**List**

In Python, lists are mutable sequences, it stores collection of value of same and different type of value and they come with a variety of built-in methods and functions to manipulate and work with them. Here's a comprehensive list of methods and functions that can be used with lists:

**Creating list in two way**

1.list() constructor

List2=list(0,1,2,3,4)

Print list2

2.using range () function

List2=list(range (0,5))

Print list2

[0,1,2,3,4]

List Methods:

1. append(x): Adds an element to the end of the list.
2. clear(): Removes all elements from the list.
3. copy(): Returns a shallow copy of the list.
4. count(x): Returns the number of times x appears in the list.
5. extend(iterable): Appends elements from an iterable to the end of the list.
6. index(x[, start[, end]]): Returns the index of the first occurrence of x in the list.
7. insert(i, x): Inserts an element at the specified position.
8. pop([i]): Removes and returns the element at the specified position. If no index is specified, removes and returns the last element.
9. remove(x): Removes the first occurrence of the specified value.
10. reverse(): Reverses the elements of the list in place.
11. sort(key=None, reverse=False): Sorts the list in ascending order by default. Custom sorting can be achieved by specifying the key function. If reverse is True, the list will be sorted in descending order.
12. del(): to delete one or more items from list .

List Functions:

1. len(list): Returns the number of elements in the list.
2. max(list): Returns the largest element in the list.
3. min(list): Returns the smallest element in the list.
4. sum(list): Returns the sum of all elements in the list (only applicable if elements are numeric).
5. sorted(iterable): Returns a new sorted list from the elements of the iterable.

Llist operator

1. Concatenation Operator:

+ (Concatenation): Concatenates two lists, creating a new list containing elements from both lists.

list1 = [1, 2, 3]

list2 = [4, 5, 6]

concatenated\_list = list1 + list2

print(concatenated\_list) # Output: [1, 2, 3, 4, 5, 6]

2. Repetition Operator:

\* (Repetition): Repeats the elements of a list by a specified number of times.

my\_list = [1, 2]

repeated\_list = my\_list \* 3

print(repeated\_list) # Output: [1, 2, 1, 2, 1, 2]

3. Indexing and Slicing:

Lists support indexing and slicing operations to access elements or sublists.

my\_list = [10, 20, 30, 40, 50]

print(my\_list[0]) # Output: 10 (accessing first element)

print(my\_list[1:3]) # Output: [20, 30] (slicing from index 1 to 2)

a. reversing of list can be done by 2 ways

1.revers() bult-in function

2.slicing technique

It displays a list of given position to end position

Syntax:- list[start,end,stepcount]

Start by default starting point

End by default ending point and take value exclusive

Step count by default forward index

e.g. A[50,45,12,89,15,45,545,49]

here 50 get index value as 0 and also by backward index it is -8

A[-2:1:-2] >>>[545,15,12]

NOTE:-

It is mutable data type and all method and function are in place

Means if we apply any method of list then original list change

traversing a list

it means accessing all the elements or items of the list it will be done by using for loop

e.g. list=[10,20,30,40,50]

for x in list:

print(x)

10

20

30

50