PYTHON REVISION TOUR – II(Cont…)

TUPLES

Tuples in Python

Tuple is a standard data type of Python that can store a sequence of values belonging to any type. Tuples are depicted through parenthesis i.e. round brackets. Tuples are **immutable sequence** i.e. element cannot be changed in place.

Example

–Tup1=(1,2,3,4,5)

–Tup2=('p','r','o','b','l','e','m‘)

–Tup3=('pan','ran','oggi','blade','lemon','egg','mango‘)

Creating a Tuple

Tuples can be created by assigning a variable with the values enclosed in round bracket separated by comma.

Example: tup1=(1,2,3,4,5)

Tuples can be created in different ways:

–**Empty Tuple:** Tuple with no item is a empty tuple. It can be created with the function T=tuple( ). it generates the empty tuple with the name T. This list is equivalent to 0 or ‘ ’.

–**Single Element Tuple:** Tuple with one item.

- T1=(9,) or T1=9, comma is required because Python treats T1=(9) as value not as tuple element.

–**Creating Tuple from Existing Sequence:**

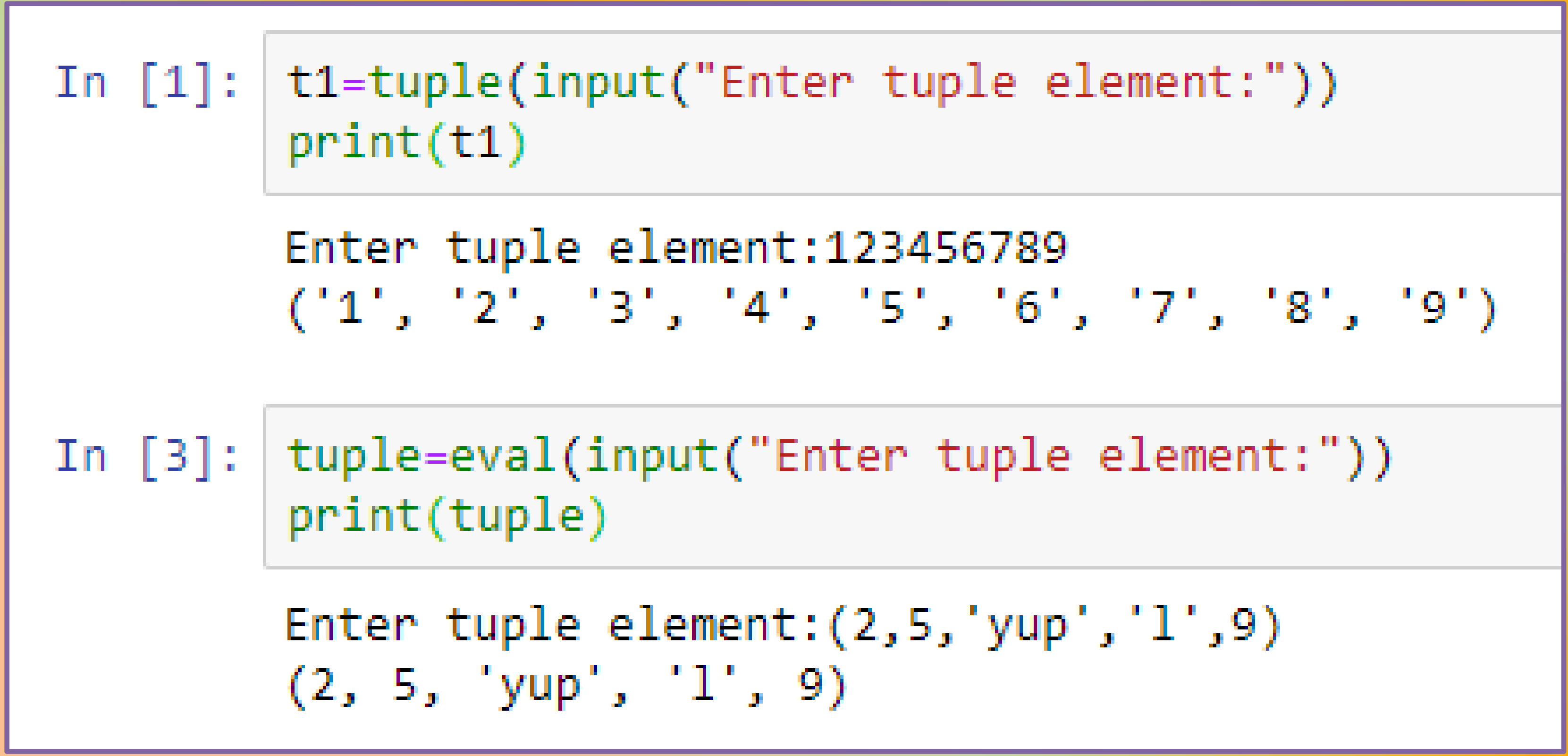
Syntax: T1=tuple(<sequence>)

