1

y = 2.8

z = 1j

print(type(x))

print(type(y))

print(type(z))

Output:

<class 'int'>  
<class 'float'>  
<class 'complex'>

[Create integers](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_int)

x = 1

y = 35656222554887711

z = -3255522

print(type(x))

print(type(y))

print(type(z))

Output:

<class 'int'>  
<class 'int'>  
<class 'int'>

[Create floating point numbers](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_float)

x = 1.10

y = 1.0

z = -35.59

print(type(x))

print(type(y))

print(type(z))

Output:

<class 'float'>  
<class 'float'>  
<class 'float'>

[Create scientific numbers with an "e" to indicate the power of 10](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_float2)

x = 35e3

y = 12E4

z = -87.7e100

print(type(x))

print(type(y))

print(type(z))

Output:

<class 'float'>  
<class 'float'>  
<class 'float'>

[Create complex numbers](https://www.w3schools.com/python/trypython.asp?filename=demo_numbers_complex)

x = 3+5j

y = 5j

z = -5j

print(type(x))

print(type(y))

print(type(z))

Output:

<class 'complex'>  
<class 'complex'>  
<class 'complex'>

[Numbers Explained](https://www.w3schools.com/python/python_numbers.asp)

## Python Casting

[Casting - Integers](https://www.w3schools.com/python/trypython.asp?filename=demo_casting_int)

x = int(1)

y = int(2.8)

z = int("3")

print(x)

print(y)

print(z)

Output:

1  
2  
3

[Casting - Floats](https://www.w3schools.com/python/trypython.asp?filename=demo_float)

x = float(1)

y = float(2.8)

z = float("3")

w = float("4.2")

print(x)

print(y)

print(z)

print(w)

Output:

1.0  
2.8  
3.0  
4.2

[Casting - Strings](https://www.w3schools.com/python/trypython.asp?filename=demo_string)

x = str("s1")

y = str(2)

z = str(3.0)

print(x)

print(y)

print(z)

Output:

s1  
2  
3.0

[Casting Explained](https://www.w3schools.com/python/python_casting.asp)

## Python Strings

[Get the character at position 1 of a string](https://www.w3schools.com/python/trypython.asp?filename=demo_string1)

a = "Hello, World!"

print(a[1])

Output:

e

[Substring. Get the characters from position 2 to position 5 (not included)](https://www.w3schools.com/python/trypython.asp?filename=demo_string2)

b = "Hello, World!"

print(b[2:5])

Output:

llo

[Remove whitespace from the beginning or at the end of a string](https://www.w3schools.com/python/trypython.asp?filename=demo_string_strip)

a = " Hello, World! "

print(a.strip())

Output:

Hello, World!

[Return the length of a string](https://www.w3schools.com/python/trypython.asp?filename=demo_string_len)

a = "Hello, World!"

print(len(a))

Output:

13

[Convert a string to lower case](https://www.w3schools.com/python/trypython.asp?filename=demo_string_lower)

a = "Hello, World!"

print(a.lower())

Output:

hello, world!

[Convert a string to upper case](https://www.w3schools.com/python/trypython.asp?filename=demo_string_upper)

a = "Hello, World!"

print(a.upper())

Output:

HELLO, WORLD!

[Replace a string with another string](https://www.w3schools.com/python/trypython.asp?filename=demo_string_replace)

a = "Hello, World!"

print(a.replace("H", "J"))

Output:

Jello, World!

[Split a string into substrings](https://www.w3schools.com/python/trypython.asp?filename=demo_string_split)

a = "Hello, World!"

b = a.split(",")

print(b)

Output:

['Hello', ' World!']

[Strings Explained](https://www.w3schools.com/python/python_strings.asp)

## Python Operators

[Addition operator](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_add)

x = 5

y = 3

print(x + y)

Output:

8

[Subtraction operator](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_sub)

x = 5

y = 3

print(x - y)

Output:

2

[Multiplication operator](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_mult)

x = 5

y = 3

print(x \* y)

Output:

15

[Division operator](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_div)

x = 12

y = 3

print(x / y)

Output:

4

[Modulus operator](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_mod)

x = 5

y = 2

print(x % y)

Output:

1

[Assignment operator](https://www.w3schools.com/python/trypython.asp?filename=demo_oper_ass1)

x = 5

print(x)

Output:

5

[Operators Explained](https://www.w3schools.com/python/python_operators.asp)

## Python Lists

[Create a list](https://www.w3schools.com/python/trypython.asp?filename=demo_list)

thislist = ["apple", "banana", "cherry"]

print(thislist)

Output:

['apple', 'banana', 'cherry']

[Access list items](https://www.w3schools.com/python/trypython.asp?filename=demo_list_access)

thislist = ["apple", "banana", "cherry"]

print(thislist[1])

Output:

banana

[Change the value of a list item](https://www.w3schools.com/python/trypython.asp?filename=demo_list_change)

thislist = ["apple", "banana", "cherry"]

thislist[1] = "blackcurrant"

print(thislist)

Output:

['apple', 'blackcurrant', 'cherry']

[Loop through a list](https://www.w3schools.com/python/trypython.asp?filename=demo_list_loop)

thislist = ["apple", "banana", "cherry"]

for x in thislist:

print(x)

Output:

apple  
banana  
cherry

[Check if a list item exists](https://www.w3schools.com/python/trypython.asp?filename=demo_list_in)

thislist = ["apple", "banana", "cherry"]

if "apple" in thislist:

print("Yes, 'apple' is in the fruits list")

Output:

Yes, 'apple' is in the fruits list

[Get the length of a list](https://www.w3schools.com/python/trypython.asp?filename=demo_list_len)

thislist = ["apple", "banana", "cherry"]

print(len(thislist))

Output:

3

[Add an item to the end of a list](https://www.w3schools.com/python/trypython.asp?filename=demo_list_append)

thislist = ["apple", "banana", "cherry"]

thislist.append("orange")

print(thislist)

Output:

['apple', 'banana', 'cherry', 'orange']

[Add an item at a specified index](https://www.w3schools.com/python/trypython.asp?filename=demo_list_insert)

thislist = ["apple", "banana", "cherry"]

thislist.insert(1, "orange")

print(thislist)

Output:

['apple', 'orange', 'banana', 'cherry']

[Remove an item](https://www.w3schools.com/python/trypython.asp?filename=demo_list_remove)

thislist = ["apple", "banana", "cherry"]

thislist.remove("banana")

print(thislist)

Output:

['apple', 'cherry']

[Remove the last item](https://www.w3schools.com/python/trypython.asp?filename=demo_list_pop)

thislist = ["apple", "banana", "cherry"]

thislist.pop()

print(thislist)

Output:

['apple', 'banana']

[Remove an item at a specified index](https://www.w3schools.com/python/trypython.asp?filename=demo_list_del)

thislist = ["apple", "banana", "cherry"]

thislist.insert(1, "orange")

print(thislist)

Output:

['apple', 'orange', 'banana', 'cherry']

[Empty a list](https://www.w3schools.com/python/trypython.asp?filename=demo_list_clear)

thislist = ["apple", "banana", "cherry"]

thislist.clear()

print(thislist)

Output:

[]

[Using the list() constructor to make a list](https://www.w3schools.com/python/trypython.asp?filename=demo_list_constructor)

thislist = list(("apple", "banana", "cherry"))

print(thislist)

Output:

['apple', 'banana', 'cherry']

[Lists Explained](https://www.w3schools.com/python/python_lists.asp)

## Python Tuples

[Create a tuple](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple)

thistuple = ("apple", "banana", "cherry")

print(thistuple)

Output:

('apple', 'banana', 'cherry')

[Access tuple items](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple1)

thistuple = ("apple", "banana", "cherry")

print(thistuple[1])

Output:

banana

[Change tuple values](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple2)

thistuple = ("apple", "banana", "cherry")

thistuple[1] = "blackcurrant"

# the value is still the same:

print(thistuple)

Output:

('apple', 'banana', 'cherry')

[Loop through a tuple](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_loop)

thistuple = ("apple", "banana", "cherry")

for x in thistuple:

print(x)

Output:

apple  
banana  
cherry

[Check if a tuple item exists](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_in)

thistuple = ("apple", "banana", "cherry")

if "apple" in thistuple:

print("Yes, 'apple' is in the fruits tuple")

