Unit-1 Introduction to Data Structure

SESSION 1	Introduction to Data Structures
SESSION 2	Abstract Data Types
SESSION 3	Static & Dynamic Implementations with examples
SESSION 4	Arrays- ordered lists
SESSION 5	Representation of Arrays in Memory
SESSION 6	Operations on Arrays
SESSION 7	Asymptotic Analysis
SESSION 8	Space & Time Complexities

Unit-1 Introduction to Data Structure

SESSION 9	Recurrence Relations
SESSION 10	Analysis of Iterative and Recurrence Relations
SESSION 11	Revision

Recapitulation (Previous Session)

Quiz

Q1: Which of the following is an example of a static data structure?

a) Linked list

b) Stack

c) Array

d) Binary tree

Q2: When using a static data structure, the memory allocation is:

a) Done at compile time

b) Done at runtime

c) Dynamically adjusted as needed

d) Not required

Recapitulation (Previous Session cont..)

Q3: Which of the following is a limitation of static data structures?

a) Efficient memory utilization
b) Ability to resize dynamically
c) Flexibility in handling varying data sizes
d) Fast access to elements

Q4: In dynamic data structures, memory allocation is typically done:

a) At compile time
b) By the operating system
c) Using malloc() or new() functions
d) Automatically by the language runtime

Recapitulation (Previous Session)

Quiz (Answers)

A1: c) Array

A2: a) Done at compile time

A3: c) Flexibility in handling varying data sizes

A4: d) Automatically by the language runtime

Array

An array is a collection of items of same data type stored at contiguous memory locations.

Fig.1: Examples of Array

Array (Cont..)

Fig.2: Examples of Array

Why array is important?

- Efficient storage
- Random Access
- Iterative Operations
- Data Organisation
- Foundation for Data Structure
- Represent many instances in one variable

Array

- Linear data structure
- Same data type elements
- Contiguous memory allocation
- Indexing start with 0
- Elements accessed using index value

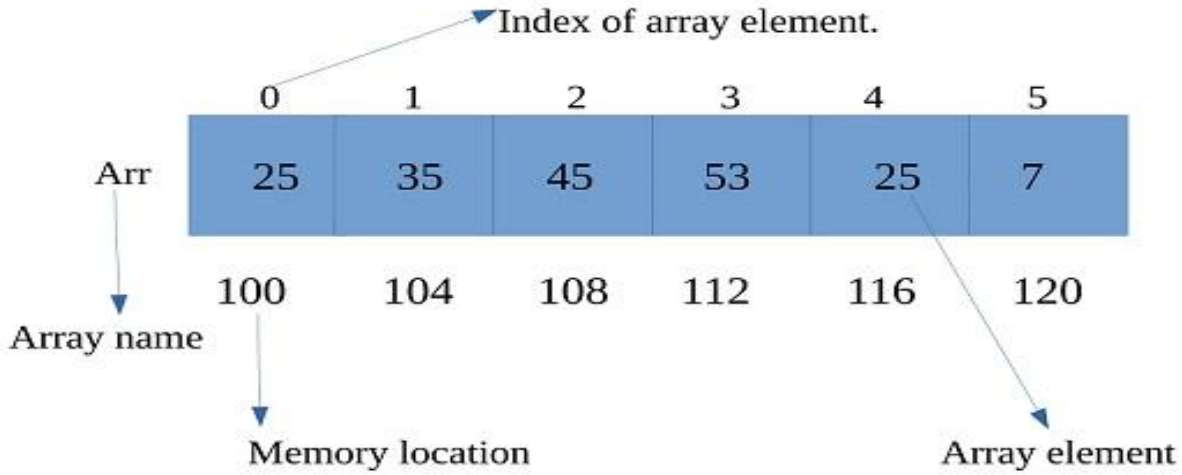


Fig.3: Representation of an Array

Test your knowledge

- Ques: What is the length of the array shown in Figure?