Example: T1=tuple(‘Computer’)

>>>T1

Output: (’C’,’o’,’m’,’p’,’u’,’t’,’e’,’r’)

Creating Tuple from Keyboard Input



```
In [1]: t1=tuple(input("Enter tuple element:"))
        print(t1)

Enter tuple element:123456789
('1', '2', '3', '4', '5', '6', '7', '8', '9')
```

```
In [3]: tuple=eval(input("Enter tuple element:"))
        print(tuple)

Enter tuple element:(2,5,'yup','l',9)
(2, 5, 'yup', 'l', 9)
```

Tuples vs List

Similarities:

–Length

–Indexing

– Slicing

–Membership operators

–Concatenation and Replication operators

–Accessing Individual elements

- **Differences**

–Mutability: Tuples are not mutable, while list are.

Tuple Operations

- **Joining Tuples :** Two Tuples can be joined through addition.

```
>>>t1=(1,2,3)
>>>t2=(4,5,6)
>>>t3=t1+t2
>>>t3
(1,2,3,4,5,6)
```

- **Repeating or Replicating Tuples:** Multiply(*) operator replicates the tuple specified number of times

```
>>>t1=(1,2,3)
>>>t1*3
>>> t1
(1,2,3,1,2,3,1,2,3)
```

SLICING THE TUPLE

- Tuple slices are the subpart of a tuple extracted out. Tuple slices can be created through the use of indexes.

Syntax: Seq=Tuple[start:stop] : creates tuple slice out of t1 with element falling in between indexes start and stop not including stop.

Example:

```
>>>t1=(1,2,3,4,5,6,7,8)
 >>>seq=t1[2:-3]
>>>seq
```

Output: (3,4,5)

- tuples also supports slice steps. Example, Seq=Tuple[start:stop:step] creates tuple slice out of tuple with element falling in between indexes start and stop not including stop, skipping step-1 element in between.

Example:

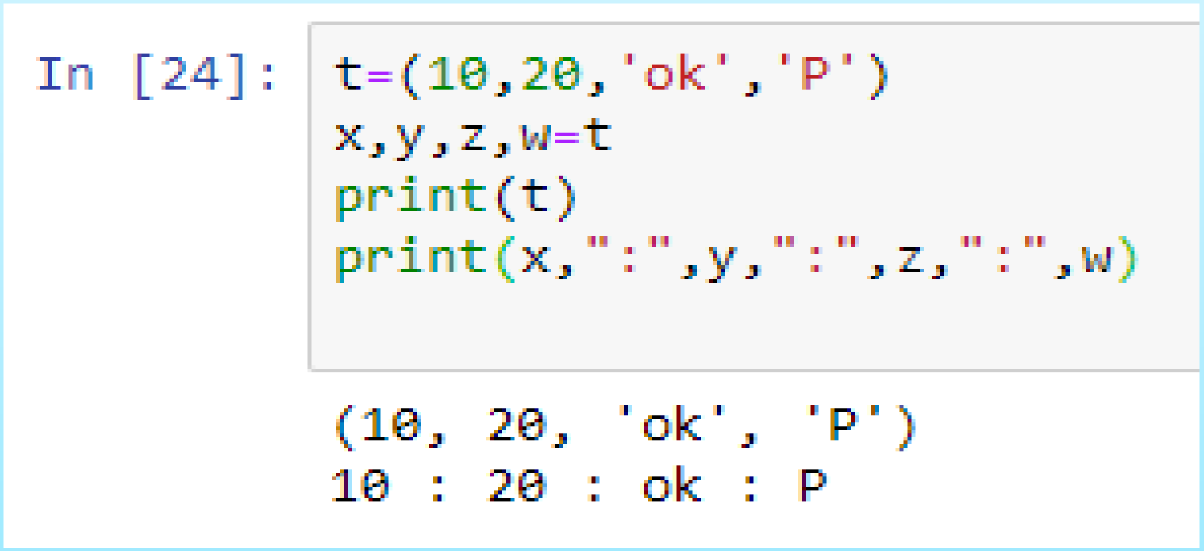
```
>>>t1=(1,2,3,4,5,6,7,8)
>>>seq=t1[2:7:2]
>>>seq
```

