# Python – Lists and its Operations (Continuation)

## The clear() method empties the list:

```
thislist = ["apple", "banana", "cherry"]
print("\nBefore Clearing")
print(thislist)
thislist.clear()
print("\nAfter Clearing")
print(thislist)
```

```
Before Clearing
['apple', 'banana', 'cherry']

After Clearing
[]
```

## 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`.

```
subject_list = ["Chemistry", "Mathematics", "Physics", "Biology", "Python", "Data Structures"]

print("\nsubject_list")

print(subject_list)

mylist = subject_list

print("\nmylist")

print(mylist)

subject_list[0]="Applied Chemistry"

print("\nmylist")

print(mylist)
```

```
subject_list
['Chemistry', 'Mathematics', 'Physics', 'Biology', 'Python', 'Data Structures']

mylist
['Chemistry', 'Mathematics', 'Physics', 'Biology', 'Python', 'Data Structures']

mylist
['Applied Chemistry', 'Mathematics', 'Physics', 'Biology', 'Python', 'Data Structures']
```

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:

```
subject_list = ["Chemistry", "Mathematics", "Physics", "Biology", "Python", "Data Structures"]
print("\nsubject_list")
print(subject_list)
mylist = subject_list.copy()

print("\nmylist")
print(mylist)
```

```
subject_list
['Chemistry', 'Mathematics', 'Physics', 'Biology', 'Python', 'Data Structures']

mylist
['Chemistry', 'Mathematics', 'Physics', 'Biology', 'Python', 'Data Structures']
```

**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:

```
solar_system_list = list(("Sun", "Moon", "Earth")) # note the double round-brackets
print(solar_system_list)
```

```
['Sun', 'Moon', 'Earth']
```

## Join Two Lists

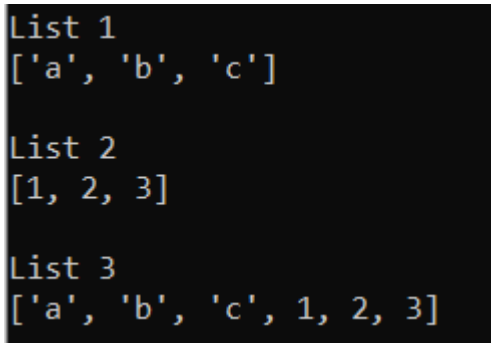
There are several ways to join, or concatenate, two or more lists in Python.

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

Example

Join two list:

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

print("\nList 1")

print(list1)

list2 = [1, 2, 3]

print("\nList 2")

print(list2)

list3 = list1 + list2

print("\nList 3")

print(list3)
```



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

Example

Append list2 into list1:

```
list1 = ["A", "B" , "C"]

print("\nList 1")

print(list1)

list2 = [11, 22, 33]

print("\nList 2")

print(list2)

for x in list2:

  list1.append(x)

print("\nList 1 after append")

print(list1)
```

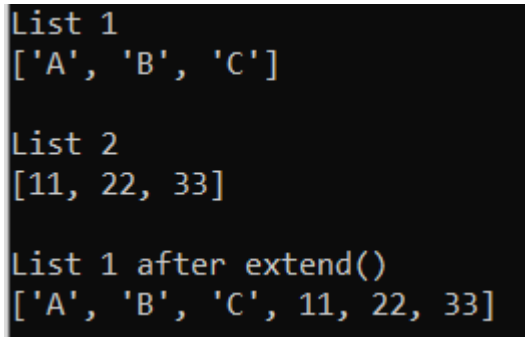
```
List 1
['A', 'B', 'C']

List 2
[11, 22, 33]

List 1 after append
['A', 'B', 'C', 11, 22, 33]
```

**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"]
print("\nList 1")
print(list1)

list2 = [11, 22, 33]
print("\nList 2")
print(list2)

print("\nList 1 after extend()")
list1.extend(list2)
print(list1)
```




**List Items - Data Types**

List items can be of any data type:

Example

String, int and boolean data types:

```
list1 = ["apple", "banana", "cherry"]
list2 = [1, 5, 7, 9, 3]
list3 = [True, False, False]
```

A list can contain different data types:

