**List, Tuple, Dictionary and Set**

**List:** A mutable (changeable) ordered collection of items, which can be of different types.

🡪They allow you to store and manipulate multiple items in a single variable.

🡪Lists are defined by square brackets [].

🡪A list can contain items of different data types (e.g., strings, integers, floats and other lists).

🡪Lists can allow duplicate values.

Ex: fruits = ["apple", "sapota", "cherry"]

o/p: ['apple', 'sapota', 'cherry']

**Append():** The append() method is used to add a single element to the end of a list.

Ex: fruits.append("banana")

o/p: ['apple', 'sapota', 'cherry', 'banana']

**Insert():** The insert() method is used to insert an element at a specific index in the list.

Ex: fruits.insert(2,”pomegranate”)

o/p: ['apple', 'sapota', 'pomegranate', 'cherry', 'banana']

**Extend():** The extend() method is used to add multiple elements to the list.

Ex: fruits.extend([“orange”,”guava”])

o/p: ['apple', 'sapota', 'pomegranate', 'cherry', 'banana', 'orange', 'guava']

**Remove():**The remove() method removes the first occurrence of a specified value from the list.

Ex: fruits.remove(“cherry”)

o/p: ['apple', 'sapota', 'pomegranate', 'banana', 'orange', 'guava']

**Clear():**The clear() method removes all the elements from the list, leaving it empty.

Ex: fruits.clear()

o/p: []

**Tuple:** An immutable (unchangeable) ordered collection of items. 🡪Tuples can hold items of different data types, such as integers, strings, floats, or even other tuples.

🡪 A tuple can contain duplicate values.

🡪You cannot modify, add, or remove elements in a tuple after creation.

Ex: person = ("Ram", 25, "Engineer")

🡪 Python has two built-in methods that you can use on tuples.

* Count(): Returns the number of times a specified value occurs in a tuple

Ex: tuple=("Ram",25,"Engineer","Ram")  
 y=tuple.count("Ram")  
 print(y)

o/p: 2

* Index(): Searches the tuple for a specified value and returns the position of where it was first found.

Ex: tuple=("Ram",25,"Engineer","Ram")  
 y=tuple.index(25)  
 print(y)

o/p: 1

**Dictionary(dict):** A mutable collection of key-value pairs.

🡪 A dictionary is defined using curly braces {} and consists of key-value pairs, separated by a colon.

🡪 Key should be immutable and unique.

🡪Value should be mutable it has duplicate values and different type of data.

Ex: student = {"Id": 99, "name": "lakshmi"}

**Set**: A mutable collection of unique items. Sets automatically remove duplicate values, ensuring that each item is unique.

Ex: numbers = {1, 2, 3, 3, 4} # {1, 2, 3, 4}

**1.Union(|):**The union combines all the elements from set1 and set2, but it removes duplicates (since sets only store unique elements).

**Syntax**: set1 | set2 or set1.union(set2)

1. **Intersection (**&**)**:The intersection finds the common elements between set1 and set2

**Syntax**: set1 & set2 or set1.intersection(set2)

1. **Difference(-)**:The difference operation removes the elements of set2 from set1 and returns the result.

**Syntax**: set1 - set2 or set1.difference(set2)

1. **Symmetric Difference (**^**)**:The symmetric difference removes the common elements (3 and 4 in this case) and returns the elements that are unique to each set.

**Syntax**: set1 ^ set2 or set1.symmetric\_difference(set2)

1. **Compares b/w set, list, tuple and dictionary ?**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Features** | **List** | **Tuple** | **Set** | **Dictionary** |
| **Order** | Ordered collection of elements | Ordered collection of elements | Unordered collection of unique elements | Unordered collection of key-value elements |
| **Syntax** | [] | () | Set() | {key: value} |
| **Mutable** | Mutable | Immutable | Mutable | Mutable |
| **Duplicates** | Allows duplicates | Allows duplicates | Doesn’t allows duplicates | Keys must be unique (values can be duplicated). |
| **Indexing** | Yes (indexable) | Yes (indexable) | Yes (indexable) | Yes (indexable) |
| **Example** | list = [1, 2, 3, 2] | tuple = (1, 2, 3,2) | set = {1, 2, 3, 2} | dict={'a': 1, 'b' 2} |