Python Arrays

Contents

[Python Arrays 3](#_Toc89185809)

[Creating a simple array 4](#_Toc89185810)

[Output 5](#_Toc89185811)

[Adding Elements to Array a 5](#_Toc89185812)

[Output : 6](#_Toc89185813)

[Accessing elements from the Array 6](#_Toc89185814)

[Output : 6](#_Toc89185815)

[Removing Elements from the Array 7](#_Toc89185816)

[Output: 7](#_Toc89185817)

[Slicing of Array a 8](#_Toc89185818)

[Output : 9](#_Toc89185819)

[Searching element in a Array 9](#_Toc89185820)

[Output: 10](#_Toc89185821)

[Updating Elements in a Array 10](#_Toc89185822)

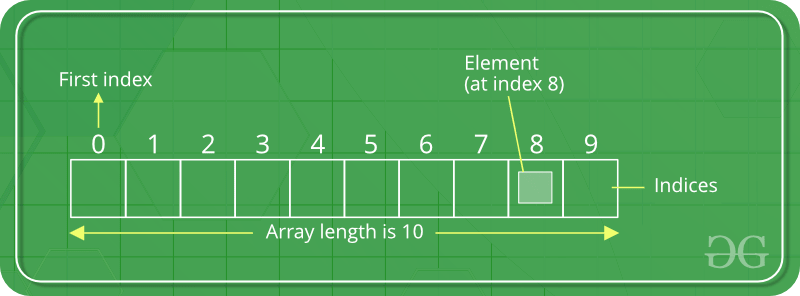
[Output: 11](#_Toc89185823)

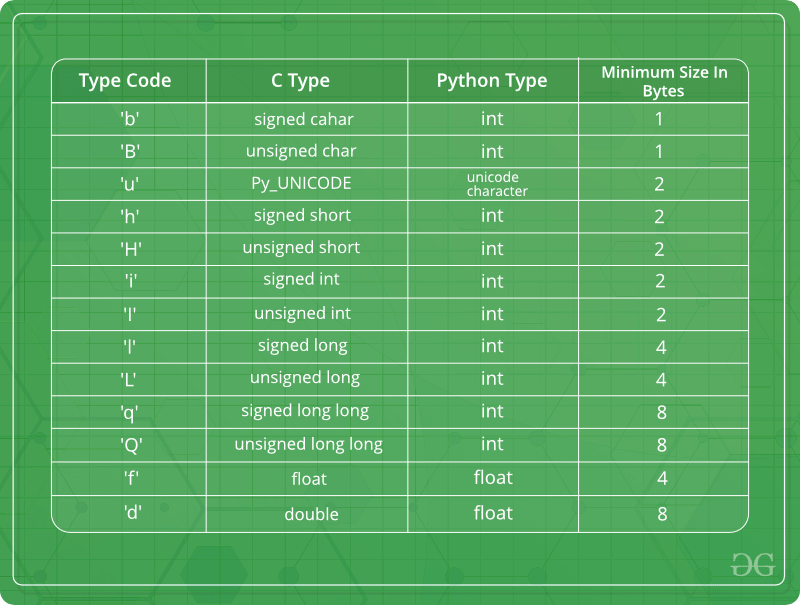
[Python Collections (Arrays) 11](#_Toc89185824)

[Reference 11](#_Toc89185825)

# Python Arrays

|  |  |
| --- | --- |
| An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array). | Array can be handled in Python by a module named array. They can be useful when we have to manipulate only a specific data type values. A user can treat lists as arrays. However, user cannot constraint the type of elements stored in a list. If you create arrays using the array module, all elements of the array must be of the same type. |
|  |  |





## Creating a simple array

# Python program to demonstrate

# Creation of Array

# importing "array" for array creations

**import array as arr**

# creating an array with integer type

**a = arr.array('i', [1, 2, 3])**

# printing original array

**print ("The new created array is : ", end =" ")**

**for i in range (0, 3):**

**print (a[i], end =" ")**

**print()**

# creating an array with float type

**b = arr.array('d', [2.5, 3.2, 3.3])**

# printing original array

**print ("The new created array is : ", end =" ")**

**for i in range (0, 3):**

**print (b[i], end =" ")**

## Output

The new created array is : 1 2 3

The new created array is : 2.5 3.2 3.3

## Adding Elements to Array a

# Python program to demonstrate

# Adding Elements to a Array

# importing "array" for array creations

**import array as arr**

# array with int type

**a = arr.array('i', [1, 2, 3])**

print ("Array before insertion : ", end =" ")

for i in range (0, 3):

print (a[i], end =" ")

print()

# inserting array using

# insert() function

a.insert(1, 4)

print ("Array after insertion : ", end =" ")

for i in (a):

print (i, end =" ")

print()

# array with float type

b = arr.array('d', [2.5, 3.2, 3.3])

print ("Array before insertion : ", end =" ")

for i in range (0, 3):

print (b[i], end =" ")

print()

# adding an element using append()

b.append(4.4)

print ("Array after insertion : ", end =" ")

for i in (b):

print (i, end =" ")

print()

## Output :

Array before insertion : 1 2 3

Array after insertion : 1 4 2 3

Array before insertion : 2.5 3.2 3.3

Array after insertion : 2.5 3.2 3.3 4.4

## Accessing elements from the Array

# Python program to demonstrate

# accessing of element from list

# importing array module

import array as arr

# array with int type

a = arr.array('i', [1, 2, 3, 4, 5, 6])

# accessing element of array

print("Access element is: ", a[0])

# accessing element of array

print("Access element is: ", a[3])

# array with float type

b = arr.array('d', [2.5, 3.2, 3.3])

# accessing element of array

print("Access element is: ", b[1])