Output:

```
[3,5,7]
```

Unpacking Tuples

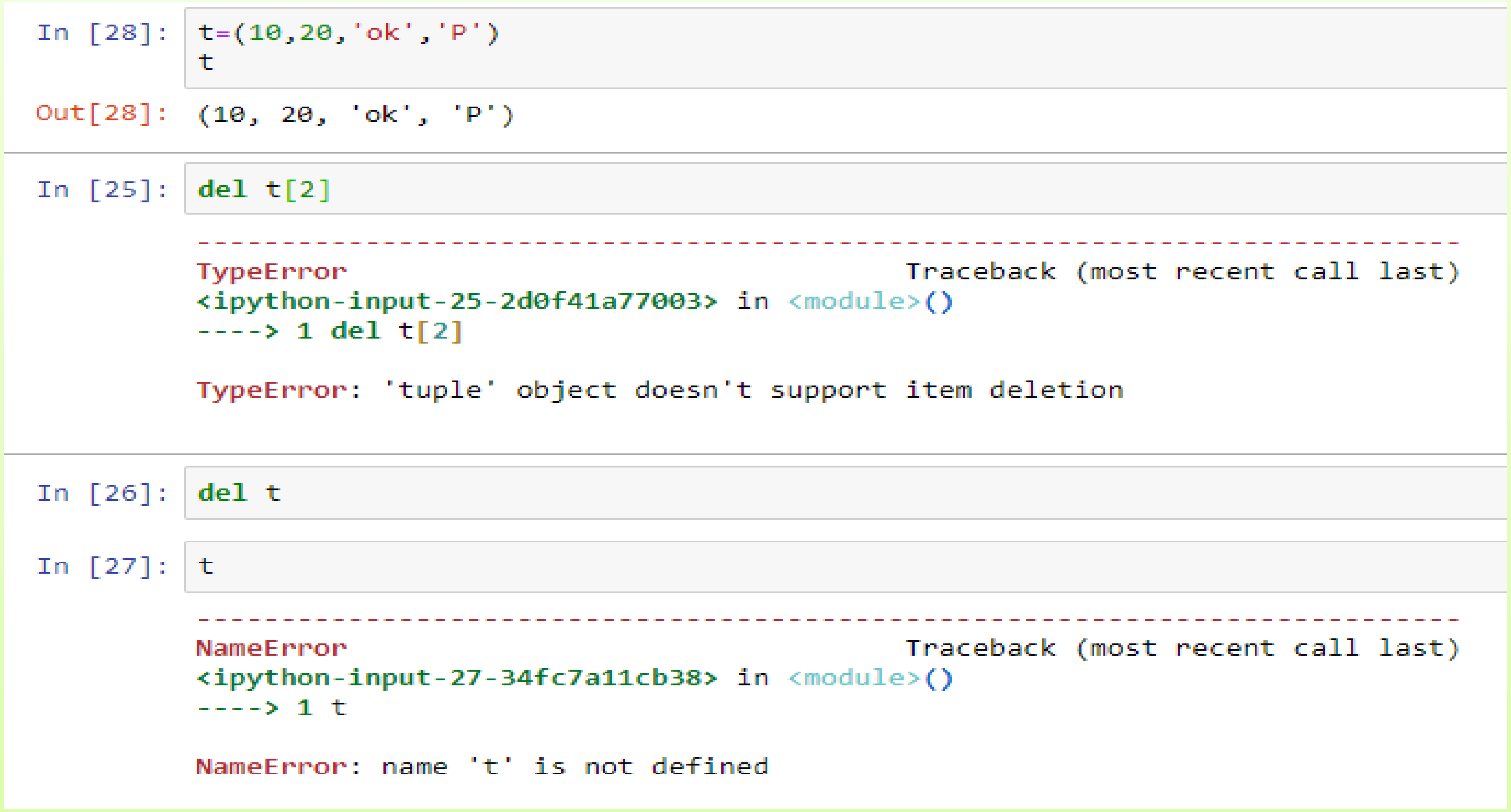
- Forming a tuple from individual values is called packing and creating individual values from a tuple’s elements is called unpacking.



```
In [24]: t=(10,20,'ok','P')
         x,y,z,w=t
         print(t)
         print(x,":",y,":",z,":",w)

         (10, 20, 'ok', 'P')
         10 : 20 : ok : P
```

