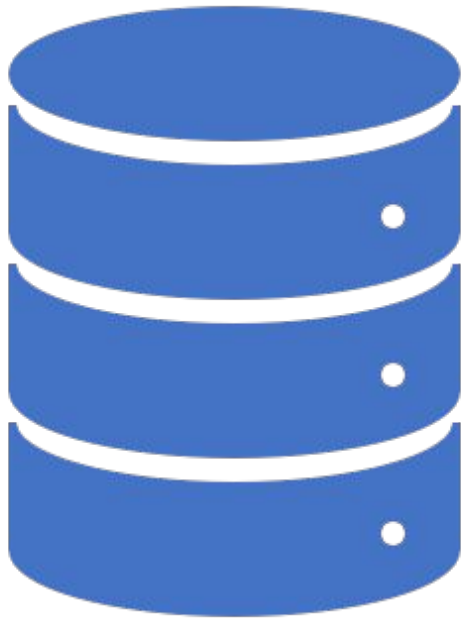
PROGRAMMING WITH PYTHON

Karuna sheel

NIELIT kurukshetra



Python Arrays

•An array is defined as a collection of items that are stored at contiguous memory locations. It is a container which can hold a fixed number of items, and these items should be of the same type. An array is popular in most programming languages like C/C++, JavaScript, etc.

•Array is an idea of storing multiple items of the same type together and it makes easier to calculate the position of each element by simply adding an offset to the base value. A combination of the arrays could save a lot of time by reducing the overall size of the code. It is used to store multiple values in single variable. If you have a list of items that are stored in their corresponding variables like this:

•car1 = "Lamborghini“

•car2 = "Bugatti"

•car3 = “BMW"

Python Arrays…

The array can be handled in Python by a module named array. It is useful when we have to manipulate only specific data values. Following are the terms to understand the concept of an array:

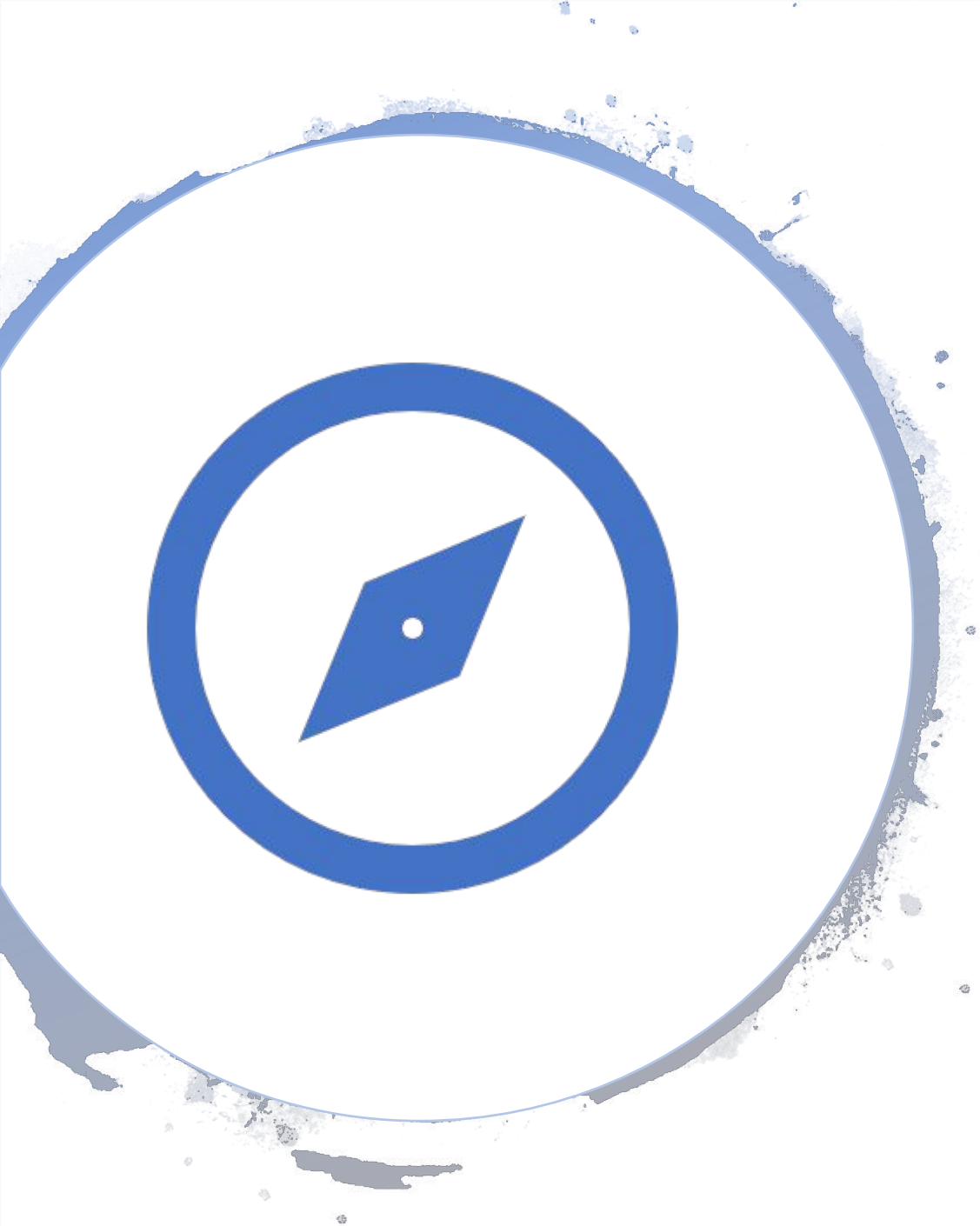
Element - Each item stored in an array is called an element.

