**Python List**

In Python, lists allow us to store multiple items in a single variable. For example, if you need to store the ages of all the students in a class, you can do this task using a list.

Lists are similar to arrays (dynamic arrays that allow us to store items of different data types) in other programming languages.

**Create a Python List**

We create a list by placing elements inside square brackets [], separated by commas. For example,

# a list of three elements

ages = [19, 26, 29]

print(ages)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

[19, 26, 29]

Here, the ages list has three items.

**List Items of Different Types**

Python lists are very flexible. We can also store data of different data types in a list. For example,

# a list containing strings, numbers and another list

student = ['Jack', 32, 'Computer Science', [2, 4]]

print(student)

# an empty list

empty\_list = []

print(empty\_list)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

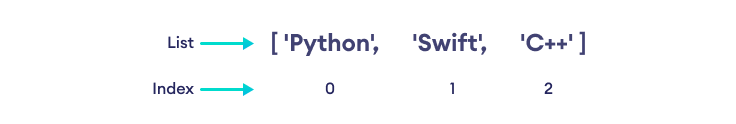
**List Characteristics**

In Python, lists are:

* **Ordered** - They maintain the order of elements.
* **Mutable** - Items can be changed after creation.
* **Allow duplicates** - They can contain duplicate values.

**Access List Elements**

Each element in a list is associated with a number, known as an **index**. The index of first item is **0**, the index of second item is **1**, and so on.

Index of List Elements

We use these indices to access items of a list. For example,

languages = ['Python', 'Swift', 'C++']

# access the first element

print('languages[0] =', languages[0])

# access the third element

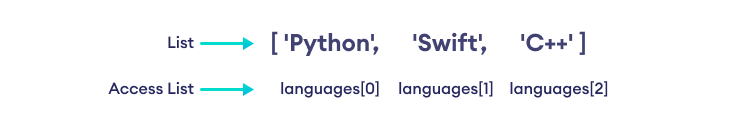
print('languages[2] =', languages[2])

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

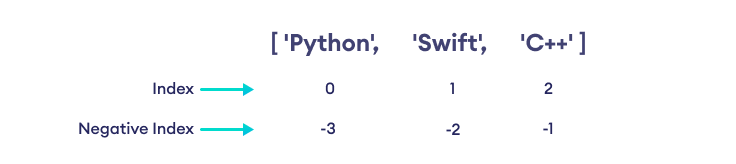
languages[0] = Python

languages[2] = C++

Access List Elements

**Negative Indexing**

In Python, a list can also have negative indices. The index of the last element is **-1**, the second last element is **-2** and so on.

Python Negative Indexing

Let's see an example.

languages = ['Python', 'Swift', 'C++']

# access the last item

print('languages[-1] =', languages[-1])

# access the third last item

print('languages[-3] =', languages[-3])

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

languages[-1] = C++

languages[-3] = Python

**Slicing of a List in Python**

If we need to access a portion of a list, we can use the slicing operator, :. For example,

my\_list = ['p', 'r', 'o', 'g', 'r', 'a', 'm']

print("my\_list =", my\_list)

# get a list with items from index 2 to index 4 (index 5 is not included)

print("my\_list[2: 5] =", my\_list[2: 5])

# get a list with items from index 2 to index -3 (index -2 is not included)

print("my\_list[2: -2] =", my\_list[2: -2])

# get a list with items from index 0 to index 2 (index 3 is not included)

print("my\_list[0: 3] =", my\_list[0: 3])

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

my\_list = ['p', 'r', 'o', 'g', 'r', 'a', 'm']

my\_list[2: 5] = ['o', 'g', 'r']

my\_list[2: -2] = ['o', 'g', 'r']

my\_list[0: 3] = ['p', 'r', 'o']

**Omitting Start and End Indices in Slicing**

If you omit the start index, the slicing starts from the first element. Similarly, if you omit the last index, the slicing ends at the last element. For example,

my\_list = ['p', 'r', 'o', 'g', 'r', 'a', 'm']

print("my\_list =", my\_list)

# get a list with items from index 5 to last

print("my\_list[5: ] =", my\_list[5: ])

# get a list from the first item to index -5

print("my\_list[: -4] =", my\_list[: -4])

# omitting both start and end index

# get a list from start to end items

print("my\_list[:] =", my\_list[:])

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

my\_list = ['p', 'r', 'o', 'g', 'r', 'a', 'm']

my\_list[5: ] = ['a', 'm']

my\_list[: -4] = ['p', 'r', 'o']

my\_list[:] = ['p', 'r', 'o', 'g', 'r', 'a', 'm']

To learn more about slicing, visit [Python program to slice lists](https://www.programiz.com/python-programming/examples/list-slicing).

