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-orientated way or a functional way.
* <https://youtu.be/WB5eMfnBI-8:-system> date and time
* <http://www.indiastudychannel.com/projects/2636-C-program-to-find-out-your-present-age-in-days-months-and-years.aspx>
* https://www.python-course.eu/tkinter\_layout\_management.php

## Features of Python

**Open source:** Python is publicly available open source software, any one can use source code that doesn't cost anything.

**Easy-to-learn:** Popular (scripting/extension) language, clear and easy syntax, no type declarations, automatic memory management, high-level data types and operations, design to read (more English like syntax) and write (shorter code compared to C, C++, and Java) fast.

**High-level Language:**   
High-level language (closer to human) refers to the higher level of concept from machine language (for example assembly languages). Python is an example of a high-level language like C, C++, Perl, and Java with low-level optimization.

**Portable:**  
High level languages are portable, which means they are able to run across all major hardware and software platforms with few or no change in source code. Python is portable and can be used on Linux, Windows, Macintosh, Solaris, FreeBSD, OS/2, Amiga, AROS, AS/400 and many more.

**Object-Oriented:** Python is a full-featured object-oriented programming language, with features such as classes, inheritance, objects, and overloading.

**Python is Interactive :**   
Python has an interactive console where you get a Python prompt (command line) and interact with the interpreter directly to write and test your programs. This is useful for mathematical programming.

**Interpreted :** Python programs are interpreted, takes source code as input, and then compiles (to portable byte-code) each statement and executes it immediately. No need to compiling or linking

**Extendable :** Python is often referred to as a "glue" language, meaning that it is capable to work in mixed-language environment. The Python interpreter is easily extended and can add a new built-in function or modules written in C/C++/Java code.

**Libraries :** Databases, web services, networking, numerical packages, graphical user interfaces, 3D graphics, others.

**Supports :**Support from online Python community

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 to 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.

## Python Indentations

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

Python uses indentation to indicate a block of code.

### Example

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

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

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

### Example

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

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

## 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!")

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

### Docstrings

Python also has extended documentation capability, called docstrings.

Docstrings can be one line, or multiline.

Python uses triple quotes at the beginning and end of the docstring:

### Example

Docstrings are also comments:

"""This is a   
multiline docstring."""  
print("Hello, World!")

## Output Variables

The Python print statement is often used to output variables.

To combine both text and a variable, Python uses the + character:

### Example

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

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_variables3)

You can also use the + character to add a variable to another variable:

### Example

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

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_variables4)

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

### Example

x = 5  
y = 10  
print(x + y)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_variables5)

If you try to combine a string and a number, Python will give you an error:

### Example

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

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

## 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))

## List

A list is a collection which is ordered and changeable. In Python lists are written with square brackets.

### Example

Create a List:

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

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

### Example

Change the second item:

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

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

## The list() Constructor

It is also possible to use the list() constructor to make a list. To add an item to the list use append() object method. To remove a specific item use the remove() object method. The len() function returns the length of the list.

### Example

Using the list() constructor to make a List:

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

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

### Example

Using the append() method to append an item:

thislist = list(("apple", "banana", "cherry"))  
thislist.append("damson")  
print(thislist)

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

### Example

Using the remove() method to remove an item:

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

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

### Example

The len() method returns the number of items in a list:

thislist = list(("apple", "banana", "cherry"))  
print(len(thislist))

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

## Tuple

A tuple is a collection which is ordered and unchangeable. In Python tuples are written with round brackets.

### Example

Create a Tuple:

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

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

### Example

Return the item in position 1:

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

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_tuple1)

### Example

You cannot change values in a tuple:

thistuple = ("apple", "banana", "cherry")  
thistuple[1] = "blackcurrant" # test changeability  
print(thistuple)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_tuple2)

## The tuple() Constructor

It is also possible to use the tuple() constructor to make a tuple. The len() function returns the length of the tuple.

### Example

Using the tuple() method to make a tuple:

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

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_tuple3)

### Example

The len() method returns the number of items in a tuple:

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

## Set

A set is a collection which is unordered and unindexed. In Python sets are written with curly brackets.

### Example

Create a Set:

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

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

**Note:** the set list is unordered, so the items will appear in a random order.

## The set() Constructor

It is also possible to use the set() constructor to make a set. You can use the add() object method to add an item, and the remove() object method to remove an item from the set. The len() function returns the size of the set.

### Example

Using the set() constructor to make a set:

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

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_set2)

### Example

Using the add() method to add an item:

thisset = set(("apple", "banana", "cherry"))  
thisset.add("damson")  
print(thisset)

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

### Example

Using the remove() method to remove an item:

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

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

### Example

Using the len() method to return the number of items:

thisset = set(("apple", "banana", "cherry"))  
print(len(thisset))

# Python Dictionaries

[❮ Previous](https://www.w3schools.com/python/python_sets.asp) [Next ❯](https://www.w3schools.com/python/python_conditions.asp)

## Dictionary

A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets, and they have keys and values.

### Example

Using the len() method to return the number of items:

thisdict = {  
  "apple": "green",  
  "banana": "yellow",  
  "cherry": "red"  
}  
print(thisdict)

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

### Example

Change the apple color to "red":

thisdict = {  
  "apple": "green",  
  "banana": "yellow",  
  "cherry": "red"  
}  
thisdict["apple"] = "red"  
print(thisdict)

[Run example »](https://www.w3schools.com/python/showpython.asp?filename=demo_dictionary2)

## The dict() Constructor

It is also possible to use the dict() constructor to make a dictionary:

### Example

thisdict = dict(apple="green", banana="yellow", cherry="red")  
# note that keywords are not string literals  
# note the use of equals rather than colon for the assignment  
print(thisdict)

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

## Adding Items

Adding an item to the dictionary is done by using a new index key and assigning a value to it:

### Example

thisdict = dict(apple="green", banana="yellow", cherry="red")  
thisdict["damson"] = "purple"  
print(thisdict)

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

## Removing Items

Removing a dictionary item must be done using the del() function in python:

### Example

thisdict = dict(apple="green", banana="yellow", cherry="red")  
del(thisdict["banana"])  
print(thisdict)

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

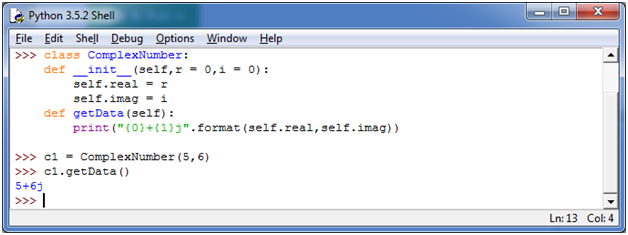
## Get the Length of a Dictionary

The len() function returns the size of the dictionary:

### Example

thisdict = dict(apple="green", banana="yellow", cherry="red")  
print(len(thisdict))

1. **class** Student:
2. **def** \_\_init\_\_(self, rollno, name):
3. self.rollno = rollno
4. self.name = name
5. **def** displayStudent(self):
6. **print** "rollno : ", self.rollno,  ", name: ", self.name
7. emp1 = Student(121, "Ajeet")
8. emp2 = Student(122, "Sonoo")
9. emp1.displayStudent()
10. emp2.displayStudent()



1. **class** Student:
2. # Constructor - non parameterized
3. **def** \_\_init\_\_(self):
4. **print**("This is non parametrized constructor")
5. **def** show(self,name):
6. **print**("Hello",name)
7. student = Student()
8. student.show("irfan")