# accessing element of array

print("Access element is: ", b[2])

### Output :

Access element is: 1

Access element is: 4

Access element is: 3.2

Access element is: 3.3

## Removing Elements from the Array

# Python program to demonstrate

# Removal of elements in a Array

# importing "array" for array operations

import array

# initializing array with array values

# initializes array with signed integers

arr = array.array('i', [1, 2, 3, 1, 5])

# printing original array

print ("The new created array is : ", end ="")

for i in range (0, 5):

print (arr[i], end =" ")

print ("\r")

# using pop() to remove element at 2nd position

print ("The popped element is : ", end ="")

print (arr.pop(2))

# printing array after popping

print ("The array after popping is : ", end ="")

for i in range (0, 4):

print (arr[i], end =" ")

print("\r")

# using remove() to remove 1st occurrence of 1

arr.remove(1)

# printing array after removing

print ("The array after removing is : ", end ="")

for i in range (0, 3):

print (arr[i], end =" ")

## Output:

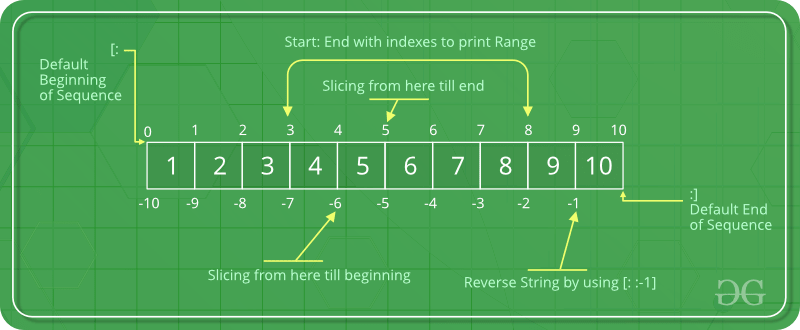
The new created array is : 1 2 3 1 5

The popped element is : 3

The array after popping is : 1 2 1 5

The array after removing is : 2 1 5

## Slicing of Array a



|  |  |
| --- | --- |
| lice operation is performed on array with the use of colon(:).  To print elements from beginning to a range use [:Index], to print elements from end use [:-Index], to print elements from specific Index till the end use [Index:], to print elements within a range, use [Start Index:End Index] and to print whole List with the use of slicing operation, use [:].  Further, to print whole array in reverse order, use [::-1]. |  |
|  |  |

# Python program to demonstrate

# slicing of elements in a Array

# importing array module

import array as arr

# creating a list

l = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

a = arr.array('i', l)

print("Initial Array: ")

for i in (a):

print(i, end =" ")

# Print elements of a range

# using Slice operation

Sliced\_array = a[3:8]

print("\nSlicing elements in a range 3-8: ")

print(Sliced\_array)

# Print elements from a

# pre-defined point to end

Sliced\_array = a[5:]

print("\nElements sliced from 5th "

"element till the end: ")

print(Sliced\_array)

# Printing elements from

# beginning till end

Sliced\_array = a[:]

print("\nPrinting all elements using slice operation: ")

print(Sliced\_array)

### Output :

Initial Array:

1 2 3 4 5 6 7 8 9 10

Slicing elements in a range 3-8:

array('i', [4, 5, 6, 7, 8])

Elements sliced from 5th element till the end:

array('i', [6, 7, 8, 9, 10])

Printing all elements using slice operation:

array('i', [1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

## Searching element in a Array

# Python code to demonstrate

# searching an element in array

# importing array module

import array

# initializing array with array values

# initializes array with signed integers

arr = array.array('i', [1, 2, 3, 1, 2, 5])

# printing original array

print ("The new created array is : ", end ="")

for i in range (0, 6):

print (arr[i], end =" ")

print ("\r")

# using index() to print index of 1st occurrenece of 2

print ("The index of 1st occurrence of 2 is : ", end ="")

print (arr.index(2))

# using index() to print index of 1st occurrenece of 1

print ("The index of 1st occurrence of 1 is : ", end ="")

print (arr.index(1))

## Output:

The new created array is : 1 2 3 1 2 5

The index of 1st occurrence of 2 is : 1

The index of 1st occurrence of 1 is : 0

## Updating Elements in a Array

In order to update an element in the array we simply reassign a new value to the desired index we want to update.

# Python code to demonstrate

# how to update an element in array

# importing array module

import array

# initializing array with array values

# initializes array with signed integers

arr = array.array('i', [1, 2, 3, 1, 2, 5])

# printing original array

print ("Array before updation : ", end ="")

for i in range (0, 6):

print (arr[i], end =" ")

print ("\r")

# updating a element in a array

arr[2] = 6

print("Array after updation : ", end ="")

for i in range (0, 6):

print (arr[i], end =" ")

print()

# updating a element in a array

arr[4] = 8

print("Array after updation : ", end ="")

for i in range (0, 6):

print (arr[i], end =" ")

## Output:

Array before updation : 1 2 3 1 2 5

Array after updation : 1 2 6 1 2 5

Array after updation : 1 2 6 1 8 5

# Python Collections (Arrays)

There are four collection data types in the Python programming language:

* [**List**](https://www.w3schools.com/python/python_lists.asp) is a collection which is ordered and changeable. Allows duplicate members.
* [**Tuple**](https://www.w3schools.com/python/python_tuples.asp) is a collection which is ordered and unchangeable. Allows duplicate members.
* [**Set**](https://www.w3schools.com/python/python_sets.asp) is a collection which is unordered, unchangeable\*, and unindexed. No duplicate members.
* **Dictionary** is a collection which is ordered\*\* and changeable. No duplicate members.

# Reference

* https://www.geeksforgeeks.org/python-arrays/