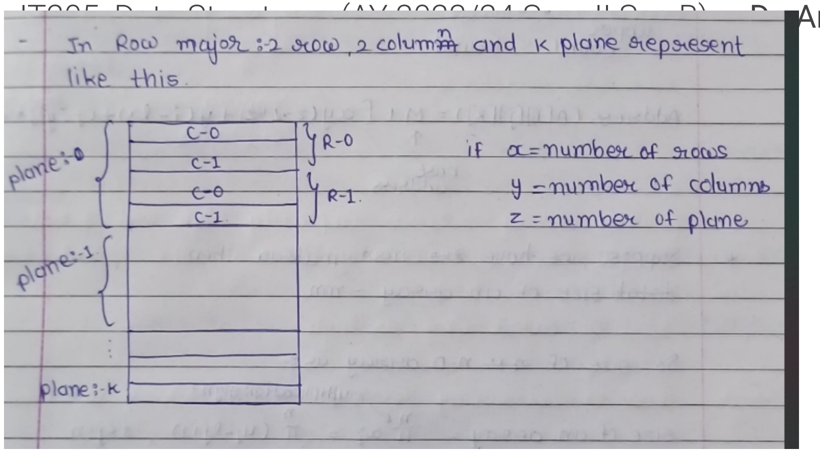
14 to 18 baki

Lecture 11

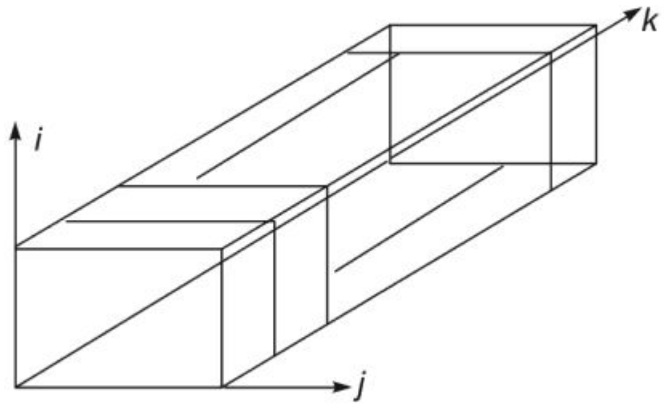
- Three Dimensional Arrays
- Pointer Arrays



Three-Dimensional (3-D) Array

A collection of ***homogeneous*** elements where the elements are specified by three subscripts:

- the first subscript specifies a **row number**, the second subscript a **column number**, and the third a **plane/page number**
- The number of elements in an array is the product of the ranges of all its dimensions.



usually we take first subscript as page,second as row and third as column.

row major for 2D = m + n*i + j (R→L)
A[i][j]

column major = m + nj + i (L→R

column | a_{31} a_{41} | a_{22} a_{32} a_{42} | a_{13} a_{23} a_{33} a_{43} |
C-1 C-2 C-3

$a_{33} = m + (2+3) + 2$ $a_{43} = m + (2+3) + 3$
$a_{32} = m + (2\cdot) + 2$

$a[i][j] = m + ((n-1) + n + n+1 + n+2 \ldots n-(i-2)) + c$

* 4-D array

$A[i_1][i_2][i_3][i_4][i_5] \Longrightarrow R \rightarrow L$

row-major type A[d1][d2][d3][d4]

$Add[i_1][i_2][i_3][i_4] = m + [i_1 * d_2 d_3 d_4 + i_2 d_3 d_4 + i_3 d_4 + i_4] * w$

4D → 3+2+1 = 6
n → n + n-1 + n-2 + n-3 + … 3+2+1
→ $\frac{n(n+1)}{2} = O(n^2)$

column major

$adr[i_1][i_2][i_3][i_4] = m + [i_4 * d_3 d_2 d_1 + i_3 * d_2 d_1 + i_2 * d_1 + i_1] * w$

* for n dimension Column major

$= L_0 + \sum_{p=1}^{n} \left[i_p \prod_{q=p+1}^{n} d_q \right] * w$ $= L_0 + \sum_{p=n}^{1} i_p \prod_{q=p-1}^{n-1} d_q]$ $d_0 = 1$

row major

→ so it is very time consuming.

