

[All Tracks](#) > [Python](#) > [Getting Started](#) > [Python Variables](#)

Python

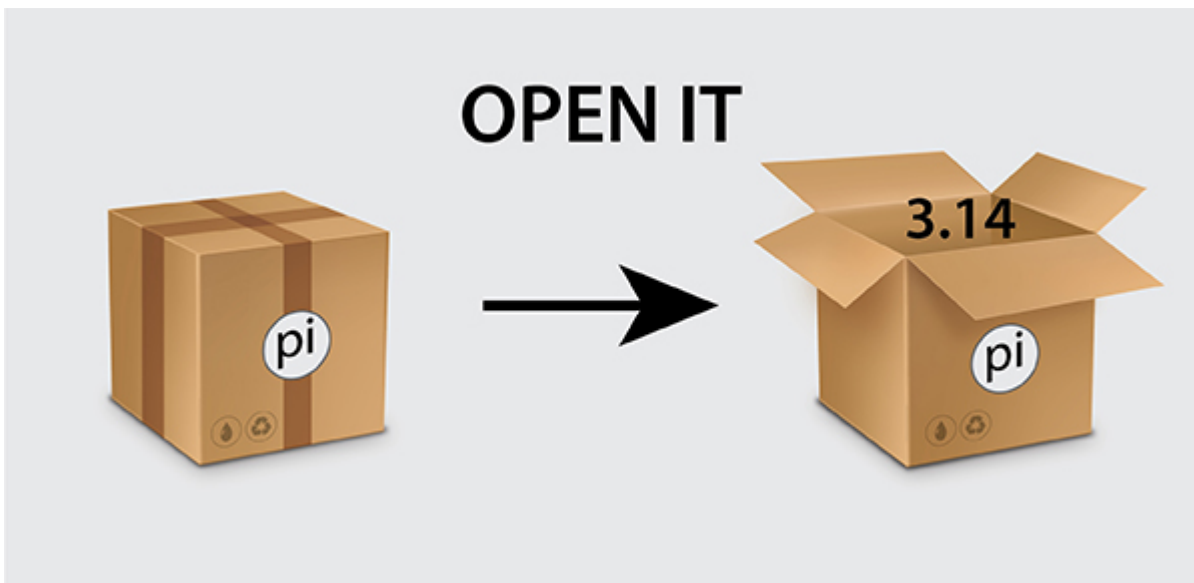
Topics: Python Variables

# Python Variables

## Tutorial

## Variables

A variable can be considered a storage container for data. Every variable will have a name. For example, we can have a variable named `speed_of_light`. A variable is a good way to store information while making it easy to refer to that information in our code later. A close analogy to variables may be a named box where you can store information.



For instance, instead of working with the number 3.14, we can assign it to a variable `pi`. You may forget that you need to use the number 3.14 when you will need to make relevant calculations later. On the other hand, it will be easier for you to remember to call `pi` when writing the code.

?

In this tutorial, you will learn how to name a variable and assign values. You will take a closer look at the methods that variables can support.

## Assignment

In Python, assignment can be done by writing the variable name followed by the value separated by an equal = symbol. The skeleton or pseudo-code is

```
“Variable name” = “ value or information ”
```

In the following examples, you assign various numbers and strings to variables.

```
>>> # assign the value 299792458 to the variable speed_of_light
>>> speed_of_light = 299792458
>>> print(speed_of_light)
299792458

>>> # assign a decimal number 3.14 to the variable pi
>>> pi = 3.14
>>> print(pi)
3.14

>>> # assign a string
>>> fav_lang = "python"
>>> print(fav_lang)
'python'
```

## Valid and invalid ways of assigning variables

Multiple words Assignment only works when the variable is a single word.

```
>>> multiple word = "multiple word"
File "<stdin>", line 1
    multiple word = "multiple word"
                ^
SyntaxError: invalid syntax
```

So, if you want to have more than one word in the name, the convention is to use underscore "\_" in the name.

```
>>> multiple_word = "multiple word" # note the variable name has an underscore _
>>> print(multiple_word)
multiple word
```

Do not start with a number

You cannot start a variable name with a number. The rest of the variable name can contain a number.