Output:

Yes, 'apple' is in the fruits tuple

[Get the length of a tuple](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_length)

thistuple = ("apple", "banana", "cherry")

print(len(thistuple))

Output:

3

[Delete a tuple](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple_del)

thistuple = ("apple", "banana", "cherry")

del thistuple

print(thistuple) #this will raise an error because the tuple no longer exists

Output:

Traceback (most recent call last):  
  File "demo\_tuple\_del.py", line 3, in <module>  
    print(thistuple) #this will raise an error because the tuple no longer exists  
NameError: name 'thistuple' is not defined

[Using the tuple() constructor to create a tuple](https://www.w3schools.com/python/trypython.asp?filename=demo_tuple3)

thistuple = tuple(("apple", "banana", "cherry"))

print(thistuple)

Output:

('apple', 'banana', 'cherry')

[Tuples Explained](https://www.w3schools.com/python/python_tuples.asp)

## Python Sets

[Create a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set)[Loop through a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set_loop)

thisset = {"apple", "banana", "cherry"}

print(thisset)

# Note: the set list is unordered, meaning: the items will appear in a random order.

# Refresh this page to see the change in the result.

Output:

{'apple', 'banana', 'cherry'}

[Check if an item exists](https://www.w3schools.com/python/trypython.asp?filename=demo_set_in)

thisset = {"apple", "banana", "cherry"}

print("banana" in thisset)

Output:

True

[Add an item to a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set_add)

thisset = {"apple", "banana", "cherry"}

thisset.add("orange")

print(thisset)

Output:

{'orange', 'cherry', 'apple', 'banana'}

[Add multiple items to a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set_update)

thisset = {"apple", "banana", "cherry"}

thisset.update(["orange", "mango", "grapes"])

print(thisset)

Output:

{'mango', 'cherry', 'grapes', 'orange', 'apple', 'banana'}

[Get the length of a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set_length)

thisset = {"apple", "banana", "cherry"}

print(len(thisset))

Output:

3

[Remove an item in a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set_remove)

thisset = {"apple", "banana", "cherry"}

thisset.remove("banana")

print(thisset)

Output:

{'apple', 'cherry'}

[Remove an item in a set by using the discard() method](https://www.w3schools.com/python/trypython.asp?filename=demo_set_discard)

thisset = {"apple", "banana", "cherry"}

thisset.discard("banana")

print(thisset)

Output:

{'apple', 'cherry'}

[Remove the last item in a set by using the pop() method](https://www.w3schools.com/python/trypython.asp?filename=demo_set_pop)

thisset = {"apple", "banana", "cherry"}

x = thisset.pop()

print(x) #removed item

print(thisset) #the set after removal

Output:

cherry  
{'apple', 'banana'}

[Empty a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set_clear)

thisset = {"apple", "banana", "cherry"}

thisset.clear()

print(thisset)

Output:

set()

[Delete a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set_del)

thisset = {"apple", "banana", "cherry"}

del thisset

print(thisset) #this will raise an error because the set no longer exists

Output:

Traceback (most recent call last):  
  File "demo\_set\_del.py", line 5, in <module>  
    print(thisset) #this will raise an error because the set no longer exists  
NameError: name 'thisset' is not defined

[Using the set() constructor to create a set](https://www.w3schools.com/python/trypython.asp?filename=demo_set2)

thisset = set(("apple", "banana", "cherry"))

print(thisset)

# Note: the set list is unordered, so the result will display the items in a random order.

Output:

{'apple', 'cherry', 'banana'}

[Sets Explained](https://www.w3schools.com/python/python_sets.asp)

## Python Dictionaries

[Create a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

print(thisdict)

Output:

{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}

[Access the items of a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_access)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

x = thisdict["model"]

print(x)

Output:

Mustang

[Change the value of a specific item in a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_change)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

thisdict["year"] = 2018

print(thisdict)

Output:

{'brand': 'Ford', 'model': 'Mustang', 'year': 2018}

[Print all key names in a dictionary, one by one](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

for x in thisdict:

print(x)

Output:

brand  
model  
year

[Print all values in a dictionary, one by one](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop2)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

for x in thisdict:

print(thisdict[x])

Output:

Ford  
Mustang  
1964

[Using the values() function to return values of a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop_values)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

for x in thisdict.values():

print(x)

Output:

Ford  
Mustang  
1964

[Loop through both keys an values, by using the items() function](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_loop_items)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

for x, y in thisdict.items():

print(x, y)

Output:

brand Ford  
model Mustang  
year 1964

[Check if a key exists](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_in)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

if "model" in thisdict:

print("Yes, 'model' is one of the keys in the thisdict dictionary")

Output:

Yes, 'model' is one of the keys in the thisdict dictionary

[Get the length of a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_length)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

print(len(thisdict))

Output:

3

[Add an item to a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_add)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

thisdict["color"] = "red"

print(thisdict)

Output:

{'brand': 'Ford', 'model': 'Mustang', 'year': 1964, 'color': 'red'}

[Remove an item from a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_pop)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

thisdict.pop("model")

print(thisdict)

Output:

{'brand': 'Ford', 'year': 1964}

[Empty a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_clear)

thisdict = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

thisdict.clear()

print(thisdict)

Output:

{}

[Using the dict() constructor to create a dictionary](https://www.w3schools.com/python/trypython.asp?filename=demo_dictionary_dict)

thisdict = dict(brand="Ford", model="Mustang", year=1964)

# note that keywords are not string literals

# note the use of equals rather than colon for the assignment

print(thisdict)

Output:

{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}

[Dictionaries Explained](https://www.w3schools.com/python/python_dictionaries.asp)

## Python If ... Else

[The if statement](https://www.w3schools.com/python/trypython.asp?filename=demo_if2)

a = 33

b = 200

if b > a:

print("b is greater than a")

Output:

b is greater than a

[The elif statement](https://www.w3schools.com/python/trypython.asp?filename=demo_if_elif)

a = 33

b = 33

if b > a:

print("b is greater than a")

elif a == b:

print("a and b are equal")

Output:

a and b are equal

[The else statement](https://www.w3schools.com/python/trypython.asp?filename=demo_if_else)

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")

Output:

a is greater than b

[Short hand if](https://www.w3schools.com/python/trypython.asp?filename=demo_if_short)

a = 200

b = 33

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

Output:

"a is greater than b"

[Short hand if ... else](https://www.w3schools.com/python/trypython.asp?filename=demo_if_else_short)

a = 2

b = 330

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

Output:

B

[The and keyword](https://www.w3schools.com/python/trypython.asp?filename=demo_if_and)

a = 200

b = 33

c = 500

if a > b and c > a:

print("Both conditions are True")

Output:

Both conditions are True

[The or keyword](https://www.w3schools.com/python/trypython.asp?filename=demo_if_or)

a = 200

b = 33

c = 500

if a > b or a > c:

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

Output:

At least one of the conditions is True

[If ... Else Explained](https://www.w3schools.com/python/python_conditions.asp)

## Python While Loop

[The while loop](https://www.w3schools.com/python/trypython.asp?filename=demo_while)

i = 1

while i < 6:

print(i)

i += 1

Output:

1  
2  
3  
4  
5

[Using the break statement in a while loop](https://www.w3schools.com/python/trypython.asp?filename=demo_while_break)

i = 1

while i < 6:

print(i)

if (i == 3):

break

i += 1

Output:

1  
2  
3

[Using the continue statement in a while loop](https://www.w3schools.com/python/trypython.asp?filename=demo_while_continue)

i = 0

while i < 6:

i += 1

if i == 3:

continue

print(i)

# Note that number 3 is missing in the result

Output:

1  
2  
4  
5  
6

[While Loop Explained](https://www.w3schools.com/python/python_while_loops.asp)

## Python For Loop

[The for loop](https://www.w3schools.com/python/trypython.asp?filename=demo_for)

fruits = ["apple", "banana", "cherry"]

for x in fruits:

print(x)

Output:

apple  
banana  
cherry

[Loop through a string](https://www.w3schools.com/python/trypython.asp?filename=demo_for_string)

for x in "banana":

print(x)

Output:

b  
a  
n  
a  
n  
a

[Using the break statement in a for loop](https://www.w3schools.com/python/trypython.asp?filename=demo_for_break)

fruits = ["apple", "banana", "cherry"]

for x in fruits:

print(x)

if x == "banana":

break

Output:

apple  
banana

[Using the continue statement in a for loop](https://www.w3schools.com/python/trypython.asp?filename=demo_for_continue)

fruits = ["apple", "banana", "cherry"]

for x in fruits:

if x == "banana":

continue

print(x)

Output:

apple  
cherry

[Using the range() function in a for loop](https://www.w3schools.com/python/trypython.asp?filename=demo_for_range)

for x in range(6):

print(x)

Output:

0  
1  
2  
3  
4  
5

[Else in for loop](https://www.w3schools.com/python/trypython.asp?filename=demo_for_else)

for x in range(6):

print(x)

else:

print("Finally finished!")