Deleting Tuples

- We cannot delete individual item of a tuple.
- del statement deletes the complete tuple.



```
In [28]: t=(10,20,'ok','P')
         t

Out[28]: (10, 20, 'ok', 'P')

In [25]: del t[2]

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-25-2d0f41a77003> in <module>()
         ----> 1 del t[2]

         TypeError: 'tuple' object doesn't support item deletion

In [26]: del t

In [27]: t

         ---------------------------------------------------------------------------
         NameError                                 Traceback (most recent call last)
         <ipython-input-27-34fc7a11cb38> in <module>()
         ----> 1 t

         NameError: name 't' is not defined
```

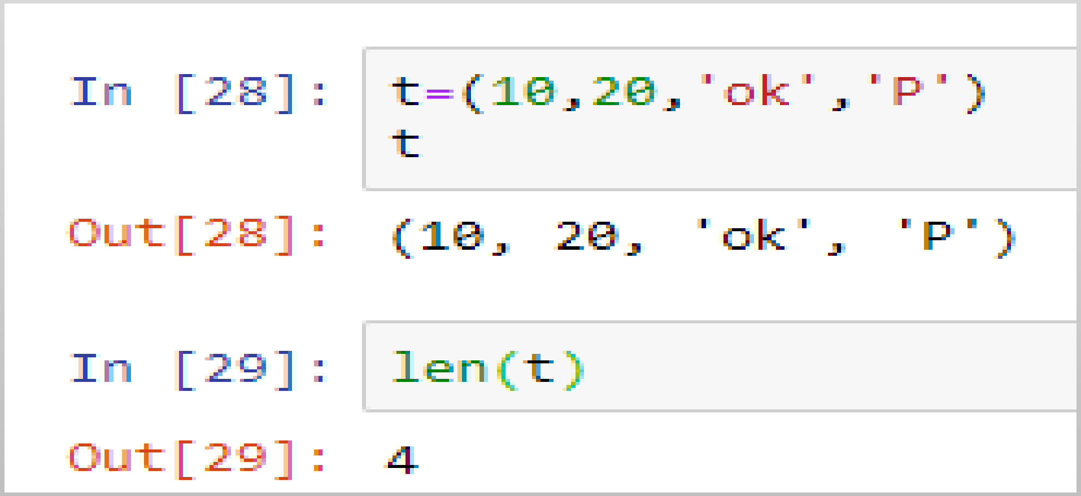
Tuple Functions and Methods

The len( ) Method

This function returns the length of the tuple, i.e. the count of elements in the tuple.

Syntax: len(<tuple>)

Example:



```
In [28]:  t=(10,20,'ok','P')
          t

Out[28]:  (10, 20, 'ok', 'P')

In [29]:  len(t)

Out[29]:  4
```

