C++ Multidimensional Arrays

In arrays, you learned about one dimensional array, that is, single variables specifies array. C++ allows programmer to create array of an array known as multidimensional arrays. Consider this example:

```
int x[3][4];
```

Here, *x* is a two dimensional array. This array can hold 12 elements. You can think this array as table with 3 row and each row has 4 column.

	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]

Three dimensional also array works in similar way. For example:

```
float x[2][4][3];
```

This array *x* can hold 24 elements. You can think this example as: Each 2 elements can hold 4 elements, which makes 8 elements and each 8 elements can hold 3 elements. Hence, total number of elements this array can hold is 24.

Multidimensional Array Initialisation

You can initialise a multidimensional array in more than one way. Consider this examples to initialise two dimensional array.

```
int test[2][3] = {2, 4, -5, 9, 0, 9};
```

Better way to initialise this array with same array elements as above.

```
int  test[2][3] = { {2, 4, 5}, {9, 0 0}};
```

Initialisation of three dimensional array

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

Better way to initialise this array with same elements as above.

```
int test[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} }
                      };
```

Example 1: Two Dimensional Array

C++ Program to display all elements of an initialised two dimensional array.

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

int main() {
     int test[3][2] = {
          {2, -5},
          {4, 0},
          {9, 1}
     };
     for(int i = 0; i < 3; ++i) {
          for(int j = 0; j < 2; ++j) {
               cout<< "test["<< i << "][" << ;j << "] = " << test[i][j]<<endl;
          }
     }
     return 0;
}
```

```
test[0][0] = 2
test[0][1] = -5
test[1][0] = 4
test[1][1] = 0
test[2][0] = 9
test[2][1] = 1
```

Example 2: Two Dimensional Array

C++ Program to store temperature of two different cities for a week and display it.

```
#include <iostream>
using namespace std;
const int CITY = 2;
const int WEEK = 7;

int main() {
     int temperature[CITY][WEEK];
     cout<<"Enter all temperature for a week of first city and then second city.
     for (int i = 0; i < CITY; ++i) {
          for(int j = 0; j < WEEK; ++j) {
               cout<<"City "<<i+1<<", Day "<<j+1<<" : ";
               cin>>temperature[i][j];
          }
     }
     cout<<"\n\nDisplaying Values:\n";
     for (int i = 0; i < CITY; ++i) {
          for(int j = 0; j < WEEK; ++j) {
               cout<<"City "<<i+1<<", Day "<<j+1<<" = "<< temperature[i][j]<<endl;
          }
     }
     return 0;
}
```

```
Enter all temperature for a week of first city and then second city.
City 1, Day 1 : 32
City 1, Day 2 : 33
City 1, Day 3 : 32
```

```
City 1, Day 4 : 34
City 1, Day 5 : 35
City 1, Day 6 : 36
City 1, Day 7 : 38
City 2, Day 1 : 23
City 2, Day 2 : 24
City 2, Day 3 : 26
City 2, Day 4 : 22
City 2, Day 5 : 29
City 2, Day 6 : 27
City 2, Day 7 : 23


Displaying Values:
City 1, Day 1 = 32
City 1, Day 2 = 33
City 1, Day 3 = 32
City 1, Day 4 = 34
City 1, Day 5 = 35
City 1, Day 6 = 36
City 1, Day 7 = 38
City 2, Day 1 = 23
City 2, Day 2 = 24
City 2, Day 3 = 26
City 2, Day 4 = 22
City 2, Day 5 = 29
City 2, Day 6 = 27
City 2, Day 7 = 23
```

Example 3: Three Dimensional Array

C++ Program to Store value entered by user in three dimensional array and display it.

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

int main() {
     int test[2][3][2];     // this array can store 12 elements
     cout<<"Enter 12 values: \n";
     for(int i = 0; i < 2; ++i) {
          for (int j = 0; j < 3; ++j) {
               for(int k = 0; k < 2; ++k ) {
                    cin>>test[i][j][k];
               }
          }
     }
     cout<<"\nDisplaying Value stored:"<<endl;
/* Displaying the values with proper index. */
     for(int i = 0; i < 2; ++i) {
          for (int j = 0; j < 3; ++j) {
               for(int k = 0; k < 2; ++k ) {
                    cout<< "test["<<i<<"]["<<j<<"]["<<k<<"] = "<< test[i][j][k]<<e
               }
          }
     }

     return 0;
}
```

Output

```
Enter 12 values:
1
2
3
4
5
```

```
6
7
8
9
10
11
12

Displaying Value stored:
test[0][0][0] = 1
test[0][0][1] = 2
test[0][1][0] = 3
test[0][1][1] = 4
test[0][2][0] = 5
test[0][2][1] = 6
test[1][0][0] = 7
test[1][0][1] = 8
test[1][1][0] = 9
test[1][1][1] = 10
test[1][2][0] = 11
test[1][2][1] = 12
```

As the number of dimension increases, the complexity also increases tremendously although the concept is quite similar.