IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 12, Issue 1, January 2024

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.379

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 1, January 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1201044 |

Arrays across Programming Languages: A Comparative Look

[1]Nidhi Sharma, [2]Shivani Bhadoria, [3]Prof. Shashi Patel

[1&2]M. Sc(CS) Student, Department of Computer Science, SAM Girls College, Bhopal, India

[3]Assistant Professor, Department of Computer Science, SAM Girls College, Bhopal, India

ABSTRACT: In programming, most cases require storing a large amount of data of a similar type. Arrays are fundamental data structures in programming, allowing developers to store multiple values of the same type in a single variable. They're essential for tasks ranging from simple data storage to complex algorithms. Recently, many programming languages have used arrays in their way to keep memories. In this research we'll try to explore arrays in four popular programming languages: C, C++, Python, and Java, highlighting their similarities and differences.

KEYWORDS: Programming Languages, Python, Java, C, C++.

I. INTRODUCTION

Early digital computers used machine languages programming to configure and access matrix calculations, and many other purposes. John von Neumann wrote the first array-sorting program (combined sort) in 1945 when building the first stored-program computer. In the 1960s, some mainframes were designed, such as the Burroughs B5000 and its successors. Array indexing was done originally by itself-modifying code and, later, index registers and indirection using memory segmentation to perform index limit checking in hardware.

Arrays can bring many benefits to the user, like it provides easy access to all elements at once, and the order of access to any element does not matter. When creating an array, you don't have to worry about memory allocation because all aspects are allocated in contiguous memory locations in the Array. There is no chance of extra memory being allocated if an array exists. This avoids the problem of overflow or insufficient memory. Data structure such as linked lists, stacks, queues, trees, and graphs can be obtained using arrays.

Arrays are one of the programs that are used by many of the top programming languages like python, Java, or C\C++.

Therefore, the users that use arrays are primarily programmers, computer designers, and software developers. It's also a good program for the beginner since it can be used as a fundamental building block in many data structures. Many applications of Array can be seen in top programming languages. Like arrays are used to implement vectors and lists, an essential part of C++ STL arrays are used to implement stacks and queues. Also, arrays are used to hold multiple variables with the same name. Data structures such as heap, map, and set use binary trees whose services can be implemented by arrays. In real life, physical devices such as computer screens, television, and phased-array radar apply array-structured. We can rotate or animate graphics using the APL language for array and vector transformations.

II. ARRAY DETAILS

An array is a homogeneous sequential collection of data items over single variable. In computer science, array programming refers to solution that allows the application of operation to an entire set of values at once. The numeric index (a non-negative value) corresponding to that element's location must be used to access an array element. The first index with the value in the Array is zero, so the Index with the value of four is used to access the fifth array element. An array element that is also an array is called a sub-array.

Characteristics:

The characteristics of arrays are as follows-

- An array is always stored in consecutive memory location.
- It can store multiple value of similar type, which can be referred with single name.
- The pointer points to the first location of memory block, which is allocated to the array name.
- An array can either be an integer, character, or float data type that can be initialised only during the declaration.
- The particular element of an array can be modified separately without changing the other elements.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 1, January 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1201044 |

- All elements of an array can be distinguishing with the help of index number.

Operations:

The operations of an array include-

- Searching – It is used to find whether particular element is present or not.
- Sorting- helps in arranging the elements in an array either in an ascending or descending order.
- Traversing – Processing every element in an array, sequentially.
- Inserting – Helps in inserting element in an array.
- Deleting – helps in deleting the elements in an array.

Types of arrays:

There are majorly three types of arrays:

- One-dimensional array (1-D arrays)
- Two-dimensional (2D) array
- Three-dimensional array

1. One-dimensional array (1-D arrays):

Imagine a 1d array as a row, where elements are stored one after another.

