Tuple Manipulation

Introduction

- In Python, tuple is also a kind of container which can store list of any kind of values.
- Tuple is an immutable data type which means we can not change any value of tuple.
- Tuple is a sequence like string and list but the difference is that list is mutable whereas string and tuple are immutable.
- In this chapter we will see manipulation on tuple i.e. creation of tuple, its use and operations on tuple with built in functions.

Creation of Tuple

- In Python, “( )” parenthesis are used for tuple creation.

( )	empty tuple
(1, 2, 3)	integers tuple
(1, 2.5, 3.7, 7)	numbers tuple
(‘a’, ’b’, ’c’)	characters tuple
(‘a’, 1, ‘b’, 3.5, ‘zero’)	mixed values tuple
(‘one’, ’two’, ’three’, ’four’)	string tuple

***Tuple is an immutable sequence whose values can not be changed.**

Creation of Tuple

Look at following examples of tuple creation carefully-

- Empty tuple:

```
>>> t=()
>>> t
()
```

- Single element tuple:

```
>>> t=(1)
>>> t
1
```

- Long tuple:

```
>>> t=(0,1,4,9,16,25,36,49,64,81,100,121)
>>> t
(0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121)
```

- Nested tuple:

```
>>> t=(1,2,3,(4,5))
>>> t
(1, 2, 3, (4, 5))
```

Creation of Tuple

tuple() function is used to create a tuple from other sequences. See examples-

Tuple creation from string

```
>>> t=tuple("Hello")
>>> t
('H', 'e', 'l', 'l', 'o')
```

Tuple creation from list

```
>>> L=['a', 'e', 'i', 'o', 'u']
>>> T=tuple(L)
>>> T
('a', 'e', 'i', 'o', 'u')
```

Tuple creation from input

```
>>> t1=tuple(input("Enter element"))
Enter element123456
>>> t1
('1', '2', '3', '4', '5', '6')
```

All these elements are of character type. To have these in different types, need to write following statement.-

Tuple=eval(input("Enter elements"))

```
>>> t1=eval(input("Enter the elements"))
Enter the elements (2,4.5,"hello")
>>> t1
(2, 4.5, 'hello')
```

Accessing a Tuple

- In Python, the process of tuple accessing is same as with list. Like a list, we can access each and every element of a tuple.
- **Similarity with List-** like list, tuple also has index. All functionality of a list and a tuple is same except mutability.

Forward index Tuple Backward	0	1	2	3	4	5	6	7	8	9	10	11	12	13
	R	E	S	P	O	N	S	I	B	I	L	I	T	Y
	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- len ( ) function is used to get the length of tuple.

```
>>> t=tuple("Hello")
>>> len(t)
5
```

Accessing a Tuple

- Indexing and Slicing:
  - T[i] returns the item present at index i.
  - T[i : j] returns a new tuple having all the items of T from index i to j.
  - T [i : j : n] returns a new tuple having difference of n elements of T from index i to j.

```
>>> T=(1,2,3,4,5,6,7,8,9,10)
>>> T[1:10:3]
(2, 5, 8)
```

- Membership operator:
  - Working of membership operator “in” and “not in” is same as in a list. (for details see the chapter- list manipulation).
  - Concatenation and Replication operators:
    - + operator adds second tuple at the end of first tuple. * operator repeats elements of tuple.

Accessing a Tuple

- Accessing Individual elements-

```
>>> L=['a', 'e', 'i', 'o', 'u']
>>> L[0]
'a'
>>> L[3]
'o'
```

- Traversal of a Tuple –

```
for <item> in <tuple>:
        #to process every element.
```

```
T=tuple("Python")
print(T,end="")
print()
for a in T:
    print(a)
```

OUTPUT

```
('P', 'y', 't', 'h', 'o', 'n')
P
y
t
h
o
n
```

Tuple Operations

- Tuple joining
  - Both the tuples should be there to add with +.

```
>>> tp1=(1,3,5)
>>> tp2=(6,7,8)
>>> tp1+tp2
(1, 3, 5, 6, 7, 8)
>>> tp3=tp1+tp2
>>> tp3
(1, 3, 5, 6, 7, 8)
```

Some errors in tuple joining-

- In Tuple + number
- In Tuple + complex number
- In Tuple + string
- In Tuple + list
- **Tuple + (5) will also generate error because when adding a tuple with a single value, tuple will also be considered as a value and not a tuple..**

- Tuple Replication-

```
>>> tp1=(1,3,5)
>>> tp1*3
(1, 3, 5, 1, 3, 5, 1, 3, 5)
>>> tp1
(1, 3, 5)
>>> tp2=tp1*3
>>> tp2
(1, 3, 5, 1, 3, 5, 1, 3, 5)
```

Tuple Slicing

```
>>> tpl=(10,12,14,20,22,24,30,32,34)
>>> seq=tpl[3:-3]
>>> seq
(20, 22, 24)
>>> tpl[3:30]
(20, 22, 24, 30, 32, 34)
>>> tpl[-15:7]
(10, 12, 14, 20, 22, 24, 30)
>>> tpl[0:10:2]
(10, 14, 22, 30, 34)
>>> tpl[0:10:3]
(10, 20, 30)
>>> tpl[::3]
(10, 20, 30)
```

Tuple will show till last element of list irrespective of upper limit.

Every alternate element will be shown.

Every third element will be shown.

Tuple Comparision

```
>>> a=(2,3)
>>> b=(2,3)
>>> a==b
True
>>> c=('2','3')
>>> a==c
False
>>> a>b
False
>>> d=(2.0,3.0)
>>> d>a
False
>>> d==a
True
>>> e=(2,3,4)
>>> a<e
True
```

Tuple unpacking

```
>>> t=(2,3,'A','B')
>>> w,x,y,z=t
>>> print(w)
2
>>> print(x)
3
>>> print(y)
A
>>> print(z)
B
```

Tuple Deletion

As we know that tuple is of immutable type, it is not possible to delete an individual element of a tuple. With del() function, it is possible to delete a complete tuple. Look at following example-

```
>>> t=(2,3,'A','B')
>>> del t[2]
Traceback (most recent call la
  File "<pyshell#65>", line 1,
    del t[2]
TypeError: 'tuple' object does
>>> del t
>>> t
Traceback (most recent call la
  File "<pyshell#67>", line 1,
    t
NameError: name 't' is not def
```

Error shown because deletion of a single element is also possible.

Complete tuple has been deleted. Now error shown on printing of tuple.

Tuple Functions

```
>>> emp=('Ram',25000,24,'LKO')
>>> len(emp)
4
>>> tpl=(10,12,14,16,18,20,22)
>>> max(tpl)
22
>>> tpl2=("Karan","Zubin","Zara","Ana")
>>> max(tpl2)
'Zubin'
>>> min(tpl)
10
>>> min(tpl2)
'Ana'
>>> tpl2.index("Zubin")
1
>>> tpl3=(10,12,14,16,10,18,20,10,22)
>>> tpl3.count(10)
3
>>> t=tuple("Hello")
>>> t
('H', 'e', 'l', 'l', 'o')
```

len( ) Function

max( ) Function

min( ) Function

index( ) Function

tuple( ) Function

Thank you