Python Tuple

Python Tuple is used to store the sequence of immutable python objects. Tuple is similar to lists since the value of the items stored in the list can be changed whereas the tuple is immutable and the value of the items stored in the tuple can not be changed.

Python Tuple Features

- Tuples are immutable Python sequences, i.e. you cannot change elements of a tuple in place.
- Tuples items are indexed.
- Tuples store a reference at each index.
- Tuples can be indexed sliced and its individual items can be indexed.
- len (T) returns count of tuple elements.
- Tuple manipulation functions are: len(), max(), min(), cmp() and tuple(). **Dictionaries**
- Dictionaries in Python are a collection of some key-value pairs.
- These are mutable and unordered collection with elements in the form of a key : value pairs that associate keys to values.
- The key of dictionaries are immutable type and unique.
- To manipulate dictionaries functions are : len(). clear(), has_key(),items(), keys(), values(), update(), and cmp().
- The membership operators in and not in work with dictionary keys only.

A tuple can be written as the collection of comma-separated values enclosed with the small brackets. A tuple can be defined as follows.

```
1. T1 = (101, "Ayush", 22)
2. T2 = ("Apple", "Banana", "Orange")
```

Example

```
1. tuple1 = (10, 20, 30, 40, 50, 60)
2. print(tuple1)
3. count = 0
4. for i in tuple1:
5.     print("tuple1[%d] = %d"%(count, i));
```

Output:

```
(10, 20, 30, 40, 50, 60)
tuple1[0] = 10
tuple1[0] = 20
tuple1[0] = 30
```

```
tuple1[0] = 40
tuple1[0] = 50
tuple1[0] = 60
```

Example 2

```
1. tuple1 = tuple(input("Enter the tuple elements ..."))
2. print(tuple1)
3. count = 0
4. for i in tuple1:
5.     print("tuple1[%d] = %s"%(count, i));
```

Output:

```
Enter the tuple elements ...12345
('1', '2', '3', '4', '5')
tuple1[0] = 1
tuple1[0] = 2
tuple1[0] = 3
tuple1[0] = 4
tuple1[0] = 5
```

However, if we try to reassign the items of a tuple, we would get an error as the tuple object doesn't support the item assignment.

An empty tuple can be written as follows.

```
1. T3 = ()
```

The tuple having a single value must include a comma as given below.

```
1. T4 = (90,)
```

A tuple is indexed in the same way as the lists. The items in the tuple can be accessed by using their specific index value.

We will see all these aspects of tuple in this section of the tutorial.

Tuple indexing and splitting

The indexing and slicing in tuple are similar to lists. The indexing in the tuple starts from 0 and goes to length(tuple) - 1.

The items in the tuple can be accessed by using the slice operator. Python also allows us to use the colon operator to access multiple items in the tuple.

Consider the following image to understand the indexing and slicing in detail.

Unlike lists, the tuple items can not be deleted by using the del keyword as tuples are immutable. To delete an entire tuple, we can use the del keyword with the tuple name.

Consider the following example.

```
1. tuple1 = (1, 2, 3, 4, 5, 6)
2. print(tuple1)
3. del tuple1[0]
4. print(tuple1)
5. del tuple1
6. print(tuple1)
```

Output:

```
(1, 2, 3, 4, 5, 6)
Traceback (most recent call last):
  File "tuple.py", line 4, in <module>
    print(tuple1)
NameError: name 'tuple1' is not defined
```

Like lists, the tuple elements can be accessed in both the directions. The right most element (last) of the tuple can be accessed by using the index -1. The elements from left to right are traversed using the negative indexing.

Consider the following example.

```
1. tuple1 = (1, 2, 3, 4, 5)
2. print(tuple1[-1])
3. print(tuple1[-4])
```

Output:

```
5
2
```

Basic Tuple operations

The operators like concatenation (+), repetition (*), Membership (in) works in the same way as they work with the list. Consider the following table for more detail.

Let's say Tuple t = (1, 2, 3, 4, 5) and Tuple t1 = (6, 7, 8, 9) are declared.

Operator	Description	Example
Repetition	The repetition operator enables the tuple elements to be repeated multiple times.	T1*2 = (1, 2, 3, 4, 5, 1, 2, 3, 4, 5)
Concatenation	It concatenates the tuple mentioned on either side of the operator.	T1+T2 = (1, 2, 3, 4, 5, 6, 7, 8, 9)
Membership	It returns true if a particular item exists in the tuple otherwise false.	print (2 in T1) prints True.

Iteration	The for loop is used to iterate over the tuple elements.	for i in T1: print(i) **Output** 1 2 3 4 5
Length	It is used to get the length of the tuple.	len(T1) = 5

Python Tuple inbuilt functions

SN	Function	Description
1	cmp(tuple1, tuple2)	It compares two tuples and returns true if tuple1 is greater than tuple2 otherwise false.
2	len(tuple)	It calculates the length of the tuple.
3	max(tuple)	It returns the maximum element of the tuple.
4	min(tuple)	It returns the minimum element of the tuple.
5	tuple(seq)	It converts the specified sequence to the tuple.

Where use tuple

Using tuple instead of list is used in the following scenario.

1. Using tuple instead of list gives us a clear idea that tuple data is constant and must not be changed.

2. Tuple can simulate dictionary without keys. Consider the following nested structure which can be used as a dictionary.

1. [(101, "John", 22), (102, "Mike", 28), (103, "Dustin", 30)]

3. Tuple can be used as the key inside dictionary due to its immutable nature.

List VS Tuple

SN	List	Tuple
1	The literal syntax of list is shown by the [].	The literal syntax of the tuple is shown by the ().
2	The List is mutable.	The tuple is immutable.
3	The List has the variable length.	The tuple has the fixed length.
4	The list provides more functionality than tuple.	The tuple provides less functionality than the list.
5	The list Is used in the scenario in which we need to store the simple collections with no	The tuple is used in the cases where we need to store the read-only collections i.e., the value of the

	constraints where the value of the items can be changed.	items can not be changed. It can be used as the key inside the dictionary.

Nesting List and tuple

We can store list inside tuple or tuple inside the list up to any number of level.

Lets see an example of how can we store the tuple inside the list.

```
1. Employees = [(101, "Ayush", 22), (102, "john", 29), (103, "james", 45), (104, "Ben", 34)]
2. print("----Printing list----");
3. for i in Employees:
4.     print(i)
5. Employees[0] = (110, "David",22)
6. print();
7. print("----Printing list after modification----");
8. for i in Employees:
9.     print(i)
```

Output:

```
----Printing list----
(101, 'Ayush', 22)
(102, 'john', 29)
(103, 'james', 45)
(104, 'Ben', 34)

----Printing list after modification----

(110, 'David', 22)
(102, 'john', 29)
(103, 'james', 45)
(104, 'Ben', 34)
```