Lists

Python lists are containers used to store a list of values of any data type. In simple words, we can say that a list is a collection of elements from any data type.

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

It works as a container that holds other objects in a given order. We can perform various operations like insertion and deletion on list.

A list can be composed by storing a sequence of different type of values separated by commas. Python list is enclosed between square [] brackets and elements are stored in the index basis with starting index 0.

How to create a list

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

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

How to access List Elements

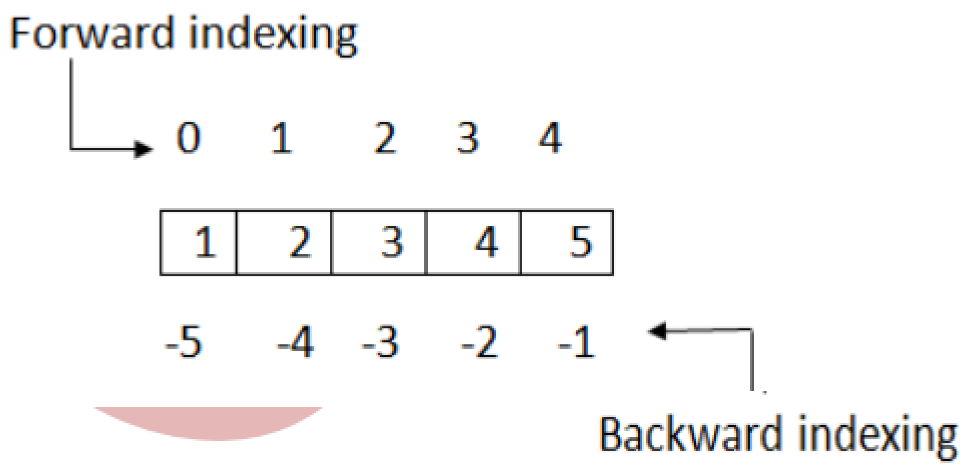
```
data1 = [1, 2, 3, 4]
data2 = ['x', 'y', 'z']

print(data1[0])
print(data1[0:2])
print(data2[-3:-1])
print(data1[0:])
print(data2[:2])
```

Elements in a List

Following are the pictorial representation of a list. We can see that it allows to access elements from both end (forward and backward).

data = [1, 2, 3, 4, 5]



Python List Operations

Apart from creating and accessing elements from the list, Python allows us to perform various other operations on the list. Some common operations are given below:-

1) **Adding Python Lists**
   In Python, lists can be added by using the concatenation operator (+) to join two lists.

```
list1 = [3, 66, 75]
list2 = ["plane", 44, "python"]

list3 = list1+list2
print(list3)

# Below will give us an error
list4 = list3+22
print(list4)
```

2) **Replicating Python Lists**

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

```
list1 = ["plane", 44, "python"]
list2 = list1*4

print(list2)
```

3) **Slicing Python List**

A subpart of a list can be retrieved on the basis of index. This subpart is known as list slice. This feature allows us to get sub-list of specified start and end index.

```
list1 = [54, 14, 47, 89, 63, 21, 44, 28]
print(list1)
print(list1[4])
print(list1[0:5])
print(list1[2:6])

# This will give us an error
print(list1[4:10])
```

4) **Updating List**

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

```
data1 = [56, "india", "python", 45, 12, 22]
print(data1)

data1[1] = "india is great"
print(data1)

data1[3] = 99
data1[-1] = 100
print(data1)

# Will give us an error
data1[8] = 22
```

5) Appending Elements to Python List

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

```
data1 = [56, "india", "python"]
print(data1)

data1.append("hello")
print(data1)

data1.append(55)
print(data1)
```

6) Deleting Elements from Python List

In Python, 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.

```
data1 = ["india", "python", 54, 12, 78, "australia"]
print(data1)

del data1[0]
print(data1)

del data1[1:4]
print(data1)
```

List Built-In Methods

Python provides various Built-in functions and methods for Lists that we can apply on the list. Following are the common list functions.

