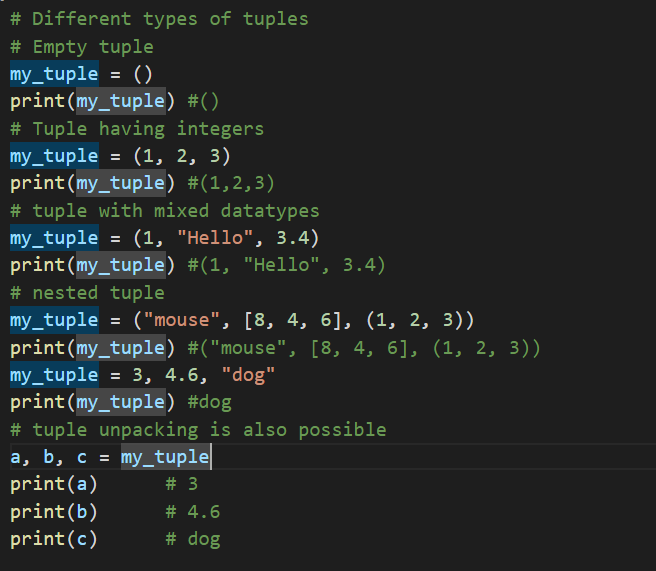
Tuple In Python

A tuple in Python is similar to a list. The difference between the two is that we cannot change the elements of a tuple once it is assigned whereas we can change the elements of a list.

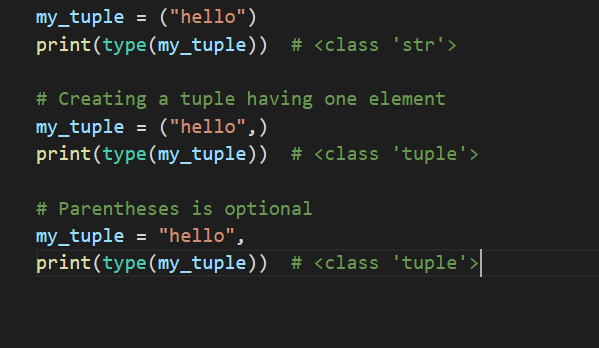
## Creating a Tuple

A tuple is created by placing all the items (elements) inside parentheses (), separated by commas. The parentheses are optional; however, it is a good practice to use them.

A tuple can have any number of items and they may be of different types (integer, float, list, string, etc.).



Having one element within parentheses is not enough. We will need a trailing comma to indicate that it is, in fact, a tuple.



## Access Tuple Elements

There are various ways in which we can access the elements of a tuple.

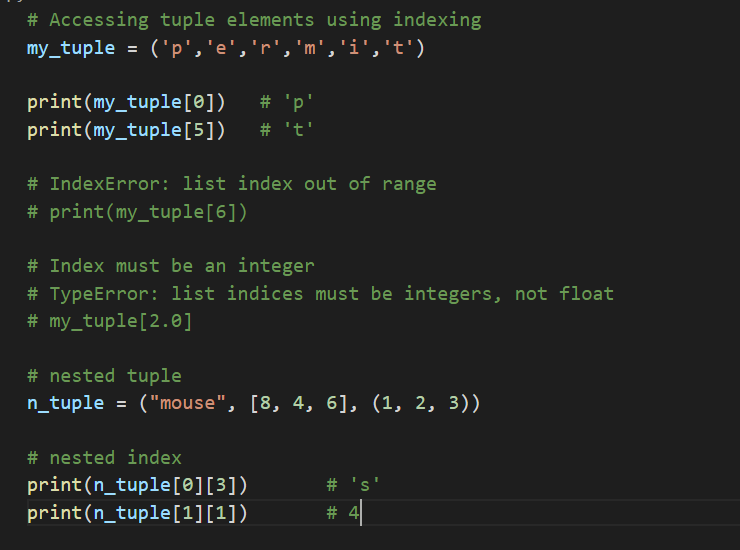
### **1. Indexing**

We can use the index operator [] to access an item in a tuple, where the index starts from 0.

So, a tuple having 6 elements will have indices from 0 to 5. Trying to access an index outside of the tuple index range(6,7,... in this example) will raise an IndexError.

The index must be an integer, so we cannot use float or other types. This will result in TypeError.

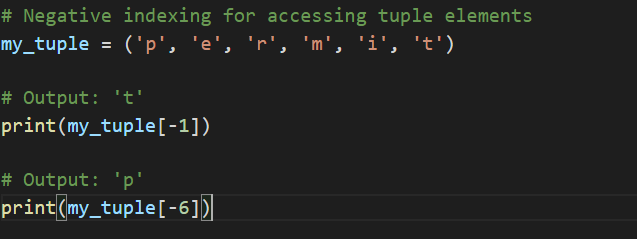
Likewise, nested tuples are accessed using nested indexing, as shown in the example below.



