Tuples

- Used to store multiple items in a single variable.
- Ordered
- unchangeable., Allow Duplicate values.
- are written with round brackets. Indexing & slicing are allowed in List & Tuple.

ex:

```
tuple = ("a", "b", "c")
```

Create tuple with one item

- TO create a tuple, you have to add a comma after the item.

ex:

```
atuple = ("apple", )
print (type (atuple))
```

#<class 'tuple'>

ex:

```
a tuple = ("apple")
print ( type (atuple))
```

<class 'str'>.

(2) ways to create an empty tuple.

Using round bracket.

```
a = ()
print (type(a))
```

O/P:

<class 'tuple'>

using tuple() fn.

```
a = tuple()
print (type(a))
```

O/P:

<class 'tuple'>

Mixed Tuple

- we can store anything in tuple like int, float, string, boolean, tuples, list, dictionary etc...

ex: a = ('agi', True, 8, 10, 8.10, [8, 10])

Methods:

a) **Index()** → Returns the index of 1st occ. of an elem. in the tuple.

ex: r = ('a', 'b', 'd', 'a', 'e')

r.index('a') → 0.

r.index('b') → 1.

ii) count()

Return the no. of occ. of value present in the tuple, else 0.

ex:- r = ('a', 'b', 'd', 'a', 'e')

r.count('a') → 2

r.count('e') → 1

r.count('c') → 0.

Unpacking a Tuple

- when we create a tuple, we normally assign values to it. That's called "packing"

- But in python, we are also allowed to extract the values back into variables. This is called "unpacking"

ex:- c = ("abc", "agi", "k").

(c1, c2, c3) = c.

print (c1) → abc

(c2) → agi

(c3) → k.

Using Asterisk (*)

If no. of var. is less than no. of values, then we can add * to the var. name & values will be assigned to the var. as a list.

ex:

```
cl = ("abc", "agi", "K", "C#",
      "Ho")

(a, b, *c) = cl.

print (a) → abc
      (b) → agi
      (c) → ['K', 'C#', 'Ho'].
```

If the * is added to another var. name than the last, then it will assign value to var. until the no. of matches left

ex: cl = ("abc", "agi", "K", "C#", "Ho")

```
(a, *b, c) = cl.

print (a) → abc
      (b) → ['agi', 'K', 'C#']
      (c) → Ho.
```