Chapter 6 Multi-dimensional Arrays

Multi-dimensional arrays can be termed as arrays of arrays. Multi-dimensional arrays are arrays that contain other arrays as their elements.

2.1 Two-dimensional Arrays

Two-dimensional arrays (2D arrays) are arrays that contain one-dimensional arrays as their elements.

2.1.1 Declaring 2D Arrays

You can create an array of arrays. These arrays are known as multi-dimensional arrays. For example,

```
float x[3][4];
```

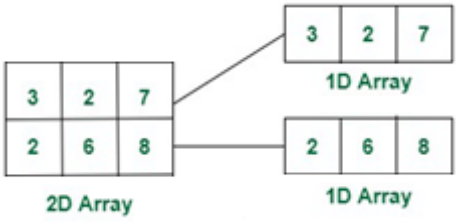
Here, x is a two-dimensional (2D) array. The array can hold 12 elements. You can think of the array as a table with 3 rows and each row has 4 columns.

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

2.1.2 Initializing 2D Arrays

You can initialize a two-dimensional array in two different ways as below:

```
int c[2][3] = {
    {3, 2, 7},
    {2, 6, 8}
};

int c[2][3] = {3, 2, 7, 2, 6, 8};
```



2.1.3 Accessing 2D Array Elements

Example 05: Access 2D array's elements

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

int main(){
    //Create a 2D array
    int arr[2][3] = {
           {1,2,3},
           {4,5,6}
    };

    // Update some elements
    arr[0][1] = 8;
    arr[1][2] = 9;

    // Display some elements
    cout << arr[0][0] << endl
         << arr[1][2];

    return 0;
}
```

Output

```
1
9
```

2.1.4 Iterating Through a 2D Array

You can use two loops to iterate through a 2D array. The first loop loops through each row of the 2D array one by one. As the first loop runs through each row, the second (nested) loop inside the first loop loops through the columns one by one. Let's look at an example below:

Example 06: Displaying the elements with index number from 2D Arrays.

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

int main(){
    //Array with 3 rows and 2 columns
    int array[3][2] = {
        {0, 1},
        {2, 3},
        {4, 5}
    };

    //Displaying the elements
    cout << "Elements with index numbers:" << endl;
    for(int i = 0; i < 3; i++)
        for(int j = 0; j < 2; j++)
            cout << "a[" << i << "][" << j << "] = " << array[i][j] << endl;

    return 0;
}
```

Output

```
Elements with index numbers:
a[0][0] = 0
a[0][1] = 1
a[1][0] = 2
a[1][1] = 3
a[2][0] = 4
a[2][1] = 5
```

2.2 Three-dimensional Arrays

Three-dimensional array (3D array) are arrays that contain two-dimensional arrays as their elements.

2.2.1 Declaring 3D Arrays

You can declare a three-dimensional (3D) array. For example,

```
float y[2][4][3];   // Here, the array y can hold 24 elements.
```

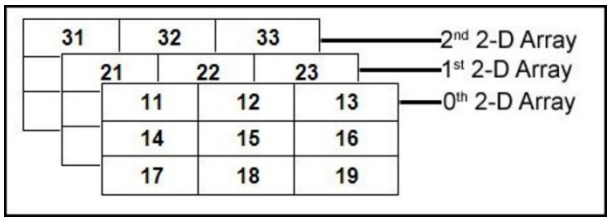
2.2.2 Initialization of 3D Arrays

You can initialize a three-dimensional array in a similar way like a two-dimensional array.

```
int a[2][3][4] = {
    {{3,  4, 2,  3}, {0, -3, 9, 11}, {23, 12, 23, 2}},
    {{13, 4, 56, 3}, {5,  9, 3, 5 }, {3,  1,  4,  9}}
};
```

Another example:

```
int b[3][3][3] = {
    {{11, 12, 13}, {14, 15, 16}, {17, 18, 19}},
    {{21, 22, 23}, {24, 25, 26}, {27, 28, 29}},
    {{31, 32, 33}, {34, 35, 36}, {37, 38, 39}}
};
```



2.2.3 Accessing 3D Array Elements

Example 07: Access 3D array's elements

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

int main(){
    //Create a 3D array
    int arr[2][3][4] = {
        {{3,  4, 2,  3}, {0, -3, 9, 11}, {23, 12, 23, 2}},
        {{13, 4, 56, 3}, {5,  9, 3, 5 }, {3,  1,  4,  9}}
    };

    // Update some elements
    arr[0][1][3] = 100;
    arr[1][2][3] = 200;

    // Display some elements
    cout << arr[0][0][1] << endl
         << arr[0][1][3];

    return 0;
}
```

Output

```
4
100
```

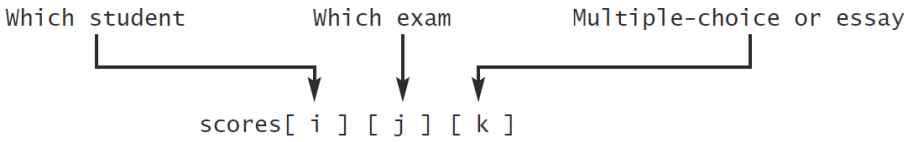
Example 08: For example, you can use a three-dimensional array to store exam scores for a class of four students with three exams, and each exam has two parts (multiple-choice and essay).

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

int main(){
    //Create a 3D array to store all scores
    float scores [4][3][2] = {
        {{7.5, 20.5}, {3.0, 12.5}, {15, 43.5}},
        {{4.5, 21.5}, {9.0, 52.5}, {35, 34.5}},
        {{3.5, 30.5}, {8.4, 10.5}, {11, 33.5}},
        {{9.5, 20.5}, {9.4, 42.5}, {23, 31.5}}
    };

    // Display the essay score of the 3rd student's 2nd exam
    cout << scores[2][1][1];

    return 0;
}
```

