Python Lists

‹ Previous Next ›

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List** is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- **Set** is a collection which is unordered and unindexed. No duplicate members.
- **Dictionary** is a collection which is unordered, changeable and indexed. No duplicate members.

When choosing a collection type, it is useful to understand the properties of that type. Choosing the right type for a particular data set could mean retention of meaning, and, it could mean an increase in efficiency or security.

List

A list is a collection which is ordered and changeable. In Python lists are written with square brackets.

Example

Create a List:

```
thislist = ["apple", "banana", "cherry"]
print(thislist)
```

Run example »

Access Items

You access the list items by referring to the index number:

Example

Print the second item of the list:

```
thislist = ["apple", "banana", "cherry"]
print(thislist[1])
```

Run example »

Change Item Value

To change the value of a specific item, refer to the index number:

Example

Change the second item:

```
thislist = ["apple", "banana", "cherry"]
thislist[1] = "blackcurrant"
print(thislist)
```

Run example »

Loop Through a List

You can loop through the list items by using a `for` loop:

Example

Print all items in the list, one by one:

```
thislist = ["apple", "banana", "cherry"]
for x in thislist:
  print(x)
```

Run example »

You will learn more about `for` loops in out Python For Loops Chapter.

Check if Item Exists

To determine if a specified item is present in a list use the `in` keyword:

Example

Check if "apple" is present in the list:

```
thislist = ["apple", "banana", "cherry"]
if "apple" in thislist:
  print("Yes, 'apple' is in the fruits list")
```

Run example »

List Length

To determine how many items a list has, use the `len()` method:

Example

Print the number of items in the list:

```
thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```

Run example »

Add Items

To add an item to the end of the list, use the `append()` method:

Example

Using the append() method to append an item:

```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")
print(thislist)
```

Run example »

To add an item at the specified index, use the `insert()` method:

Example

Insert an item as the second position:

```
thislist = ["apple", "banana", "cherry"]
thislist.insert(1, "orange")
print(thislist)
```

Run example »

Remove Item

There are several methods to remove items from a list:

Example

The `remove()` method removes the specified item:

```
thislist = ["apple", "banana", "cherry"]
thislist.remove("banana")
print(thislist)
```

Run example »

Example

The `pop()` method removes the specified index, (or the last item if index is not specified):

```
thislist = ["apple", "banana", "cherry"]
thislist.pop()
print(thislist)
```

Run example »

Example

The `del` keyword removes the specified index:

```
thislist = ["apple", "banana", "cherry"]
del thislist[0]
print(thislist)
```

Run example »

Example

The `del` keyword can also delete the list completely:

```
thislist = ["apple", "banana", "cherry"]
del thislist
print(thislist) #this will cause an error because "thislist" no longer exists.
```

Run example »

Example

The `clear()` method empties the list:

```
thislist = ["apple", "banana", "cherry"]
thislist.clear()
print(thislist)
```

Run example »

The list() Constructor

It is also possible to use the `list()` constructor to make a list.

Example

Using the list() constructor to make a List:

```
thislist = list(("apple", "banana", "cherry")) # note the double round-brackets
print(thislist)
```

Run example »

List Methods

Python has a set of built-in methods that you can use on lists.

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position

pop()	Removes the element at the specified position
remove()	Removes the item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

Test Yourself With Exercises

Exercise:

Print the second item in the `fruits` list.

```
fruits = ["apple", "banana", "cherry"]
print(          )
```

Submit Answer »

Start the Exercise

‹ Previous

Next ›

HOW TO

Tabs
Dropdowns
Accordions
Side Navigation
Top Navigation
Modal Boxes
Progress Bars
Parallax
Login Form
HTML Includes
Google Maps
Range Sliders
Tooltips
Slideshow
Filter List
Sort List

SHARE

CERTIFICATES

HTML
CSS
JavaScript
PHP
jQuery
Bootstrap
XML

Read More »

REPORT ERROR

PRINT PAGE

Top 10 Tutorials

HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
SQL Tutorial
PHP Tutorial
jQuery Tutorial
Python Tutorial

Top 10 References

HTML Reference
CSS Reference
JavaScript Reference
W3.CSS Reference
Bootstrap Reference
SQL Reference
PHP Reference
HTML Colors
jQuery Reference
Python Reference

Top 10 Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
jQuery Examples
Angular Examples
XML Examples

Web Certificates

HTML Certificate
CSS Certificate
JavaScript Certificate
jQuery Certificate
PHP Certificate
Bootstrap Certificate
XML Certificate