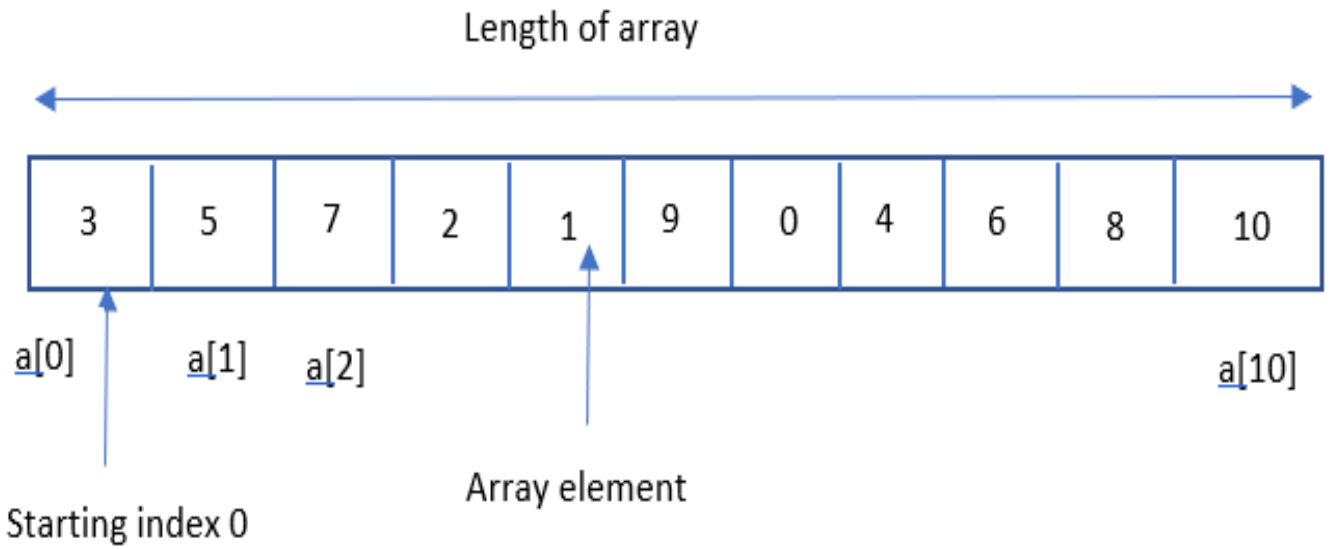


Fig.4: Question on Array

Basic Terminologies of array

- **Array Index** - Each item in an array is indexed starting with 0. Each element in an array is accessed through its index.
- **Array Element** - An array element is a single data item located at a specific index within an array.
- **Array Length** - Array length is the total number of elements that an array can hold.

Declaration of array

- int arr[5]; // This array will store integer type element
- char arr[10]; // This array will store char type element
- float arr[20]; // This array will store float type element

Initialization of array

- int arr[] = { 1, 2, 3, 4, 5 };
- char arr[5] = { 'a', 'b', 'c', 'd', 'e' };
- float arr[10] = { 1.4, 2.0, 24, 5.0, 0.0 };
- String names[5] = { “Bill”, “Susan”, “Jean”, “Piaget”, “Maria” }

