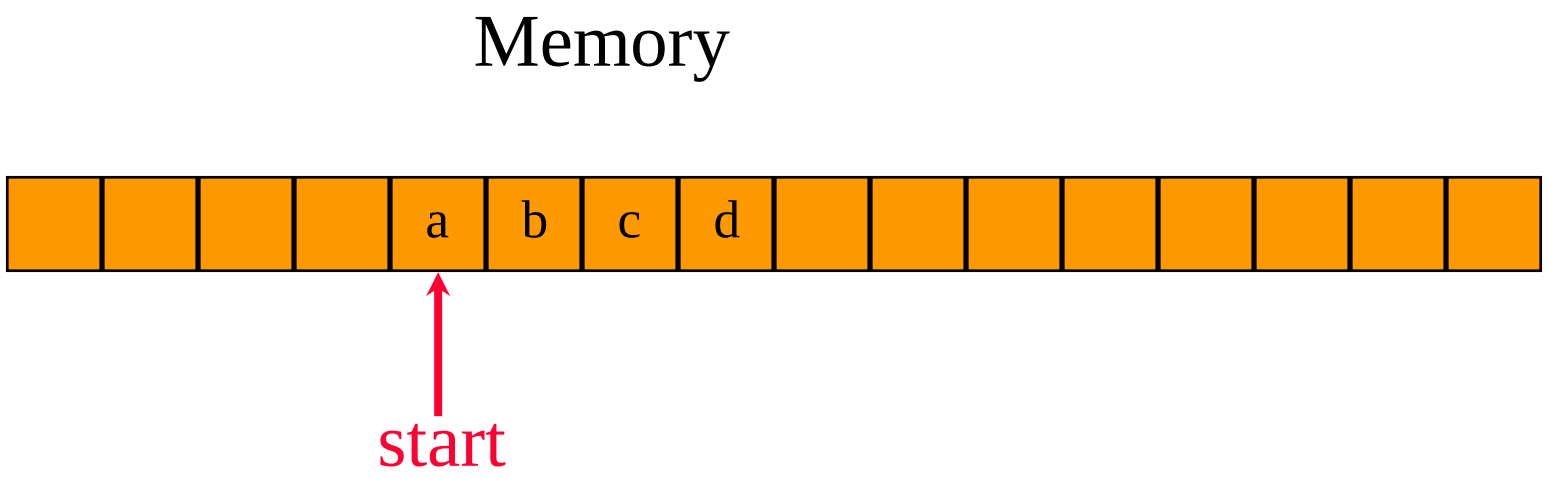
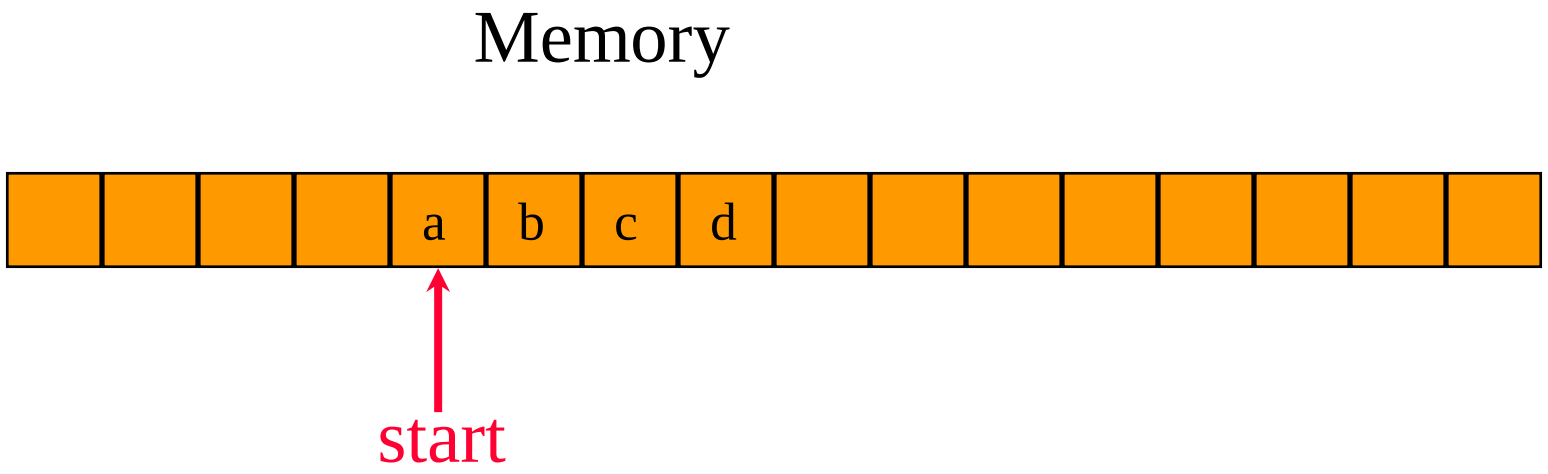
1D Array Representation In C++



