8. *Two − Dimensional Array*

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

$Time\ Complexity : O(1)$

$[here\ p, q\ are\ constants, hence\ O(1)].$

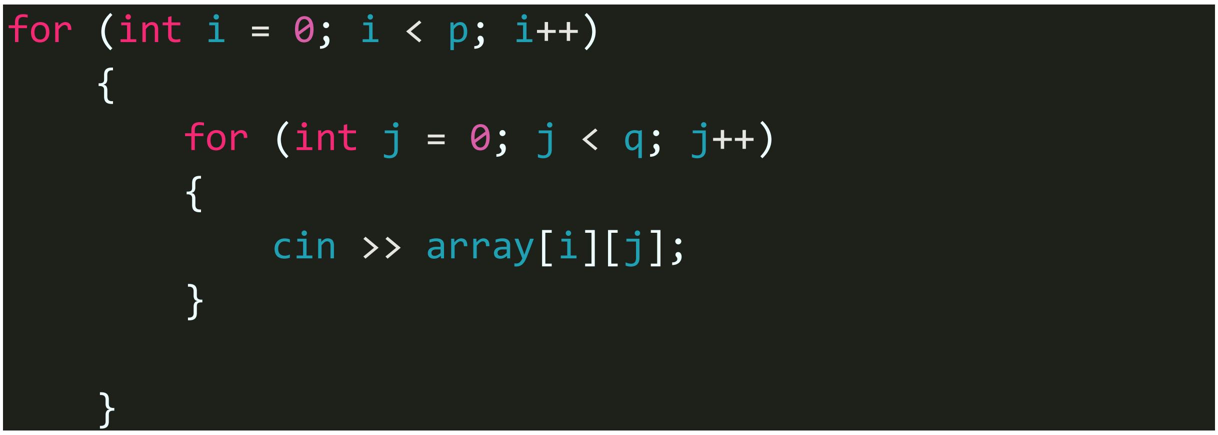
Traversal of Array

```
for (int i = 0; i < p; i++)
    {
        for (int j = 0; j < q; j++)
        {
            cout << array[i][j] << " ";
        }
        cout << endl;

    }
```

$Time\ Complexity : O(PQ)$

Insertion of Elements in Array



```
for (int i = 0; i < p; i++)
    {
        for (int j = 0; j < q; j++)
        {
            cin >> array[i][j];
        }

    }
```

Time Complexity : O(PQ)

Two − Dimensional Array − Time Complexity

The time complexity of entering elements into a two-dimensional array depends on the number of rows and columns in the array, as well as the method used to input the elements.

If we are entering each element manually, the time complexity would be O(NM), where N is the number of rows and M is the number of columns. This is because we would need to iterate through each element in the array to input its value, resulting in NM operations.

If we are reading in elements from a file or some other data source, the time complexity would depend on the algorithm used to read in the data. In the best case, the time complexity would be O(N*M), but in the worst case, it could be much higher if the data source is very large or requires complex parsing.

Overall, the time complexity of entering elements into a two-dimensional array is primarily determined by the number of elements that need to be entered, and the method used to input them.

****