CS 111
- Kangmei Yang
2D-Array

Why do we make such a fuzz to create two dimensional array?
-It helps us to better manage the data by grouping them

Ex:
Store grades for 10 students, each student has 3 exam grades.

int grades[30];
2D-Array

- A collection of data of the same type
- in 1D array, each cell stores 1 data
- in 2D array, each cell stores 1D array of data

Ex:
2 students, each student has 3 grades
- group by students

int grades[2][3]

[Blank grid for 2D-array representation]
2D-Array

- A collection of data of the same type
- For better visualization, we may see the 2D array as following, store the data in a table/matrix

Model:
```
datatype name[ row_size] [column_size]
```

Ex:
```
int grades[2][3]
```

- row 0, first student
- row 1, second student
Accessing Array

- only ONE data can be accessed at a time.
- we need to specify the index/position of this element in the array
- Specify row and column

```c
int grades[10]; ← declaration
grades[0][0]; grades[0][1]; grades[1][2];
```

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>89</td>
<td>93</td>
</tr>
<tr>
<td>100</td>
<td>67</td>
<td>75</td>
</tr>
</tbody>
</table>
```

grades[row_index][col_index]
index from 0 – capacity - 1
Accessing Array

Ex: int grades[2][3];

Assign first data in first row, first column to 100
  grades[0][0] = 100;

Output first data to console
  cout << grades[0][0];

Input first data from console
  cin >> grades[0][0];
Use loops to access entire Array

To access all the elements in the 2D-array, the entire table, we need a nested loop!

Ex: int grades[2][3]

Output the data from entire array
for (int r = 0; r < 2; r++) {
    for (int c = 0; c < 3; c++) {
        cout << grades[r][c] << endl;
    }
}
Initializing Array

One {} for each row,
int grades[2][3] = {{100, 90, 88}, {97, 64, 92}};