



**New
syllabus
2020-21**

**Chapter 12
List**

**Computer Science
Class XI (As per CBSE Board)**

LIST



It is a collections of items and each item has its own index value.

Index of first item is 0 and the last item is $n-1$. Here n is number of items in a list.

Indexing of list

0	1	2	3	4	index
80	60	70	85	75	value
-5	-4	-3	-2	-1	Negative index



Creating a list

Lists are enclosed in square brackets [] and each item is separated by a comma.

Initializing a list

Passing value in list while declaring list is initializing of a list

e.g.

```
list1 = ['English', 'Hindi', 1997, 2000]
```

```
list2 = [11, 22, 33, 44, 55 ]
```

```
list3 = ["a", "b", "c", "d"]
```

Blank list creation

A list can be created without element

```
List4=[ ]
```



Access Items From A List

List items can be accessed using its index position.

e.g.

```
list = [3,5,9]
```

```
print(list[0])
```

```
print(list[1])
```

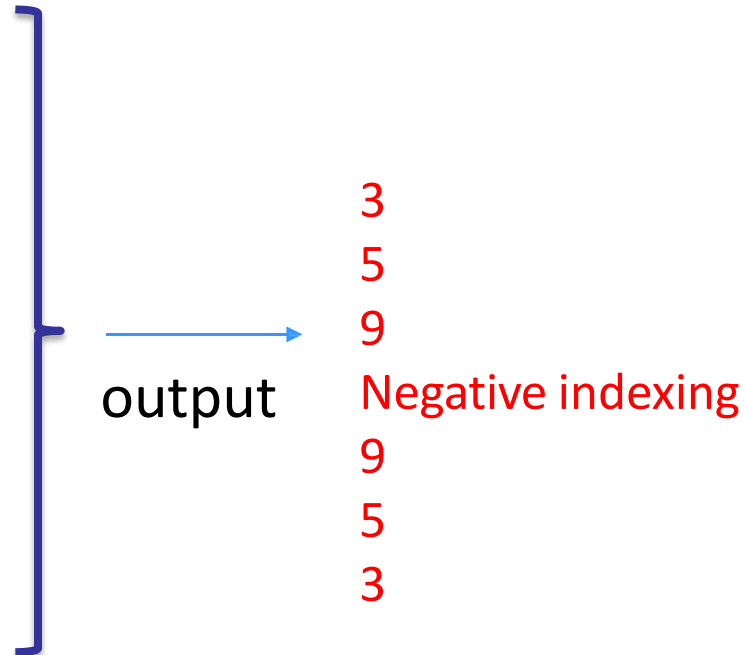
```
print(list[2])
```

```
print('Negative indexing')
```

```
print(list[-1])
```

```
print(list[-2])
```

```
print(list[-3])
```





LIST

Iterating/Traversing Through A List

List elements can be accessed using looping statement.

e.g.

```
list = [3,5,9]
for i in range(0, len(list)):
    print(list[i])
```

Output

3

5

9



Slicing of A List

List elements can be accessed in subparts.

e.g.

```
list = ['I', 'N', 'D', 'I', 'A']
```

```
print(list[0:3])
```

```
print(list[3:])
```

```
print(list[:])
```

Output

```
['I', 'N', 'D']
```

```
['I', 'A']
```

```
['I', 'N', 'D', 'I', 'A']
```

Updating / Manipulating Lists

We can update single or multiple elements of lists by giving the slice on the left-hand side of the assignment operator.

e.g.

```
list = ['English', 'Hindi', 1997, 2000]
print ("Value available at index 2 : ", list[2])
list[2:3] = 2001,2002 #list[2]=2001 for single item update
print ("New value available at index 2 : ", list[2])
print ("New value available at index 3 : ", list[3])
```

Output

```
('Value available at index 2 : ', 1997)
('New value available at index 2 : ', 2001)
('New value available at index 3 : ', 2002)
```



Add Item to A List

`append()` method is used to add an Item to a List.

e.g.

```
list=[1,2]
```

```
print('list before append', list)
```

```
list.append(3)
```

```
print('list after append', list)
```

Output

```
('list before append', [1, 2])
```

```
('list after append', [1, 2, 3])
```

NOTE :- `extend()` method can be used to add multiple item at a time in list.eg - `list.extend([3,4])`

LIST



Add Item to A List

`append()` method is used to add an Item to a List.