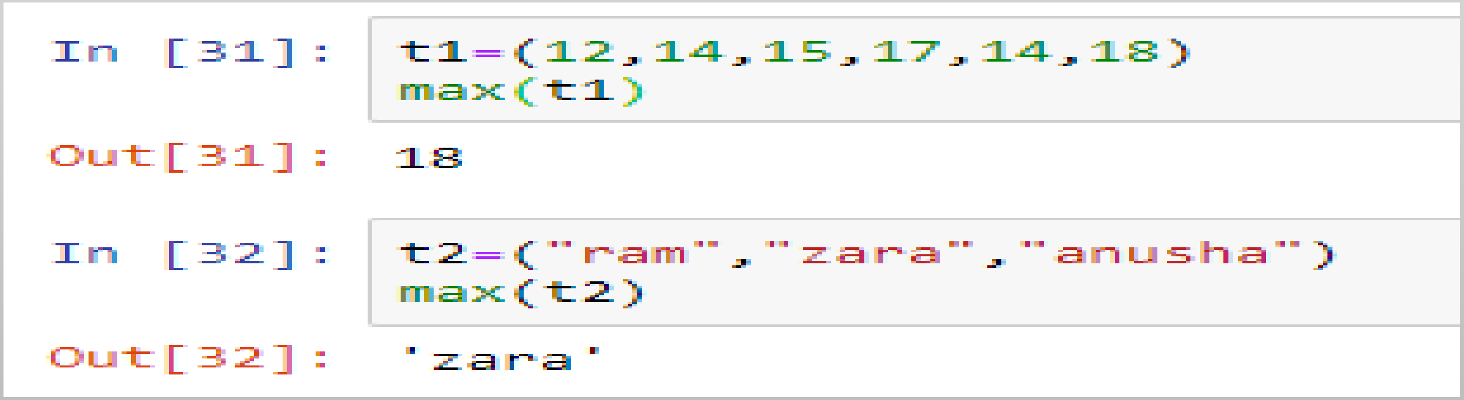
The max( ) Method

This function returns the element from the tuple having maximum value .

Syntax:

max(<tuple>) .

Example:



```
In [31]:  t1=(12,14,15,17,14,18)
          max(t1)

Out[31]:  18

In [32]:  t2=("ram","zara","anusha")
          max(t2)

Out[32]:  'zara'
```

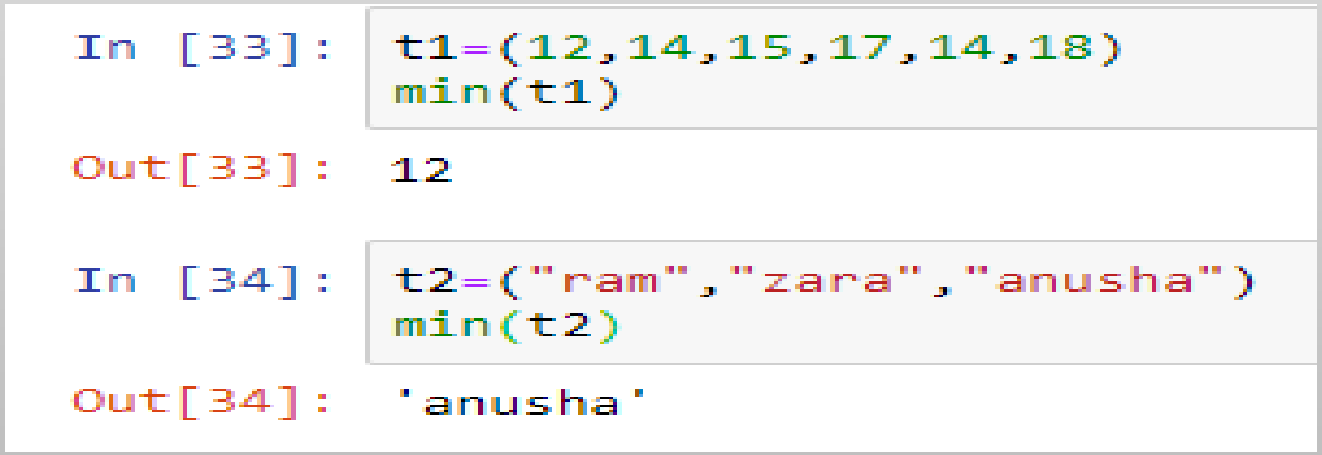
The min( ) Method

This function returns the element from the tuple having minimum value .

Syntax:

min(<tuple>)

Example:



```
In [33]:  t1=(12,14,15,17,14,18)
          min(t1)

Out[33]:  12

In [34]:  t2=("ram","zara","anusha")
          min(t2)

Out[34]:  'anusha'
```

