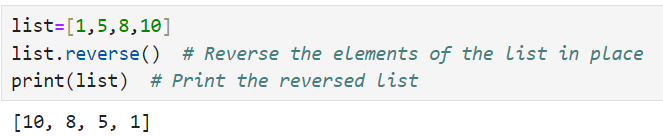
Day: 11

20th November, 2023

**reverse():**

reverse() is used to reverse a list.

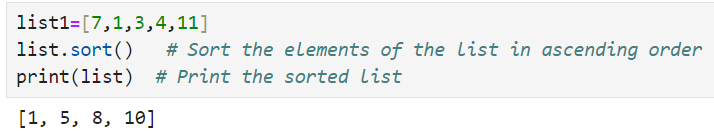
Example:



**sort():**

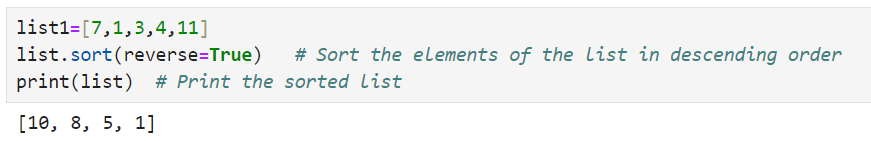
sort() is used to sort the list by default in ascending order.

Example:



In order to sort list in descending order, a property called “reverse=True” should be set.

Example:



**all/any:**

**all()** method returns :

True – if all elements of list are True.

False – if any element in a list is false.

Anything apart from 0 “zero” is True, otherwise False.

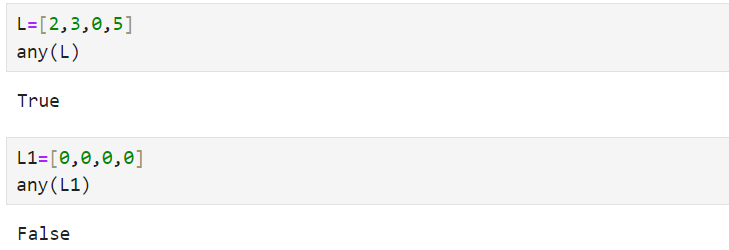
Example:



**any()** function returns True if any element in the list is True. If not, any() returns False.

Example:

Example:

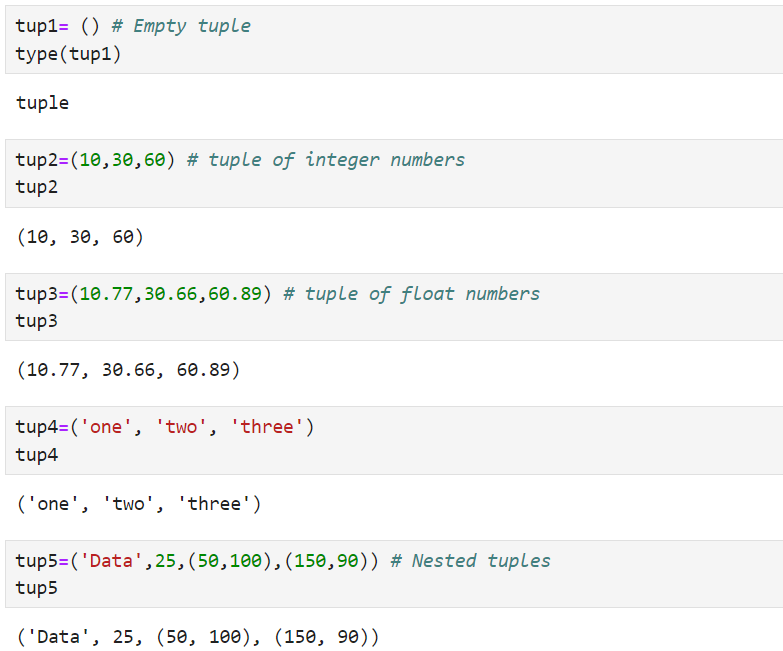


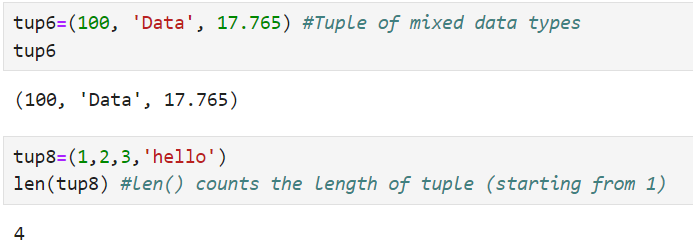
**2. Tuple:**

Tuple is a data structure where elements are stored inside the parenthesis ‘()’. Tuples are immutable(unchangeable). That means elements once assigned, you cannot update them or delete them from the tuple.

Tuples are preferred if there is no requirement to update the data. Iterating over the elements of a tuple is faster compared to iterating over a list.

Example of Tuple creation:

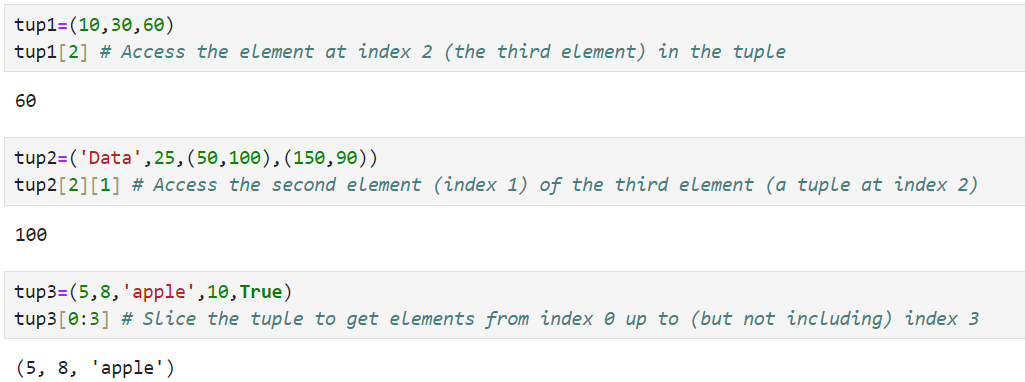




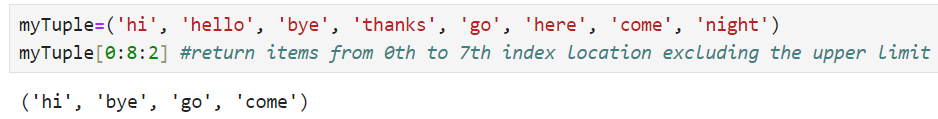
**Tuple Indexing:**

Tuple indexing is very similar to List Indexing.

Example:



**Tuple Slicing:**

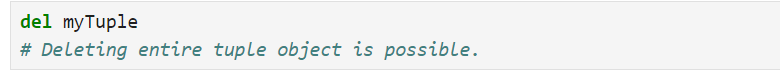
****

**Remove and change items in Tuple:**

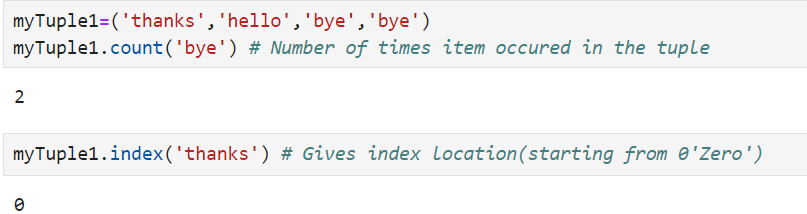
Tuples in Python are immutable, meaning their items cannot be modified (i.e., you cannot directly add, remove, or change the elements of a tuple). However, there are workarounds to achieve this behavior by converting the tuple into a list, modifying the list, and then converting it back to a tuple.

