# Tuple

Tuple – A tuple contains a group of elements which can be same or different types.

Tuples are immutable.

It is similar to List but Tuples are read-only which means we can not modify it’s element.

Tuples are used to store data which should not be modified.

It occupies less memory compare to list.

Tuples are represented using parentheses ( ).

Ex:- a = (10, 20, -50, 21.3, ‘Geekyshows’)

# Creating Empty Tuple

Syntax:- tuple_name = ( )

Ex:- a = ( )

# Creating Tuple

We can create tuple by writing elements separated by commas inside parentheses.

## With one Element

b = (10)

It will become integer

c  = (10, )

## With Multiple Elements

d = (10, 20, 30, 40)

e = (10, 20, -50, 21.3, ‘GeekyShows’)

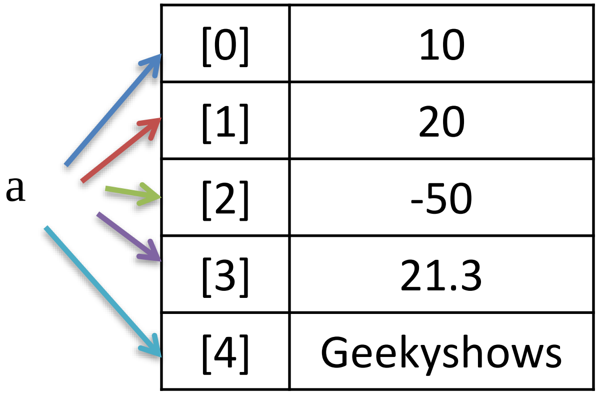
f = 10, 20, -50, 21.3, ‘GeekyShows’

It will become a tuple

# Index

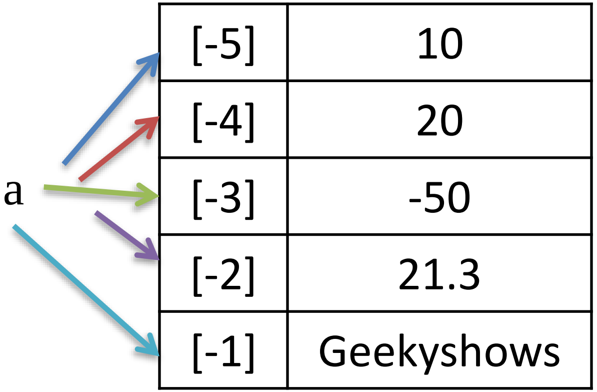
An index represents the position number of an tuple’s element. The index start from 0 on wards and written inside square braces.

Ex:- a = (10, 20, -50, 21.3, ‘Geekyshows’)



a

|  |  |
|---|---|
| [0] | 10 |
| [1] | 20 |
| [2] | -50 |
| [3] | 21.3 |
| [4] | Geekyshows |



a

|  |  |
|---|---|
| [-5] | 10 |
| [-4] | 20 |
| [-3] | -50 |
| [-2] | 21.3 |
| [-1] | Geekyshows |

# Accessing Tuple’s Element

```
a = (10, 20, -50, 21.3, ‘Geekyshows’)

print(a[0])

print(a[1])

print(a[2])

print(a[3])

print(a[4])
```

| | | | | |
|---|---|---|---|---|
| 10 | 20 | -50 | 21.3 | Geekyshows |
| a[0] | a[1] | a[2] | a[3] | a[4] |

# Accessing using for loop

```
a = (10, 20, -50, 21.3, ‘Geekyshows’)
```

Without index

```
for element in a:
		print(element)
```

With index

```
n = len(a)
for i in range(n):
		print(a[i])
```

# Accessing using while loop

```
a = (10, 20, -50, 21.3, ‘Geekyshows’)

n = len(a)
i = 0
while i < n :
            print(a[i])
            i+=1
```

# Slicing on Tuple

Slicing on tuple can be used to retrieve a piece of the tuple that contains a group of elements. Slicing is useful to retrieve a range of elements.

Syntax:-

new_tuple_name = tuple_name[start:stop:stepsize]

# Tuple Concatenation

+ operator is used to do concatenation the tuple.

