
LAB SESSION 5

2D Arrays and Vectors in C++

Objective:

The objective of this lab session is to introduce students to the concept of 2D arrays in C++, their declaration, initialization, and manipulation. By the end of the session, participants should be able to understand the fundamentals of 2D arrays and solve basic problems using them.

Introduction:

A two-dimensional array, also known as a matrix, is a collection of elements arranged in rows and columns. In C++, 2D arrays provide a convenient way to store and manipulate data in a tabular format. Each element in a 2D array is identified by its row and column index. Understanding how to declare, initialize, and access elements in 2D arrays is essential for solving problems involving matrices and grids.

```
dataType arrayName[rows][columns];
```

Applications:

2D arrays are widely used in various applications, including:

- Image processing: Representing pixels of an image.
- Matrices: Performing mathematical operations like addition, multiplication, etc.
- Grid-based problems: Solving problems related to grids or game boards.

Memory Representation:

In memory, elements of a 2D array are stored in a contiguous block of memory. The elements are arranged row-wise or column-wise based on the programming language and compiler. Understanding 2D arrays is fundamental in programming as they are used in many real-world scenarios for data storage and manipulation. Mastering the concepts of 2D arrays enables programmers to solve a wide range of problems efficiently.

Details about 2D Arrays:

To declare a 2D array in C++, you specify the data type of the elements and the dimensions of the array.

```
int matrix[ROW][COL];
```

You can initialize a 2D array at the time of declaration or later using nested loops

```
int matrix[2][3] = {{1, 2, 3}, {4, 5, 6}};
```

Elements in a 2D array can be accessed using row and column indices

```
int element = matrix[i][j];
```

To traverse through all elements of a 2D array, nested loops are commonly used.

```
for (int i = 0; i < ROW; ++i) {
  for (int j = 0; j < COL; ++j) {
    // Access matrix[i][j] here
  }
}
```

Exercis

Questio

Explana
To find
add it to
consider

```
matrix
  [1,
  [4,
  [7,
]
```

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}$$

To check if this matrix is symmetric, we need to compare it with its transpose:

$$A^T = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}$$

As you can see, the original matrix A is equal to its transpose A^T element-wise. Therefore, matrix A is symmetric.

The sum of all elements in the matrix is 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 = 45.

Question: Calculate the transpose of a given matrix.

Explanation:
The transpose of a matrix involves switching its rows with its columns. For each element at position (i, j) in the original matrix, the corresponding element in the transpose matrix is at position (j, i).
Matrix Representation:
Let's consider a 3x3 matrix

```
Original Matrix:
1 2 3
4 5 6
7 8 9
```

Solution: The transpose of the given matrix would be:

```
Transpose:
1 4 7
2 5 8
3 6 9
```

Question:Check if a matrix is symmetric or not.

Explanation:
A matrix is symmetric if it is equal to its transpose. To check symmetry, we compare each element of the original matrix with its corresponding element in the transpose matrix.Let's consider a 3x3 matrix matrix.The given matrix is symmetric if it is equal to its transpose.

Question:Multiply two matrices and print the result.

Explanation:
Matrix multiplication involves multiplying the elements of each row of the first matrix by the corresponding elements of each column of the second matrix and summing the results.Let's consider two matrices matrix1 and matrix2 of suitable dimensions.The resulting matrix will have dimensions equal to the number of rows of the first matrix and the number of columns of the second matrix.

Question:Find the largest element in each row of a matrix.

Explanation:
For each row of the matrix, we iterate through its elements and find the maximum value. We store the maximum value of each row in a separate array.Let's consider a 3x3 matrix matrix.We find the maximum element in each row of the matrix and store it in an array maxInRow.

Vectors

Vectors in C++ (from the Standard Template Library, **STL**) are dynamic arrays that can grow and shrink automatically. Here are some common questions on vectors in C++:

A `vector` is a sequence container in the C++ STL that dynamically resizes as elements are added or removed.

```
#include <vector>
vector<int> v;

vector<int> v = {1, 2, 3, 4, 5};
```

```
v.size(); // Returns the number of elements
v.capacity(); // Returns the allocated storage capacity
```

```
for (size_t i = 0; i < v.size(); i++)
{
```

cout << v[i] << " "; }

---

for (int num : v)

{ cout << num << " "; }

---

v.insert(v.begin() + 2, 10); // Inserts 10 at index 2

How do you find an element in a vector?

```
auto it = std::find(v.begin(), v.end(), 5);
if (it != v.end()) std::cout << "Found!";
```

How do you use `emplace_back()`?

```
cpp
CopyEdit
v.emplace_back(42);  // Constructs element in-place
(better than push_back for objects)
```

Question Reverse the elements of a given vector without using the `reverse()` function.
Example Input: `{1, 2, 3, 4, 5}`
Output: `{5, 4, 3, 2, 1}`