$A[i_1][i_2][i_3][i_4] = m + (i_1 * d_2 * d_3 * d_4 + i_2 * d_3 * d_4 + i_3 * d_4 + i_4] * w$

→ $i_4 + i_3 * d_4 + i_2 * d_3 * d_4 + i_1 * d_2 * d_3 * d_4$
$= i_4 + d_4 * [i_3 + d_3 * [i_2 + i_1 d_2]]$
1 1 1 → 3

4D → 3
So for nD → n-1 → O(n)

* for 3-D

int A[l][m][n]

$adr(A[i][j][k]) = m + [i \times m \times n + j \times n + k] * w$
row →

column major $adr A[i][j][k] = m + [k \times m \times l + j \times l + i] * w$

* Link-List

Three-Dimensional (3-D) Array

We will use the following specifications for a 3-D array:

- Number of rows = x (number of elements in a column)
- Number of columns = y (number of elements in a row)
- Number of pages (planes) = z

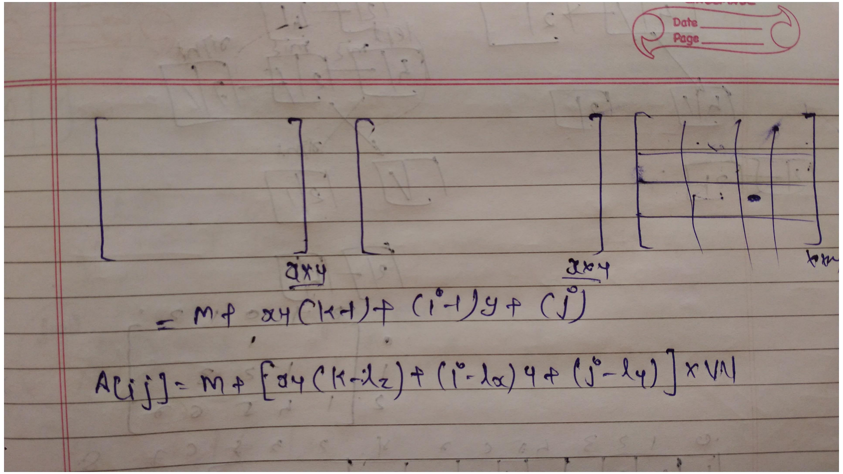
Storing a 3-D array means storing pages one-by-one. A row-major representation denotes that the elements of a plane are stored in a row-major fashion.

Three-Dimensional (3-D) Array

Row-major Order

- Address (A[i][j][k]) = number of elements in the first (k - 1) pages + the number of elements in the k^{th} page up to (i - 1) rows + the number of elements in the k^{th} page, in the i^{th} row, up to the j^{th} column

$$= xy\,(k-1) + (i-1)\,y + j$$



Three-Dimensional (3-D) Array

Row-major Order

If we assume that i varies from l_x to u_x, j varies from l_y to u_y, and k varies from l_z to u_z; such that

$$x = u_x - l_x + 1,\ y = u_y - l_y + 1,\ z = u_z - l_z + 1$$

- Address (A[i][j][k]) = number of elements in the first (k - 1) pages +
  the number of elements in the k^{th} page up to (i - 1) rows +
  the number of elements in the k^{th} page, in the i^{th} row, up to the j^{th} column

$$= M + [xy\,(k - l_z) + (i - l_x)\,y + (j - l_y)] \times w$$

Here, M is the base address of the array.

N - Dimensional Array

In *n*-dimensional array, we need ***n*** indices to identify an element, $i_1, i_2, \ldots, i_n$

Let $\mathbf{x}_j$ be the number of elements in $\mathbf{j}^{th}$ dimension and suppose the range of index for $\mathbf{i}_j$, say, varies between $\mathbf{l}_j$ and $\mathbf{u}_j$, where 1 <= j <= n.