Output:

0  
1  
2  
3  
4  
5  
Finally finished!

[Nested for loop](https://www.w3schools.com/python/trypython.asp?filename=demo_for_nested)

adj = ["red", "big", "tasty"]

fruits = ["apple", "banana", "cherry"]

for x in adj:

for y in fruits:

print(x, y)

Output:

red apple  
red banana  
red cherry  
big apple  
big banana  
big cherry  
tasty apple  
tasty banana  
tasty cherry

[For Loop Explained](https://www.w3schools.com/python/python_for_loops.asp)

## Python Functions

[Create and call a function](https://www.w3schools.com/python/trypython.asp?filename=demo_function)

def my\_function():

print("Hello from a function")

my\_function()

Output:

Hello from a function

[Function parameters](https://www.w3schools.com/python/trypython.asp?filename=demo_function_param)

def my\_function(fname):

print(fname + " Refsnes")

my\_function("Emil")

my\_function("Tobias")

my\_function("Linus")

Output:

Emil Refsnes  
Tobias Refsnes  
Linus Refsnes

[Default parameter value](https://www.w3schools.com/python/trypython.asp?filename=demo_function_param2)

def my\_function(country = "Norway"):

print("I am from " + country)

my\_function("Sweden")

my\_function("India")

my\_function()

my\_function("Brazil")

Output:

I am from Sweden  
I am from India  
I am from Norway  
I am from Brazil

[Let a function return a value](https://www.w3schools.com/python/trypython.asp?filename=demo_function_return)

def my\_function(x):

return 5 \* x

print(my\_function(3))

print(my\_function(5))

print(my\_function(9))

Output:

15  
25  
45

[Recursion](https://www.w3schools.com/python/trypython.asp?filename=demo_recursion)

def tri\_recursion(k):

if(k > 0):

result = k + tri\_recursion(k - 1)

print(result)

else:

result = 0

return result

print("\n\nRecursion Example Results")

tri\_recursion(6)

Output:

Recursion Example Results  
1  
3  
6  
10  
15  
21

[Functions Explained](https://www.w3schools.com/python/python_functions.asp)

## Python Lambda

[A lambda function that adds 10 to the number passed in as an argument](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda)

x = lambda a: a + 10

print(x(5))

Output:

15

[A lambda function that multiplies argument a with argument b](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda2)

x = lambda a, b: a \* b

print(x(5, 6))

Output:

30

[A lambda function that sums argument a, b, and c](https://www.w3schools.com/python/trypython.asp?filename=demo_lambda3)

x = lambda a, b, c: a + b + c

print(x(5, 6, 2))

Output:

13

[Lambda Explained](https://www.w3schools.com/python/python_lambda.asp)

## Python Arrays

[Create an array](https://www.w3schools.com/python/trypython.asp?filename=demo_array1)

cars = ["Ford", "Volvo", "BMW"]

print(cars)

Output:

['Ford', 'Volvo', 'BMW']

[Access the elements of an array](https://www.w3schools.com/python/trypython.asp?filename=demo_array2)

cars = ["Ford", "Volvo", "BMW"]

x = cars[0]

print(x)

Output:

Ford

[Change the value of an array element](https://www.w3schools.com/python/trypython.asp?filename=demo_array3)

cars = ["Ford", "Volvo", "BMW"]

cars[0] = "Toyota"

print(cars)

Output:

['Toyota', 'Volvo', 'BMW']

[Get the length of an array](https://www.w3schools.com/python/trypython.asp?filename=demo_array4)

cars = ["Ford", "Volvo", "BMW"]

x = len(cars)

print(x)

Output:

3

[Loop through all elements of an array](https://www.w3schools.com/python/trypython.asp?filename=demo_array5)

cars = ["Ford", "Volvo", "BMW"]

for x in cars:

print(x)

Output:

Ford  
Volvo  
BMW

[Add an element to an array](https://www.w3schools.com/python/trypython.asp?filename=demo_array6)

cars = ["Ford", "Volvo", "BMW"]

cars.append("Honda")

print(cars)

Output:

['Ford', 'Volvo', 'BMW', 'Honda']

[Remove an element from an array](https://www.w3schools.com/python/trypython.asp?filename=demo_array7)

cars = ["Ford", "Volvo", "BMW"]

cars.pop(1)

print(cars)

Output:

['Ford', 'BMW']

[Arrays Explained](https://www.w3schools.com/python/python_arrays.asp)

## Python Classes and Objects

[Create a class](https://www.w3schools.com/python/trypython.asp?filename=demo_class1)

class MyClass:

x = 5

print(MyClass)

Output:

<class '\_\_main\_\_.MyClass'>

[Create an object](https://www.w3schools.com/python/trypython.asp?filename=demo_class2)

class MyClass:

x = 5

p1 = MyClass()

print(p1.x)

Output:

5

[The \_\_init()\_\_ Function](https://www.w3schools.com/python/trypython.asp?filename=demo_class3)

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

p1 = Person("John", 36)

print(p1.name)

print(p1.age)

Output:

John  
36

[Create object methods](https://www.w3schools.com/python/trypython.asp?filename=demo_class4)

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def myfunc(self):

print("Hello my name is " + self.name)

p1 = Person("John", 36)

p1.myfunc()

Output:

Hello my name is John

[The self parameter](https://www.w3schools.com/python/trypython.asp?filename=demo_class5)

class Person:

def \_\_init\_\_(mysillyobject, name, age):

mysillyobject.name = name

mysillyobject.age = age

def myfunc(abc):

print("Hello my name is " + abc.name)

p1 = Person("John", 36)

p1.myfunc()

Output:

Hello my name is John

[Modify object properties](https://www.w3schools.com/python/trypython.asp?filename=demo_class6)

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def myfunc(self):

print("Hello my name is " + self.name)

p1 = Person("John", 36)

p1.age = 40

print(p1.age)

Output:

40

[Delete object properties](https://www.w3schools.com/python/trypython.asp?filename=demo_class7)

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def myfunc(self):

print("Hello my name is " + self.name)

p1 = Person("John", 36)

del p1.age

print(p1.age)

Output:

Traceback (most recent call last):  
  File "demo\_class7.py", line 13, in <module>  
    print(p1.age)  
AttributeError: 'Person' object has no attribute 'age'

[Delete an object](https://www.w3schools.com/python/trypython.asp?filename=demo_class8)

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def myfunc(self):

print("Hello my name is " + self.name)

p1 = Person("John", 36)

del p1

print(p1)

Output:

Traceback (most recent call last):  
  File "demo\_class8.py", line 13, in <module>  
    print(p1)  
NameError: 'p1' is not defined

[Classes/Objects Explained](https://www.w3schools.com/python/python_classes.asp)

## Python Iterators

[Return an iterator from a tuple](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator)

mytuple = ("apple", "banana", "cherry")

myit = iter(mytuple)

print(next(myit))

print(next(myit))

print(next(myit))

Output:

apple  
banana  
cherry

[Return an iterator from a string](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator2)

mystr = "banana"

myit = iter(mystr)

print(next(myit))

print(next(myit))

print(next(myit))

print(next(myit))

print(next(myit))

print(next(myit))

Output:

b  
a  
n  
a  
n  
a

[Loop through an iterator](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator_loop)

mytuple = ("apple", "banana", "cherry")

for x in mytuple:

print(x)

