









Data Structures-II

Session Objectives

-  Understand what tuples are.
-  Understand common methods and operations associated with tuples.
-  Understand what sets are.
-  Understand common methods and operations associated with sets.
-  Understand the comparison between lists, tuples and sets.
-  Understand what dictionaries are.
-  Understand common methods and operations associated with dictionaries.
-  Understand the comparison between lists, tuples, sets and dictionaries.

```
# Immutability of a Tuple
# We Can't change or add any elements directly
# Tuples don't have : - append, extend , remove, insert
# But we can do it indirectly by modifying it into List
country_tuple = ('India', 'America', 'Russia', 'China', 'Canada', 'Japan',
                 'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia',
                 'Spain', 'New Zealand', 'Finland')
conv_tuple_to_list = list(country_tuple)
print(type(conv_tuple_to_list)) # <class List>
conv_tuple_to_list.append('Germany')
print(conv_tuple_to_list)
country_tuple = tuple(conv_tuple_to_list)
print(country_tuple)
print(type(country_tuple))

<class 'list'>
['India', 'America', 'Russia', 'China', 'Canada', 'Japan', 'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia', 'Spain', 'New Zealand', 'Finland', 'Germany']
('India', 'America', 'Russia', 'China', 'Canada', 'Japan', 'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia', 'Spain', 'New Zealand', 'Finland', 'Germany')
<class 'tuple'>
```

```
country_tuple = ('India', 'America', 'Russia', 'China', 'Canada', 'Japan',
                 'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia',
                 'Spain', 'New Zealand', 'Finland')
country_tuple = ('India', 'America', 'Russia', 'China', 'Canada', 'Japan',
                 'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia',
                 'Spain', 'New Zealand', 'Finland', 'Germany')
```

```
country_tuple
```

```
('India',
 'America',
 'Russia',
 'China',
 'Canada',
 'Japan',
 'Vietnam',
 'Sri-Lanka',
 'France',
 'Singapore',
 'Australia',
 'Spain',
 'New Zealand',
 'Finland',
 'Germany')
```

```
# Concatenating A tuples
```

```
weekday_tuple = ('Mon', 'Tues', 'Wed', 'Thurs', 'Fri')
```

```
weekend_tuple = ('Sat', 'Sun')
```

```
week_tuple = weekday_tuple + weekend_tuple
```

```
print(week_tuple)
```

```
('Mon', 'Tues', 'Wed', 'Thurs', 'Fri', 'Sat', 'Sun')
```

```
weekday_tuple = (['Mon', 'Tues'],)
```

```
weekend_tuple = ('Sat', 'Sun')
```

```
week_tuple = weekday_tuple + weekend_tuple
```

```
print(week_tuple)
```

```
(['Mon', 'Tues'], 'Sat', 'Sun')
```

```
# week_tuple[0] ~ ['Mon', 'Tues']
```

```
week_tuple[0].extend(['Wed', 'Thurs', 'Fri'])
```

```
print(week_tuple)
```

```
(['Mon', 'Tues', 'Wed', 'Thurs', 'Fri'], 'Sat', 'Sun')
```

```
weekday_tuple = ('Mon', 'Tues', 'Wed', 'Thurs', 'Fri')
```

```
weekend_tuple = ('Sat', 'Sun')
```

```
week_tuple = weekday_tuple[:3] + weekend_tuple
```

```
print(week_tuple)
```

```
('Mon', 'Tues', 'Wed', 'Sat', 'Sun')
```

```

week_tuple[2] = 'Wednesday' # TypeError: 'tuple' object does not support item assignment

country_list = ['India', 'America', 'Russia', 'China', 'Canada', 'Japan',
                'Vietnam', 'Sri-Lanka', 'france', 'singapore', 'australia',
                'Spain', 'New Zealand', 'Finland', 'India', 'Japan', 'America']
country_list[-5:] = ['Germany']
print(country_list)

['India', 'America', 'Russia', 'China', 'Canada', 'Japan', 'Vietnam', 'Sri-Lanka', 'france', 'singapore', 'australia', 'Spain', 'Germany']