Index - The location of an element in an array has a numerical index, which is used to identify the position of the element.

Array Representation

An array can be declared in various ways and different languages. The important points that should be considered are as follows:

- Index starts with 0.
- We can access each element via its index.
- The length of the array defines the capacity to store the elements.



Array operations

Some of the basic operations supported by an array are as follows:

- Traverse - It prints all the elements one by one.
- Insertion - It adds an element at the given index.
- Deletion - It deletes an element at the given index.
- Search - It searches an element using the given index or by the value.
- Update - It updates an element at the given index.

The Array can be created in Python by importing the array module to the python program.

import array as arr

arrayName = arr.array(typecode, [initializers])

Type code List

Typecode are the codes that are used to define the type of value the array will hold. Some common typecodes used are:

Type code	C Type	Python Type	Minimum size in bytes
'b'	signed char	int	1
'B'	unsigned char	int	1
'u'	Py_UNICODE	Unicode character	2
'h'	signed short	int	2
'H'	unsigned short	int	2
'i'	signed int	int	2
'I'	unsigned int	int	2
'l'	signed long	int	4
'L'	unsigned long	int	4
'q'	signed long long	int	8
'Q'	unsigned long long	int	8
'f'	float	float	4
'd'	double	float	8

Accessing array elements

We can access the array elements using the respective indices of those elements.

```
import array as arr
a = arr.array('i', [2, 4, 6, 8])
print("First element:", a[0])
print("Second element:", a[1])
print("Second last element:", a[-1])
```

Output:

First element: 2

Second element: 4

Second last element: 8

Explanation: In the above example, we have imported an array, defined a variable named as "a" that holds the elements of an array and print the elements by accessing elements through indices of an array.

How to change or add elements

Arrays are mutable, and their elements can be changed in a similar way like lists.

```
import array as arr
numbers = arr.array('i', [1, 2, 3, 5, 7, 10])

# changing first element
numbers[0] = 0
print(numbers)    # Output: array('i', [0, 2, 3, 5, 7, 10])

# changing 3rd to 5th element
numbers[2:5] = arr.array('i', [4, 6, 8])
print(numbers)    # Output: array('i', [0, 2, 4, 6, 8, 10])
```

Explanation: In the given example, we have imported an array and defined a variable named as "numbers" which holds the value of an array. If we want to change or add the elements in an array, we can do it by defining the particular index of an array where you want to change or add the elements.

