

`int myArray[6];`
↑
size of the array

`int myArray[2][3];`
↑ ↑
number number
of rows of columns

`myArray[1][0] = 4;`

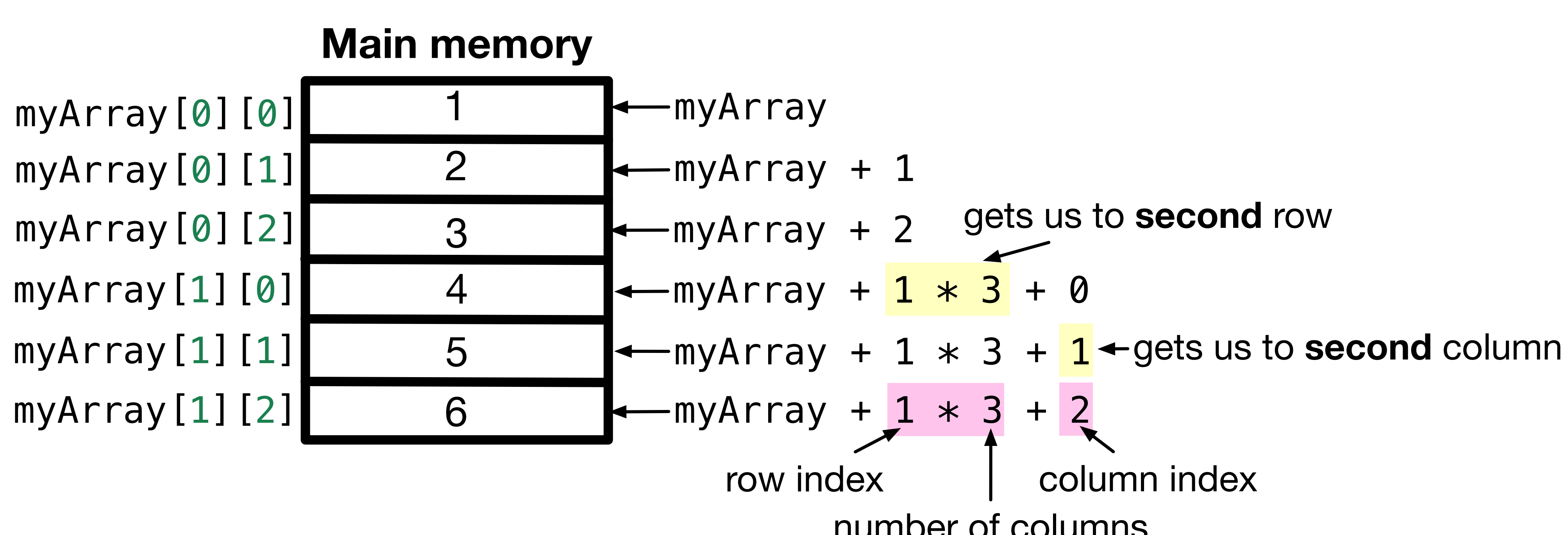
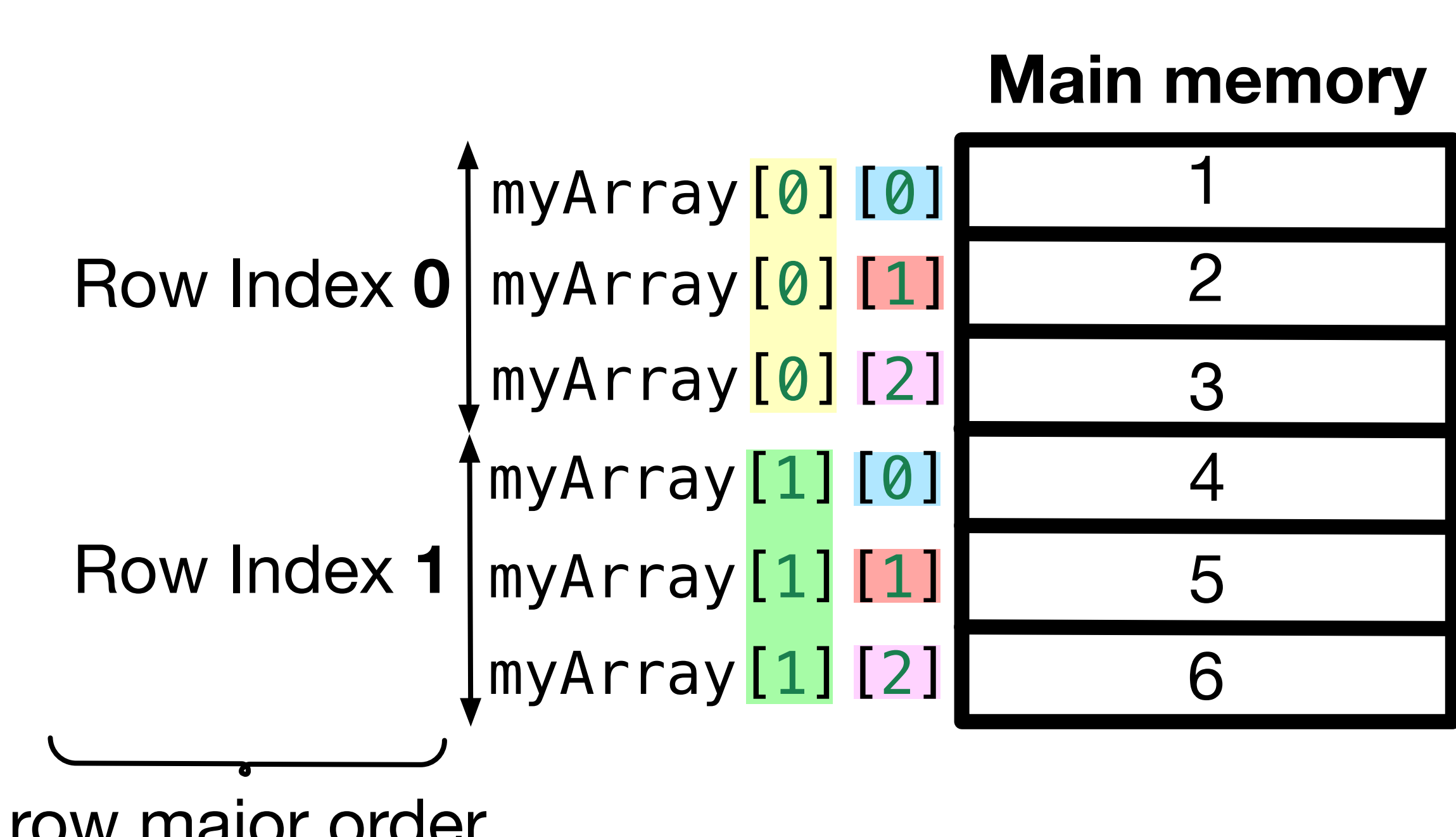
	0	1	2
0	1 [0][0]	2 [0][1]	3 [0][2]
1	4 [1][0]	5 [1][1]	6 [1][2]