Classification of arrays

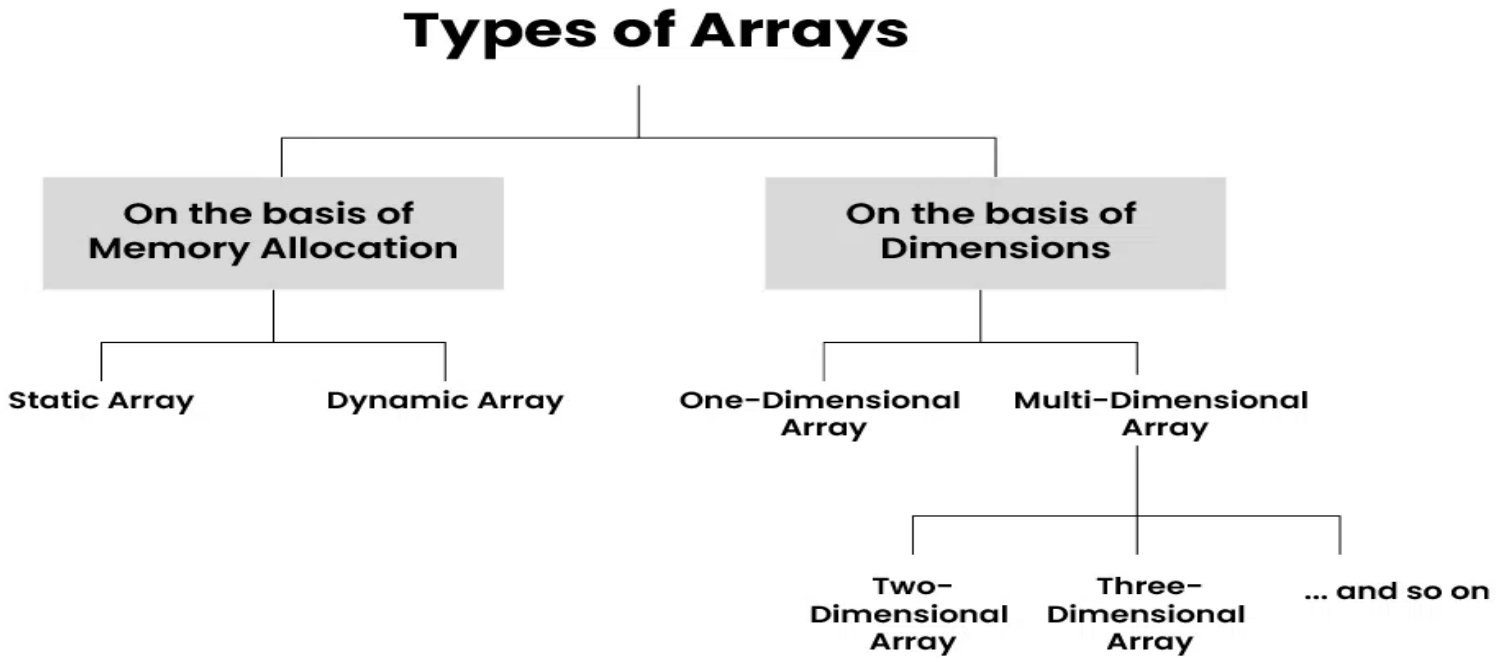
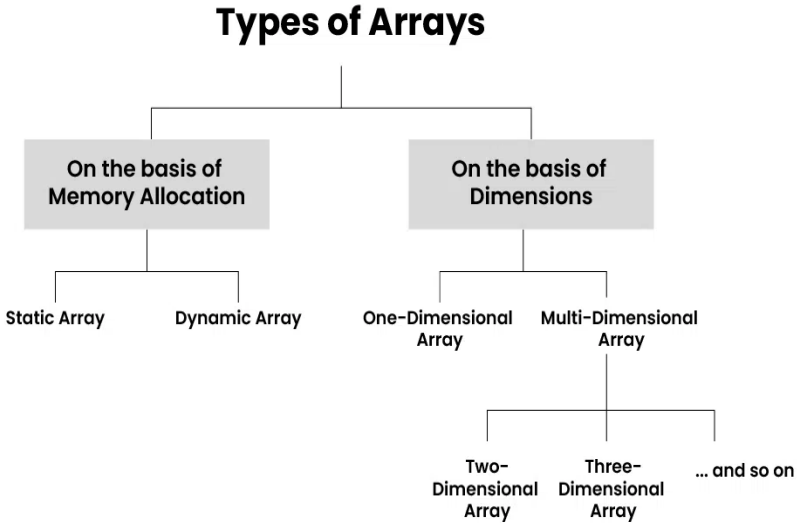


Fig.5: Types of Arrays

Static Arrays

- Memory allocated at compile time.
- Array is of fixed size.
- Size in square brackets [ ].
- Example:

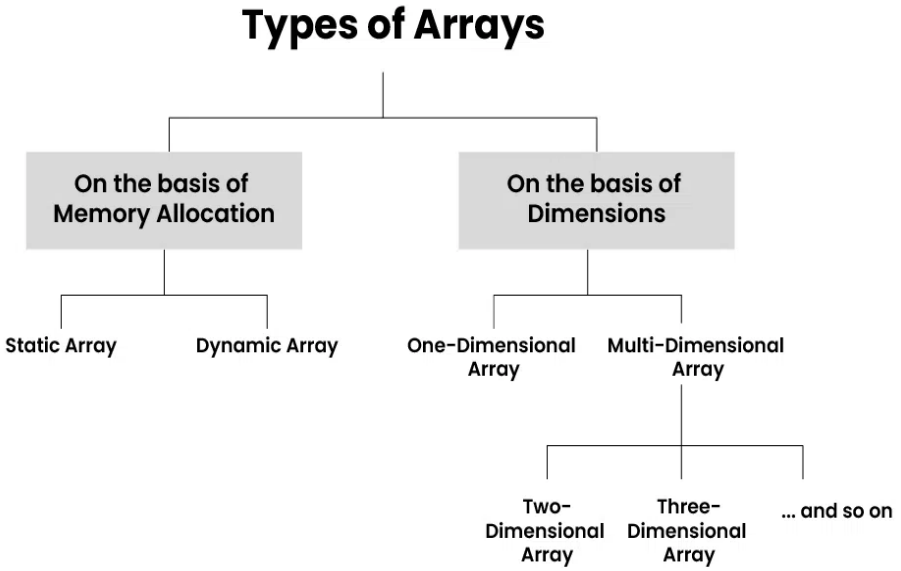
```
int arr[5] = {1, 2, 3, 4, 5};
```



Dynamic Arrays

- Memory allocated at run time
- Array is not of fixed size.
- Specify size during run time.
- Example:

```
int* arr = new int[5];
```



Reference: GeeksforGeeks

Example of Static and Dynamic Arrays

Static and Dynamic Arrays Example:

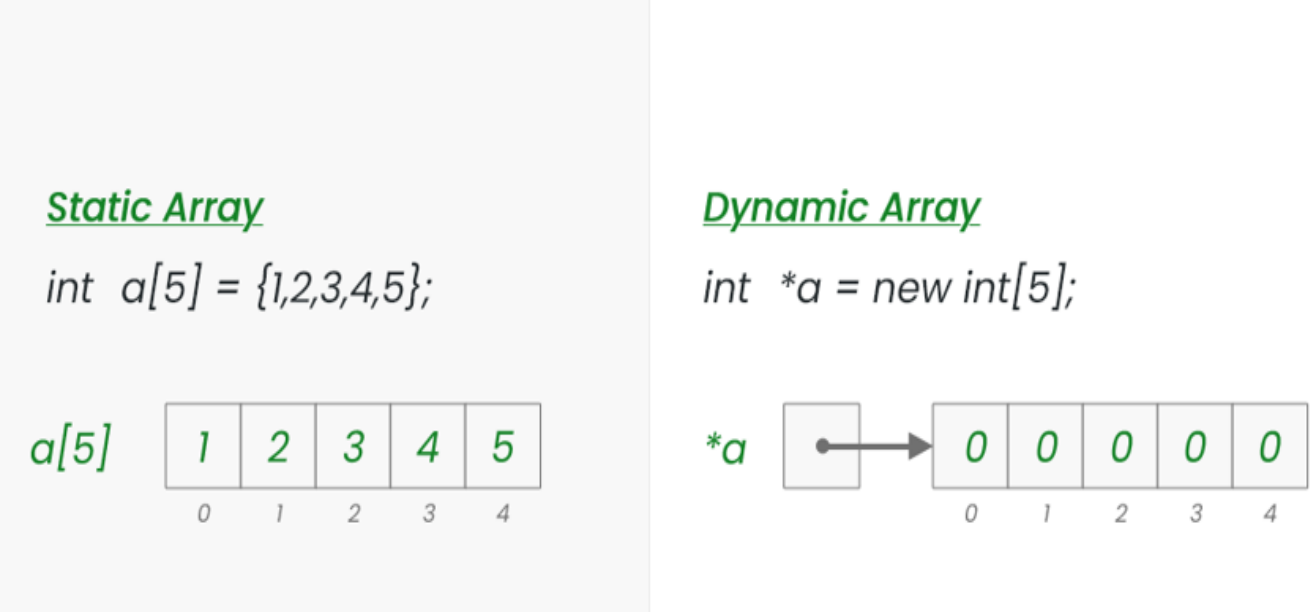
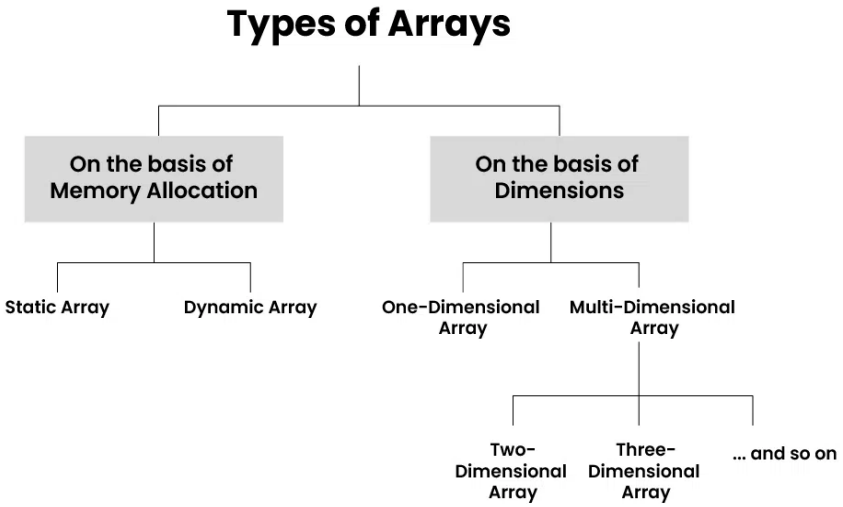