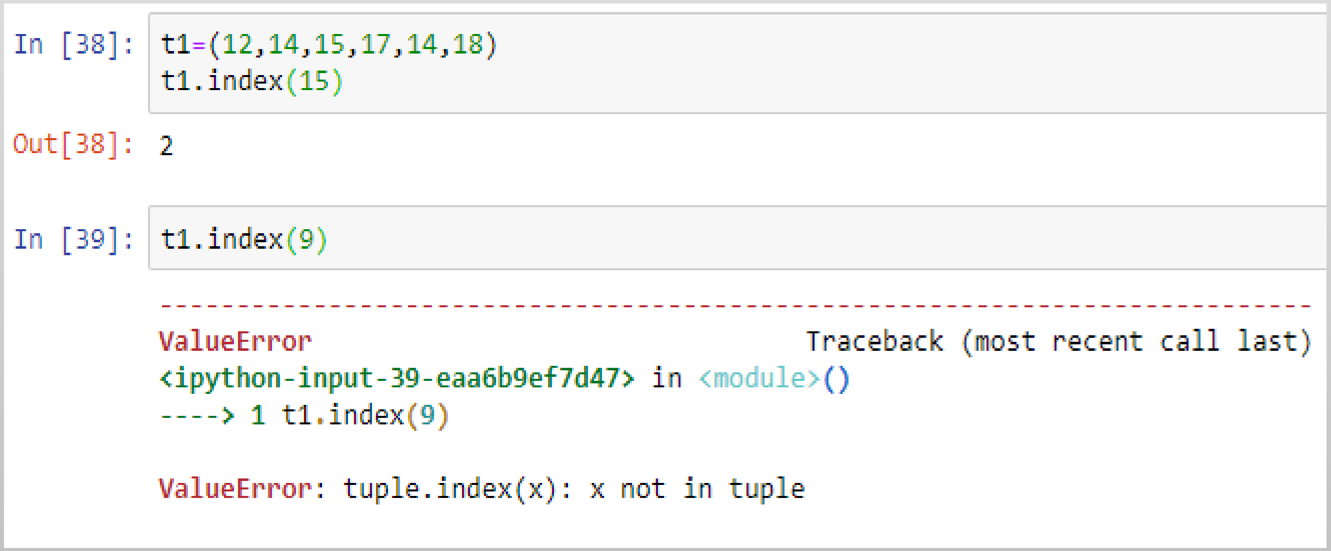
The index( ) Method

This function returns the index of first matched item from the tuple.

Syntax:

Tuple.index(<item>)

Example:



```
In [38]: t1=(12,14,15,17,14,18)
         t1.index(15)

Out[38]: 2

In [39]: t1.index(9)

         ---------------------------------------------------------------------------
         ValueError                                Traceback (most recent call last)
         <ipython-input-39-eaa6b9ef7d47> in <module>()
         ----> 1 t1.index(9)

         ValueError: tuple.index(x): x not in tuple
```

Note: If item is not in the list it raises exception value

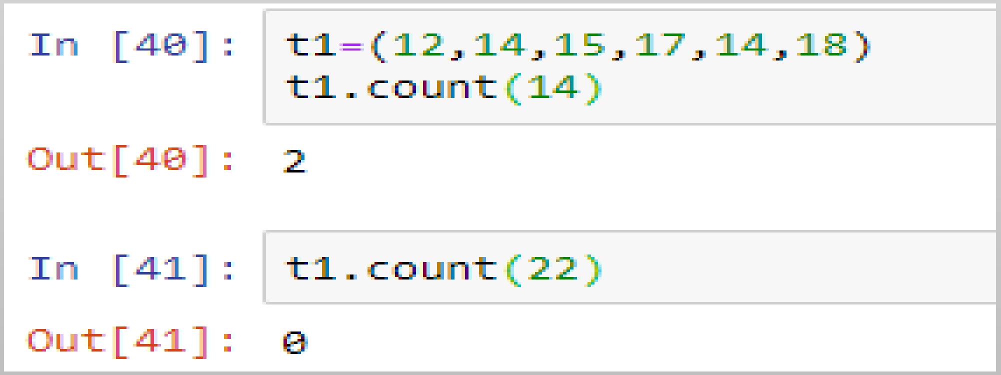
The count( ) Method

- This function returns the count of the item passed as argument. If given item is not in the tuple it returns zero.

