Python Programming

- Presentation By Uplatz
- Contact us: https://training.uplatz.com
- Email: info@uplatz.com
- Phone: +44 7836 212635

Lists

Learning outcomes:

Python Lists

Accessing Values in Lists

Updating Lists

Deleting List Elements

Basic List Operations

Built-in List Functions and Methods

List

The list is a most versatile **data type** available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type.

A single list may contain **Data Types** like Integers, Strings, as well as Objects. Lists are mutable, and hence, they can be altered even after their creation. List in Python are ordered and have a definite count. The elements in a list are indexed according to a definite sequence and the indexing of a list is done with 0 being the first index.

List

❖Collection of Multiple Data.
❖Holds Multiple Data types.
❖Lists are Mutable.
❖Store multiple data at same time.
❖Creating a list is as simple as putting different comma-separated values between square brackets. For example:

```
a = [1,  12.5,  ‘a’,  “python”]
list1 = ['physics', 'chemistry', 1997, 2000];
```

Accessing Values in Lists

To access values in lists, use the square brackets for slicing along with the index or indices to obtain value available at that index.

For example:

```
a = [1,  12.5,  ‘a’,  “python”, “Programming”]
print(a[1])    # Output is 12.5
print(a[3])   # Output is python
print(a[1:4]) # Output is [12.5, ’a’, ‘python’]
```

Accessing Values in Lists

Example:

```
L = [9,18,'Hi',12,"Hello",15.55,'Programming',100,125.5]
print(L[5])
print(L[1:5])
print(L[2:8])
print(L[2:9:3])
print(L[-1])
print(L[-5])
```

How to take every nth-element of a list?

What if we want to have only every 2-nd element of **L**? This is where the ***step parameter*** comes into play.

Accessing Values in Lists

Example:

L = [9,18,'Hi',12,"Hello",15.55,'Programming',100,125.5]

print(L[0:9:3]) # Here ‘**3**’ is **step** parameter

Now you also write the above code as

print(L[::3])

Here both **print(L[0:9:3])** and **print(L[::3])** gives an output as [9, 12, 'Programming']

Accessing Values in Lists

Example:

```
L = [9,18,'Hi',12,"Hello",15.55,'Programming',100,125.5]
print (L[2::2]) # ['Hi', 'Hello', 'Programming', 125.5]
print (L[2::3]) # ['Hi', 15.55, 125.5]
print (L[3::3]) #[12, 'Programming']
print(L[5:-2]) # [15.55, 'Programming']
print(L[-2:-5:-1]) # [100, 'Programming', 15.55]
print(L[-2:-7:-2]) # [100, 15.55, 12]
```

Updating Lists

Lists in Python are **mutable**. All that means is that after defining a list, it is possible to update the individual items in a list.

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

You can also add to elements in a list with the **append()** method.

Updating Lists

Example:

```
z = [30, 17, 40, 2]
# Update the item at index 1 with the string
“python"
z[1] = "Python"
print(z) #  Output->[30, 'Python', 40, 2]
z.append(100) #Use of append method to add 100
print(z) # Output->[30, 'Python', 40, 2, 100]
```

Deleting List Elements

To remove a list element, you can use either the ***del statement*** *or* ***remove()*** **method.**

For Example:

```
b = ['Python', 100, 'Programming', 2, 'is']
del b[1] # deleting element of 1st index
print(b) # ['Python', 'Programming', 2, 'is']
b.remove(2) # Removing the element ‘2’
print(b) # ['Python', 'Programming', 'is']
```

Basic List Operations

Lists respond to the **+** and ***** operators much like strings; they mean concatenation and repetition here too, except that the result is a new list, not a string.

Let’s see some of the basic list operations in Python.

Built-in List Functions and Methods

Python includes the following list **functions**:

cmp(list1, list2) : Compares elements of both lists.
Please note cmp() does not support in python 3.

len(list): Gives the total length of the list.
max(list):Returns item from the list with max value.
min(list): Returns item from the list with min value.

Let’s see the example of list functions.

Built-in List Functions and Methods

Python includes following list **methods**:

list.append(obj): Appends object obj to list
list.count(obj): Returns count of how many times obj occurs in list
list.index(obj): Returns the lowest index in list that obj appears
list.insert(index, obj): Inserts object obj into list at offset index
list.pop(obj=list[-1]) : Removes and returns last object or obj from list

Built-in List Functions and Methods

Python includes following list **methods**:

list.remove(obj): Removes object obj from list
list.reverse(): Reverses objects of list in place
list.sort([func]): Sorts objects of list, use compare func if given
list.extend(seq): Appends the contents of seq to list

Let’s see the example of list method.

Thank you

Uplatz