Fig.6: Example of Static and Dynamic Arrays

One-dimensional Array(1-D Array)

- 1-D as a row
- Elements stored one after other.



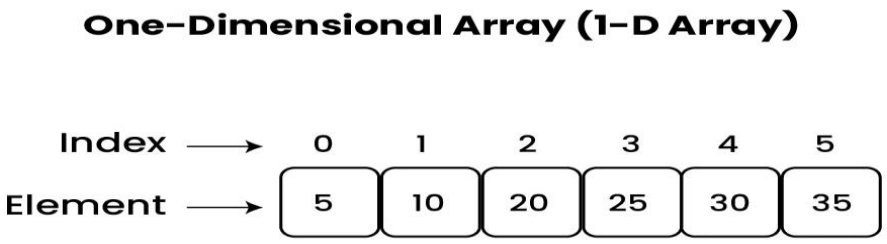


Fig.7: Example of One-dimensional Array

One-dimensional Array(1-D Array)

```
#include <iostream>
using namespace std;
int main() {
   // Declare and initialize a 1D array with 5 elements
   int numbers[5] = {10, 20, 30, 40, 50};
   int sum = 0;  // Variable to store the sum of the array
elements

   // Loop through the array using the size of the array
   for (int i = 0; i < 5; i++) {
      sum += numbers[i];  // Add each element to sum
   }
   // Output the sum of the array elements
   cout << "Sum of the array elements is: " << sum <<
endl;

   return 0;
}
```

Explanation:

Array Declaration and Initialization:

The array numbers is declared with a size of 5, meaning it can hold 5 integers. It is initialized with the values {10, 20, 30, 40, 50}.

Sum Calculation: A for loop iterates through each element of the array. The variable sum accumulates the total of the array elements as the loop progresses.

Output: After the loop completes, the total sum of the array elements is printed.