Output:

apple  
banana  
cherry

[Create an iterator](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator_create)

class MyNumbers:

def \_\_iter\_\_(self):

self.a = 1

return self

def \_\_next\_\_(self):

x = self.a

self.a += 1

return x

myclass = MyNumbers()

myiter = iter(myclass)

print(next(myiter))

print(next(myiter))

print(next(myiter))

print(next(myiter))

print(next(myiter))

Output:

1  
2  
3  
4  
5

[Stop iteration](https://www.w3schools.com/python/trypython.asp?filename=demo_iterator_create2)

class MyNumbers:

def \_\_iter\_\_(self):

self.a = 1

return self

def \_\_next\_\_(self):

if self.a <= 20:

x = self.a

self.a += 1

return x

else:

raise StopIteration

myclass = MyNumbers()

myiter = iter(myclass)

for x in myiter:

print(x)

Output:

1  
2  
3  
4  
5

[Iterators Explained](https://www.w3schools.com/python/python_iterators.asp)

## Python Modules

[Use a module](https://www.w3schools.com/python/trypython.asp?filename=demo_module1)

import mymodule

mymodule.greeting("Jonathan")

Output:

Hello, Jonathan

[Variables in module](https://www.w3schools.com/python/trypython.asp?filename=demo_module2)

import mymodule

a = mymodule.person1["age"]

print(a)

Output:

36

[Re-naming a module](https://www.w3schools.com/python/trypython.asp?filename=demo_module3)

import mymodule as mx

a = mx.person1["age"]

print(a)

Output:

36

[Built-in modules](https://www.w3schools.com/python/trypython.asp?filename=demo_module4)

import platform

x = platform.system()

print(x)

Output:

Windows

[Using the dir() function](https://www.w3schools.com/python/trypython.asp?filename=demo_module5)

import platform

x = dir(platform)

print(x)

Output:

['DEV\_NULL', '\_UNIXCONFDIR', 'WIN32\_CLIENT\_RELEASES', 'WIN32\_SERVER\_RELEASES', '\_\_builtins\_\_', '\_\_cached\_\_', '\_\_copyright\_\_', '\_\_doc\_\_', '\_\_file\_\_', '\_\_loader\_\_', '\_\_name\_\_', '\_\_package \_\_', '\_\_spec\_\_', '\_\_version\_\_', '\_default\_architecture', '\_dist\_try\_harder', '\_follow\_symlinks', '\_ironpython26\_sys\_version\_parser', '\_ironpython\_sys\_version\_parser', '\_java\_getprop', '\_libc\_search', '\_linux\_distribution', '\_lsb\_release\_version', '\_mac\_ver\_xml', '\_node', '\_norm\_version', '\_perse\_release\_file', '\_platform', '\_platform\_cache', '\_pypy\_sys\_version\_parser', '\_release\_filename', '\_release\_version', '\_supported\_dists', '\_sys\_version', '\_sys\_version\_cache', '\_sys\_version\_parser', '\_syscmd\_file', '\_syscmd\_uname', '\_syscmd\_ver', '\_uname\_cache', '\_ver\_output', 'architecture', 'collections', 'dist', 'java\_ver', 'libc\_ver', 'linux\_distribution', 'mac\_ver', 'machine', 'node', 'os', 'platform', 'popen', 'processor', 'python\_branch', 'python\_build', 'python\_compiler', 'python\_implementation', 'python\_revision', 'python\_version', 'python\_version\_tuple', 're', 'release', 'subprocess', 'sys', 'system', 'system\_aliases', 'uname', 'uname\_result', 'version', 'warnings', 'win32\_ver']

[Import from module](https://www.w3schools.com/python/trypython.asp?filename=demo_module6)

from mymodule import person1

print(person1["age"])

Output:

36

[Modules Explained](https://www.w3schools.com/python/python_modules.asp)

## Python Dates

[Import the datetime module and display the current date](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime1)

import datetime

x = datetime.datetime.now()

print(x)

Output:

2022-10-07 11:24:48.807911

[Return the year and name of weekday](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime2)

import datetime

x = datetime.datetime.now()

print(x.year)

print(x.strftime("%A"))

Output:

2022  
Friday

[Create a date object](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime3)

import datetime

x = datetime.datetime(2020, 5, 17)

print(x)

Output:

2020-05-17 00:00:00

[The strftime() Method](https://www.w3schools.com/python/trypython.asp?filename=demo_datetime_strftime)

import datetime

x = datetime.datetime(2018, 6, 1)

print(x.strftime("%B"))

Output:

June

[Dates Explained](https://www.w3schools.com/python/python_datetime.asp)

## Python Math

[Find the lowest and highest value in an iterable](https://www.w3schools.com/python/trypython.asp?filename=demo_math_min_max)

x = min(5, 10, 25)

y = max(5, 10, 25)

print(x)

print(y)

Output:

5  
25

[Return the absolute value of a number](https://www.w3schools.com/python/trypython.asp?filename=demo_math_abs)

x = abs(-7.25)

print(x)

Output:

7.25

[Return the value of x to the power of y (xy)](https://www.w3schools.com/python/trypython.asp?filename=demo_math_pow)

x = pow(4, 3)

print(x)

Output:

64

[Return the square root of a number](https://www.w3schools.com/python/trypython.asp?filename=demo_math_sqrt)

import math

x = math.sqrt(64)

print(x)

Output:

8.0

[Round a number upwards and downwards to its nearest integer](https://www.w3schools.com/python/trypython.asp?filename=demo_math_ceil_floor)

#Import math library

import math

#Round a number upward to its nearest integer

x = math.ceil(1.4)

#Round a number downward to its nearest integer

y = math.floor(1.4)

print(x)

print(y)

Output:

2  
1

[Return the value of PI](https://www.w3schools.com/python/trypython.asp?filename=demo_math_pi)

import math

x = math.pi

print(x)

Output:

3.141592653589793

[Math Explained](https://www.w3schools.com/python/python_math.asp)

## Python JSON

[Convert from JSON to Python](https://www.w3schools.com/python/trypython.asp?filename=demo_json)

import json

# some JSON:

x = '{ "name":"John", "age":30, "city":"New York"}'

# parse x:

y = json.loads(x)

# the result is a Python dictionary:

print(y["age"])

Output:

30

[Convert from Python to JSON](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python)

import json

# a Python object (dict):

x = {

"name": "John",

"age": 30,

"city": "New York"

}

# convert into JSON:

y = json.dumps(x)

# the result is a JSON string:

print(y)

Output:

{"name": "John", "age": 30, "city": "New York"}

[Convert Python objects into JSON strings](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_all)

import json

print(json.dumps({"name": "John", "age": 30}))

print(json.dumps(["apple", "bananas"]))

print(json.dumps(("apple", "bananas")))

print(json.dumps("hello"))

print(json.dumps(42))

print(json.dumps(31.76))

print(json.dumps(True))

print(json.dumps(False))

print(json.dumps(None))

Output:

{"name": "John", "age": 30}  
["apple", "bananas"]  
["apple", "bananas"]  
"hello"  
42  
31.76  
true  
false  
null

[Convert a Python object containing all the legal data types](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_all_in_one)

import json

x = {

"name": "John",

"age": 30,

"married": True,

"divorced": False,

"children": ("Ann","Billy"),

"pets": None,

"cars": [

{"model": "BMW 230", "mpg": 27.5},

{"model": "Ford Edge", "mpg": 24.1}

]

}

# convert into JSON:

y = json.dumps(x)

# the result is a JSON string:

print(y)

Output:

{"name": "John", "age": 30, "married": true, "divorced": false, "children": ["Ann","Billy"], "pets": null, "cars": [{"model": "BMW 230", "mpg": 27.5}, {"model": "Ford Edge", "mpg": 24.1}]}

[Use the indent parameter to define the numbers of indents](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_indent)

import json

x = {

"name": "John",

"age": 30,

"married": True,

"divorced": False,

"children": ("Ann","Billy"),

"pets": None,

"cars": [

{"model": "BMW 230", "mpg": 27.5},

{"model": "Ford Edge", "mpg": 24.1}

]

}

# use four indents to make it easier to read the result:

print(json.dumps(x, indent=4))

Output:

