# Python Tuples

**Tuple** is a collection of Python objects much like a list. The sequence of values stored in a tuple can be of any type, and they are indexed by integers.
Values of a tuple are syntactically separated by ‘commas’. Although it is not necessary, it is more common to define a tuple by closing the sequence of values in parentheses. This helps in understanding the Python tuples more easily.

### Creating a Tuple

In Python, tuples are created by placing a sequence of values separated by ‘comma’ with or without the use of parentheses for grouping the data sequence.

**Note:** Creation of Python tuple without the use of parentheses is known as Tuple Packing.

```
# Creating an empty Tuple

Tuple1 = ()

print("Initial empty Tuple: ")

print(Tuple1)

# Creating a Tuple with the use of string

Tuple1 = ('Disha', 'Computer')

print("\nTuple with the use of String: ")

print(Tuple1)

# Creating a Tuple with the use of list

list1 = [1, 2, 4, 5, 6]

print("\nTuple using List: ")

print(tuple(list1))

# Creating a Tuple with the use of built-in function

Tuple1 = tuple('Disha')

print("\nTuple with the use of function: ")

print(Tuple1)
```

```
# Creating a Tuple with Mixed Datatype
Tuple1 = (5, 'Welcome', 7, 'Geeks')
print("\nTuple with Mixed Datatypes: ")
print(Tuple1)
# Creating a Tuple with nested tuples
Tuple1 = (0, 1, 2, 3)
Tuple2 = ('python', 'geek')
Tuple3 = (Tuple1, Tuple2)
print("\nTuple with nested tuples: ")
print(Tuple3)
# Creating a Tuple with repetition
Tuple1 = ('Geeks',) * 3
print("\nTuple with repetition: ")
print(Tuple1)


# Creating a Tuple  with the use of loop
Tuple1 = ('Geeks')
n = 5
print("\nTuple with a loop")
for i in range(int(n)):
        Tuple1 = (Tuple1,)
        print(Tuple1)
```

## Accessing of Tuples

```
# Accessing Tuple
# with Indexing
Tuple1 = tuple("Geeks")
print("\nFirst element of Tuple: ")
print(Tuple1[0])
```

## Concatenation of Tuples

Concatenation of tuple is the process of joining two or more Tuples. Concatenation is done by the use of '+' operator. Concatenation of tuples is done always from the end of the original tuple. Other arithmetic operations do not apply on Tuples.

**Note-** Only the same datatypes can be combined with concatenation, an error arises if a list and a tuple are combined.

```
# Concatenation of tuples

Tuple1 = (0, 1, 2, 3)

Tuple2 = ('Disha', 'Computer', 'Institute')

Tuple3 = Tuple1 + Tuple2

print("Tuple 1: ")

print(Tuple1)

print("\nTuple2: ")

print(Tuple2)

print("\nTuples after Concatenation: ")

print(Tuple3)
```

## Slicing of Tuple

```
# Slicing of a Tuple with Numbers

Tuple1 = tuple('DISHACOMPUTERINSTITUTE')

print("Removal of First Element: ")

print(Tuple1[1:])

print("\nTuple after sequence of Element is reversed: ")

print(Tuple1[::-1])

print("\nPrinting elements between Range 4-9: ")

print(Tuple1[4:9])
```

**Deleting a Tuple**

Tuples are immutable and hence they do not allow deletion of a part of it. The entire tuple gets deleted by the use of del() method.

**Note-** Printing of Tuple after deletion results in an Error.

```
# Deleting a Tuple

Tuple1 = (0, 1, 2, 3, 4)

del Tuple1

print(Tuple1)
```

Example 1: Len() function with tuples and string

```
# Python program to demonstrate the use of len() method

tup = (1,2,3)

print(len(tup))
```