So, the total number of elements in the array is -

$$\prod_{j=1}^{n} x_j = \prod_{j=1}^{n} (u_j - l_j + 1), \qquad 1 \le j \le n$$

N - Dimensional Array

Any element can be referenced using the following formula -

Row-major Order

$$\text{Address } A[i_1][i_2]...[i_n] = (i_n - l_n)x_{n-1}\, x_{n-2}\, x_{n-3} \ldots x_3x_2x_1 + (i_{n-1} - l_{n-1})x_{n-2}x_{n-3} \ldots x_3x_2x_1 + \cdots + (i_2 - l_2)\, x_1 + (i_1 - l_1)$$

Sir make formula on basis of frequence so both formula so both looks different

N - Dimensional Array

Any element can be referenced using the following formula -

Row-major Order

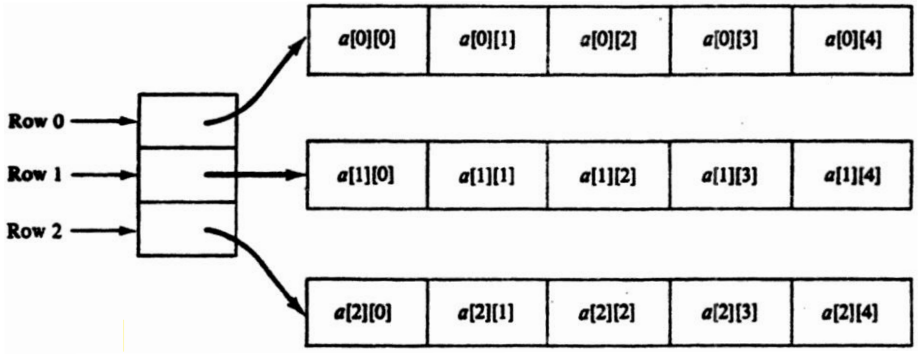
$$\text{Address } A[i_1][i_2]...[i_n] = (i_n - l_n)x_{n-1}\, x_{n-2}\, x_{n-3} \dots x_3x_2x_1 + (i_{n-1} - l_{n-1})x_{n-2}x_{n-3} \dots x_3x_2x_1 + \cdots + (i_2 - l_2)\, x_1 + (i_1 - l_1)$$

Column-major Order

$$\text{Address } A[i_1][i_2]...[i_n] = (i_1 - l_1)\, x_2x_3 \dots x_{n-2}x_{n-1}x_n + (i_2 - l_2)\, x_3x_4 \dots x_{n-2}x_{n-1}x_n + (i_{n-1} - l_{n-1})\, x_{n-2}\, x_{n-1}\, x_n + (i_n - l_n)$$

Pointer Array: Two-Dimensional

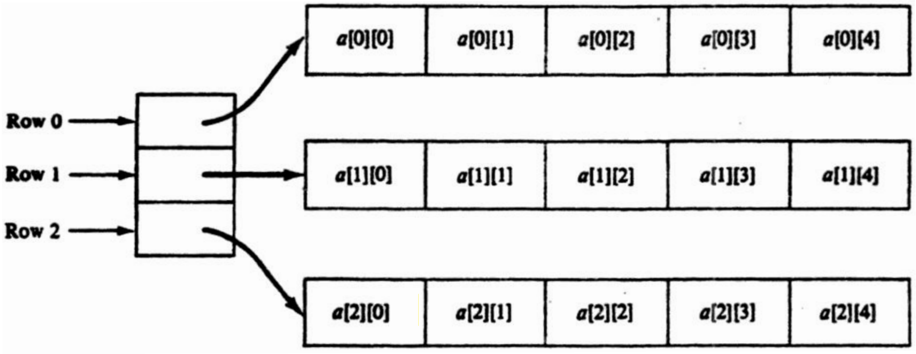
A 2-D array "a" declared with upper bounds u_1 and u_2, consists of $u_1 - l_1 + 1$ one-dimensional arrays.

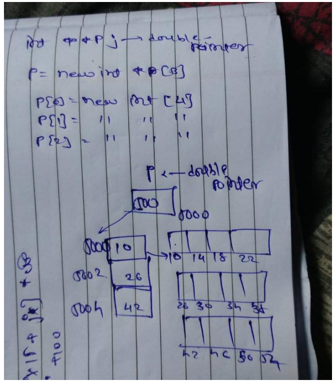


- The first is an array "ap" of u_1 pointers.
- The i^{th} element of "ap" is a pointer to a one-dimensional array a[i]
- To reference a[i][j], the array "ap" is accessed to obtain the pointer a[i], then array at that pointer is accessed to subsequently obtain the element a[i][j].

