**Python List**

1).Python lists are the data structure that is capable of holding different type of data.

2).Python lists are mutable i.e., Python will not create a new list if we modify an element in the list.

3).It is a container that holds other objects in a given order. Different operation like insertion and deletion can be performed on lists.

4).A list can be composed by storing a sequence of different type of values separated by commas.

5).A python list is enclosed between square([]) brackets.

6).The elements are stored in the index basis with starting index as 0.

**eg:**

1. data1=[1,2,3,4];
2. data2=['x','y','z'];
3. data3=[12.5,11.6];
4. data4=['raman','rahul'];
5. data5=[];
6. data6=['abhinav',10,56.4,'a'];

**Accessing Lists**

A list can be created by putting the value inside the square bracket and separated by comma.

**Syntax:**

1. <list\_name>=[value1,value2,value3,...,valuen];

For accessing list :

1. <list\_name>[index]

Different ways to access list:

**Eg:**

1. data1=[1,2,3,4];
2. data2=['x','y','z'];
3. print data1[0]
4. print data1[0:2]
5. print data2[-3:-1]
6. print data1[0:]
7. print data2[:2]

**Output:**

>>>

>>>

1

[1, 2]

['x', 'y']

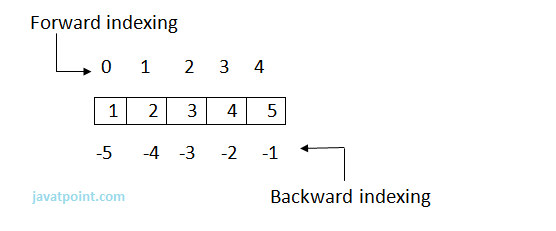
[1, 2, 3, 4]

['x', 'y']

>>>

**Elements in a Lists:**

1. Data=[1,2,3,4,5];



1. Data[0]=1=Data[-5] , Data[1]=2=Data[-4] , Data[2]=3=Data[-3] ,
2. =4=Data[-2] , Data[4]=5=Data[-1].

Note: Internal Memory Organization:  
List do not store the elements directly at the index. In fact a reference is stored at each index which subsequently refers to the object stored somewhere in the memory. This is due to the fact that some objects may be large enough than other objects and hence they are stored at some other memory location.

**List Operations:**

Various Operations can be performed on List. Operations performed on List are given as:

**a) Adding Lists:**

Lists can be added by using the concatenation operator(+) to join two lists.

**Eg:**

1. list1=[10,20]
2. list2=[30,40]
3. list3=list1+list2
4. print list3

**Output:**

1. >>>
2. [10, 20, 30, 40]
3. >>>

Note: '+'operator implies that both the operands passed must be list else error will be shown.

**Eg:**

1. list1=[10,20]
2. list1+30
3. print list1

**Output:**

1. Traceback (most recent call last):
2. File "C:/Python27/lis.py", line 2, in <module>
3. list1+30

**b) Replicating lists:**

Replicating means repeating . It can be performed by using '\*' operator by a specific number of time.

**Eg:**

1. list1=[10,20]
2. print list1\*1

**Output:**

1. >>>
2. [10, 20]
3. >>>

**c) List slicing:**

A subpart of a list can be retrieved on the basis of index. This subpart is known as list slice.

**Eg:**

1. list1=[1,2,4,5,7]
2. print list1[0:2]
3. print list1[4]
4. list1[1]=9
5. print list1

**Output:**

1. >>>
2. [1, 2]
3. 7
4. [1, 9, 4, 5, 7]
5. >>>

Note: If the index provided in the list slice is outside the list, then it raises an IndexError exception.

**Other Operations:**

Apart from above operations various other functions can also be performed on List such as Updating, Appending and Deleting elements from a List:

**a) Updating elements in a List:**

To update or change the value of particular index of a list, assign the value to that particular index of the List.

**Syntax:**

1. <list\_name>[index]=<value>

**Eg:**

1. data1=[5,10,15,20,25]
2. print "Values of list are: "
3. print data1
4. data1[2]="Multiple of 5"
5. print "Values of list are: "
6. print data1

**Output:**

1. >>>
2. Values of list are:
3. [5, 10, 15, 20, 25]
4. Values of list are:
5. [5, 10, 'Multiple of 5', 20, 25]
6. >>>

**b) Appending elements to a List:**

append() method is used to append i.e., add an element at the end of the existing elements.

**Syntax:**

1. <list\_name>.append(item)

**Eg:**

1. list1=[10,"rahul",'z']
2. print "Elements of List are: "
3. print list1
4. list1.append(10.45)
5. print "List after appending: "
6. print list1

**Output:**

1. >>>
2. Elements of List are:
3. [10, 'rahul', 'z']
4. List after appending:
5. [10, 'rahul', 'z', 10.45]
6. >>>

**c) Deleting Elements from a List:**

del statement can be used to delete an element from the list. It can also be used to delete all items from startIndex to endIndex.

