Module 7: Python Data Types

List: List is group of elements which can be of same or different data types.

- List is created by placing the elements inside square bracket [ ] separated by comma.
- Lists are mutable meaning we can append, insert , remove etc.

List Operations:

- **+** ➔ to perform concatenation between lists else we get TypeError.
- ***** ➔ To repeat elements in the list int number of times. If one of them is not int then we get TypeError:
- **in** to check the membership in List.

Changing & Adding Elements in List:

- Changing the element via =

Note: if index is not in range then we get IndexError.

- **append(element):** Adds element to the end of the List.

Note: if we supply more then one element then we get TypeError: append() takes as input only one argument

- **extend([e]/(e)/{e}):** Appends all the element of the List/Tuple/Set to the main List

Note: if we supply more then one element then we get TypeError: append() takes as input only one argument

- **insert():** Inserts an item at the defined index

Note: if index is too high then element is inserted at last position and if index is too low then element is inserted at first position.

Removing Elements from List:

- **remove(ele):** Removes an item specified from the List.

Note: If element is not present in the List then we get ValueError.

- **del ln[pos]:** Deletes the specified index element from the List

Note: if index is not in range then we get IndexError.

- **pop():** Removes the last element from the List.

Note: if List is empty then we get IndexError: pop from empty list.

- **pop(index):** Removes the element present at the index position.

Note: if index is not in range then we get Ind-exError: pop index out of range.

- **clear():** Removes all item from the List.

Occurrence of items & Sorting Items in List:

- **index(ele):**

Returns the index of first matched item.

Note: If item is not present in the List then we get ValueError.

- **count(ele):**

Returns the count of the item passed as argument in the List.

- **sort():**

Sorts item in a list in ascending order.

Note: if the items are of mixed types then we get TypeError < not supported.

- **reverse():**

It reverses the order of items in the List

Tuple:

- Tuple is similar to a List with the main difference of Tuples are immutable i.e. we cannot change the elements of a tuple once it is assigned whereas Lists are mutable.
- Advantages:
  - It guarantees that the data will remain write-protected.
  - Tuples can be used in Dictionary (while List cannot)
  - Iterating becomes faster.
- Tuples are created by placing the elements in ().

Occurrence of items in Tuple:

- **index(ele):** Returns the index of first matched item.

Note: If item is not present in the Tuple then we get ValueError.

- **count(ele):** Returns the count of the item passed as argument in the Tuple.

Set:

- Set is an unordered collection of elements in which every element is unique(no duplicates).
- Sets can be created by placing elements in { }.
- To access the elements we should not use indexing or slicing.
- Set is mutable meaning we can add or remove items.

Set Operations:

- **| performs Union➔** set of all elements from both sets.
- **& performs Intersection ➔** elements common in both sets.
- **- performs Difference ➔** elements in set1 not in set2
- **^ performs Symmetric Diff ➔** elements in both sets except the common

Adding to Set:

- **add(ele):** Adds element to the set if its not present.
- **update([e]/(e)/{e}):** Updates the set with List or Tuple or Set.

Removing from set:

- **discard(ele):** Removes element from the set if it is a member else nothing.
- **remove(ele):** Removes element from the set else KeyError
- **pop():** Removes and returns an arbitrary set element. If the set is empty we get KeyError: pop from an empty set.
- **clear():** Removes all elements from set.

Dictionary:

- Python dictionary is an ordered collection of items where elements are in the key:value pairs.
- Keys should be unique and immutable(number, string or tuple).
- If we enter same key again then old key will be overwritten.
- All key value pairs are inserted in curly braces { }.

Accessing Dictionary:

- **dn['key']:** it will return the value of the key if its present else KeyError.
- **get('key') / get('key', dv):** It returns value for the specified key if key is in dictionary. None if the key is not found. We can specify default value also.

Modifying Dictionary:

- **dn['key'] = value:** it will add element if it is not present and if present then update the value of the key.
- **update(dict):** updates the dictionary with the elements from the another dictionary object

Removing from Dictionary:

- **pop('key'):** removes and returns an element from a dictionary having the given key. If element is not present then we get KeyError.
- **popitem():** returns and removes an arbitrary element (key, value) pair from the dictionary.
- **clear():** removes all items from the dictionary.