Tuples

Immutable list-like collection.

Characteristics:

- Ordered elements.
- Unmodifiable post-creation.
- Allows duplicates.

1. Create
2. Access
3. Edit
4. Add
5. Delete
6. Operations
7. Functions

1. Create

In [36]:
```
# empty
T1 = ()
T1
```

Out[36]:
```
()
```

In [11]:
```
# homo
T2 = (1, 2, 3, 4, 5)
T2
```

Out[11]:
```
(1, 2, 3, 4, 5)
```

In [12]:
```
# hetro
T3 = ("Hello", 4, 5, 6)
T3
```
Out[12]:
```
('Hello', 4, 5, 6)
```

In [13]:
```
# tuple
T4 = (1, 2, 3, (4, 5))
T4
```
Out[13]:
```
(1, 2, 3, (4, 5))
```

In [14]:
```
T5 = (1,)
T5
```
Out[14]:
```
(1,)
```

In [15]:
```
type(T5)
```
Out[15]:
```
tuple
```

In [16]:
```
# Type Conversion
T5 = ("Hello")
type(T5)
```
Out[16]:
```
str
```

In [17]:
```
# Single-item tuple creation
T5 = ("Hello",)
type(T5)
```
Out[17]:
```
tuple
```

In [18]:
```
T6 = tuple("Goa")
T6
```
Out[18]:
```
('G', 'o', 'a')
```

In [19]:
```
T6 = tuple([ 1, 2, 3, 4])
T6
```

Out[19]:
```
(1, 2, 3, 4)
```

2. Access

- Indexing
- Slicing

In [20]:
```
T2
```

Out[20]:
```
(1, 2, 3, 4, 5)
```

In [21]:
```
T2[3]
```

Out[21]:
```
4
```

In [22]:
```
T2[-3]
```

Out[22]:
```
3
```

In [23]:
```
T2[:3]
```

Out[23]:
```
(1, 2, 3)
```

In [24]:
```
T4
```

Out[24]:
```
(1, 2, 3, (4, 5))
```

In [30]:
```
T4[3][-2]
```

Out[30]:
```
4
```

3. Edit

In [31]:
```
L = [1, 2, 3, 4, 5]
```

In [32]:
```
L[0] = 100
L
```

Out[32]:
```
[100, 2, 3, 4, 5]
```

In [33]:
```
T2
```

Out[33]:
```
(1, 2, 3, 4, 5)
```

In [34]:
```
T2[0] = 100
# Immutable, like strings
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[34], line 1
----> 1 T2[0] = 100

TypeError: 'tuple' object does not support item assignment
```

In [21]:
```
# Tuples = immutable (like strings)
```

4. Add

In [29]:
```
# not possible
# Tuples: immutable
```

5. Delete

```
In [37]: T1
```

```
Out[37]: ()
```

```
In [38]: del T1
         T1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[38], line 2
      1 del T1
----> 2 T1

NameError: name 'T1' is not defined
```

```
In [39]: T3
```

```
Out[39]: ('Hello', 4, 5, 6)
```

```
In [40]: T2
```

```
Out[40]: (1, 2, 3, 4, 5)
```

```
In [41]: del T2(-1)
```

```
  Cell In[41], line 1
    del T2(-1)
        ^
SyntaxError: cannot delete function call
```

```
In [28]: # Tuples are immutable
```

6. Operations

```
In [42]: T2
```

```
Out[42]: (1, 2, 3, 4, 5)
```

```
In [43]: T3
```

```
Out[43]: ('Hello', 4, 5, 6)
```

```
In [ ]: # + and *
```

```
In [44]: T2 + T3
```

```
Out[44]: (1, 2, 3, 4, 5, 'Hello', 4, 5, 6)
```

```
In [45]: T2 * 3
```

```
Out[45]: (1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5)
```

```
In [46]: # iteration
         for i in T2:
             print(i)
```

```
1
2
3
4
5
```

In [50]:
```
# membership
2 in T2
```
Out[50]:
```
True
```

7. Functions

In [51]:
```
len(T2)
```
Out[51]:
```
5
```

In [52]:
```
min(T2)
```
Out[52]:
```
1
```

In [53]:
```
max(T2)
```
Out[53]:
```
5
```

In [15]:
```
sum(T2)
```
Out[15]:
```
15
```

In [56]:
```
sorted(T2)
```
Out[56]:
```
[1, 2, 3, 4, 5]
```

In [57]:
```
sorted(T2, reverse = True)
```
Out[57]:
```
[5, 4, 3, 2, 1]
```

In [60]:
```
# count
t = (1, 2, 2, 3, 4, 5)
t.count(2)
```
Out[60]:
```
1
```

In [61]:
```
# index
t.index(3)
```

Out[61]:
```
3
```

Lists vs Tuples

Syntax:

- Lists: `[ ]`
- Tuples: `( )`

Mutability:

- Lists: Mutable
- Tuples: Immutable

Speed:

- Lists: Slower (mutable)
- Tuples: Faster (immutable)

Memory:

- Lists: Higher
- Tuples: Lower

Functionality:

- Both: Indexing, slicing
- Lists: More methods

Error-Prone:

- Lists: Modifiable
- Tuples: Safer

Use Case:

In [63]:

```python
import time
L = list(range(10000))
T = tuple(range(10000))

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

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

```
List time 0.009129047393798828
Tuple time 0.0020101070404052734
```

In [2]:

```python
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 [64]:
```
a = [1, 2, 3]
b = a
a.append(4)
print(a)
print(b)
```

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

In [22]:
```
a = (1, 2, 3)
b = a
a = a + (4,)
print(a)
print(b)
```

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

Q. Why use Tuples?

- Immutable; prevents changes.
- Ensures data integrity.
- Use for fixed collections.
- Example:

```
college_database = ('CS', 'Math', 'Physics')
# college_database[0] = 'Electronics'  # TypeError
```

- Use for static data; lists for mutable.

Special Syntax

In [23]:
```
# tuple unpacking
a, b, c = (1, 2, 3)
print(a, b, c)
```
```
1 2 3
```

In [24]:
```
a, b = (1, 2, 3)
print(a, b)
```
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[24], line 1
----> 1 a, b = (1, 2, 3)
      2 print(a, b)

ValueError: too many values to unpack (expected 2)
```

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

In [68]:
```
a, b, *others = (1, 2, 3,4)
print(a, b)
print(others)
```
```
1 2
[3, 4]
```

In [77]:
```
# zipping tuples
a = (1, 2, 3, 4)
b = (5, 6, 7, 8)
tuple(zip(a,b))
```

Out[77]:
```
((1, 5), (2, 6), (3, 7), (4, 8))
```