Tuples

1. Tuple is similar to List except tuple are immutable which means we cannot change the elements of a tuple once assigned.
2. When we do not want to change the data over time, tuple is a preferred data type.
3. Iterating over the elements of a tuple is faster compared to iterating over a list.

Empty tuple

In [1]:

```
#option-1

tup1 = ()
print(tup1)
```

```
()
```

In [2]:

```
#option-2

tup2=tuple()
print(tup2)
```

```
()
```

Tuple is heterogenous

In [3]:

```
t=(1,0.2,True,'new',[1,4])
t
```

Out[3]:

```
(1, 0.2, True, 'new', [1, 4])
```

In [4]:

```
len(t)
```

Out[4]:

```
5
```

Nested tuple

In [5]:

```
tup = ('siva', 25 ,25.0,True, [50, 100], (99,22,33))
tup
```

Out[5]:

```
('siva', 25, 25.0, True, [50, 100], (99, 22, 33))
```

Tuple Indexing

In [6]:

```
tup[1]
```

Out[6]:

```
25
```

In [7]:

```
tup[-1][1]
```

Out[7]:

```
22
```

Tuple Slicing

In [8]:

```
tup[0:5]
```

Out[8]:

```
('siva', 25, 25.0, True, [50, 100])
```

Tuple Methods

In [9]:

```
mytuple1 =(1,1,2,3,4,5,1,2,3,4,1,1)
mytuple1
```

Out[9]:

```
(1, 1, 2, 3, 4, 5, 1, 2, 3, 4, 1, 1)
```

tuple.count() --> Number of times item occurred in the tuple.

In [10]:

```
mytuple1.count(1) # Number of times item 1 occurred in the tuple.
```

Out[10]:

```
5
```

tuple.index()

In [11]:

```
mytuple1.index(1) # Index of first element equal to 'one'
```

Out[11]:

```
0
```

Tuples are immutable

can't replace a item in a tuple

In [12]:

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

t[1] = 40
print(t)
```

```
---------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[12], line 3
      1 t=(1,2,3,4)
----> 3 t[1] = 40
      4 print(t)

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

can't remove a item in a tuple

In [13]:

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

del t[0]
print(t)
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[13], line 3
      1 t = (1,2,3,4)
----> 3 del t[0]
      4 print(t)

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

delete entire tuple

In [14]:

```
mytuple = (1,2,3)

del mytuple            # Deleting entire tuple object is possible
```

sorting

In [15]:

```
mytuple = (43,67,99,12,6,90,67)

a = tuple(sorted(mytuple))
print(a)
```

```
(6, 12, 43, 67, 67, 90, 99)
```

convert tuple to list

In [16]:

```
t=(1,2,3,4)
print(t)

l=list(t)
print(l)
```

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

if we want to modify item in tuple

- convert the tuple to list

- do the modification
- again convert back to tuple

In [17]:

```
t=(1,2,3,4,5)
print(t)
print(id(t))

a = list(t)
del a[0]
t = tuple(a)
print(t)
print(id(t))
```

```
(1, 2, 3, 4, 5)
2048104185056
(2, 3, 4, 5)
2048118243392
```

interview Question

In [18]:

```
t=(1,40,60,[20,15,5],80,100)
```

can we add a value in a list within a tuple?

In [19]:

```
t[3].append(25)
t
```

Out[19]:

```
(1, 40, 60, [20, 15, 5, 25], 80, 100)
```

can we repalce a value in a list within a tuple?

In [20]:

```
t[3][1] = 55
t
```

Out[20]:

```
(1, 40, 60, [20, 55, 5, 25], 80, 100)
```

can we remove a value in a list within a tuple?

In [21]:

```
t[3].remove(55)
t
```

Out[21]:

```
(1, 40, 60, [20, 5, 25], 80, 100)
```