Python Tuples

A tuple in Python is similar to a list. The difference between the two is that we cannot change the elements of a tuple once it is assigned whereas we can change the elements of a list.

In short, a tuple is an immutable list. A tuple can not be changed in any way once it is created.

Characterstics

- Ordered
- Unchangeble
- Allows duplicate

Creating tuple

In [16]:
```
# empty
t1 = ()
print(t1)
# create a tuple with a single item
t2 = ('hello',)
print(t2)
print(type(t2))
# homo
t3 = (1,2,3,4)
print(t3)
# hetro
t4 = (1,2.5,True,[1,2,3])
print(t4)
# tuple
t5 = (1,2,3,(4,5))
print(t5)
# using type conversion
t6 = tuple('hello')
print(t6)
```

```
()
('hello',)
<class 'tuple'>
(1, 2, 3, 4)
(1, 2.5, True, [1, 2, 3])
(1, 2, 3, (4, 5))
('h', 'e', 'l', 'l', 'o')
```

Accessing items

- Indexing
- Slicing

In [39]:
```
# indexing
t=(1,2.34,'hello',[1,3,4,5])
print(t[0],t[3])
print(t[-1],t[2][1])
```

```
1 [1, 3, 4, 5]
[1, 3, 4, 5] e
```

In [54]:
```
# slicing
t=(1,2.34,'hello',[1,3,4,5])
print(t[::-1])
print(t[-1:-4:-1])
```

```
([1, 3, 4, 5], 'hello', 2.34, 1)
([1, 3, 4, 5], 'hello', 2.34)
```

Editing items

Since tuple is immutable so we cannot do any type of editing

Adding items

Immutability not allow to adding any item in a tuple

Deleting items

We can only delete whole tuple using del keyword like strings

In [37]:
```
t=(3,4,5)
del t
```

Operations on tuples

In [61]:
```
# can apply only * and + opertation on tuple
t1 = (1,2,3,4)
t2 = (5,6,7,8)

print(t1 + t2)

print(t1*3)
# membership
print(1 in t1)
# iteration
for i in t1:
  print(i)
```

```
(1, 2, 3, 4, 5, 6, 7, 8)
(1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4)
True
1
2
3
4
```

Tuples Functions

In [65]:
```
# len/sum/min/max/sorted
t = (1,2,3,4)
print(len(t),sum(t),min(t),max(t),sorted(t,reverse=True))
```

```
4 10 1 4 [4, 3, 2, 1]
```

In [73]:
```
# count
t = (1,2,3,4,5)
print(t.count(50))
# index
print(t.index(5))
```

```
0
4
```

Difference between lists and tuples

- Syntax
- Mutability
- Speed
- Memory
- Built in functionality - more in list
- Error prone - list is more error prone
- Usability

In [81]:
```
# tuple is fast than list
import time

L = list(range(100000000))
T = tuple(range(100000000))

start = time.time()
for i in L:
  i*5
print('List time',time.time()-start)

start = time.time()
for i in T:
  i*5
print('Tuple time',time.time()-start)
```

```
List time 6.208082914352417
Tuple time 5.773765802383423
```

In [84]:
```python
# tuple takes less space than list
import sys

L = list(range(1000))
T = tuple(range(1000))

print('List size',sys.getsizeof(L))
print('Tuple size',sys.getsizeof(T))
```

```
List size 8056
Tuple size 8040
```

In [89]:
```python
a = [1,2,3]
b = a

a.append(4)
print(a)
print(b)
```

```
[1, 2, 3, 4]
[1, 2, 3, 4]
```

In [87]:
```python
a = (1,2,3)
b = a

a = a + (4,)
print(a)
print(b)
```

```
(1, 2, 3, 4)
(1, 2, 3)
```

Special Syntax

In [93]:
```python
# tuple unpacking
a,b,c = (1,2,3)
print(a,b,c)
```

```
1 2 3
```

In [95]:
```
a = 1
b = 2
a,b = b,a

print(a,b)
```

```
2 1
```

In [98]:
```
a,b,*others = (1,2,3,4)
print(a,b)
print(others)
```

```
1 2
[3, 4]
```

Zipping in tuple

In [102…
```
# zipping tuples
a = (1,2,3,4)
b = (5,6,7,8)

tuple(zip(a,b))
```

Out[102…
```
((1, 5), (2, 6), (3, 7), (4, 8))
```

END