Output:

array('i', [0, 2, 3, 5, 7, 10])
array('i' ,[0, 2, 4, 6, 8, 10])

Why to use arrays in Python?

A combination of arrays saves a lot of time. The array can reduce the overall size of the code.

How to delete elements from an array?

The elements can be deleted from an array using Python's del statement. If we want to delete any value from the array, we can do that by using the indices of a particular element.

```
import array as arr
number = arr.array('i', [1, 2, 3, 3, 4])
del number[2]                    # removing third element
print(number)                    # Output: array('i', [1, 2, 3, 4])
```

Explanation: In the above example, we have imported an array and defined a variable named as "number" which stores the values of an array. Here, by using del statement, we are removing the third element [3] of the given array.

Explanation

In the above example, we have defined variables named as "a, b, c" that hold the values of an array.

Finding the length of an array

The length of an array is defined as the number of elements present in an array. It returns an integer value that is equal to the total number of the elements present in that array.

Syntax

len(array_name)

Array Concatenation

We can easily concatenate any two arrays using the + symbol.

Example

```
a=arr.array('d',[1.1 , 2.1 ,3.1,2.6,7.8])
b=arr.array('d',[3.7,8.6])
c=arr.array('d')
c=a+b
print("Array c = ",c)
```

Output:

Array c= array('d', [1.1, 2.1, 3.1, 2.6, 7.8, 3.7, 8.6])

Example

```
import array as arr
x = arr.array('i', [4, 7, 19, 22])
print("First element:", x[0])
print("Second element:", x[1])
print("Second last element:", x[-1])
```

Output:

First element: 4
Second element: 7
Second last element: 22

Explanation: In the above example, first, we have imported an array and defined a variable named as "x" which holds the value of an array and then, we have printed the elements using the indices of an array.

Python Lists Vs Arrays

We can treat lists as arrays. However, we cannot constrain the type of elements stored in a list. For example:

```
a = [1, 3.5, "Hello"]
```

If you create arrays using the array module, all elements of the array must be of the same numeric type.

```
import array as arr
# Error
a = arr.array('d', [1, 3.5, "Hello"])
```

Output

```
Traceback (most recent call last):
  File "<string>", line 3, in <module>
    a = arr.array('d', [1, 3.5, "Hello"])
TypeError: must be real number, not str
```

Slicing Python Arrays

We can access a range of items in an array by using the slicing operator **:**

```
import array as arr
numbers_list = [2, 5, 62, 5, 42, 52, 48, 5]
numbers_array = arr.array('i', numbers_list)
print(numbers_array[2:5]) # 3rd to 5th
print(numbers_array[:-5]) # beginning to 4th
print(numbers_array[5:])  # 6th to end
print(numbers_array[:])   # beginning to end
```

Output

```
array('i', [62, 5, 42])
array('i', [2, 5, 62])
array('i', [52, 48, 5])
array('i', [2, 5, 62, 5, 42, 52, 48, 5])
```

Removing Python Array Elements

We can delete one or more items from an array using Python's del statement.

```
import array as arr
```

```
number = arr.array('i', [1, 2, 3, 3, 4])
```

```
del number[2]  # removing third element
```

```
print(number)  # Output: array('i', [1, 2, 3, 4])
```

```
del number  # deleting entire array
```

```
print(number)  # Error: array is not defined
```

Output

```
array('i', [1, 2, 3, 4])
Traceback (most recent call last):
  File "<string>", line 9, in <module>
    print(number)  # Error: array is not defined
NameError: name 'number' is not defined
```

We can use the remove() method to remove the given item, and pop() method to remove an item at the given index.

```
import array as arr

numbers = arr.array('i', [10, 11, 12, 12, 13])

numbers.remove(12)
print(numbers)   # Output: array('i', [10, 11, 12, 13])

print(numbers.pop(2))   # Output: 12
print(numbers)   # Output: array('i', [10, 11, 13])
```

Output

```
array('i', [10, 11, 12, 13])
12
array('i', [10, 11, 13])
```

Array Methods

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 first item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

Note: Python does not have built-in support for Arrays, but Python Lists can be used instead.