{  
    "name": "John",  
    "age": 30,  
    "married": true,  
    "divorced": false,  
    "children": [  
        "Ann",  
        "Billy"  
    ],  
    "pets": null,  
    "cars": [  
        {  
            "model": "BMW 230",  
            "mpg": 27.5  
        },  
        {  
            "model": "Ford Edge",  
            "mpg": 24.1  
        }  
    ]  
}

[Use the separators parameter to change the default separator](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_separators)

import json

x = {

"name": "John",

"age": 30,

"married": True,

"divorced": False,

"children": ("Ann","Billy"),

"pets": None,

"cars": [

{"model": "BMW 230", "mpg": 27.5},

{"model": "Ford Edge", "mpg": 24.1}

]

}

# use . and a space to separate objects, and a space, a = and a space to separate keys from their values:

print(json.dumps(x, indent=4, separators=(". ", " = ")))

Output:

{  
    "name" = "John".  
    "age" = 30.  
    "married" = true.  
    "divorced" = false.  
    "children" = [  
        "Ann".  
        "Billy"  
    ].  
    "pets" = null.  
    "cars" = [  
        {  
            "model" = "BMW 230".  
            "mpg" = 27.5  
        }.  
        {  
            "model" = "Ford Edge".  
            "mpg" = 24.1  
        }  
    ]  
}

[Use the sort\_keys parameter to specify if the result should be sorted or not](https://www.w3schools.com/python/trypython.asp?filename=demo_json_from_python_sort_keys)

import json

x = {

"name": "John",

"age": 30,

"married": True,

"divorced": False,

"children": ("Ann","Billy"),

"pets": None,

"cars": [

{"model": "BMW 230", "mpg": 27.5},

{"model": "Ford Edge", "mpg": 24.1}

]

}

# sort the result alphabetically by keys:

print(json.dumps(x, indent=4, sort\_keys=True))

Output:

{  
    "age": 30,  
    "cars": [  
        {  
            "model": "BMW 230",  
            "mpg": 27.5  
        },  
        {  
            "model": "Ford Edge",  
            "mpg": 24.1  
        }  
    ],  
    "children": [  
        "Ann",  
        "Billy"  
    ],  
    "divorced": false,  
    "married": true,  
    "name": "John",  
    "pets": null  
}

[JSON Explained](https://www.w3schools.com/python/python_json.asp)

## Python RegEx

[Search a string to see if it starts with "The" and ends with "Spain"](https://www.w3schools.com/python/trypython.asp?filename=demo_regex)

import re

#Check if the string starts with "The" and ends with "Spain":

txt = "The rain in Spain"

x = re.search("^The.\*Spain$", txt)

if x:

print("YES! We have a match!")

else:

print("No match")

Output:

YES! We have a match!

[Using the findall() function](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_findall)

import re

#Return a list containing every occurrence of "ai":

txt = "The rain in Spain"

x = re.findall("ai", txt)

print(x)

Output:

['ai', 'ai']

[Using the search() function](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_search)

import re

txt = "The rain in Spain"

x = re.search("\s", txt)

print("The first white-space character is located in position:", x.start())

Output:

The first white-space character is located in position: 3

[Using the split() function](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_split)

import re

#Split the string at every white-space character:

txt = "The rain in Spain"

x = re.split("\s", txt)

print(x)

Output:

['The', 'rain', 'in', 'Spain']

[Using the sub() function](https://www.w3schools.com/python/trypython.asp?filename=demo_regex_sub)

import re

#Replace all white-space characters with the digit "9":

txt = "The rain in Spain"

x = re.sub("\s", "9", txt)

print(x)

Output:

The9rain9in9Spain

[RegEx Explained](https://www.w3schools.com/python/python_regex.asp)

## Python PIP

[Using a package](https://www.w3schools.com/python/trypython.asp?filename=demo_camelcase)

import camelcase

c = camelcase.CamelCase()

txt = "lorem ipsum dolor sit amet"

print(c.hump(txt))

#This method capitalizes the first letter of each word.

Output:

Lorem Ipsum Dolor Sit Amet

[PIP Explained](https://www.w3schools.com/python/python_pip.asp)

## Python Try Except

[When an error occurs, print a message](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except)

#The try block will generate an error, because x is not defined:

try:

print(x)

except:

print("An exception occurred")

Output:

An exception occurred

[Many exceptions](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except2)

#The try block will generate a NameError, because x is not defined:

try:

print(x)

except NameError:

print("Variable x is not defined")

except:

print("Something else went wrong")

Output:

Variable x is not defined

[Use the else keyword to define a block of code to be executed if no errors were raised](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except3)

#The try block does not raise any errors, so the else block is executed:

try:

print("Hello")

except:

print("Something went wrong")

else:

print("Nothing went wrong")

Output:

Hello  
Nothing went wrong

[Use the finally block to execute code regardless if the try block raises an error or not](https://www.w3schools.com/python/trypython.asp?filename=demo_try_except4)

#The finally block gets executed no matter if the try block raises any errors or not:

try:

print(x)

except:

print("Something went wrong")

finally:

print("The 'try except' is finished")

Output:

Something went wrong  
The 'try except' is finished

[Try Except Explained](https://www.w3schools.com/python/python_try_except.asp)

## Python File Handling

[Read a file](https://www.w3schools.com/python/trypython.asp?filename=demo_file_open)

f = open("demofile.txt", "r")

print(f.read())

Output:

Hello! Welcome to demofile.txt  
This file is for testing purposes.  
Good Luck!

[Read only parts of a file](https://www.w3schools.com/python/trypython.asp?filename=demo_file_open2)

f = open("demofile.txt", "r")

print(f.read(5))

Output:

Hello

[Read one line of a file](https://www.w3schools.com/python/trypython.asp?filename=demo_file_readline)

f = open("demofile.txt", "r")

print(f.readline())

Output:

Hello! Welcome to demofile.txt

[Loop through the lines of a file to read the whole file, line by line](https://www.w3schools.com/python/trypython.asp?filename=demo_file_readline3)

f = open("demofile.txt", "r")

for x in f:

print(x)

Output:

Hello! Welcome to demofile.txt  
This file is for testing purposes.  
Good Luck!

[File Handling Explained](https://www.w3schools.com/python/python_file_handling.asp)

## Python MySQL

[Create a connection to a database](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_connection)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword"

)

print(mydb)

Output:

<mysql.connector.connection.MySQLConnection object ar 0x016645F0>

[Create a database in MySQL](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_create_db)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword"

)

mycursor = mydb.cursor()

mycursor.execute("CREATE DATABASE mydatabase")

#If this page is executed with no error, you have successfully created a database.

Output:

[Check if a database exist](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_show_databases)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword"

)

mycursor = mydb.cursor()

mycursor.execute("SHOW DATABASES")

for x in mycursor:

print(x)

Output:

('information\_scheme',)  
('mydatabase',)  
('performance\_schema',)  
('sys',)

[Create a table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_create_table)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("CREATE TABLE customers (name VARCHAR(255), address VARCHAR(255))")

#If this page is executed with no error, you have successfully created a table named "customers".

Output:

[Check if a table exist](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_show_tables)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("SHOW TABLES")

for x in mycursor:

print(x)

Output:

('customers',)

[Create primary key when creating a table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_primary_key)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="yourusername",

password="yourpassword",

database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("CREATE TABLE customers (id INT AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(255), address VARCHAR(255))")

#If this page is executed with no error, the table "customers" now has a primary key

Output:

[Insert a record in a table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_insert)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"

val = ("John", "Highway 21")

mycursor.execute(sql, val)

mydb.commit()

print(mycursor.rowcount, "record inserted.")

Output:

1 record inserted.

[Insert multiple rows](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_insert_many)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"

val = [

('Peter', 'Lowstreet 4'),

('Amy', 'Apple st 652'),

('Hannah', 'Mountain 21'),

('Michael', 'Valley 345'),

('Sandy', 'Ocean blvd 2'),

('Betty', 'Green Grass 1'),

('Richard', 'Sky st 331'),

('Susan', 'One way 98'),

('Vicky', 'Yellow Garden 2'),

('Ben', 'Park Lane 38'),

('William', 'Central st 954'),

('Chuck', 'Main Road 989'),

('Viola', 'Sideway 1633')

]

mycursor.executemany(sql, val)

mydb.commit()

print(mycursor.rowcount, "record was inserted.")