Output:

```
10.5
```

The `scores[0][1][0]` refers to the multiple-choice score for the first student's second exam, which is `9.0`. `[0][1][1]` refers to the essay score for the first student's second exam, which is `22.5`. This is depicted in the following figure:



2.2.4 Iterating Through a 3D Array

You can use three loops to iterate through a 3D array. Let's look at an example below:

Example 09: Enter and display the elements of 3D Arrays.

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

int main(){
    int arr[2][3][4];

    printf("Enter elements into a 3D array:\n");
    for(int i = 0; i < 2; i++)
        for(int j = 0; j < 3; j++)
            for(int k = 0; k < 4; k++)
                cin >> arr[i][j][k];

    printf("Display the elements of the 3D array:\n");
    for(int i = 0; i < 2; i++)
        for(int j = 0; j < 3; j++)
            for(int k = 0; k < 4; k++)
                cout << arr[i][j][k] << " ";

    return 0;
}
```

Output

```
Enter elements into a 3D array:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
Display the elements of the 3D array:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
```

3. Justifying the Output

`setw`() function is a part of `iomaip` library and is defined in `std` namespace. It stands for set width. This function specifies the minimum number of characters an output will consume.

Example 10: Using `setw`() function.

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

int main(){
    cout << left << setw(10) << "Hello" << endl;     // left justify
    cout << right << setw(10) << "Hello" << endl;    // right justify

    return 0;
}
```

Output:

```
Hello
      Hello
```

Practice

Display the following table:

ID	Name	Salary	Department
1	Chan Dara	80	ITE
2	Sok Sophea	900	BioE
3	Keo Tola	2000	TEE

Exercises

Write the following programs:

1. (Sort a list of points on y-coordinates) Write a program that sorts an 2D array of points on their y-coordinates. Each point has two values for x- and y-coordinates.

   For example, the points `{{4, 2}, {1, 7}, {4, 5}, {1, 2}, {1, 1}, {4, 1}}` will be sorted to `{{1, 1}, {4, 1}, {1, 2}, {4, 2}, {4, 5}, {1, 7}}`.

2. *(Points farthest to each other)* Write a program that finds two points farthest to each other. Use a 2D array store the points. Test the program using the following points:

   ```
   {{-1, 0}, {-1, -1}, {4, 1}, {2, 0.5}, {3.5, 2}, {3, 1.5}, {-1.5, 4}, {5.5, 4}};
   ```

   The formula for computing the distance between two points is: $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

3. A table below shows the seven-day work hours of six employees. The normal working hours is 30 hours per week. Rate per hour is 5$, while rate per hour for OT (overtime) is 10$.

	Su	M	T	W	Th	F	Sa
Employee 1	0	5	8	3	2	7	3
Employee 2	5	2	3	2	4	2	2
Employee 3	7	6	2	3	8	7	6
Employee 4	4	7	8	5	7	2	8
Employee 5	2	4	3	6	4	6	7
Employee 6	6	5	8	6	7	5	4

   Write a program that calculates the total working hours and the weekly salary for each employee. Display the result in a tabular format, for example:

```
--------------------------------------------------------------------------------
Name                    Total Hours             Salary
--------------------------------------------------------------------------------
Employee 1              28                      140
Employee 2              20                      100
…
Employee 6              41                      260
--------------------------------------------------------------------------------
```

4. There are 2 departments, ITE and DSE. Each department has 3 employees. Each employee works three days a week. The normal working hours is 15 hours per week. Rate per hour is 5$, while rate per hour for OT (overtime) is 10$.

   Write a program that calculates the total working hours and the weekly salary for each employee. Ask the user to enter the department name, and display the result in a tabular format, for example:

```
Enter a department: DSE
--------------------------------------------------------------------------------
Name                    Total Hours             Salary
--------------------------------------------------------------------------------
Employee 1              28                      140
Employee 2              20                      100
Employee 3              41                      260
--------------------------------------------------------------------------------
```

5. Ask a user for scores of 4 students. Each student studies 3 subjects, so each of them gets 3 scores. You must store those scores in a two-dimensional array first, then access the elements of the array to calculate the total score of each student and the total score of all students. Then, display the result in a tabular format, for example:

Name	Math	Physics	Chemistry	Total
Student 1	50	30	70	150
Student 2	60	40	80	180
Student 3	70	50	90	210
Student 4	80	60	100	240
			Total Score:	780

Reference

[1] Y. Daniel Liang. 'Introduction to Programming with C++', 3e – 2014

[2] Dey, P., & Ghosh, M. 'Computer Fundamentals and Programming in C', 2e – 2013

[3] https://cplusplus.com/doc/tutorial/

[4] https://www.programiz.com/cpp-programming

[5] https://www.studytonight.com/cpp/