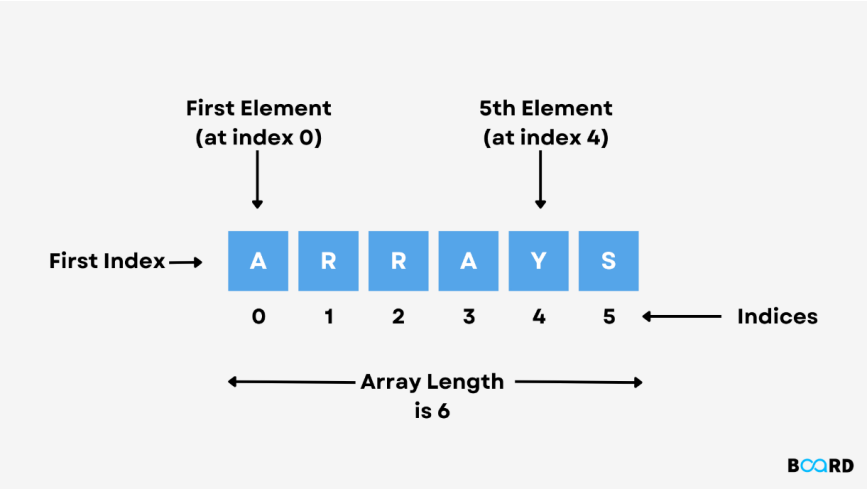

Figure 1: 1-D Array

Syntax:

data_type array_name[array_size];
Where,

- data_type array_name[array_size];
- array_name: is the name of the array using which we can refer to it.
- Array_size: is the number of blocks of memory array going to have.

For example
int nums[5];

2. Two-dimensional(2D) array:

Multidimensional arrays can be considered as an array of arrays or as a matrix consisting of rows and columns.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 1, January 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1201044 |

	Column 1	Column 2	Column 3	Column 4
Row 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Figure 2: 2D-Array

Syntax:

Data_type array_name[sizeof_1st_dimension][sizeof_2nd_dimension];
Where,

- Sizeof_dimension:is the number of blocks of memory array going to have in the corresponding dimension.

For example
Int nums[5][10];

3. Three-dimensional array:

A 3-D multidimensional array contains three dimensions, so it can be considered an array of two-dimensional arrays.

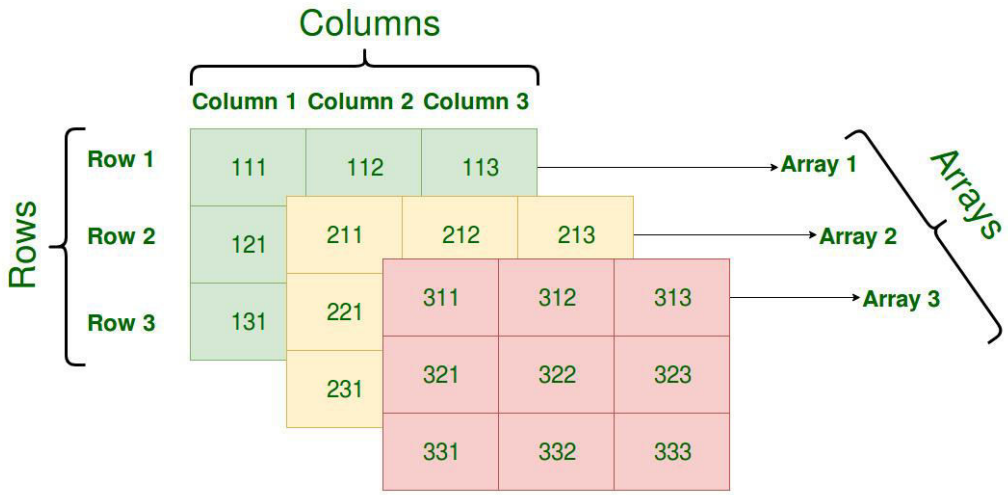

Figure 3: 3D-Array

Syntax:
Data_type array_name[size1] [size2] [size3];

- Size1, size2, size3: size of each dimension.

Arrays in C:
Declaration:
int arr[5];
Initialization:
int arr[5] ={1,2,3,4,5}
Accessing elements:
int firstElement =arr[0];

Key points:

- The size of the array must be known at compile time.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 1, January 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1201044 |

- The array index starts with 0 and goes up to the array size minus 1.
- Arrays in C don't have built-in bounds checking, which can lead to buffer overflows.
- Arrays decay into pointers, which means passing an array to a function passes a pointer to its first element.
- Example: Declare an array of size 3 and initialize it with the values 10,20, and 30. Print the second value of the array.
- Answer

```
#include <stdio.h>
int main()
{int arr[3] =[10,20,30}
Printf("second value :%d\n",arr[1]);
Return 0;
}
```

Arrays in C++:

C++ inherits the array structure from C but also introduce the Standard Library, which offers dynamic arrays(vectors).

Standard Array:

```
int array[5] = {1,2,3,4,5);
```

Using Vectors:

In C++, a vector is a dynamic array provided by the standard Library. Unlike the traditional array, which has a fixed size, vectors can grow or shrink dynamically as needed. This means you can add or remove elements from a vector without worrying about its size beforehand.