Output:

13 record was inserted.

[Get inserted ID](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_insert_id)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"

val = ("Michelle", "Blue Village")

mycursor.execute(sql, val)

mydb.commit()

print("1 record inserted, ID:", mycursor.lastrowid)

Output:

1 record inserted, ID: 15

[Select all records from a table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_select)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("SELECT \* FROM customers")

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

(1, 'John', 'Highway 21')  
(2, 'Peter', 'Lowstreet 27')  
(3, 'Amy', 'Apple st 652')  
(4, 'Hannah', 'Mountain 21')  
(5, 'Michael', 'Valley 345')  
(6, 'Sandy', 'Ocean blvd 2')  
(7, 'Betty', 'Green Grass 1')  
(8, 'Richard', 'Sky st 331')  
(9, 'Susan', 'One way 98')  
(10, 'Vicky', 'Yellow Garden 2')  
(11, 'Ben', 'Park Lane 38')  
(12, 'William', 'Central st 954')  
(13, 'Chuck', 'Main Road 989')  
(14, 'Viola', 'Sideway 1633')  
(15, 'Michelle', 'Blue Village')

[Select only some of the columns in a table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_select_columns)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("SELECT name, address FROM customers")

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

('John', 'Highway 21')  
('Peter', 'Lowstreet 27')  
('Amy', 'Apple st 652')  
('Hannah', 'Mountain 21')  
('Michael', 'Valley 345')  
('Sandy', 'Ocean blvd 2')  
('Betty', 'Green Grass 1')  
('Richard', 'Sky st 331')  
('Susan', 'One way 98')  
('Vicky', 'Yellow Garden 2')  
('Ben', 'Park Lane 38')  
('William', 'Central st 954')  
('Chuck', 'Main Road 989')  
('Viola', 'Sideway 1633')  
('Michelle', 'Blue Village')

[Use the fetchone() method to fetch only one row in a table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_select_fetchone)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("SELECT \* FROM customers")

myresult = mycursor.fetchone()

print(myresult)

Output:

(1, 'John', 'Highway 21')

[Select with a filter](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_where)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \* FROM customers WHERE address = 'Park Lane 38'"

mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

(11, 'Ben', 'Park Lane 38')

[Wildcards characters](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_where_wildcard)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \* FROM customers WHERE address Like '%way%'"

mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

(1, 'John', 'Highway 21')  
(9, 'Susan', 'One way 98')  
(14, 'Viola', 'Sideway 1633')

[Prevent SQL injection](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_where_escape)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \* FROM customers WHERE address = %s"

adr = ("Yellow Garden 2", )

mycursor.execute(sql, adr)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

(10, 'Vicky', 'Yellow Garden 2')

[Sort the result of a table alphabetically](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_orderby)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \* FROM customers ORDER BY name"

mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

(3, 'Amy', 'Apple st 652')  
(11, 'Ben', 'Park Lane 38')  
(7, 'Betty', 'Green Grass 1')  
(13, 'Chuck', 'Main Road 989')  
(4, 'Hannah', 'Mountain 21')  
(1, 'John', 'Highway 21')  
(5, 'Michael', 'Valley 345')  
(15, 'Michelle', 'Blue Village') (2, 'Peter', 'Lowstreet 27')  
(8, 'Richard', 'Sky st 331')  
(6, 'Sandy', 'Ocean blvd 2')  
(9, 'Susan', 'One way 98')  
(10, 'Vicky', 'Yellow Garden 2')  
(14, 'Viola', 'Sideway 1633')  
(12, 'William', 'Central st 954')

[Sort the result in a descending order (reverse alphabetically)](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_orderby_desc)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \* FROM customers ORDER BY name DESC"

mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

(12, 'William', 'Central st 954') (14, 'Viola', 'Sideway 1633')  
(10, 'Vicky', 'Yellow Garden 2')  
(9, 'Susan', 'One way 98')  
(6, 'Sandy', 'Ocean blvd 2')  
(8, 'Richard', 'Sky st 331')  
(2, 'Peter', 'Lowstreet 27')  
(15, 'Michelle', 'Blue Village') (5, 'Michael', 'Valley 345')  
(1, 'John', 'Highway 21')  
(4, 'Hannah', 'Mountain 21')  
(13, 'Chuck', 'Main Road 989')  
(7, 'Betty', 'Green Grass 1')  
(11, 'Ben', 'Park Lane 38')  
(3, 'Amy', 'Apple st 652')

[Delete records from an existing table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_delete)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "DELETE FROM customers WHERE address = 'Mountain 21'"

mycursor.execute(sql)

mydb.commit()

print(mycursor.rowcount, "record(s) deleted")

Output:

1 record(s) deleted

[Prevent SQL injection](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_delete_escape)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "DELETE FROM customers WHERE address = %s"

adr = ("Yellow Garden 2", )

mycursor.execute(sql, adr)

mydb.commit()

print(mycursor.rowcount, "record(s) deleted")

Output:

1 record(s) deleted

[Delete an existing table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_drop_table)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "DROP TABLE customers"

mycursor.execute(sql)

#If this page was executed with no error(s), you have successfully deleted the "customers" table.

Output:

[Delete a table if it exist](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_drop_table2)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "DROP TABLE IF EXISTS customers"

mycursor.execute(sql)

#If this page was executed with no error(s), you have successfully deleted the "customers" table.

Output:

[Update existing records in a table](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_update)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "UPDATE customers SET address = 'Canyon 123' WHERE address = 'Valley 345'"

mycursor.execute(sql)

mydb.commit()

print(mycursor.rowcount, "record(s) affected")

Output:

1 record(s) affected

[Prevent SQL injection](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_update_escape)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "UPDATE customers SET address = %s WHERE address = %s"

val = ("Valley 345", "Canyon 123")

mycursor.execute(sql, val)

mydb.commit()

print(mycursor.rowcount, "record(s) affected")

Output:

1 record(s) affected

[Limit the number of records returned from a query](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_limit)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("SELECT \* FROM customers LIMIT 5")

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

(1, 'John', 'Highway 21')  
(2, 'Peter', 'Lowstreet 27')  
(3, 'Amy', 'Apple st 652')  
(4, 'Hannah', 'Mountain 21')  
(5, 'Michael', 'Valley 345')

[Combine rows from two or more tables, based on a related column between them](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_inner_join)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \

users.name AS user, \

products.name AS favorite \

FROM users \

INNER JOIN products ON users.fav = products.id"

mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

('John', 'Chocolate Heaven')  
('Peter', 'Chocolate Heaven')  
('Amy', 'Tasty Lemon')

[LEFT JOIN](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_left_join)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \

users.name AS user, \

products.name AS favorite \

FROM users \

LEFT JOIN products ON users.fav = products.id"

mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

('John', 'Chocolate Heaven')  
('Peter', 'Chocolate Heaven')  
('Amy', 'Tasty Lemon')  
('Hannah', None)  
('Michael', None)

[RIGHT JOIN](https://www.w3schools.com/python/trypython.asp?filename=demo_mysql_right_join)

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="myusername",

password="mypassword",

database="mydatabase"

)

mycursor = mydb.cursor()

sql = "SELECT \

users.name AS user, \

products.name AS favorite \

FROM users \

RIGHT JOIN products ON users.fav = products.id"

mycursor.execute(sql)

myresult = mycursor.fetchall()

for x in myresult:

print(x)

Output:

('John', 'Chocolate Heaven')  
('Peter', 'Chocolate Heaven')  
('Amy', 'Tasty Lemon')  
(None, 'Vanilla Dreams')

[MySQL Explained](https://www.w3schools.com/python/python_mysql_getstarted.asp)

## Python MongoDB

[Create a database](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_create_db)

import pymongo

myclient = pymongo.MongoClient('mongodb://localhost:27017/')

mydb = myclient['mydatabase']

# database created!

Output:

import pymongo

myclient = pymongo.MongoClient('mongodb://localhost:27017/')

mydb = myclient['mydatabase']

# database created!

[Check if a database exist](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_check_db)

import pymongo

myclient = pymongo.MongoClient('mongodb://localhost:27017/')

mydb = myclient['mydatabase']

mycol = mydb["customers"]

