

Python Lists

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.

LISTS - INDEXING AND SLICING

List Slicing

```
a[2:5]
```



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)
```

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])
```

Negative Indexing

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

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.

By leaving out the start value, the range will start at the first item

Example

This example returns the items from the beginning to "orange":

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

By leaving out the end value, the range will go on to the end of the list:

Example

This example returns the items from "cherry" and to the end:

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

Range of Negative Indexes

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

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)
```

List Length

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

Example

Print the number of items in the list:

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

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)
```

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

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)
```

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)
```

Example

The `del` keyword removes the specified index:

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

Example

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

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

Example

The `clear()` method empties the list:

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

Copy a List

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()`.

Example

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

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

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

Example

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

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

Join Two Lists

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)
```

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

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

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
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

I HOPE YOU WILL ENJOY THIS VIDEO,

IF YOU SHARE THIS VIDEO WITH YOU FRIENDS

THIS ENCOURAGE ME A LOT TO
UPLOAD

A VIDEO VERY EFFICIENTLY & QUICKLY.

PLEASE SUBSCRIBE MY CHANNEL METAEDUCATORS

SHARE + LIKE + COMMENT