- Key features of vectors:
- Dynamic size: As mentioned, vectors can grow or shrink. Thismakes them incredibly versatile, especially when you don't know the number of elements in advance.
- Contiguous storage: Just like arrays, vectors ensure that elements are stored in contiguous memory locations. This allows efficient using indices.
- Random Access: Vectors support direct access to any element using an index, just like arrays.
- Insertion &Deletion: Vectors provide functions to insert or delete elements from any position.
- STL Benefits: Being part of the Standard Template Library (STL) vectors can be used with other STL components like iterators and algorithms.

Basic Usage:

Declaration and initialization:

```
#include <vector>
Std::vector<int> vec;
Std::vector<int> vec(5,10);
Adding elements:
vec.push_back(1);
vec.push_back(2);
Accessing elements:
int firstelement =vec[0];
int secondeelement = vec.capacity();
Removing elements:
vec.popback();
```

Why use vectors over traditional arrays?

- Flexibility: Vectors can grow or shrink, making them suitable for situations where the number of a elements is unknown or can change.
- Safety: Vectors provide member function like at () that perform bounds checking. This can prevent out-of-bounds access, a common issue with arrays.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 1, January 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1201044 |

- STL Integration: Vectors are integrated with the C++ standard Library, allowing them to be used seamlessly with algorithms, iterators, and other STL containers.

#include<vector>
Std: :vector<int> vec = {1,2,3,4,5}

Key points:

- Like C, the size of standard arrays must be known at compile time.
- Vectors are dynamic and can grow or shrink in size.
- Vectors provide bounds checking with the at() method.

Example: Using vectors in C++, create a dynamic array and add the values 5,10, and 15 to it.Print the first value.
#include<iostream>
#include<vector>
int main() {
Std::vector<int> vec;
Vec.push_back(5);
Vec.puch_back(10);
Vec.push_back(15);
Std::cout<<"first value:" <<vec[0] << std::endl;
Return 0;
}
Arrays in python:
Python offers a flexible and dynamic list type, which is similar to arrays in other languages but with more capabilities.
Lists:
Arr =[1,2,3,4,5]
Accessing elements:
Firstelement =arr[0]

Key points:

- Lists are dynamin and can store mixed data types.
- Python provides comprehensive methods to manipulate lists, like append (), remove (), and slicing.
- For array-like operations and better performance, one can use the array module or libraries like NumPy, also Python has tuples, sets, dictionaries, string, bytes, byte arrays that represent similar to array structures.

Example: Create a list with the value's "apple", "banana", and "cherry". Remove "banana" from the list and print the list.
Fruits = ["apple", "banana", "cherry"]
Fruits.remove("banana")
Print(fruits)

Arrays in Java:

Java provides both static and dynamic arrays through its standard library.
Standard arrays:
int[] arr = new int[5];
arr[0] =1;
using arraylist:
import java.util.arraylist;
arraylist<integer> arrlist = new arraylist<>();
arrlist.add(1);

key points:

- standard arrays in Java have a fixed size once declared.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 1, January 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1201044 |

- Array lists are dynamic and can grow or shrink.
- Java arrays have built-in bounds checking, throwing an ArrayIndexOutofBoundsException if accessed improperly.

Example: declare an Arraylist and add the values 1.1,2.2, and 3.3 to it. Print the third value.

```
Import java.util.Arraylist;
Public class main {
Public static void main(string[] args) {
Arraylist<Double> arrlist =new arraylist<>();
arrlist.add(1.1);
arrlist.add(2.2);
arrlist.add(3.3);
System.out,println("Third value: " + arrlist.get(2));
}}
```

III. CONCLUSION

To sum up, arrays are common to the top programming languages like python, Java, and C/C++. Even though they are not precisely similar to each other in some way, they also have different potential. It's a good program used by many developers, programmers, and designers. Arrays have many applications in data structure and real life. Therefore, it brought to the programming by helping program runs faster and can be utilized anywhere. Even though it also has some downside. They store data of similar data types together and can be used anywhere in the code. Hence, they are more efficient regarding memory allocation and are most advisable to be used in all modern languages. Additionally, they can be used to perform various CPU programming techniques. They also help to maximize the code. We can store many values in a single array by writing a small piece of code.

REFERENCES

1. https://www.tutorialspoint.com/explain-the-characteristics-and-operations-of-arrays-in-c-languages
2. https://images.app.goo.gl/YH9kLukXrqgyaV6
3. https://images.app.goo.gl/HyRtdt1NU38mG4HF9
4. https://images.app.goo.gl/EJ2vcVCxLvnMDJCt6

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor: 8.379

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details