Two-dimensional Array(2-D Array)

- Array of arrays
- Consists of rows and columns

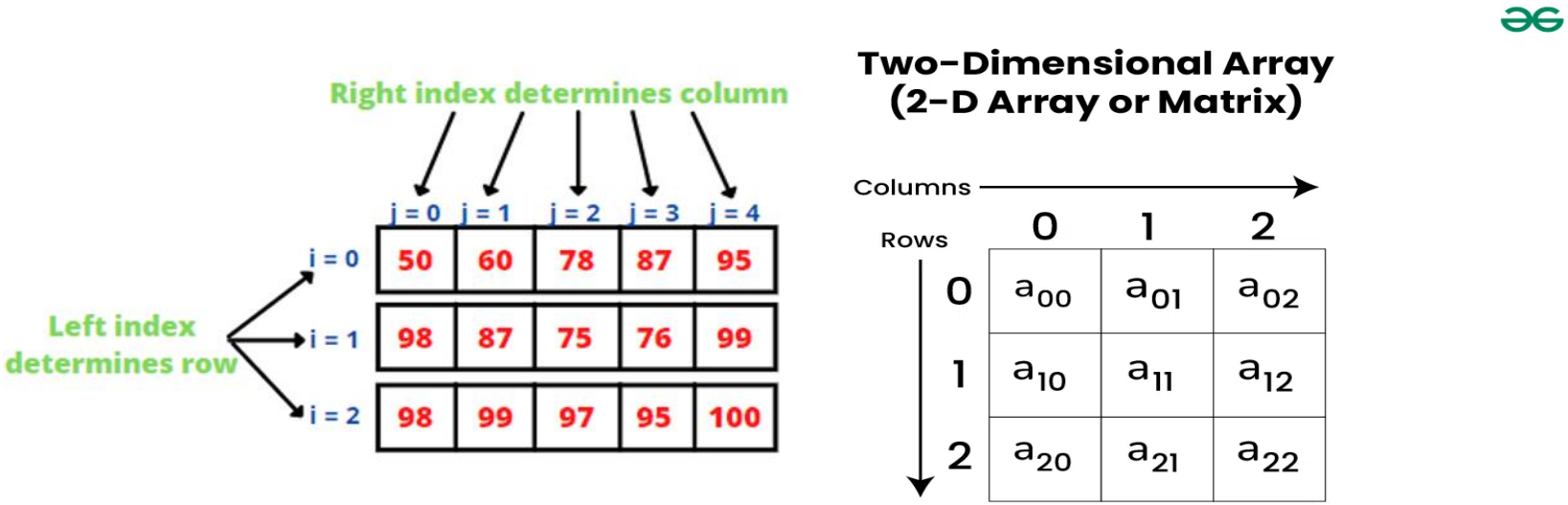


Fig.8: Examples of Two-dimensional Array

Two-dimensional Array (2-D Array)


```
#include <iostream>
using namespace std;

int main() {
    // Declare and initialize a 2D array (3x3 matrix)
    int matrix[3][3] = {
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9}
    };

    // Declare an array to store the transpose
    int transpose[3][3];

    // Compute the transpose of the matrix
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }
```

Explanation:

2D Array Declaration and Initialization: The matrix variable is declared as a 2D array and initialized with integers in a 3x3 layout. This represents a simple square matrix.

Transpose Calculation: A nested loop iterates through each element of the matrix. Elements are swapped between rows and columns to create the transpose, which is stored in the transpose array.

Output: The program first prints the original matrix and then the transposed matrix. Each element is accessed using its row (i) and column (j) indices.

Three-dimensional Array(3-D Array)

- Three dimensions
- An array of 2-dimensional arrays.