Syntax:

tuple.count(<item>)

Example:

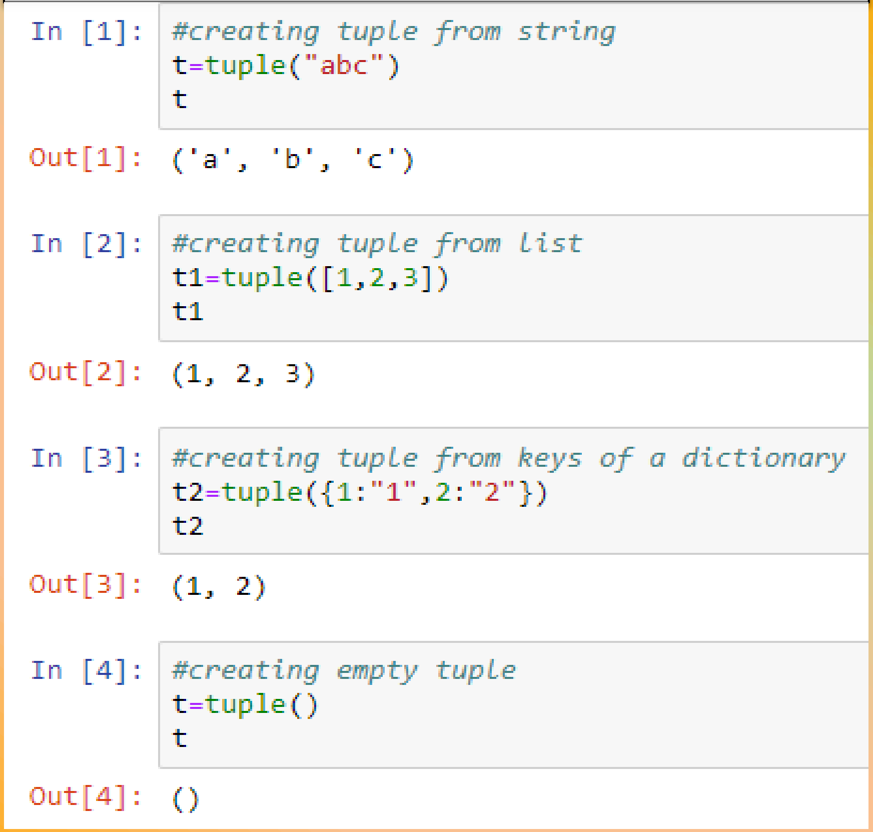


```
In [40]:  t1=(12,14,15,17,14,18)
          t1.count(14)

Out[40]:  2

In [41]:  t1.count(22)

Out[41]:  0
```

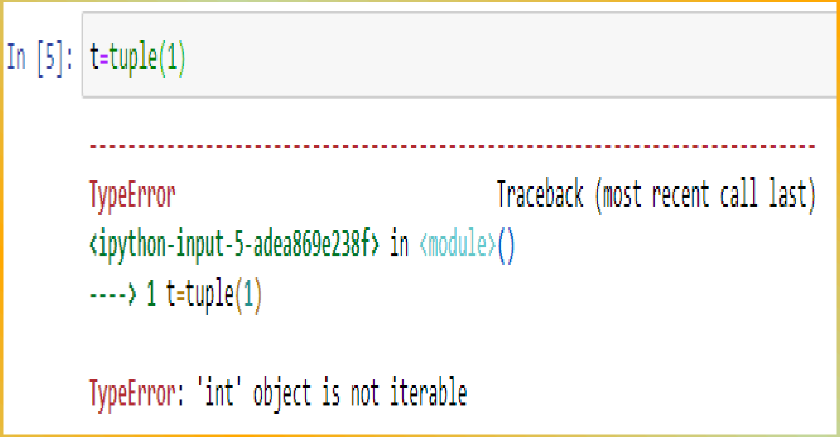
The tuple( ) Method

This function creates tuples from different types of values.

Syntax: tuple(<sequence>)



```
In [1]:  #creating tuple from string
         t=tuple("abc")
         t

Out[1]:  ('a', 'b', 'c')

In [2]:  #creating tuple from list
         t1=tuple([1,2,3])
         t1

Out[2]:  (1, 2, 3)

In [3]:  #creating tuple from keys of a dictionary
         t2=tuple({1:"1",2:"2"})
         t2

Out[3]:  (1, 2)

In [4]:  #creating empty tuple
         t=tuple()
         t

Out[4]:  ()
```



```
In [5]:  t=tuple(1)

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-5-adea869e238f> in <module>()
         ----> 1 t=tuple(1)

         TypeError: 'int' object is not iterable
```

Note: tuple( ) can receive argument of sequence type only, like string or list or dictionary.

Any other type of value will lead to an error