# collection created!

Output:

[Create a collection](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_create_collection)

import pymongo

myclient = pymongo.MongoClient('mongodb://localhost:27017/')

mydb = myclient['mydatabase']

mycol = mydb["customers"]

# collection created!

Output:

[Check if a collection exist](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_check_collection)

import pymongo

myclient = pymongo.MongoClient('mongodb://localhost:27017/')

mydb = myclient['mydatabase']

mycol = mydb["customers"]

print(mydb.list\_collection\_names())

Output:

['customers']

[Insert into collection](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_insert)

import pymongo

myclient = pymongo.MongoClient('mongodb://localhost:27017/')

mydb = myclient['mydatabase']

mycol = mydb["customers"]

mydict = { "name": "John", "address": "Highway 37" }

x = mycol.insert\_one(mydict)

print(x)

Output:

<pymongo.results.InsertOneResult object at 0x03D62918>

[Return the id field](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_insert_id)

import pymongo

myclient = pymongo.MongoClient('mongodb://localhost:27017/')

mydb = myclient['mydatabase']

mycol = mydb["customers"]

mydict = { "name": "Peter", "address": "Lowstreet 27" }

x = mycol.insert\_one(mydict)

print(x.inserted\_id)

Output:

5b1910482ddb101b7042fcd7

[Insert multiple documents](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_insert_many)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

mylist = [

{ "name": "Amy", "address": "Apple st 652"},

{ "name": "Hannah", "address": "Mountain 21"},

{ "name": "Michael", "address": "Valley 345"},

{ "name": "Sandy", "address": "Ocean blvd 2"},

{ "name": "Betty", "address": "Green Grass 1"},

{ "name": "Richard", "address": "Sky st 331"},

{ "name": "Susan", "address": "One way 98"},

{ "name": "Vicky", "address": "Yellow Garden 2"},

{ "name": "Ben", "address": "Park Lane 38"},

{ "name": "William", "address": "Central st 954"},

{ "name": "Chuck", "address": "Main Road 989"},

{ "name": "Viola", "address": "Sideway 1633"}

]

x = mycol.insert\_many(mylist)

#print list of the \_id values of the inserted documents:

print(x.inserted\_ids)

Output:

[ObjectId('5b19112f2ddb101964065487'), ObjectId('5b19112f2ddb101964065488'), ObjectId('5b19112f2ddb101964065489'), ObjectId('5b19112f2ddb10196406548a'), ObjectId('5b19112f2ddb10196406548b'), ObjectId('5b19112f2ddb10196406548c'), ObjectId('5b19112f2ddb10196406548d'), ObjectId('5b19112f2ddb10196406548e'), ObjectId('5b19112f2ddb10196406548f'), ObjectId('5b19112f2ddb101964065490'), ObjectId('5b19112f2ddb101964065491'), ObjectId('5b19112f2ddb101964065492')]

[Insert multiple documents with specified IDs](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_insert_many2)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

mylist = [

{ "\_id": 1, "name": "John", "address": "Highway 37"},

{ "\_id": 2, "name": "Peter", "address": "Lowstreet 27"},

{ "\_id": 3, "name": "Amy", "address": "Apple st 652"},

{ "\_id": 4, "name": "Hannah", "address": "Mountain 21"},

{ "\_id": 5, "name": "Michael", "address": "Valley 345"},

{ "\_id": 6, "name": "Sandy", "address": "Ocean blvd 2"},

{ "\_id": 7, "name": "Betty", "address": "Green Grass 1"},

{ "\_id": 8, "name": "Richard", "address": "Sky st 331"},

{ "\_id": 9, "name": "Susan", "address": "One way 98"},

{ "\_id": 10, "name": "Vicky", "address": "Yellow Garden 2"},

{ "\_id": 11, "name": "Ben", "address": "Park Lane 38"},

{ "\_id": 12, "name": "William", "address": "Central st 954"},

{ "\_id": 13, "name": "Chuck", "address": "Main Road 989"},

{ "\_id": 14, "name": "Viola", "address": "Sideway 1633"}

]

x = mycol.insert\_many(mylist)

#print a list of the \_id values of the inserted documents:

print(x.inserted\_ids)

Output:

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]

[Find the first document in the selection](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_find_one)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

x mycol.find\_one()

print(x)

Output:

{'\_id': 1, 'name': 'John', 'address': 'Highway37'}

[Find all documents in the selection](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_find)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

for x in mycol.find():

print(x)

Output:

{'\_id': 1, 'name': 'John', 'address': 'Highway37'}  
{'\_id': 2, 'name': 'Peter', 'address': 'Lowstreet 27'}  
{'\_id': 3, 'name': 'Amy', 'address': 'Apple st 652'}  
{'\_id': 4, 'name': 'Hannah', 'address': 'Mountain 21'}  
{'\_id': 5, 'name': 'Michael', 'address': 'Valley 345'}  
{'\_id': 6, 'name': 'Sandy', 'address': 'Ocean blvd 2'}  
{'\_id': 7, 'name': 'Betty', 'address': 'Green Grass 1'}  
{'\_id': 8, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 9, 'name': 'Susan', 'address': 'One way 98'}  
{'\_id': 10, 'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'\_id': 11, 'name': 'Ben', 'address': 'Park Lane 38'}  
{'\_id': 12, 'name': 'William', 'address': 'Central st 954'}  
{'\_id': 13, 'name': 'Chuck', 'address': 'Main Road 989'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}

[Find only some fields](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_find_some)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

for x in mycol.find({},{ "\_id": 0, "name": 1, "address": 1 }):

print(x)

Output:

{'name': 'John', 'address': 'Highway37'}  
{'name': 'Peter', 'address': 'Lowstreet 27'}  
{'name': 'Amy', 'address': 'Apple st 652'}  
{'name': 'Hannah', 'address': 'Mountain 21'}  
{'name': 'Michael', 'address': 'Valley 345'}  
{'name': 'Sandy', 'address': 'Ocean blvd 2'}  
{'name': 'Betty', 'address': 'Green Grass 1'}  
{'name': 'Richard', 'address': 'Sky st 331'}  
{'name': 'Susan', 'address': 'One way 98'}  
{'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'name': 'Ben', 'address': 'Park Lane 38'}  
{'name': 'William', 'address': 'Central st 954'}  
{'name': 'Chuck', 'address': 'Main Road 989'}  
{'name': 'Viola', 'address': 'Sideway 1633'}

[Filter the result](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_query)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

myquery = { "address": "Park Lane 38" }

mydoc = mycol.find(myquery)

for x in mydoc:

print(x)

Output:

{'\_id': 11, 'name': 'Ben', 'address': 'Park Lane 38'}

[Advanced query](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_query_modifier)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

#address greater than S:

myquery = { "address": {"$gt": "S"} }

mydoc = mycol.find(myquery)

for x in mydoc:

print(x)

Output:

{'\_id': 5, 'name': 'Michael', 'address': 'Valley 345'}  
{'\_id': 8, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 10, 'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}

[Filter with regular expressions](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_query_regex)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

#address starts with S:

myquery = { "address": { "$regex": "^S" } }

mydoc = mycol.find(myquery)

for x in mydoc:

print(x)

Output:

{'\_id': 10, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}

[Sort the result alphabetically](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_sort)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

mydoc = mycol.find().sort("name")

for x in mydoc:

print(x)

Output:

{'\_id': 3, 'name': 'Amy', 'address': 'Apple st 652'}  
{'\_id': 11, 'name': 'Ben', 'address': 'Park Lane 38'}  
{'\_id': 7, 'name': 'Betty', 'address': 'Green Grass 1'}  
{'\_id': 13, 'name': 'Chuck', 'address': 'Main Road 989'}  
{'\_id': 4, 'name': 'Hannah', 'address': 'Mountain 21'}  
{'\_id': 1, 'name': 'John', 'address': 'Highway37'}  
{'\_id': 5, 'name': 'Michael', 'address': 'Valley 345'}  
{'\_id': 2, 'name': 'Peter', 'address': 'Lowstreet 27'}  
{'\_id': 8, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 6, 'name': 'Sandy', 'address': 'Ocean blvd 2'}  
{'\_id': 9, 'name': 'Susan', 'address': 'One way 98'}  
{'\_id': 10, 'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}  
{'\_id': 12, 'name': 'William', 'address': 'Central st 954'}