- 1-dimensional array x = [a, b, c, d]
- map into contiguous memory locations

- location(x[i]) = start + i

Space Overhead



space overhead = 4 bytes for start

(excludes space needed for the elements of x)

2D Arrays

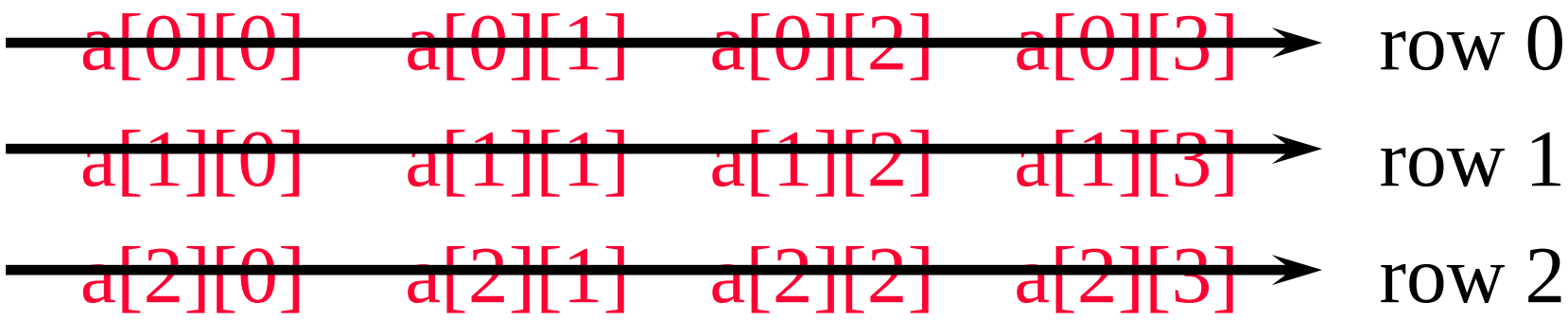
The elements of a 2-dimensional array a declared as:

```
int [][]a = new int[3][4];
```