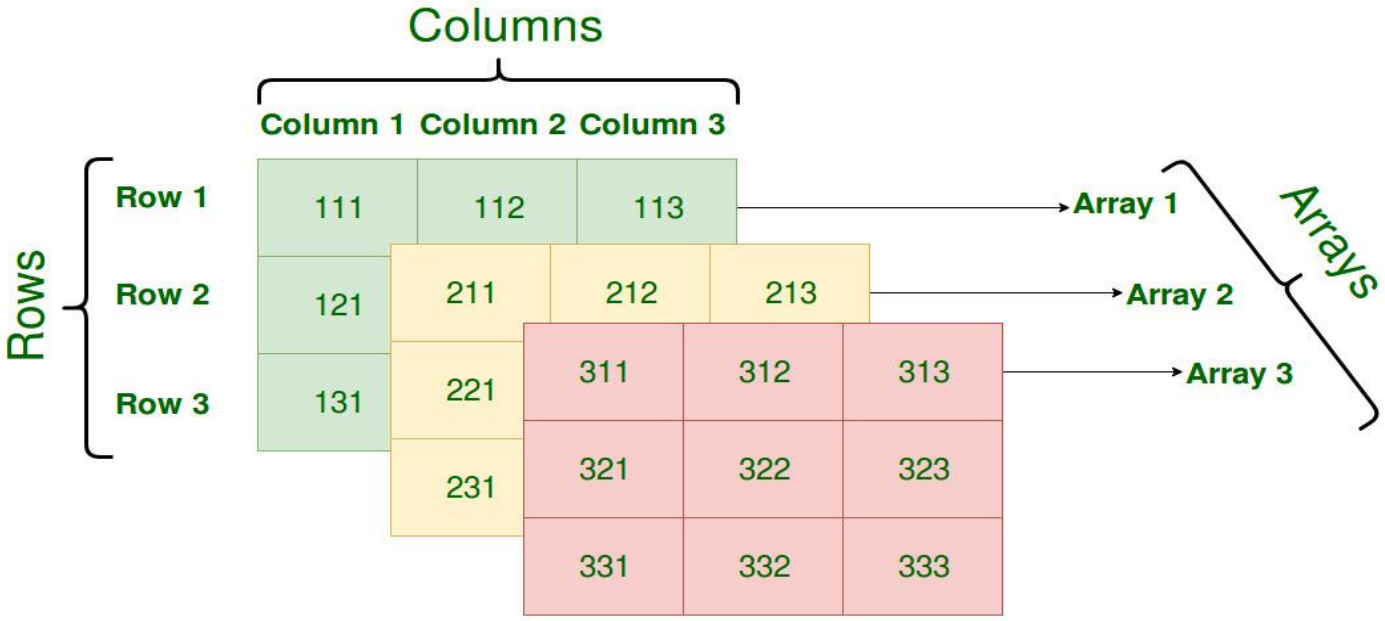


Fig.9: Examples of Three -dimensional Array

Multiple Choice Questions

Q1 What is an array?

a) A linear data structure

b) A collection of elements of the same type

c) A container that stores elements in a contiguous memory block

d) All of the above

Q2 What is the index of the first element in an array?

a) 0

b) 1

c) -1

d) It depends on the programming language

Multiple Choice Questions (Cont..)

Q3 Which of the following statements is true about arrays?

a) Arrays can store elements of different data types.

b) Arrays can automatically resize themselves to accommodate more elements.

c) Arrays have a fixed size determined at compile time.

d) Arrays are not supported in programming languages.

Q4 Which of the following is an example of a dynamic array?

a) ArrayList in Java

b) int array[10] in C++

c) Arrays.copyOf() in Python

d) None of the above

Situation- Based Questions

Q1 You are working on a scientific simulation that involves modeling weather patterns across a geographical area. Each grid cell in the simulation has various weather attributes, but many cells have default values due to the sparse nature of weather phenomena. Which type of array would you use to represent the weather data efficiently?

Q2 You are developing a game where players move around on a game board represented by a grid. Each cell in the grid can either be empty or contain an obstacle. How would you represent this game board using arrays, and which type of array would be most appropriate?

Situation- Based Questions

Q3 You are developing a program to manage a library's collection of books. Each book has a different number of authors, and you want to efficiently store this information. Which type of array would you select for representing the authors of each book?

Multiple Choice Questions (Answers)

A1: d) All of the above

A2: a) 0a) Array

A3: c) Arrays have a fixed size determined at compile time.

A4: a) ArrayList in Java

Situation- Based Questions (Answers)

A1: Sparse array, as it efficiently handles datasets with many default or empty values, which is common in weather simulations.

A2: 2D array, as it provides a grid-like structure that matches the layout of the game board.

A3: Dynamic array (ArrayList in Java, list in Python), as it can dynamically resize to accommodate varying numbers of authors for each book.

THANK YOU