### **2. Negative Indexing**

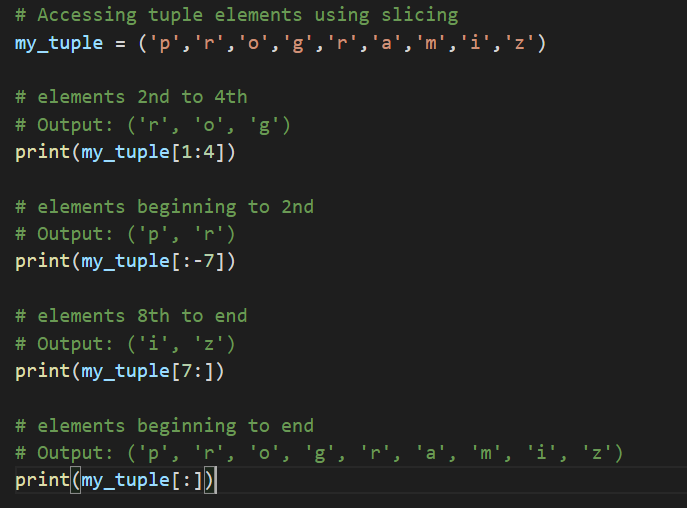
Python allows negative indexing for its sequences.

The index of -1 refers to the last item, -2 to the second last item and so on.



### **3. Slicing**

We can access a range of items in a tuple by using the slicing operator colon :.

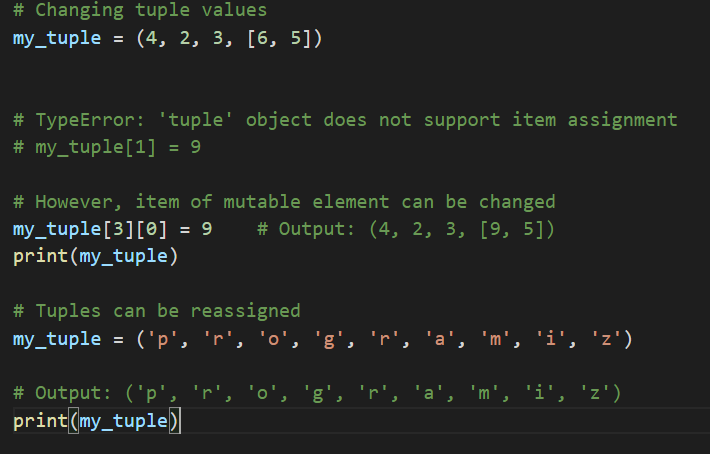


## Changing a Tuple

Unlike lists, tuples are immutable.

This means that elements of a tuple cannot be changed once they have been assigned. But, if the element is itself a mutable data type like a list, its nested items can be changed.

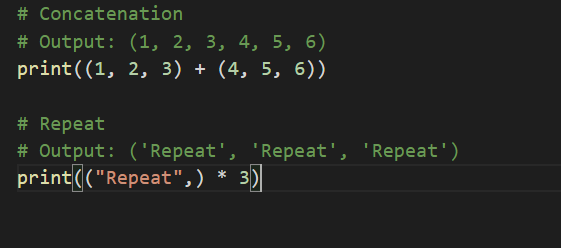
We can also assign a tuple to different values (reassignment).



We can use + operator to combine two tuples. This is called **concatenation**.

We can also **repeat** the elements in a tuple for a given number of times using the \* operator.

Both + and \* operations result in a new tuple.



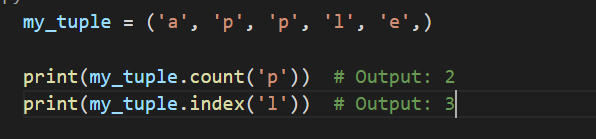
## Deleting a Tuple

As discussed above, we cannot change the elements in a tuple. It means that we cannot delete or remove items from a tuple.

Deleting a tuple entirely, however, is possible using the keyword del.

## Tuple Methods

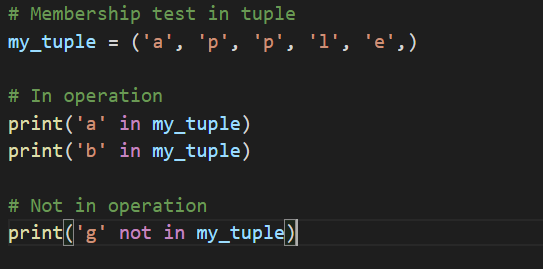
Methods that add items or remove items are not available with tuple. Only the following two methods are available.



## Other Tuple Operations

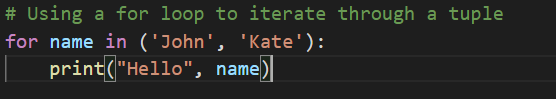
### **1. Tuple Membership Test**

We can test if an item exists in a tuple or not, using the keyword in.



### **2. Iterating Through a Tuple**

We can use a for loop to iterate through each item in a tuple.



### **Advantages of Tuple over List**

* We generally use tuples for heterogeneous (different) data types and lists for homogeneous (similar) data types.
* Since tuples are immutable, iterating through a tuple is faster than with list. So there is a slight performance boost.
* Tuples that contain immutable elements can be used as a key for a dictionary. With lists, this is not possible.
* If you have data that doesn't change, implementing it as tuple will guarantee that it remains write-protected.