

---

**PG – DESD**

**Module – Embedded C Programming**

Trainer - Devendra Dhande

Email – [devendra.dhande@sunbeaminfo.com](mailto:devendra.dhande@sunbeaminfo.com)

Mobile No - 9890662093



## 2-D array

- Logically 2-D array represents m x n matrix i.e. m rows and n columns.

- `int arr[3][4] = { {1, 2, 3, 4}, {10, 20, 30, 40}, {11, 22, 33, 44} };`

- Array declaration:

- `int arr[3][4] = { {1, 2, 3, 4}, {10, 20, 30, 40}, {11, 22, 33, 44} };`
- `int arr[3][4] = { {1, 2 }, {10}, {11, 22, 33 } };`
- `int arr[3][4] = { 1, 2, 10, 11, 22, 33 };`
- `int arr[ ][4] = { 1, 2, 10, 11, 22, 33 };`

	0	1	2	3
0	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
1	<b>10</b>	<b>20</b>	<b>30</b>	<b>40</b>
2	<b>11</b>	<b>22</b>	<b>33</b>	<b>44</b>



## 2-D array

- 2-D array is collection of 1-D arrays in contiguous memory locations.
  - Each element is 1-D array.
- `int arr[3][4] = { {1, 2, 3, 4}, {10, 20, 30, 40}, {11, 22, 33, 44} };`

arr	0				1				2			
	0	1	2	3	0	1	2	3	0	1	2	4
	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>10</b>	<b>20</b>	<b>30</b>	<b>40</b>	<b>11</b>	<b>22</b>	<b>33</b>	<b>44</b>
	400	404	408	412	416	420	424	428	436	440	444	448
400					416				436			



## 2-D array and Pointer

- Pointer to array is pointer to 0<sup>th</sup> element of the array.
  - Scale factor of the pointer = number of columns \* sizeof(data-type).
- `int arr[3][4] = { {1, 2, 3, 4}, {10, 20, 30, 40}, {11, 22, 33, 44} };`
- `int (*ptr)[4] = arr;`

		0				1				2			
		0	1	2	3	0	1	2	3	0	1	2	4
		1	2	3	4	10	20	30	40	11	22	33	44
		400	404	408	412	416	420	424	428	436	440	444	448
ptr	arr	400				416				436			

Diagram illustrating a 2D array `arr` of type `int` (4 columns) and a pointer `ptr` of type `int (*) [4]`. The pointer `ptr` is located at memory address 1000 and contains the value 400. An arrow points from the value 400 in `ptr` to the first row of the array `arr`, which starts at address 400. The array `arr` is organized into 3 rows and 4 columns. The first row (index 0) contains values 1, 2, 3, 4. The second row (index 1) contains values 10, 20, 30, 40. The third row (index 2) contains values 11, 22, 33, 44. The memory addresses for the first row are 400, 404, 408, 412. The addresses for the second row are 416, 420, 424, 428. The addresses for the third row are 436, 440, 444, 448.



# Passing 2-D array to Functions

- 2-D array is passed to function by address.
- It can be collected in formal argument using array notation or pointer notation.
- While using array notation, giving number of rows is optional. Even though mentioned, will be ignored by compiler.





Thank you!

Devendra Dhande <[devendra.dhande@sunbeaminfo.com](mailto:devendra.dhande@sunbeaminfo.com)>