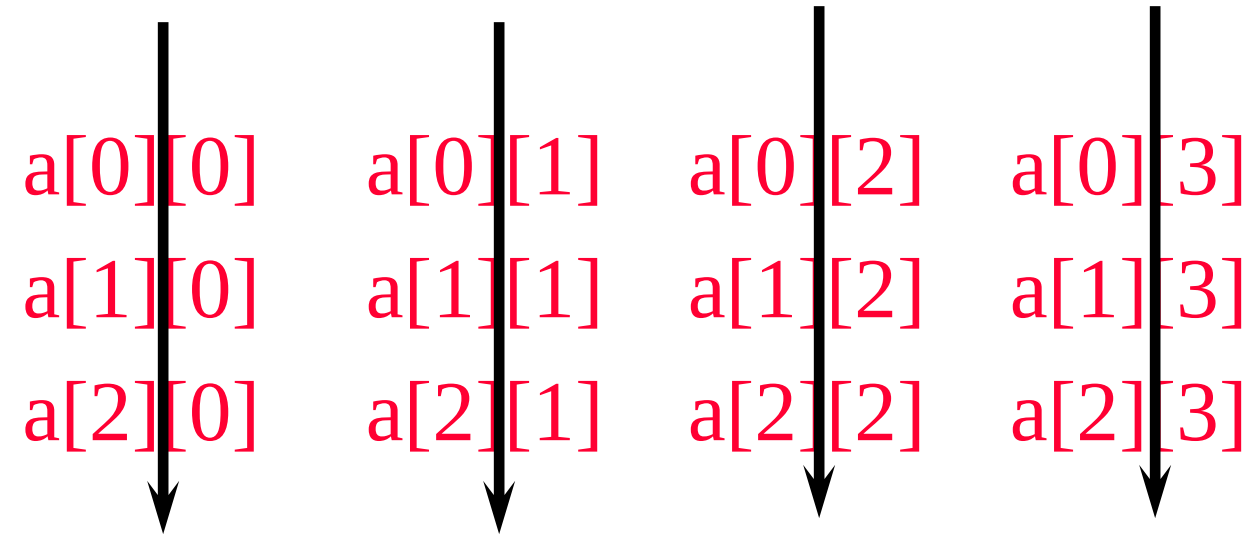
may be shown as a table

a[0][0]	a[0][1]	a[0][2]	a[0][3]
a[1][0]	a[1][1]	a[1][2]	a[1][3]
a[2][0]	a[2][1]	a[2][2]	a[2][3]

Rows Of A 2D Array



Columns Of A 2D Array



2D Array Representation In C++

2-dimensional array x

a, b, c, d

e, f, g, h

i, j, k, l

view 2D array as a 1D array of rows

x = [row0, row1, row 2]

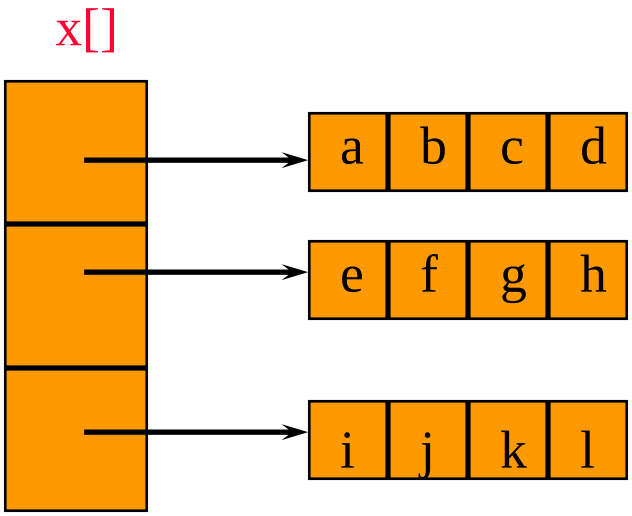
row 0 = [a,b, c, d]

row 1 = [e, f, g, h]

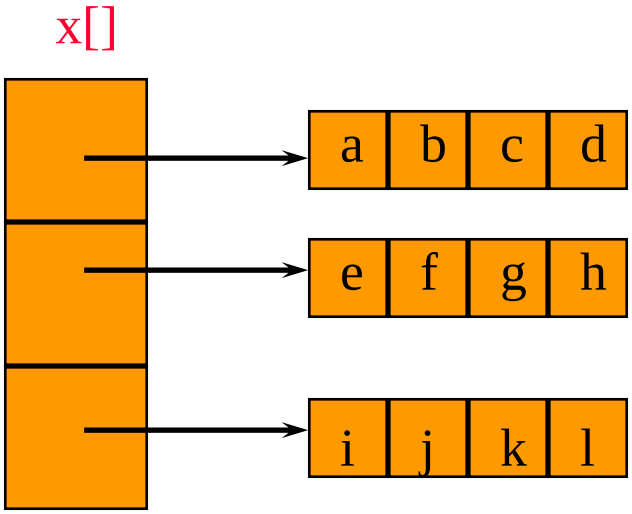
row 2 = [i, j, k, l]