```

```

# Deleting Element in Tuple Vs Deleting a Tuple
# You can't delete individual items in a tuple with 'del'
# But you can delete the entire tuple object.
weekday_tuple = ('Mon', 'Tues', 'Wed', 'Thurs', 'Fri')
print(weekday_tuple)
print(type(weekday_tuple))
# del weekday_tuple[3] # 'Thurs' # TypeError: 'tuple' object doesn't support item deletion
del weekday_tuple
# print(weekday_tuple) # NameError: name 'weekday_tuple' is not defined

('Mon', 'Tues', 'Wed', 'Thurs', 'Fri')
<class 'tuple'>

```

```

# Unpacking a Tuple
a,b,c = (1,2,3)
print(a)
print(b)
print(c)

1
2
3

a,*b,c = (1,2,3,4,5,6,7,8,9)
print(a)
print(b)
print(c)

1
[2, 3, 4, 5, 6, 7, 8]
9

```

```

a,*b,*c = (1,2,3,4,5,6,7,8,9) # SyntaxError: multiple starred expressions in assignment
print(a)
print(b)
print(c)

*a,b,c = (1,2,3,4,5,6,7,8,9)
print(tuple(a))
print(b) # 8
print(c) # 9

(1, 2, 3, 4, 5, 6, 7)
8
9

a,b,c = (1,2,3,4,5,6,7,8,9) # ValueError: too many values to unpack (expected 3)
print(a)
print(b)
print(c)

```



```
# Repeating tuples
_tuple = ('a','b','c','d','e')
print(_tuple * 3)

('a', 'b', 'c', 'd', 'e', 'a', 'b', 'c', 'd', 'e', 'a', 'b', 'c', 'd', 'e')

# Combining a Tuple
_tuple1 = (1,2,3,4,5)
_tuple2 = (False, True, 19.99, 'Coding','a','b','c')
new_tuple = _tuple1 + _tuple2
print(new_tuple)

(1, 2, 3, 4, 5, False, True, 19.99, 'Coding', 'a', 'b', 'c')
```

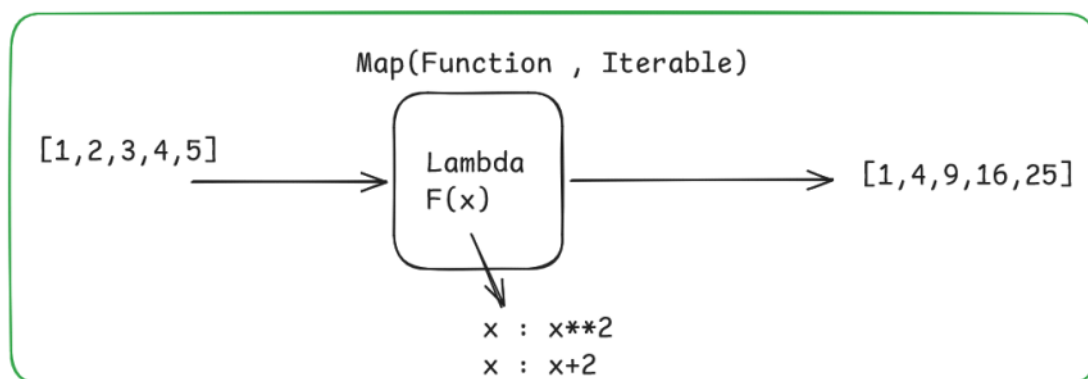
```
# .count()
country_tuple = ('India', 'America', 'Russia', 'China', 'Canada', 'Japan',
                 'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia',
                 'Spain', 'New Zealand', 'Finland', 'Germany', 'india', 'India')
country_tuple.count('India')

2

# .index()
country_tuple.index('Japan')

5

country_tuple.index('Norway') # ValueError: tuple.index(x): x not in tuple
```



Using map() in Python ¶

What is map()?