Example

A list with strings, integers and boolean values:

```
list1 = ["abc", 34, True, 40, "male"]

print(list1)
```

```
['abc', 34, True, 40, 'male']
```

## Sort List Alphanumerically

List objects have a sort() method that will sort the list alphanumerically, ascending, by default:

Example

Sort the list alphabetically:

```
list_of_fruits = ["orange", "mango", "kiwi", "pineapple", "banana"]
print("Before Sorting")
print(list_of_fruits)
list_of_fruits.sort()

print("\nAfter Sorting")
print(list_of_fruits)
```

```
Before Sorting
['orange', 'mango', 'kiwi', 'pineapple', 'banana']

After Sorting
['banana', 'kiwi', 'mango', 'orange', 'pineapple']
```

**Sort the list numerically:**

```
number_list = [100, 50, 65, 82, 23, -8, 0, 2, 66]

print("Before Sorting")
print(number_list)

number_list.sort()

print("\nAfter Sorting")
print(number_list)
```

```
Before Sorting
[100, 50, 65, 82, 23, -8, 0, 2, 66]

After Sorting
[-8, 0, 2, 23, 50, 65, 66, 82, 100]
```

## Sort Descending

To sort descending, use the keyword argument `reverse = True`:

```
list_of_fruits = ["orange", "mango", "kiwi", "pineapple", "banana"]
print("Before Sorting")
print(list_of_fruits)
list_of_fruits.sort(reverse = True)

print("\nAfter Sorting")
print(list_of_fruits)
```

```
Before Sorting
['orange', 'mango', 'kiwi', 'pineapple', 'banana']

After Sorting
['pineapple', 'orange', 'mango', 'kiwi', 'banana']
```

Sort the list descending:

```
thislist = [100, 50, 65, 82, 23]
thislist.sort(reverse = True)
print(thislist)
```

```
Before Sorting
[100, 50, 65, 82, 23, -8, 0, 2, 66]

After Sorting
[100, 82, 66, 65, 50, 23, 2, 0, -8]
```

## Case Insensitive Sort

- By default the sort() method is case sensitive.
- So if you want a case-insensitive sort function, use **str.lower** as a key function:

```
thislist = ["B", "c", "a", "E", "d", "a"]
print("\nThe given list")
print(thislist)

thislist.sort()
print("\nThe given list after sorting")
print(thislist)

thislist.sort(key = str.lower)
print("\nThe given list after sorting")
print(thislist)
```

```
The given list
['B', 'c', 'a', 'E', 'd', 'a']

The given list after sorting
['B', 'E', 'a', 'a', 'c', 'd']

The given list after sorting
['a', 'a', 'B', 'c', 'd', 'E']
```

## Reverse Order

The reverse() method reverses the current order of the elements.

Example

Reverse the order of the list items:

```
number_list = [100, 50, 65, 82, 23, -8, 0, 2, 66]
print("Before reverse()")
print(number_list)
number_list.reverse()
print("\nAfter reverse()")
print(number_list)
```

```
Before reverse()
[100, 50, 65, 82, 23, -8, 0, 2, 66]

After reverse()
[66, 2, 0, -8, 23, 82, 65, 50, 100]
```

### 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 |

Activity:

1) Read values from users and create a list of cars (say **list_of_cars**).
    a. Print the **list_of_cars** without sorting (as given by the users)
    b. Print the **list_of_cars** in ascending order.
    c. Print the **list_of_cars** in descending order.

2) Read a list of numbers from users and add them to a list (say **number_list)**.
    a. Print the **number_list**.
    b. Print the **number_list** in ascending order.
    c. Print the **number_list** in **descending** order.
    d. Create a new list (say **even_number_list**) from **number_list.** This **even_number_list** must contain only even numbers.
    e. Create a new list (say **postive_number_list**) from **number_list.** This **postive _number_list** must contain only non-negative numbers.

3) Read a list of numbers/Strings from users and add them to a list (say **string_number_list)** by strictly following the following constraints.
    a) Strings starting only with vowel character are permitted.
    b) Only positive numbers are permitted.
    c) Only even numbers are permitted.
    d) Numbers between 100 – 800 only should be permitted.