Lecture-17

Updating Tuples:

Tuples are immutable which means you cannot update or change the values of tuple elements. You are able to take portions of existing tuples to create new tuples as the following example demonstrates −

```
tup1 = (12, 34.56);
tup2 = ('abc', 'xyz');
 # Following action is not valid for tuples
# tup1[0] = 100;
 # So let's create a new tuple as follows
tup3 = tup1 + tup2;
print tup3;
```

When the above code is executed, it produces the following result −

```
(12, 34.56, 'abc', 'xyz')
```

Delete Tuple Elements:

Removing individual tuple elements is not possible. There is, of course, nothing wrong with putting together another tuple with the undesired elements discarded.

To explicitly remove an entire tuple, just use the **del** statement. For example −

```
tup = ('physics', 'chemistry', 1997, 2000);
print tup;
del tup;
print "After deleting tup : ";
print tup;
```

This produces the following result. Note an exception raised, this is because after **del tup** tuple does not exist any more −

```
('physics', 'chemistry', 1997, 2000)
After deleting tup :
Traceback (most recent call last):
 File "test.py", line 9, in <module>
  print tup;
NameError: name 'tup' is not defined
```

Basic Tuples Operations:

Tuples respond to the + and * operators much like strings; they mean concatenation and repetition here too, except that the result is a new tuple, not a string.

In fact, tuples respond to all of the general sequence operations we used on strings in the prior chapter

Python Expression	Results	Description
len((1, 2, 3))	3	Length
(1, 2, 3) + (4, 5, 6)	(1, 2, 3, 4, 5, 6)	Concatenation
('Hi!',) * 4	('Hi!', 'Hi!', 'Hi!', 'Hi!')	Repetition
3 in (1, 2, 3)	True	Membership
for x in (1, 2, 3): print x,	1 2 3	Iteration

Indexing, Slicing, and Matrixes

Because tuples are sequences, indexing and slicing work the same way for tuples as they do for strings. Assuming following input −

L = ('spam', 'Spam', 'SPAM!')

Python Expression	Results	Description
L[2]	'SPAM!'	Offsets start at zero
L[-2]	'Spam'	Negative: count from the right
L[1:]	['Spam', 'SPAM!']	Slicing fetches sections

Built-in Tuple Functions:

Python includes the following tuple functions –

Sr.No.	Function with Description
1	cmp(tuple1, tuple2) Compares elements of both tuples.
2	len(tuple) Gives the total length of the tuple.
3	max(tuple) Returns item from the tuple with max value.
4	min(tuple) Returns item from the tuple with min value.
5	tuple(seq) Converts a list into tuple.

1. cmp(tuple1, tuple2): Python tuple method **cmp()** compares elements of two tuples.

Syntax

cmp(tuple1, tuple2)

Example

```
tuple1, tuple2 = (123, 'xyz'), (456, 'abc')
print cmp(tuple1, tuple2)
print cmp(tuple2, tuple1)
tuple3 = tuple2 + (786,);
print cmp(tuple2, tuple3)
```

When we run above program, it produces following result −

```
-1
1
-1
```

2. len(tuple): Python tuple method **len()** returns the number of elements in the tuple.

Syntax

len(tuple)

Example

```
tuple1, tuple2 = (123, 'xyz', 'zara'), (456, 'abc')
print "First tuple length : ", len(tuple1)
print "Second tuple length : ", len(tuple2)
```

When we run above program, it produces following result –

```
First tuple length :  3
Second tuple length :  2
```

3. **max(tuple):** Python tuple method **max()** returns the elements from the tuple with maximum value.

Syntax

max(tuple)

Example

```
tuple1, tuple2 = (123, 'xyz', 'zara', 'abc'), (456, 700, 200)
print "Max value element : ", max(tuple1)
print "Max value element : ", max(tuple2)
```

When we run above program, it produces following result −

```
Max value element :  zara
Max value element :  700
```

4. min(tuple): Python tuple method **min()** returns the elements from the tuple with minimum value.

Syntax

min(tuple)

Example

```
tuple1, tuple2 = (123, 'xyz', 'zara', 'abc'), (456, 700, 200)
print "min value element : ", min(tuple1
)print "min value element : ", min(tuple2)
```

When we run above program, it produces following result −

```
min value element :  123
min value element :  200
```

5. **tuple(seq):** Python tuple method **tuple()** converts a list of items into tuples

Syntax

tuple(seq)

Example

```
aList = [123, 'xyz', 'zara', 'abc']
aTuple = tuple(aList)
print "Tuple elements : ", aTuple
```

When we run above program, it produces following result –

```
Tuple elements :  (123, 'xyz', 'zara', 'abc')
```

References:

1.Introduction to Computation and Programming using Python, by John Guttag, PHI Publisher

2. Fundamentals of Python first Programmes by Kenneth A Lambert, Copyrighted material Course Technology Inc. 1 st edition (6th February 2009)

3. https://www.tutorialspoint.com/python/index.htm

****END OF THE LECTURE***

THANK YOU