**Note**: If the specified index does not exist in a list, Python throws the IndexError exception.

**Add Elements to a Python List**

As mentioned earlier, lists are mutable and we can change items of a list. To add an item to the end of a list, we can use the list [append() method](https://www.programiz.com/python-programming/methods/list/append). For example,

fruits = ['apple', 'banana', 'orange']

print('Original List:', fruits)

fruits.append('cherry')

print('Updated List:', fruits)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Original List: ['apple', 'banana', 'orange']

Updated List: ['apple', 'banana', 'orange', 'cherry']

**Add Elements at the Specified Index**

We can insert an element at the specified index to a list using the [insert()](https://www.programiz.com/python-programming/methods/list/insert) method. For example,

fruits = ['apple', 'banana', 'orange']

print("Original List:", fruits)

fruits.insert(2, 'cherry')

print("Updated List:", fruits)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Original List: ['apple', 'banana', 'orange']

Updated List: ['apple', 'banana', 'cherry', 'orange']

**Add Elements to a List From Other Iterables**

The list [extend() method](https://www.programiz.com/python-programming/methods/list/extend) method all the items of the specified iterable, such as list, tuple, dictionary or string , to the end of a list. For example,

numbers = [1, 3, 5]

print('Numbers:', numbers)

even\_numbers = [2, 4, 6]

print('Even numbers:', numbers)

# adding elements of one list to another

numbers.extend(even\_numbers)

print('Updated Numbers:', numbers)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Numbers: [1, 3, 5]

Even numbers: [2, 4, 6]

Updated Numbers: [1, 3, 5, 2, 4, 6]

**Change List Items**

We can change the items of a list by assigning new values using the = operator. For example,

colors = ['Red', 'Black', 'Green']

print('Original List:', colors)

# change the first item to 'Purple'

colors[2] = 'Purple'

# change the third item to 'Blue'

colors[2] = 'Blue'

print('Updated List:', colors)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Original List: ['Red', 'Black', 'Green']

Updated List: ['Purple', 'Black', 'Blue']

Here, we have replaced

* the element at index **0** to 'Purple'
* the element at index **2** to 'Blue'

**Remove an Item From a List**

We can remove the specified item from a list using the [remove()](https://www.programiz.com/python-programming/methods/list/remove) method. For example,

numbers = [2,4,7,9]

# remove 4 from the list

numbers.remove(4)

print(numbers)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

[2, 7, 9]

**Remove One or More Elements of a List**

Instead of using the remove() method, we can delete an item from a list using the [del statement](https://www.programiz.com/python-programming/del). The del statement can also be used to delete multiple elements or even the entire list.

names = ['John', 'Eva', 'Laura', 'Nick', 'Jack']

# delete the item at index 1

del names[1]

print(names)

# delete items from index 1 to index 2

del names[1: 3]

print(names)

# delete the entire list

del names

# Error! List doesn't exist.

print(names)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

['John', 'Laura', 'Nick', 'Jack']

['John', 'Jack']

Traceback (most recent call last):

File "", line 15, in

NameError: name 'names' is not defined

**Python List Length**

To find the number of elements (length) of a list, we can use the built-in [len() function](https://www.programiz.com/python-programming/methods/built-in/len). For example,

cars = ['BMW', 'Mercedes', 'Tesla']

print('Total Elements:', len(cars))

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

Total Elements: 3

**Iterating Through a List**

We can use a [for loop](https://www.programiz.com/python-programming/for-loop) to iterate over the elements of a list. For example,

fruits = ['apple', 'banana', 'orange']

# iterate through the list

for fruit in fruits:

print(fruit)

[Run Code](https://www.programiz.com/python-programming/online-compiler)

**Output**

apple

banana

orange

**Python List Methods**

Python has many useful [list methods](https://www.programiz.com/python-programming/methods/list) that make it really easy to work with lists.

|  |  |
| --- | --- |
| Method | Description |
| [append()](https://www.programiz.com/python-programming/methods/list/append) | Adds an item to the end of the list |
| [extend()](https://www.programiz.com/python-programming/methods/list/extend) | Adds items of lists and other iterables to the end of the list |
| [insert()](https://www.programiz.com/python-programming/methods/list/insert) | Inserts an item at the specified index |
| [remove()](https://www.programiz.com/python-programming/methods/list/remove) | Removes the specified value from the list |
| [pop()](https://www.programiz.com/python-programming/methods/list/pop) | Returns and removes item present at the given index |
| [clear()](https://www.programiz.com/python-programming/methods/list/clear) | Removes all items from the list |
| [index()](https://www.programiz.com/python-programming/methods/list/index) | Returns the index of the first matched item |
| [count()](https://www.programiz.com/python-programming/methods/list/count) | Returns the count of the specified item in the list |
| [sort()](https://www.programiz.com/python-programming/methods/list/sort) | Sorts the list in ascending/descending order |
| [reverse()](https://www.programiz.com/python-programming/methods/list/reverse) | Reverses the item of the list |
| [copy()](https://www.programiz.com/python-programming/methods/list/copy) | Returns the shallow copy of the list |