Ex:-

```
a = (10, 20, 30)
b = (1, 2, 3)
result = a + b
print(result)
```

# Modifying Element

Tuples are immutable so it is not possible to modify, update or delete it’s element.

a = (10, 20, -50, 21.3, ‘Geekyshows’)

*a*[1] = 40

| | | | | |
|---|---|---|---|---|
| 10 | 20 | -50 | 21.3 | Geekyshows |
| a[0] | a[1] | a[2] | a[3] | a[4] |

b = (101, 102, 103)

t = a + b

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 10 | 20 | -50 | 21.3 | GeekyShows | 101 | 102 | 103 |
| t[0] | t[1] | t[2] | t[3] | t[4] | t[5] | t[6] | t[7] |

# Deleting Tuple

You can delete entire tuple but not an element of tuple.

a = (10, 20, -50, 21.3, ‘Geekyshows’)

del a

| | | | | |
|---|---|---|---|---|
| 10 | 20 | -50 | 21.3 | Geekyshows |
| a[0] | a[1] | a[2] | a[3] | a[4] |

# Repetition of Tuple

* Operator is used to repeat the elements of tuple.

Ex:-

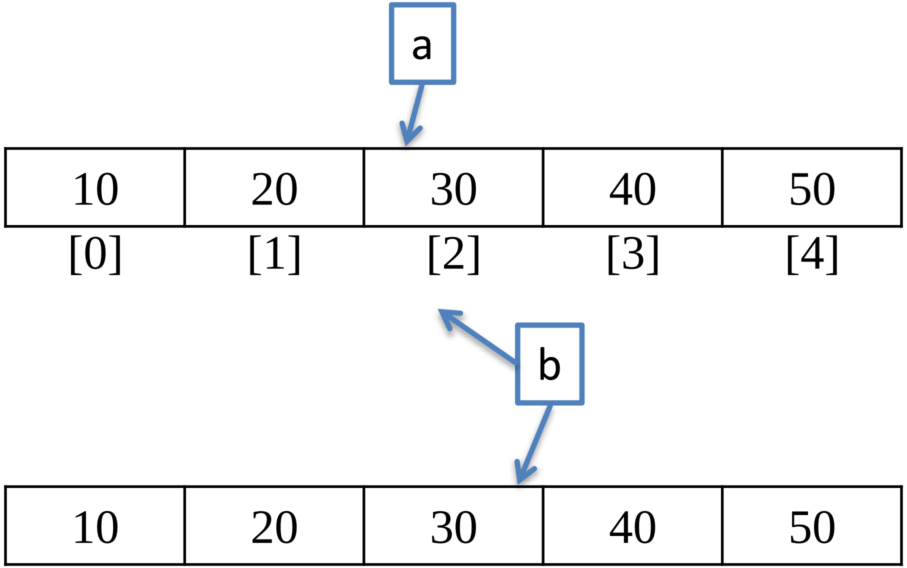
```
b = (1, 2, 3)
result = b * 3
print(result)
```

# Aliasing Tuple

Aliasing means giving another name to the existing object. It doesn’t mean copying.

a = (10, 20, 30, 40, 50)

b = a



a

| 10 | 20 | 30 | 40 | 50 |
|---|---|---|---|---|
| [0] | [1] | [2] | [3] | [4] |

b

| 10 | 20 | 30 | 40 | 50 |
|---|---|---|---|---|

# Copying Tuple

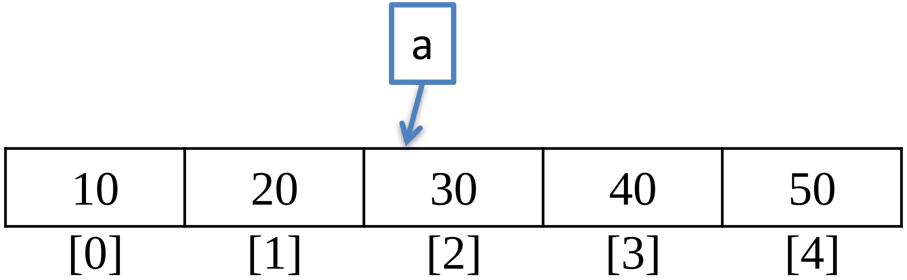
We can copy elements of tuple into another tuple using slice.

a = (10, 20, 30, 40, 50)

b = a

b = a[0:5]

b = a[1:4]



| 10 | 20 | 30 | 40 | 50 |
|---|---|---|---|---|
| [0] | [1] | [2] | [3] | [4] |