e.g.

```
list=[1,2]
```

```
print('list before append', list)
```

```
list.append(3)
```

```
print('list after append', list)
```

Output

```
('list before append', [1, 2])
```

```
('list after append', [1, 2, 3])
```

NOTE :- `extend()` method can be used to add multiple item at a time in list.eg - `list.extend([3,4])`

LIST



Add Two Lists

e.g.

```
list = [1,2]
```

```
list2 = [3,4]
```

```
list3 = list + list2
```

```
print(list3)
```

OUTPUT

```
[1,2,3,4]
```



Delete Item From A List

e.g.

```
list=[1,2,3]
print('list before delete', list)
del list [1]
print('list after delete', list)
```

Output

```
('list before delete', [1, 2, 3])
('list after delete', [1, 3])
```

e.g.

```
del list[0:2] # delete first two items
del list # delete entire list
```



LIST

Basic List Operations

Python Expression	Results	Description
<code>len([4, 2, 3])</code>	3	Length
<code>[4, 2, 3] + [1, 5, 6]</code>	<code>[4, 2, 3, 1, 5, 6]</code>	Concatenation
<code>['cs!'] * 4</code>	<code>['cs!', 'cs!', 'cs!', 'cs!']</code>	Repetition
<code>3 in [4, 2, 3]</code>	True	Membership
<code>for x in [4,2,3]: print (x,end = ' ')</code>	4 2 3	Iteration

Important methods and functions of List

Function	Description
<code>list.append()</code>	Add an Item at end of a list
<code>list.extend()</code>	Add multiple Items at end of a list
<code>list.insert()</code>	insert an Item at a defined index
<code>list.remove()</code>	remove an Item from a list
<code>del list[index]</code>	Delete an Item from a list
<code>list.clear()</code>	empty all the list
<code>list.pop()</code>	Remove an Item at a defined index
<code>list.index()</code>	Return index of first matched item
<code>list.sort()</code>	Sort the items of a list in ascending or descending order
<code>list.reverse()</code>	Reverse the items of a list
<code>len(list)</code>	Return total length of the list.
<code>max(list)</code>	Return item with maximum value in the list.
<code>min(list)</code>	Return item with min value in the list.
<code>list(seq)</code>	Converts a tuple, string, set, dictionary into list.
<code>Count(element)</code>	Counts number of times an element/object in the list

LIST



Some Programs on List

* find the largest/max number in a list #Using sort

```
a=[]
n=int(input("Enter number of elements:"))
for i in range(1,n+1):
    b=int(input("Enter element:"))
    a.append(b)
a.sort()
print("Largest element is:",a[n-1])
```

#using function definition

```
def max_num_in_list( list ):
    max = list[ 0 ]
    for a in list:
        if a > max:
            max = a
    return max
print(max_num_in_list([1, 2, -8, 0]))
```

```
list1, list2 = [123, 'xyz', 'zara', 'abc'], [456, 700, 200]
print "Max value element : ", max(list1)
print "Max value element : ", max(list2)
Output
Max value element : zara
Max value element : 700
```

LIST



Some Programs on List

* find the mean of a list

```
def Average(lst): #finding mean of a number
    return sum(lst) / len(lst)
```

Driver Code

```
lst = [15, 9, 55, 41, 35, 20, 62, 49]
```

```
average = Average(lst)
```

Printing average of the list

```
print("Average of the list =", round(average, 2))
```

Output

Average of the list = 35.75

Note : The inbuilt function `mean()` can be used to calculate the mean(average) of the list.e.g. `mean(list)`



Some Programs on List

* Linear Search

```
list_of_elements = [4, 2, 8, 9, 3, 7]
```

```
x = int(input("Enter number to search: "))
```

```
found = False
```

```
for i in range(len(list_of_elements)):
    if(list_of_elements[i] == x):
        found = True
        print("%d found at %dth position"%(x,i))
        break
if(found == False):
    print("%d is not in list"%x)
```




Some Programs on List

* Frequency of an element in list

```
import collections
```

```
my_list = [101,101,101,101,201,201,201,201]
```

```
print("Original List : ",my_list)
```

```
ctr = collections.Counter(my_list)
```

```
print("Frequency of the elements in the List : ",ctr)
```

OUTPUT

Original List : [101, 101,101, 101, 201, 201, 201, 201]

Frequency of the elements in the List : Counter({101: 4, 201:4})

NOTE :SAME CAN BE DONE USING COUNT FUNCTION.E.G. lst.count(x)