`int myArray[2][3] = {{1, 2, 3}, {4, 5, 5}};`

`int myArray[2][3] = {{1, 2, 3}, {4, 5, 6}};`
↑ ↑ ↑ ↑
encloses individual rows
↑
encloses the entire array

`int myArray[2][3] = {1, 2, 3, 4, 5, 6};`
↑
encloses the entire array

`int myArray[][3] = {1, 2, 3, 4, 5, 6};`
↑
number of rows is unnecessary
while initializing

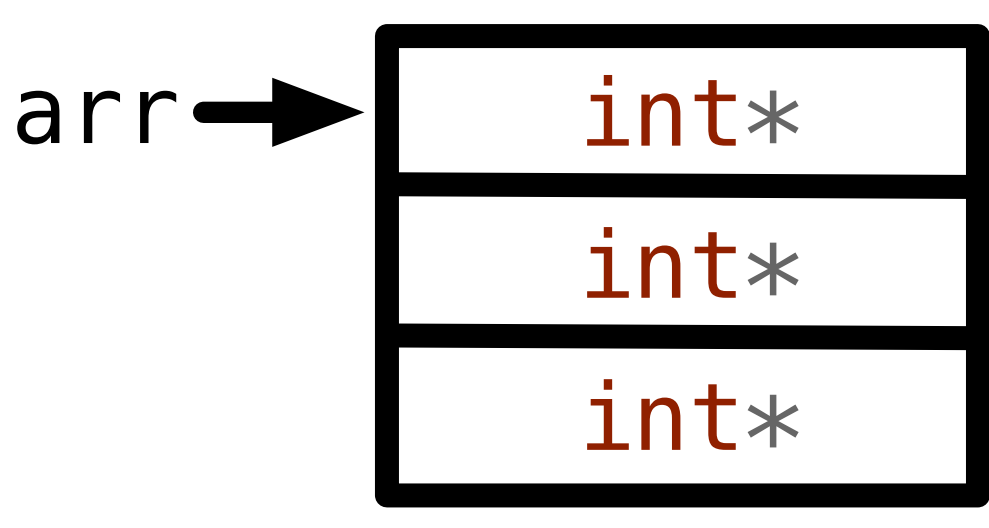


`myArray[i][j] ↔ *(myArray + i * <num of columns> + j)`

`&myArray[i][j] ↔ myArray + i * <num of columns> + j`

column index
row index

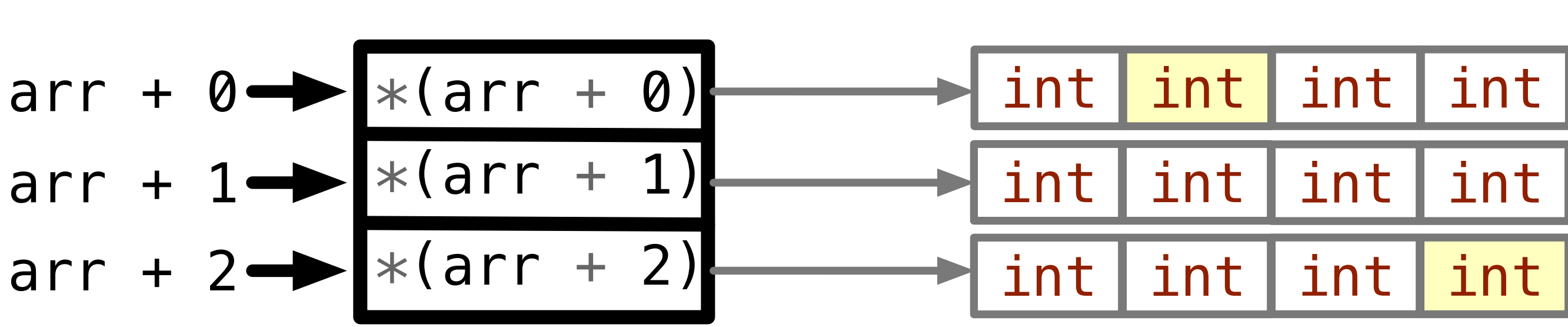
	0	1	2	3	4	5
0	0	1	1	0	0	0
1	0	1	0	1	1	0
2	0	1	1	1	0	0
3	0	0	1	1	1	0
4	0	0	0	1	1	0
5	1	1	1	0	1	1



`int** arr = (int**) malloc(3 * sizeof(int*));`

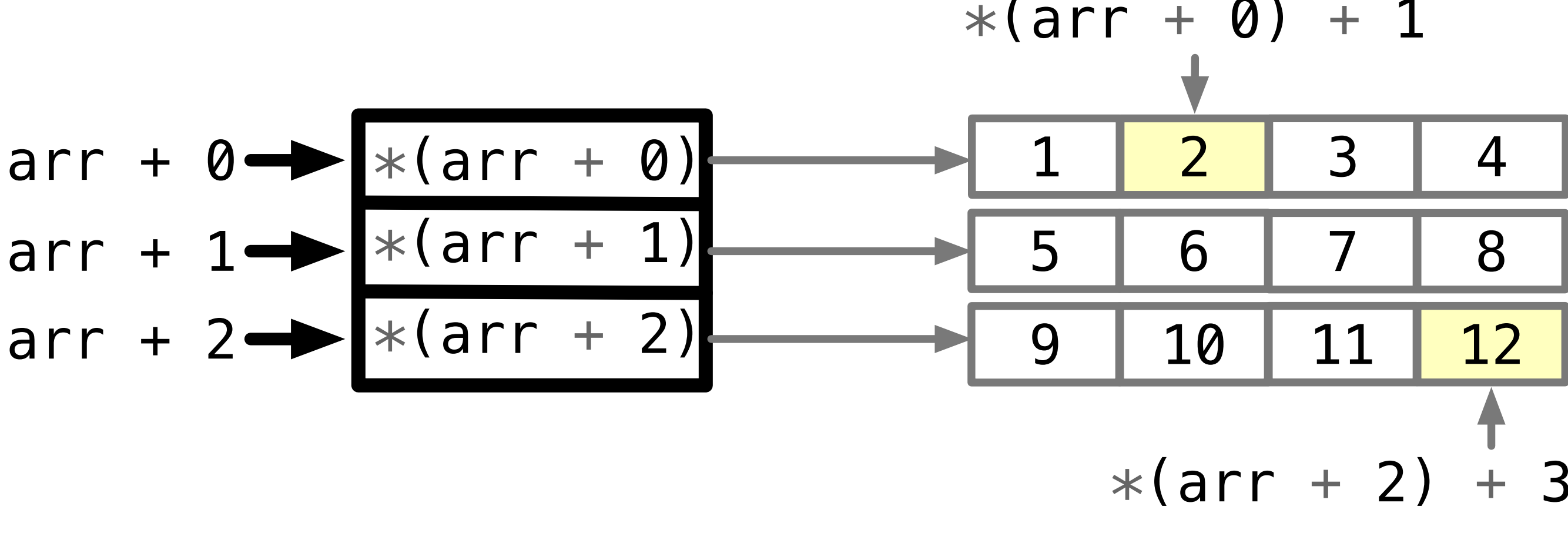
↓ ↓ ↓ ↓
A pointer that points to a We have Each element of the
pointer is a double pointer, 3 rows array **will point to**
hence we need ** **the first element in**
 the row, which is
 an **int**

	0	1	2	3
0	1 [0][0]	2 [0][1]	3 [0][2]	4 [0][3]
1	5 [1][0]	6 [1][1]	7 [1][2]	8 [1][3]
2	9 [2][0]	10 [2][1]	11 [2][2]	12 [2][3]



`for (int row = 0; row < 3; row++) {
 (arr + row) = (int) malloc(4 * sizeof(int));
}`

↓ ↓ ↓ ↓
The pointer to the Pointer to Number of Each
first element of the first element columns in element in
1D array having of the array a row is 4 the row is
elements of a row an **int**



`for (int row = 0; row < 3; row++) {
 for (int col = 0; col < 4; col++) {
 ((arr + row) + col) = row * 4 + col + 1;
 }
}`

Address to the first
element of row Address of element at
row index row and
column index col

// OR
// arr[row][col] = row * 4 + col + 1;