Syntax : map(function, iterable) where function uses (lambda function internally)

The map() function is a very handy tool in Python that let's you:

- Apply a Function to each item in a list , tuple or any other iterable
- Return a map object

```
# Square each elements
num_list = [1,2,3,4,5,6,7]
# map(function, iterable)
squared = map(lambda x : x ** 2 , num_list)
print(squared)
print(tuple(squared)) # (1,4,9,16,25,36,49)
print(list(squared)) # []
```

```
<map object at 0x0000025074C07040>
(1, 4, 9, 16, 25, 36, 49)
[]
```

```
# Adding 10 in an element
num_list = [1,2,3,4,5,6,7]
# map(function, iterable)
add_10 = map(lambda x : x + 10 , num_list)
print(add_10)
print(list(add_10)) # [11,12,13,14,15,16,17]
print(tuple(add_10)) # ()
```

```
<map object at 0x0000025074B321A0>
[11, 12, 13, 14, 15, 16, 17]
()
```

```
# Taking input from the user ->
numbers = list(map(int, input("Enter the multiple numbers : ").split()))
print(numbers)
```

```
Enter the multiple numbers : 10 20 30 40 50 60 70
[10, 20, 30, 40, 50, 60, 70]
```

```
# Taking input from the user ->
numbers = list(map(int, input("Enter the multiple numbers : ").split()))
print(numbers)
```

```
Enter the multiple numbers : 10 20 30 40 50 60 70
[10, 20, 30, 40, 50, 60, 70]
```

```
# Taking input from the user ->
string = list(map(str, input("Enter the multiple string : ").split()))
print(string)
```

```
Enter the multiple string : coding ninja python programming
['coding', 'ninja', 'python', 'programming']
```

What are Sets?

A Set in Python is:

- Unordered : No guaranteed order of elements
- Changeable (mutable): You can add or remove elements
- Unindexed : No way to access items by position
- Unique Items only : Automatically removes Duplicates

They can hold different data types together, like numbers, strings or boolean

```
_dict = {} # dictionary
# You can't assign {} to a set as by default empty curly braces represent dictionary {}
_set = set()
print(type(_dict))
print(type(_set))

<class 'dict'>
<class 'set'>
```

```
_set = {1}
print(type(_set))
```

```
<class 'set'>
```

```
_set = {1,2,3,'a','b','c',False,True,9.99,"coding"}
print(_set)
print(type(_set))
```

```
{False, 1, 2, 3, 9.99, 'a', 'coding', 'b', 'c'}
<class 'set'>
```

```
_set = {1,2,3,'a','b','c',False,True,9.99,"coding","Python","programming"}
print(_set)
```

```
{False, 1, 2, 3, 9.99, 'programming', 'a', 'coding', 'b', 'c', 'Python'}
```

```
_duplicate_set = {1,2,1,2,1,1,1,2,2,2,1,2}
print(_duplicate_set) # only store unique elements
```

```
{1, 2}
```

```
_duplicate_set = {0,1,0,0,1,2,1,2,1,1,1,2,2,2,1,2,False,True}
print(_duplicate_set) # only store unique elements
```

```
{0, 1, 2}
```

```
_duplicate_set = {False,True,0,1,0,0,1,2,1,2,1,1,1,2,2,2,1,2,False,True}
print(_duplicate_set) # only store unique elements
```

```
{False, True, 2}
```

```
# set() constructor to do the typecasting
_list = ['India', 'America', 'Russia', 'China', 'Canada', 'Japan',
         'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia',
         'Vietnam', 'Sri-Lanka', 'France', 'Singapore', 'Australia',
         'Spain', 'New Zealand', 'Finland', 'Germany', 'india', 'India']
_tuple = (False, True, 19.99, 'Coding', 'a', 'b', 'c', 0, 1, 2, 3, 4, 5)
print(set(_list))
print(set(_tuple))
```

{'India', 'Singapore', 'Australia', 'Germany', 'France', 'Canada', 'Finland', 'Sri-Lanka', 'Japan', 'Spain', 'Vietnam', 'China', 'Russia', 'New Zealand', 'india', 'America'}

{False, True, 2, 3, 4, 5, 'a', 'b', 19.99, 'Coding', 'c'}

```
# Immutable elements are allowed in sets
nested_set = {
    'coding',
    99,
    ('a', 'b', 'c', 'd'),
    ('coding',),
    ('coding'),
    (True, False)
}
nested_set
```

{('a', 'b', 'c', 'd'), ('coding',), (True, False), 99, 'coding'}

Key Takeways for Sets:

1. Unordered : You can't access elements by index.
2. Unique : Removes Duplicates Automatically.
3. Mutable Container : Can add or remove items.
4. Immutable items: Elements must be hashable & immutable (like numbers, strings , tuples)