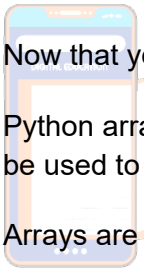
Array data structure in Python

Array

Hello, learners, You are going to learn a new module on the array data structure. I'm delighted to see your progress in learning the Python programming language.

Let us begin this module by learning the meaning of an array.
A Python Array is a collection of data structures of the same type with the same data type. It is used to store data collections. The in-built “array” module in Python is necessary for interacting with arrays. When using the array module to generate arrays, all of the array's elements must be of the same numeric type.



Now that you know what arrays are, you must also understand when to use them.

Python arrays are in handy when you need to use a lot of variables of the same type. It can also be used to maintain track of a significant amount of data.

Arrays are useful when you need to process data in a dynamic way.

Because they use less memory, arrays in Python are much faster than lists.

This brings us to the syntax of writing an array

arrayName = array.array(type code for data type, [array,items])

The array name is the name of the array, the first array is the module to import.

The second array is the method to call the array. In the bracket is the type code of data type followed by the elements in square brackets separated by commas.

Let’s understand better array creation and accessing elements.

1. Array Name: Specify an array name like a variables or Identifier
2. Module: Python has a special module for creating an array in Python, called “array” . You must import it before using it.
3. Method: the array module has a method for initializing the array. It takes two arguments, type code, and elements.
4. Type Code: Use the type codes to indicate the data type.

5. Elements: Within the square brackets, specify the array elements, for example [10,20,30].

Now look at the example to see array creation and traversal or accessing.

```
#Array creation and traversal
from array import *
#The index of the array starts with 0
array1 = array('i', [100,200,3,4,5])
# i is a type code for integer of 2 bytes size
for x in array1:
  print(x)
```

Here the whole array is imported by using the * method that we learned in the earlier module

Also, remember the array index starts with zero like in the case of a list.

Now using the syntax, array1 as array name, the module array is imported. Note, since the whole of the array is imported therefore .array which is to define the method won't be written in the code. Followed by i which is the type code that is an integer of 2 bytes size. The square brackets contain the elements, in this case, 100,200, 3, 4, 5 separated by commas.

Once the function is executed, the output is

100, 200, 3, 4, 5

Wondering what is type of code in the syntax and in the example?

a shot of dev knowledge

Arrays in python behave very similar to lists but they have the same data type of values stored in it. The data type is specified at the array creation time by using a single character called the type code. Here is the table for type codes, used for defining arrays in python.

Type code	Python type	Min size (bytes)
‘u’	Unicode character	2
‘b’	Int	1
‘B’	Int	1

‘h’	Int	2
‘l’	Int	4
‘L’	Int	4

Type code	Python type	Min size (bytes)
q	Int	8
Q	Int	8
‘H’	int	2
‘f’	float	4
‘d’	float	8
‘i’	Int	2
‘I’	Int	2

Now let us look at some of the array methods using different data types.

Starting with the float array, this function is to form an array using float data type.

```
import array
floatarray = array.array('d', [1.5, 2.5, 3.5,4.5])
print("Array first element is:",floatarray[0]) #accessing first element
print("Array last element is:",floatarray[-1]) #accessing last element using negative index
```

Since the array indexes elements like list, therefore the first element will be indexed as zero. Similarly the last element can be accessed by writing -1.

The output is.

Array first element is: 1.5

Array last element is: 4.5

Now let us look at the example of character array

```
import array
chararray = array.array('u', 'Python')
print("Array first element is:",chararray[0]) #accessing first element
print("Array last element is:",chararray[-1]) #accessing last element using -VE index
print("Entire Array", chararray)
```

Similar to the float array, here the data type of the element is character.

The first and last elements can be accessed using the indexes. Also, you can print the entire array.

The output is

Output

Array first element is: P

Array last element is: n

Entire Array array('u', 'Python')

Array module supports a rich set of methods. Let's understand with an example of insertion.

```
#Array Insertion
from array import *
array1 = array('i', [10,20,30,40,50])
array1.insert(2,600) #insert method
for x in array1:
   print(x)
```

Here the insert method is used to insert 600 at the index value of 2.

Output

10 20 600 30 40 50

Similar to the insert method, you can remove an element from an array by specifying the array element.

```
#Element Removal
from array import *
array1 = array('i', [10,20,30,40,50])
array1.remove(40)
for x in array1:
 print(x)
```

In this case, array1.remove method will remove 40.
The output is 10 20 30 50

This array method lets you determine the length of an array.
Look at the example.

```
#Program to Find an Array Length
import array as A
arr = A.array('i',[1,2,3,4,5])
print("Array elements: ",arr)
print("Length of array:",len(arr))
```

The array has been given an alias name as A.
Now this has been used in the place of module followed by integers as elements.
The output is

```
Array elements:  array('i', [1, 2, 3, 4, 5])
Length of array: 5
```

I hope you understand the concept of an array and feel confident using it in your programming.