Tuples

-> A tuple is similar to list

-> The diffence between the two is that we can't change the elements of tuple once it is assigned whereas in the list, elements can be changed

Tuple creation

In [2]:

```
#empty tuple
t = ()

#tuple having integers
t = (1, 2, 3)
print(t)

#tuple with mixed datatypes
t = (1, 'Milan', 28, 'abc')
print(t)

#nested tuple
t = (1, (2, 3, 4), [1, 'raju', 28, 'abc'])
print(t)
```

```
(1, 2, 3)
(1, 'raju', 28, 'abc')
(1, (2, 3, 4), [1, 'raju', 28, 'abc'])
```

In [1]:

```
#only parenthesis is not enough
t = ('Milan')
type(t)
```

Out[1]:

```
str
```

In [4]:

```
#need a comma at the end
t = ('satish',)
type(t)
```

Out[4]:

```
tuple
```

In [7]:

```
#parenthesis is optional
t = "Milan",
print(type(t))

print(t)
```

```
<class 'tuple'>
('satish',)
```

Accessing Elements in Tuple

In [4]:

```
t = ('Milan', 'Prashant', 'Dhananjay', 'Suraj', 'You')

print(t[1])
```

```
Prashant
```

In [5]:

```
#negative index
print(t[-1]) #print last element in a tuple
```

```
You
```

In [6]:

```
#nested tuple
t = ('ABC', ('Me', 'You', 'We'))

print(t[1])
```

```
('Me', 'You', 'We')
```

In [7]:

```
print(t[1][2])
```

```
We
```

In [1]:

```
#Slicing
t = (1, 2, 3, 4, 5, 6)

print(t[1:4])

#print elements from starting to 2nd last elements
print(t[:-2])

#print elements from starting to end
print(t[:])
```

```
(2, 3, 4)
(1, 2, 3, 4)
(1, 2, 3, 4, 5, 6)
```

Changing a Tuple

#unlike lists, tuples are immutable

#This means that elements of a tuple cannot be changed once it has been assigned. But, if the element is itself a mutable datatype like list, its nested items can be changed.

In [3]:

```
#creating tuple
t = (1, 2, 3, 4, [5, 6, 7])

t[2] = 'x' #will get TypeError
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-3-9f4cbf6ee0de> in <module>()
      2 t = (1, 2, 3, 4, [5, 6, 7])
      3 
----> 4 t[2] = 'x' #will get TypeError

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

In [4]:

```
t[4][1] = 'satish'
print(t)
```

```
(1, 2, 3, 4, [5, 'satish', 7])
```

In [15]:

```
#concatinating tuples

t = (1, 2, 3) + (4, 5, 6)
print(t)
```

```
(1, 2, 3, 4, 5, 6)
```

In [16]:

```
#repeat the elements in a tuple for a given number of times using the * operator.
t = (('satish', ) * 4)
print(t)
```

```
('satish', 'satish', 'satish', 'satish')
```

Tuple Deletion

In [18]:

```
#we cannot change the elements in a tuple.
# That also means we cannot delete or remove items from a tuple.

#delete entire tuple using del keyword
t = (1, 2, 3, 4, 5, 6)

#delete entire tuple
del t
```

Tuple Count

In [19]:

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

#get the frequency of particular element appears in a tuple
t.count(1)
```

Out[19]:

```
3
```

Tuple Index

In [22]:

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

print(t.index(3)) #return index of the first element is equal to 3

#print index of the 1
```

```
2
```

Tuple Memebership

In [23]:

```
#test if an item exists in a tuple or not, using the keyword in.
t = (1, 2, 3, 4, 5, 6)

print(1 in t)
```

```
True
```

In [24]:

```
print(7 in t)
```

```
False
```

Built in Functions

Tuple Length

In [25]:

```
t = (1, 2, 3, 4, 5, 6)
print(len(t))
```

```
6
```

Tuple Sort

In [26]:

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

new_t = sorted(t)
print(new_t) #Take elements in the tuple and return a new sorted list
             #(does not sort the tuple itself).
```

```
[1, 2, 3, 4, 5]
```

In [27]:

```
#get the largest element in a tuple
t = (2, 5, 1, 6, 9)

print(max(t))
```

```
9
```

In [28]:

```
#get the smallest element in a tuple
print(min(t))
```

```
1
```

In [29]:

```
#get sum of elments in the tuple
print(sum(t))
```

```
23
```