**Eg:**

1. list1=[10,'rahul',50.8,'a',20,30]
2. print list1
3. del list1[0]
4. print list1
5. del list1[0:3]
6. print list1

**Output:**

1. >>>
2. [10, 'rahul', 50.8, 'a', 20, 30]
3. ['rahul', 50.8, 'a', 20, 30]
4. [20, 30]
5. >>>

**Functions and Methods of Lists:**

There are many Built-in functions and methods for Lists. They are as follows:

**There are following List functions:**

|  |  |
| --- | --- |
| **Function** | **Description** |
| min(list) | Returns the minimum value from the list given. |
| max(list) | Returns the largest value from the given list. |
| len(list) | Returns number of elements in a list. |
| cmp(list1,list2) | Compares the two list. |
| list(sequence) | Takes sequence types and converts them to lists. |

**1) min(list):**

**Eg:**

1. list1=[101,981,'abcd','xyz','m']
2. list2=['aman','shekhar',100.45,98.2]
3. print "Minimum value in List1: ",min(list1)
4. print "Minimum value in List2: ",min(list2)

**Output:**

1. >>>
2. Minimum value in List1:  101
3. Minimum value in List2:  98.2
4. >>>

**2) max(list):**

**Eg:**

1. list1=[101,981,'abcd','xyz','m']
2. list2=['aman','shekhar',100.45,98.2]
3. print "Maximum value in List : ",max(list1)
4. print "Maximum value in List : ",max(list2)

**Output:**

1. >>>
2. Maximum value in List :  xyz
3. Maximum value in List :  shekhar
4. >>>

**3) len(list):**

**Eg:**

1. list1=[101,981,'abcd','xyz','m']
2. list2=['aman','shekhar',100.45,98.2]
3. print "No. of elements in List1: ",len(list1)
4. print "No. of elements in List2: ",len(list2)

**Output:**

1. >>>
2. No. of elements in List1 :  5
3. No. of elements in List2 :  4
4. >>>

**4) cmp(list1,list2):**

Explanation: If elements are of the same type, perform the comparison and return the result. If elements are different types, check whether they are numbers.

* If numbers, perform comparison.
* If either element is a number, then the other element is returned.
* Otherwise, types are sorted alphabetically .

If we reached the end of one of the lists, the longer list is "larger." If both list are same it returns 0.

**Eg:**

1. list1=[101,981,'abcd','xyz','m']
2. list2=['aman','shekhar',100.45,98.2]
3. list3=[101,981,'abcd','xyz','m']
4. print cmp(list1,list2)
5. print cmp(list2,list1)
6. print cmp(list3,list1)

**Output:**

1. >>>
2. -1
3. 1
4. 0
5. >>>

**5) list(sequence):**

**Eg:**

1. seq=(145,"abcd",'a')
2. data=list(seq)
3. print "List formed is : ",data

**Output:**

1. >>>
2. List formed is :  [145, 'abcd', 'a']
3. >>>

**There are following built-in methods of List:**

|  |  |
| --- | --- |
| **Methods** | **Description** |
| index(object) | Returns the index value of the object. |
| count(object) | It returns the number of times an object is repeated in list. |
| pop()/pop(index) | Returns the last object or the specified indexed object. It removes the popped object. |
| insert(index,object) | Insert an object at the given index. |
| extend(sequence) | It adds the sequence to existing list. |
| remove(object) | It removes the object from the given List. |
| reverse() | Reverse the position of all the elements of a list. |
| sort() | It is used to sort the elements of the List. |

**1) index(object):**

**Eg:**

1. data = [786,'abc','a',123.5]
2. print "Index of 123.5:", data.index(123.5)
3. print "Index of a is", data.index('a')

**Output:**

1. >>>
2. Index of 123.5 : 3
3. Index of a is 2
4. >>>

**2) count(object):**

**Eg:**

1. data = [786,'abc','a',123.5,786,'rahul','b',786]
2. print "Number of times 123.5 occured is", data.count(123.5)
3. print "Number of times 786 occured is", data.count(786)

**Output:**

1. >>>
2. Number of times 123.5 occured is 1
3. Number of times 786 occured is 3
4. >>>

**3) pop()/pop(int):**

**Eg:**

1. data = [786,'abc','a',123.5,786]
2. print "Last element is", data.pop()
3. print "2nd position element:", data.pop(1)
4. print data

**Output:**

1. >>>
2. Last element is 786
3. 2nd position element:abc
4. [786, 'a', 123.5]
5. >>>

**4) insert(index,object):**

**Eg:**

1. data=['abc',123,10.5,'a']
2. data.insert(2,'hello')
3. print data

**Output:**

1. >>>
2. ['abc', 123, 'hello', 10.5, 'a']
3. >>>

**5) extend(sequence):**

**Eg:**

1. data1=['abc',123,10.5,'a']
2. data2=['ram',541]
3. data1.extend(data2)
4. print data1
5. print data2

**Output:**

1. >>>
2. ['abc', 123, 10.5, 'a', 'ram', 541]
3. ['ram', 541]
4. >>>

**6) remove(object):**

**Eg:**

1. data1=['abc',123,10.5,'a','xyz']
2. data2=['ram',541]
3. print data1
4. data1.remove('xyz')
5. print data1
6. print data2
7. data2.remove('ram')
8. print data2

**Output:**

1. >>>
2. ['abc', 123, 10.5, 'a', 'xyz']
3. ['abc', 123, 10.5, 'a']
4. ['ram', 541]
5. [541]
6. >>>

**7) reverse():**

**Eg:**

1. list1=[10,20,30,40,50]
2. list1.reverse()
3. print list1

**Output:**

1. >>>
2. [50, 40, 30, 20, 10]
3. >>>

**8) sort():**

**Eg:**

1. list1=[10,50,13,'rahul','aakash']
2. list1.sort()
3. print list1

**Output:**

1. >>>
2. [10, 13, 50, 'aakash', 'rahul']
3. >>>