Introduction to Arrays

Introduction to Arrays

- An array is a collection of elements of the same type stored in contiguous memory locations.
- Arrays make it easier to manage large amounts of data using a single name (the array name). Instead of having multiple variables, arrays allow you to store and process data systematically.

Declaration

```
int arr[5];  // An array of 5 integers
```

- **Initialization:** Arrays can be initialized in different ways:
- By Specifying Values Directly:

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

- By Using Default Initialization:

```
int arr[5] = {};  // All elements initialized to 0
```

- Partial Initialization:

```
int arr[5] = {1, 2};  // First two elements are 1 and 2, others are 0
```

Operations on Arrays

- Accessing Elements:

```
arr[2] = 10;  // Assign 10 to the third element
```

- Example: Looping through Arrays

```
for(int i = 0; i < 5; i++) {
    std::cout << arr[i] << " ";
}
```

- Finding the Size of an Array:

```
int n = sizeof(arr) / sizeof(arr[0]);
```

Problem Related with Array

- Average of an integer array
- Find the maximum or minimum element of an array
- Search a specific item in an array
- Sort an array

2D Arrays (Multidimensional Arrays)

- A 2D array is an array of arrays, represented as a grid or table.

1	2	3	4
5	6	7	8
9	10	11	12

- Declaration:

```
int arr[3][4];  // A 2D array with 3 rows and 4 columns
```

- Initialization:

```
int arr[3][4] = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}};
int arr[3][4] = {
        {1, 2, 3, 4},
        {5, 6, 7, 8},
        {9, 10, 11, 12}
    };
```

Operations on 2D Arrays

- Accessing Elements:

```
arr[1][2] = 7;  // Set the element in the second row, third column to 7
```

- Looping through 2D Arrays:

```
int arr[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
for(int i = 0; i < 3; i++) {
    for(int j = 0; j < 3; j++) {
        std::cout << arr[i][j] << " ";
    }
    std::cout << std::endl;
}
```

- Finding the Size of a 2D Array

```
int rows = sizeof(arr) / sizeof(arr[0]);
int columns = sizeof(arr[0]) / sizeof(arr[0][0]);
```

Passing Arrays to Functions

```
void printArray(int arr[], int size) {
    for(int i = 0; i < size; i++) {
        std::cout << arr[i] << " ";
    }
}
```

```
int arr[5] = {1, 2, 3, 4, 5};
int size = sizeof(arr) / sizeof(arr[0]);
printArray(arr, size);
```