## **What is a List in Python?**

The Python list is the most basic built-in data type that enables you to store sequences of items in a single variable. For example, we can create a list of letters ‘a’, ‘b’, ‘c’ and store it in a variable called letters. Lists are created using square brackets.

letters = ["a", "b", "c"]

In a list, each item is indexed with a number denoting its position in the list. Python starts counting from 0.

You can access single elements in the list using the index. The following code basically says, go to the zeroth and the second position in the list and retrieve the letters stored there.

letters[0] # returns "a"

letters[2] # returns "c"

**What Are The Properties of a List?**

* Mutable: You can add and remove items from a list.
* Ordered: A list will always maintain the order in which you’ve added items to the list
* Object-Agnostic: You can append any object to a list ranging from simple numbers to complex custom classes.
* Indexed: All items have a unique associated index that can be used to access objects.
* Lists can be nested.
* Dynamic: If you delete an object from a list, it will shrink. If you add an object, it will grow.

## **List Functions and List Methods in Python**

You can perform various operations on lists such as appending, adding, removing, indexing, slicing, and multiplying, primarily using inbuilt functions.

### **How to Append to a List in Python**

You append a single item at a time to the end of a list using the “append” method on the list.

letters.append("d")

print(letters) # ['a', 'b', 'c', 'd']

If you want to append another list of several items at once you can use the extend method.

letters = ["a", "b", "c"]

letters.extend(["d", "e", "f"])

print(letters) # ['a', 'b', 'c', 'd', 'e', 'f']

### **How to Add To a List in Python**

Rather than appending to the end of a list, you also have the option to add to a list by using the Python insert method. Insert takes the index and the item you want to insert as arguments. It will insert the item before the provided index and dynamically update the indices of the following items.

letters = ["a", "b", "c"]

letters.insert(2, "a")

print(letters) # ['a', 'b', 'a', 'c']

### **How to Concatenate/Join a List in Python**

When you are appending or adding to a list, you are modifying an existing list. But if you want to join a list in Python to another list, you also have the option of concatenating two lists using the plus operator. This will produce a new list.

letters = ["a", "b", "c"]

letters\_2 = ["d", "e", "f"]

letters = letters\_1 + letters\_2

print(letters) #['a', 'b', 'c', 'd', 'e', 'f']

### **How to Check the Length of a List in Python**

Here it is easy to see straight away that the letters have been correctly added. Once your lists contain hundreds or thousands of items, determining list length manually is not as straightforward anymore. Luckily, you can calculate the length of a list in Python quite easily.

len(letters) # 6

If you simply want to count items in a Python list, the length function is the way to go.

### **How to Remove an Element from a List in Python**

If you want to remove a specific item from a list, the “remove” method is the most straightforward way. You just specify the image that you want to remove in the method call.

letters = ["a", "b", "c"]

letters.remove('b')

print(letters) # ['a', 'c']

This will only remove the first specified element in the list. If you have the same element multiple times in the list, the following items will remain.

letters = ['a', 'b', 'a', 'c']

letters.remove('a')

letters # ['b', 'a', 'c']

To remove all occurrences of the same value from a list you can use the lambda operator.

letters = ['a', 'b', 'a', 'c']

letters\_new = list(filter(lambda l: l != 'a', letters))

print(letters\_new) # ['b', 'c']

Note that this creates a new list rather than modifying the existing list in place.

### **How to Remove Duplicates from a List in Python**

In Python, you remove duplicates from a list by using the set function. This gives you a set of unique items.

letters = ['a', 'b', 'a', 'c', 'c']

letters\_unique = set(letters)

letters\_unique # {'a', 'b', 'c'}

### **How to Sort a List in Python**

Python lists come with a sort method that sorts the list alphabetically or numerically in ascending order. You can sort a list of letters or strings like this:

letters = ['b', 'c', 'a', 'd' ]

letters.sort()

print(letters) #['a', 'b', 'c', 'd']

If the list contains numbers rather than strings, the sort method recognizes this and will order the list numerically.

digits = ['2', '3', '1', '4' ]

digits.sort()

print(digits) #['1', '2', '3', '4']

If your list contains both strings and numbers, Python’s sort list command will list the numbers first, followed by the strings.

letters = ['b', 'c', 'a', 'd', '2', '3', '1', '4' ]

letters.sort()

print(letters) #['1', '2', '3', '4', 'a', 'b', 'c', 'd']

Alternatively, you can perform a Python list sort using the inbuilt sorted function, which works similarly to the method you call directly on the list object.

letters = ['b', 'c', 'a', 'd' ]

sorted(letters) #['a', 'b', 'c', 'd']

### **How to Reverse a List in Python**

With the reverse method you can directly reverse a list in place.

letters = ['b', 'c', 'a', 'd' ]

letters.reverse()

letters #['d', 'a', 'c', 'b']

You can also add an argument to the sort function, telling it to reverse the sorted list. This will sort the list first and then reverse it.

letters = ['b', 'c', 'a', 'd' ]

letters.sort(reverse = **True**)

letters

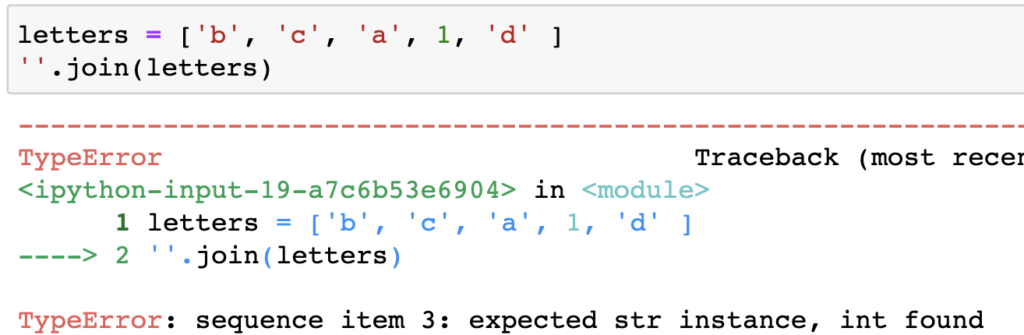
### **How to Convert a List to a String In Python?**

If the elements in your list are all strings, you can use the join function. The join function is called on a character that will separate the list elements in the newly created string. If we simply want to join all letters in a list together, we call join on empty quotation marks.

letters = ['b', 'c', 'a', 'd' ]

''.join(letters) # 'bcad'

If you try to do this with a list that includes non-string characters, you’ll get an error.



In this case, you have to convert each element to a string before you apply Python’s join method. Thanks to Python’s list comprehensions, you can do the string conversion and the join in one line of code.

letters = ['b', 'c', 'a', 1, 'd' ]

''.join([str(l) **for** l **in** letters]) # 'bca1d'

### **How to Count Items in a List in Python**

To count occurrences of an item in a list, you use the count method. You pass the item whose occurrence you want to count to the method call.

letters = ['a', 'b', 'a', 'c']

letters.count('a') # 2