

PYTHON DATATYPES



Python Collections (Arrays)

2

- **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

3

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

Example

Create a List:

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

Access Items

4

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])
```

Negative Indexing

5

Negative indexing means beginning from the end, -1 refers to the last item, -2 refers to the second last item etc.

Example

Print the last item of the list:

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

Range of Indexes(Slice)

6

You can specify a range of indexes by specifying where to start and where to end the range.

When specifying a range, the return value will be a new list with the specified items.

Example

Return the third, fourth, and fifth item:

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]  
print(thislist[2:5])
```

Note: The search will start at index 2 (included) and end at index 5 (not included). Remember that the first item has index 0.

Range of Negative Indexes

7

Specify negative indexes if you want to start the search from the end of the list:

Example

This example returns the items from index -4 (included) to index -1 (excluded)

```
thislist = ["apple", "banana", "cherry", "orange", "kiwi",  
"melon", "mango"]  
print(thislist[-4:-1])
```

Change Item Value

8

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)
```


Loop Through a List

9

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)
```

Check if Item Exists

10

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")
```

List Length

11

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))
```

Add Items

12

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)
```

continued

13

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)
```

Remove Item

14

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)
```

pop()

15

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

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

del

16

The del keyword removes the specified index:

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

The del keyword can also delete the list completely:

```
thislist = ["apple", "banana", "cherry"]  
del thislist
```


clear()

17

The `clear()` method empties the list:

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

Copy a List

18

You cannot copy a list simply by typing `list2 = list1`, because: `list2` will only be a *reference* to `list1`, and changes made in `list1` will automatically also be made in `list2`.

There are ways to make a copy, one way is to use the built-in List method `copy()`.

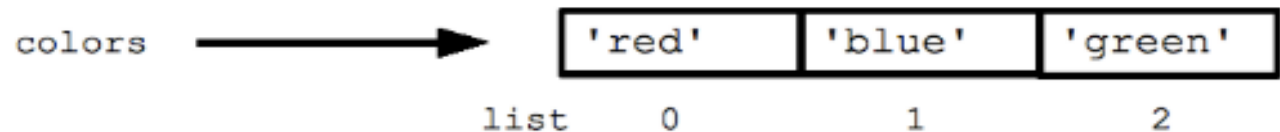
Make a copy of a list with the `copy()` method:

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

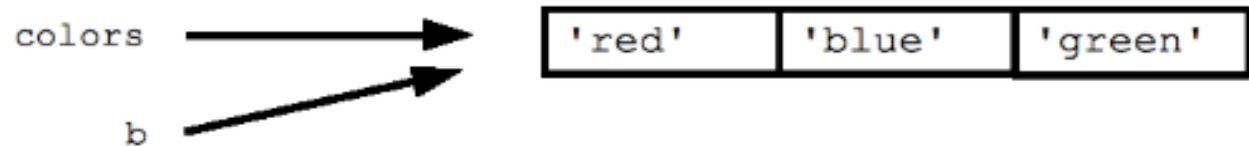
Copy a List

19

```
colors = ['red', 'blue', 'green']  
print colors[0]    ## red  
print colors[2]    ## green  
print len(colors)  ## 3
```



```
b = colors    ## Does not copy the list
```



Another way to make a copy is to use the built-in method `list()`.

20

Example

Make a copy of a list with the `list()` method:

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

Join Two Lists

21

There are several ways to join, or concatenate, two or more lists in Python.

One of the easiest ways are by using the `+` operator.

Example

Join two list:

```
list1 = ["a", "b" , "c"]
```

```
list2 = [1, 2, 3]
```

```
list3 = list1 + list2
```

```
print(list3)
```

Another way to join two lists are by appending all the items from list2 into list1, one by one:

22

Example

Append list2 into list1:

```
list1 = ["a", "b", "c"]
```

```
list2 = [1, 2, 3]
```

```
for x in list2:
```

```
    list1.append(x)
```

```
print(list1)
```

Or you can use the `extend()` method, which purpose is to add elements from one list to another list:

23

Example

Use the `extend()` method to add `list2` at the end of `list1`:

```
list1 = ["a", "b" , "c"]
```

```
list2 = [1, 2, 3]
```

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

```
print(list1)
```

The list() Constructor

24

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

Example

Using the list() constructor to make a List:

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


List Methods

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

Method	Description
<code>append()</code>	Adds an element at the end of the list
<code>clear()</code>	Removes all the elements from the list
<code>copy()</code>	Returns a copy of the list
<code>count()</code>	Returns the number of elements with the specified value
<code>extend()</code>	Add the elements of a list (or any iterable), to the end of the current list
<code>index()</code>	Returns the index of the first element with the specified value
<code>insert()</code>	Adds an element at the specified position
<code>pop()</code>	Removes the element at the specified position
<code>remove()</code>	Removes the item with the specified value
<code>reverse()</code>	Reverses the order of the list
<code>sort()</code>	Sorts the list