Function	Description
min(list)	It returns the minimum value from the list given.
max(list)	It returns the largest value from the given list.
len(list)	It returns number of elements in a list.
cmp(list1,list2)	It compares the two list.

1) **Python List min() method**
This method is used to get min value from the list.

```
list1 = [56, 43, 212, 17, 891, 90, 94, 34]
print("Minimum value in List 1 = %d" % (min(list1)))

# This will give us error
list2 = [45, 67, "Python", "India", 1]
print("Minimum value in List 2 = %d" % (min(list2)))
```

2) **Python List max() method**
This method is used to get max value from the list.

```
list1 = [56, 43, 212, 17, 891, 90, 94, 34]
print("Maximum value in List 1 = %d" % (max(list1)))

# This will give us error
list2 = [45, 67, "Python", "India", 1]
print("Maximum value in List 2 = %d" % (max(list2)))
```

3) **Python List len() method**

This method is used to get total length of list.

```
list1 = [56, 43, 212, 17, 891, 90, 94, 34]
print("Number of elements in List 1 = %d" % (len(list1)))

# This will give us error
list2 = [45, 67, "Python", "India", 1]
print("Number of elements in List 2 = %d" % (len(list2)))
```

More Built-In List Methods

Methods	Description
index(object)	It returns the index value of the object.
count(object)	It returns the number of times an object is repeated in list.
pop()/pop(index)	It returns the last object or the specified indexed object. It removes the popped object.
insert(index,object)	It inserts 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()	It reverses the position of all the elements of a list.
sort()	It is used to sort the elements of the List.

1) **Python List index() Method**

```
list1 = ["python", 56, 123, "india", "hello", 56]

print("Index of 56 = %d" % (list1.index(56)))
print("Index of hello = %d" % (list1.index("hello")))

# Will give us error because 4545 is not in List
print("Index of 4545 = %d" % (list1.index(4545)))
```

2) Python List count(object) Method

```
list1 = ["python", 56, 123, "india", "hello", 56, 56]

print("56 has occurred %d times." % (list1.count(56)))
print("python has occurred %d times." % (list1.count("python")))

print("4545 has occurred %d times." % (list1.count(4545)))
```

3) Python List pop()/pop(int) Method

```
list1 = ["python", 56, 123, "india", "hello", 67, "world"]

print("Last Element is ", list1.pop())
print(list1)

print("Third Element is ", list1.pop(2))
print(list1)
```

4) Python List insert(index,object) Method

```
list1 = ["python", 56, 123]

var = "hello"
list1.insert(1, var)
print(list1)

# Adds element to last position if Position does not exists
list1.insert(10, 22)
print(list1)
```

5) Python List extend(sequence) Method

```
list1 = ["python", 56, 123]
list2 = [33, "Hello", "language"]

print(list1)
list1.extend(list2)
print(list1)
print(list2)
```

6) Python List remove(object) Method

```
data1 = ['abc', 123, 10.5, 'a', 'xyz']

data1.remove(123)
print(data1)

# This will give us error because "python" is not in list
data1.remove('python')
print(data1)
```

7) Python List reverse() Method

```
list1 = ["python", 56, 123]
print(list1)

list1.reverse()
print(list1)
```

8) Python List sort() Method

```
list1 = [56, 123, -56, 2111, 3]
list1.sort()
print(list1)

list2 = ["python", "world", "elonmusk", "language"]
list2.sort()
print(list2)


list3 = ["python", 56, 123, "world", -56, 2111, 3]
# This will give us error because we cannot compare integers and string
list3.sort()
print(list3)
```

Iterating List

Iterating lists mean going through each and every value present in that list.

```
data1 = [54, "Computer", 98, "Python", 12]

for i in data1:
    print("Value = %s" % (i))

# -----OUTPUT-----
Value = 54
Value = Computer
Value = 98
Value = Python
Value = 12
```