and store as 4 1D arrays

2D Array Representation In C++



Space Overhead



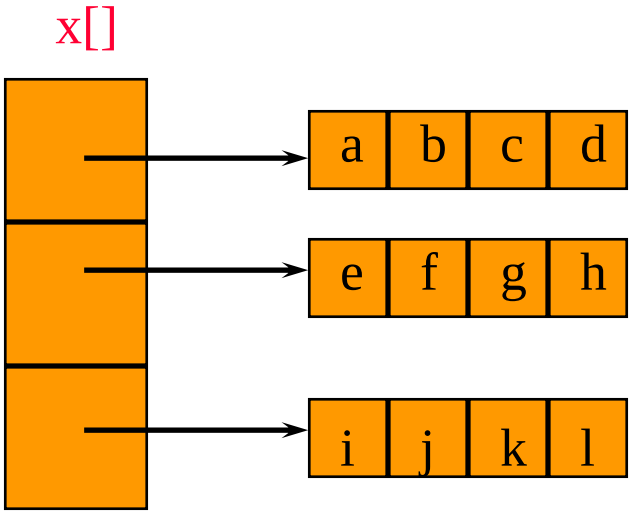
space overhead = overhead for 4 1D arrays

= 4 * 4 bytes

= 16 bytes

= (number of rows + 1) x 4 bytes

Array Representation In C++



- This representation is called the array-of-arrays representation.
- Requires contiguous memory of size 3, 4, 4, and 4 for the 4 1D arrays.
- 1 memory block of size number of rows and number of rows blocks of size number of columns

Row-Major Mapping

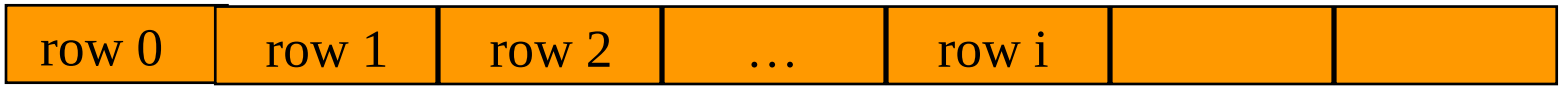
- Example 3 x 4 array:

a b c d

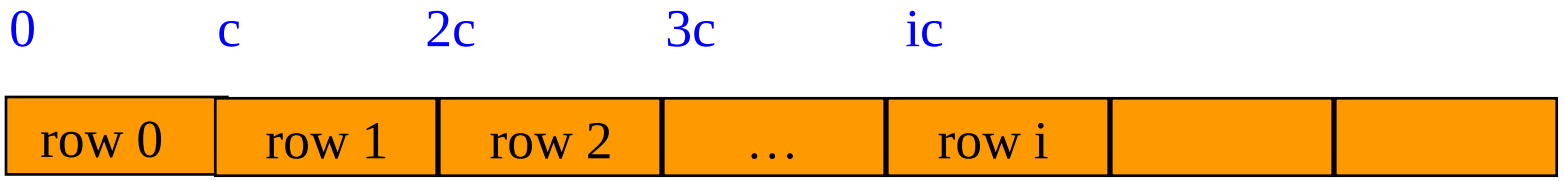
e f g h

i j k l

- Convert into 1D array y by collecting elements by rows.
- Within a row elements are collected from left to right.
- Rows are collected from top to bottom.
- We get y[] = {a, b, c, d, e, f, g, h, i, j, k, l}



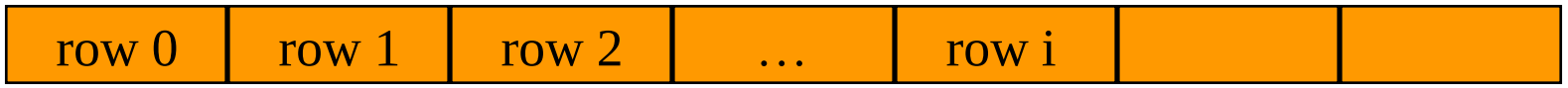
Locating Element x[i][j]



- assume x has r rows and c columns
- each row has c elements
- i rows to the left of row i
- so ic elements to the left of x[i][0]
- so x[i][j] is mapped to position

  ic + j of the 1D array

Space Overhead



4 bytes for start of 1D array +

4 bytes for c (number of columns)

= 8 bytes

Advantage

- The space overhead for the row-major scheme is 8 bytes regardless of the number of rows in the array.
- In the array of arrays scheme, the space overhead is 4 * (r+1) bytes.

So the overhead for an array with 10000 rows is 40004 bytes when the array of array scheme is used and only 8 bytes when the row-major scheme is used.