****

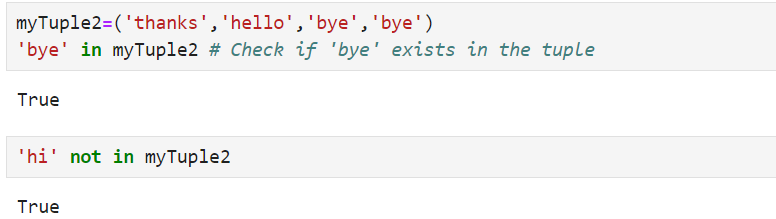
Deletion of entire tuple is possible by using del and tuple name.



**Count and Index function:**

****

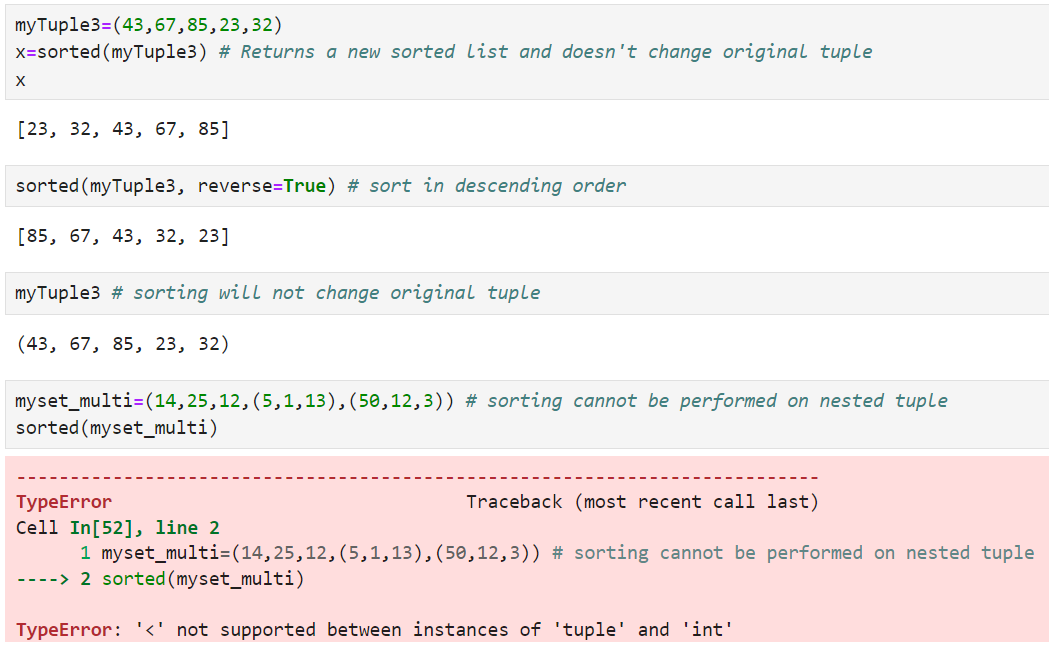
**Tuple Membership:**

****

**Sorting:**

When you perform sorting on a tuple, it will return a sorted list and it will not change the original tuple.

Example:



**Comparison between List and Tuple:**

|  |  |  |
| --- | --- | --- |
| **Feature** | **List** | **Tuple** |
| **Definition** | A mutable, ordered collection of items. | An immutable, ordered collection of items. |
| **Syntax** | Defined using square brackets []. | Defined using parenthesis (). |
| **Changeability** | Mutable (can be modified). | Immutable (cannot be modified after creation). |
| **Performance** | Generally slower due to mutability and overhead. | Generally faster due to immutability. |

**3. Set:**

An unordered collection of distinct elements is called a set which is enclosed inside curly braces ‘{}’. In particular, sets are useful for actions requiring unique values, such eliminating duplicates from a list or carrying out mathematical set operations like union, intersection, and difference. Sets are used to hold several things in a single variable.

Example:

