PG – DESD

Module – Embedded C Programming

Trainer - Devendra Dhande

Email – 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 };

$$\begin{array}{c|c|c|c|c|}
 & 0 & 1 & 2 & 3 \\ \hline
0 & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} \\ \hline
1 & \mathbf{10} & \mathbf{20} & \mathbf{30} & \mathbf{40} \\ \hline
2 & \mathbf{11} & \mathbf{22} & \mathbf{33} & \mathbf{44} \\ \hline
\end{array}$$

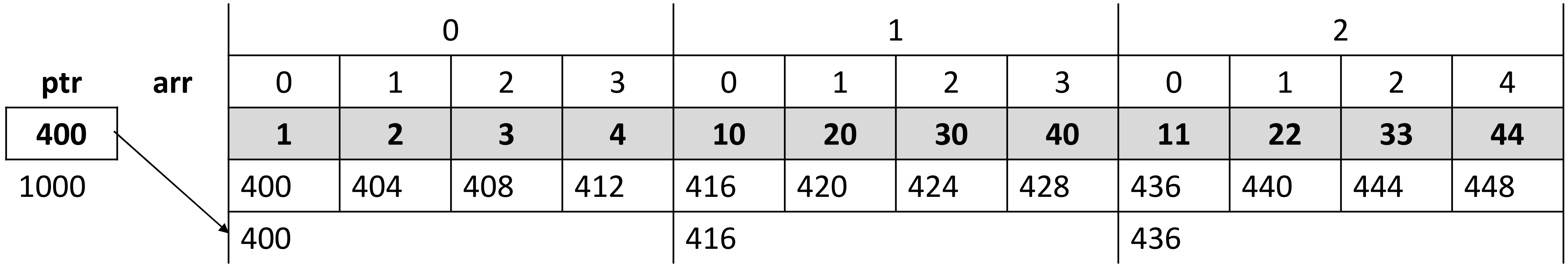
2-D array

- 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} };

	0				1				2			
	0	1	2	3	0	1	2	3	0	1	2	4
arr	**1**	**2**	**3**	**4**	**10**	**20**	**30**	**40**	**11**	**22**	**33**	**44**
	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 0th 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
400				416				436			

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>