# C++ Programming

# Multidimensional Arrays 1

**Mostafa S. Ibrahim**
*Teaching, Training and Coaching since more than a decade!*

*Artificial Intelligence & Computer Vision Researcher*
*PhD* from Simon Fraser University - Canada
*Bachelor / Msc* from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

# 2D Arrays Motivation

- Write a program that reads grades for students
  - 100 students
  - 20 subjects
- How can we code that?
  - Create 20 arrays grade1[100], grade2[100], …..grade20[100];
  - So impractical!
- Let’s visualize the data

# Grades visualization: 7 students x 4 subjects

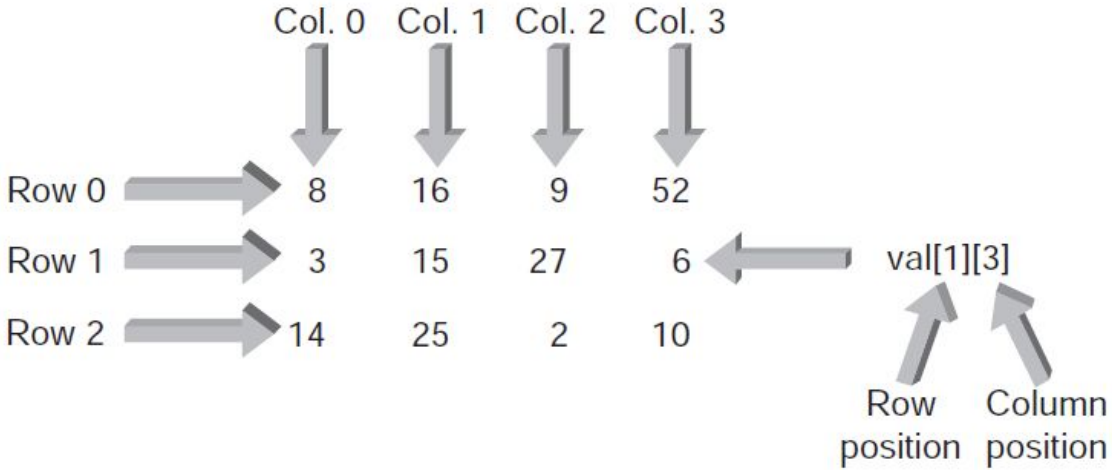
| | Math | Science | History | Arts |
|---|---|---|---|---|
| Mostafa | 50 | 33 | 40 | 30 |
| Asmaa | 35 | 50 | 44 | 17 |
| Belal | 30 | 35 | 50 | 37 |
| Ziad | 50 | 35 | 44 | 22 |
| Safa | 50 | 44 | 50 | 30 |
| Ashraf | 50 | 36 | 18 | 50 |
| Mona | 35 | 30 | **47** | 16 |

- This is called a matrix/table
  - The blue numbers
- 7 rows
  - Row 0, 1, 2, … 6
  - Row 0 for mostafa
  - Row 6 for mona
- 4 Columns
  - Column 0, 1, 2, 3
  - Column 0 for Math
- Value of table: row 6, col 2
  - 47 (Mona & History)
  - Notation: [6][2]

# 2D Arrays

- C++ saves our time by using 2D arrays
  - 2D = Table: rows x columns
- Same rules as 1D Arrays
- We create it as
  - double grades[7][4];
    - For 7 rows and 4 columns
  - To access in 2D arrays:
    - grades[6][2]

# 2D Arrays Visualization




```
int val[3][4] = {
        {8, 16, 9, 52},
        {3, 15, 27, 6},
        {14, 25, 2, 10}
};
cout<<val[1][3]<<"\n";  // 6
```

# Let’s put the values

```
4  int main() {
5
6      double grades[7][6] = {0};
7
8      // Mostafa Grades
9      grades[0][0] = 50, grades[0][1] = 33, grades[0][2] = 40, grades[0][3] = 30;
10
11     // Asmaa Grades
12     grades[1][0] = 35, grades[1][1] = 50, grades[1][2] = 40, grades[1][3] = 30;
13
14     // And so on
15
16     // Mona Grades
17     grades[6][0] = 35, grades[6][1] = 30, grades[6][2] = 47, grades[6][3] = 16;
18
19     return 0;
20 }
21
22
```

- Notice
- All mostafa data has grades[0]
- All Asmaa data has grades[1]
- All mona data has grades[6]
- Notice all inces
  - 0-6 for rows
  - 0-3 for columns

# Let’s print it

```
4  int main() {
5      double grades[7][6] = { 0 };
6
7      // Mostafa Grades
8      grades[0][0] = 50, grades[0][1] = 33, grades[0][2] = 40, grades[0][3] = 30;
9
10     // Asmaa Grades
11     grades[1][0] = 35, grades[1][1] = 50, grades[1][2] = 40, grades[1][3] = 30;
12
13     for (int row = 0; row < 7; ++row) {
14         cout << "Row " << row << ": ";
15         for (int col = 0; col < 4; ++col) {
16             cout << grades[row][col] << " ";
17         }
18         cout << "\n";
19     }
20     return 0;
21 }
22
```

```
<terminated> ztemp [C/C
Row 0: 50 33 40 30
Row 1: 35 50 40 30
Row 2: 0 0 0 0
Row 3: 0 0 0 0
Row 4: 0 0 0 0
Row 5: 0 0 0 0
Row 6: 0 0 0 0
```

- To print
  - Loop over every row
  - Then for this row
    - Loop on its columns
- We will loop this way typically
- We can also loop on columns then loop on rows

# Easier: Let's read then print!

```
4  int main() {
5      double grades[7][6] = { 0 };
6
7      for (int row = 0; row < 7; ++row)
8          for (int col = 0; col < 4; ++col)
9              cin >> grades[row][col];
10
11     for (int row = 0; row < 7; ++row) {
12         cout << "Row " << row << ": ";
13         for (int col = 0; col < 4; ++col) {
14             cout << grades[row][col] << " ";
15         }
16         cout << "\n";
17     }
18     return 0;
19 }
20
```

```
50 33 40 30 35 50 44 17 30 35 50 37 50 35 44
22 50 44 50 30 50 36 18 50 35 30 47 16
Row 0: 50 33 40 30
Row 1: 35 50 44 17
Row 2: 30 35 50 37
Row 3: 50 35 44 22
Row 4: 50 44 50 30
Row 5: 50 36 18 50
Row 6: 35 30 47 16
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."