[Sort the result descending (reverse alphabetically)](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_sort2)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

mydoc = mycol.find().sort("name", -1)

for x in mydoc:

print(x)

Output:

{'\_id': 12, 'name': 'William', 'address': 'Central st 954'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}  
{'\_id': 10, 'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'\_id': 9, 'name': 'Susan', 'address': 'One way 98'}  
{'\_id': 6, 'name': 'Sandy', 'address': 'Ocean blvd 2'}  
{'\_id': 8, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 2, 'name': 'Peter', 'address': 'Lowstreet 27'}  
{'\_id': 5, 'name': 'Michael', 'address': 'Valley 345'}  
{'\_id': 1, 'name': 'John', 'address': 'Highway37'}  
{'\_id': 4, 'name': 'Hannah', 'address': 'Mountain 21'}  
{'\_id': 13, 'name': 'Chuck', 'address': 'Main Road 989'}  
{'\_id': 7, 'name': 'Betty', 'address': 'Green Grass 1'}  
{'\_id': 11, 'name': 'Ben', 'address': 'Park Lane 38'}  
{'\_id': 3, 'name': 'Amy', 'address': 'Apple st 652'}

[Delete document](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_delete_one)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

myquery = { "address": "Mountain 21" }

mycol.delete\_one(myquery)

#print the customers collection after the deletion:

for x in mycol.find():

print(x)

Output:

{'\_id': 1, 'name': 'John', 'address': 'Highway37'}  
{'\_id': 2, 'name': 'Peter', 'address': 'Lowstreet 27'}  
{'\_id': 3, 'name': 'Amy', 'address': 'Apple st 652'}  
{'\_id': 5, 'name': 'Michael', 'address': 'Valley 345'}  
{'\_id': 6, 'name': 'Sandy', 'address': 'Ocean blvd 2'}  
{'\_id': 7, 'name': 'Betty', 'address': 'Green Grass 1'}  
{'\_id': 8, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 9, 'name': 'Susan', 'address': 'One way 98'}  
{'\_id': 10, 'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'\_id': 11, 'name': 'Ben', 'address': 'Park Lane 38'}  
{'\_id': 12, 'name': 'William', 'address': 'Central st 954'}  
{'\_id': 13, 'name': 'Chuck', 'address': 'Main Road 989'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}

[Delete many documents](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_delete_many)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

myquery = { "address": {"$regex": "^S"} }

x = mycol.delete\_many(myquery)

print(x.deleted\_count, "documents deleted")

Output:

2 documents deleted.

[Delete all documents in a collection](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_delete_all)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

x = mycol.delete\_many({})

print(x.deleted\_count, "documents deleted")

Output:

11 documents deleted.

[Delete a collection](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_drop_collection)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

mycol.drop()

Output:

[Update a document](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_update_one)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

myquery = { "address": "Valley 345" }

newvalues = { "$set": { "address": "Canyon 123" } }

mycol.update\_one(myquery, newvalues)

#print "customers" after the update:

for x in mycol.find():

print(x)

Output:

{'\_id': 1, 'name': 'John', 'address': 'Highway37'}  
{'\_id': 2, 'name': 'Peter', 'address': 'Lowstreet 27'}  
{'\_id': 3, 'name': 'Amy', 'address': 'Apple st 652'}  
{'\_id': 4, 'name': 'Hannah', 'address': 'Mountain 21'}  
{'\_id': 5, 'name': 'Michael', 'address': 'Canyon 123'}  
{'\_id': 6, 'name': 'Sandy', 'address': 'Ocean blvd 2'}  
{'\_id': 7, 'name': 'Betty', 'address': 'Green Grass 1'}  
{'\_id': 8, 'name': 'Richard', 'address': 'Sky st 331'}  
{'\_id': 9, 'name': 'Susan', 'address': 'One way 98'}  
{'\_id': 10, 'name': 'Vicky', 'address': 'Yellow Garden 2'}  
{'\_id': 11, 'name': 'Ben', 'address': 'Park Lane 38'}  
{'\_id': 12, 'name': 'William', 'address': 'Central st 954'}  
{'\_id': 13, 'name': 'Chuck', 'address': 'Main Road 989'}  
{'\_id': 14, 'name': 'Viola', 'address': 'Sideway 1633'}

[Update many/all documents](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_update_many)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

myquery = { "address": { "$regex": "^S" } }

newvalues = { "$set": { "name": "Minnie" } }

x = mycol.update\_many(myquery, newvalues)

print(x.modified\_count, "documents updated.")

Output:

2 documents updated.

[Limit the result](https://www.w3schools.com/python/trypython.asp?filename=demo_mongodb_limit)

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]

mycol = mydb["customers"]

myresult = mycol.find().limit(5)

#print the result:

for x in myresult:

print(x)

Output:

{'\_id': 1, 'name': 'John', 'address': 'Highway37'}  
{'\_id': 2, 'name': 'Peter', 'address': 'Lowstreet 27'}  
{'\_id': 3, 'name': 'Amy', 'address': 'Apple st 652'}  
{'\_id': 4, 'name': 'Hannah', 'address': 'Mountain 21'}  
{'\_id': 5, 'name': 'Michael', 'address': 'Valley 345'}

[MongoDB Explained](https://www.w3schools.com/python/python_mongodb_getstarted.asp)

# Python Online Compiler

## Python Compiler (Editor)

With our online Python compiler, you can edit Python code, and view the result in your browser.

[Run »](https://www.w3schools.com/python/trypython.asp?filename=demo_compiler)

print("Hello, World!")  
  
x = "Python"  
y = "is"  
z = "awesome"  
print(x, y, z)

Hello, World!  
Python is awesome

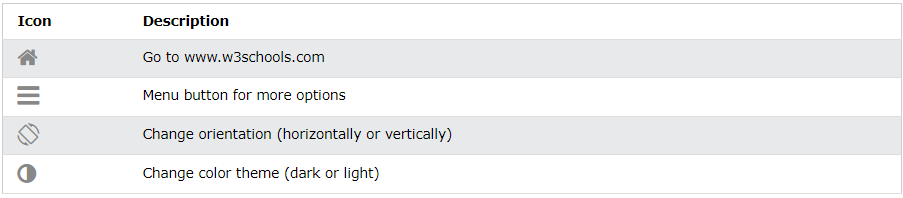
[Try it Yourself »](https://www.w3schools.com/python/trypython.asp?filename=demo_compiler)

Click on the "Try it Yourself" button to see how it works.

## Python Compiler Explained

The window to the left is editable - edit the code and click on the "Run" button to view the result in the right window.

The icons are explained in the table below:



If you don't know Python, we suggest that you read our [Python Tutorial](https://www.w3schools.com/python/default.asp) from scratch.

# Python Exercises

You can test your Python skills with W3Schools' Exercises.

## Exercises

We have gathered a variety of Python exercises (with answers) for each Python Chapter.

Try to solve an exercise by filling in the missing parts of a code. If you're stuck, hit the "Show Answer" button to see what you've done wrong.

## Count Your Score

You will get 1 point for each correct answer. Your score and total score will always be displayed.

## Start Python Exercises

Good luck!

[Start Python Exercises ❯](https://www.w3schools.com/python/exercise.asp)

If you don't know Python, we suggest that you read our [Python Tutorial](https://www.w3schools.com/python/default.asp) from scratch.

# Python Quiz

You can test your Python skills with W3Schools' Quiz.

## The Test

The test contains 25 questions and there is no time limit.

The test is not official, it's just a nice way to see how much you know, or don't know, about Python.

## Count Your Score

You will get 1 point for each correct answer. At the end of the Quiz, your total score will be displayed. Maximum score is 25 points.

## Start the Quiz

Good luck!

[Start the Python Quiz ❯](https://www.w3schools.com/quiztest/quiztest.asp?qtest=PYTHON)

If you don't know Python, we suggest that you read our [Python Tutorial](https://www.w3schools.com/python/default.asp) from scratch.

<https://www.w3schools.com/python/python_ml_getting_started.asp>