For example, 1var is wrong.

```
>>> 1var = 1
File "<stdin>", line 1
    1var = 1
    ^
SyntaxError: invalid syntax
```

But var1 is fine.

```
>>> var1 = 1
>>> print(var1)
1
```

More points to remember while deciding a variable name

You can only include a-z, A-Z, \_, and 0-9 in your variable names. Other special characters are not permitted.

For example, you cannot have hash key # in your variable names.

```
>>> # a_var_containing_# will not work as it has # in the name
>>> a_var_containing_# = 1
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'a_var_containing_' is not defined

>>> # but if we remove the # then it works
>>> a_var_containing_ = 1
>>> print(a_var_containing_)
1
```

Interestingly, you can have a variable name in your local language.

?

```
>>> 零 = 0 # chinese
>>> print(零)
0

>>> ශ්‍රී ලංකා = 0 # sinhala
>>> print(ශ්‍රී ලංකා)
0
```

## More on Assignments

Python supports assigning all data structures to variables.

For example, we can assign a list to a variable like you see in the following example, where we assign the list of names, denoted by [...], to the variable fav\_writers.

```
>>> # assigning list
>>> fav_writers = ["Mark Twain", "Fyodor Dostoyevsky"]
>>> print(fav_writers)
['Mark Twain', 'Fyodor Dostoyevsky']
```

Here is another example where you can assign dicts, shown by {...}, to a variable birthdays.

```
>>> # assign dicts
...
>>> birthdays = {"mom": "9Jan", "daughter": "24Dec"}
>>> print(birthdays)
{'mom': '9Jan', 'daughter': '24Dec'}
```

Data structures such as lists and dicts will be discussed in later tutorials.

You can also assign functions and classes to variables.

You will know more about functions and classes in later tutorials.

```
>>> # assigning functions
...
>>> import functools
>>> memoize = functools.lru_cache
>>> print(memoize)
<function lru_cache at 0x7fb2a6b42f28>
```

?

```
>>> # class assignment
...
>>> class MyClass:
...     pass
...
>>> give_me_more = MyClass()
>>> print(give_me_more)
<__main__.MyClass object at 0x7f512e65bfd0>
```

## Working with variables

Variables will support any method the underlying type supports. For example, if an integer value is stored in a variable, then the variable will support integer functions such as addition.

In the following example, you assign the number 2 to the variable `var` and then add 3 to `var`. This will print 5, the result of 3 being added to the value stored in `var` which is 2.

```
>>> var = 2
>>> print(var + 3)
5
```

You can make a change in a variable and assign it to the same variable. This is done generally when some kind of data type change is done.

For example, you can take a number as input. This will take in the digit as a string. You can then take the string number and convert it to `int` and assign it to the same number.

```
>>> number = input()
2
>>> type(number)
<class 'str'>
>>> number = int(number)
>>> type(number)
<class 'int'>
```


We will use a function `range(3)` which returns three values.

```
>>> print(range(3))
[0, 1, 2]
```

Something that returns three values can be unpacked to three variables. This is like saying take whatever is in `range(3)` and instead of assigning it to a single variable, break it up and assign individual values to the three variables. This is done using a comma between the variables.

```
>>> id1, id2, id3 = range(3)
>>> print(id1)
0
>>> print(id2)
1
>>> print(id3)
2
```

*Contributed by: Joydeep Bhattacharjee*

 [View all comments](#)

	For Developers	Developer Resources	For Business	Company
	Practice programming			About us
	Complete reference to competitive programming	Developers blog	Assess developers	Press
+1-650-461-4192	Competitive coding challenges	Learn to code by competitive programming	Conduct remote interviews	Careers
contact@hackerearth.com	Code Monk	Developers wiki	Assess university talent	Contact us
  	Start a programming club	How to conduct a hackathon	Organize hackathon	Technical support
				