Pointer Array: Two-Dimensional

A 2-D array “a” declared with upper bounds u_1 and u_2, consists of $u_1 - l_1 + 1$ one-dimensional arrays.



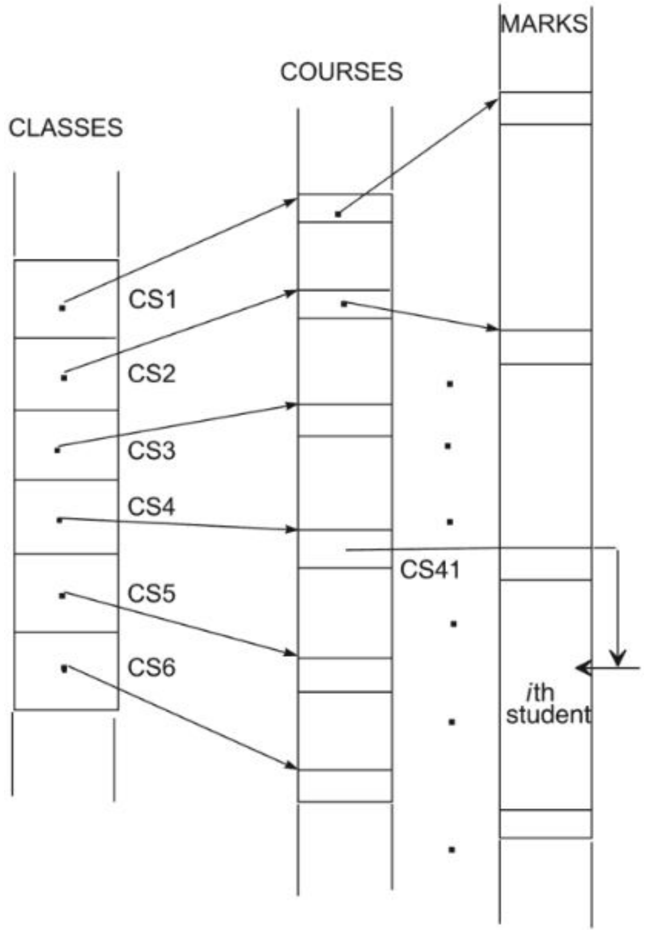


The first implementation (which we studied in the earlier lectures) avoids allocating the extra pointer array, “op”. and computing the value of an explicit pointer to the desired row array. It is therefore more efficient in both space and time.

Pointer Array: Example

Suppose we want to store the marks of all the students of CS department for a year. There are 6 classes and for each class there are at most 5 courses. We assume that there are at most 30 students in each class.

- In our classical representation, we need to have a one-dimensional array of size 6 x 5 x 30 = 900.
- Instead, we will use pointer arrays to keep track of marks the i^{th} year student in a course.
- We have to maintain arrays for Classes, Courses, and Marks of sizes 6, 30, and 900 respectively.



Exercise

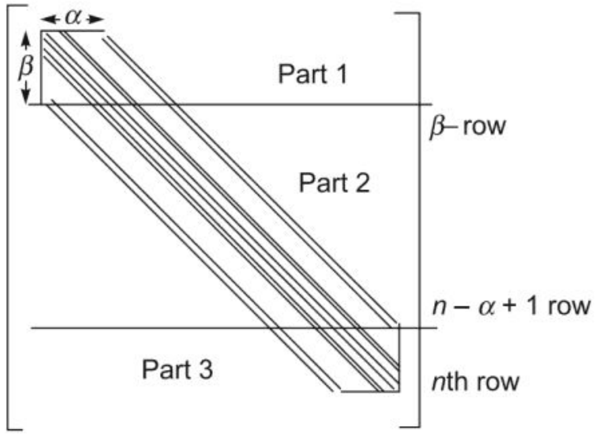
(a) Using only a single array of marks having size 900, give an idea as to how the same information can be maintained. You must consider the case that there may be less than 5 courses in a class or may be less than 30 students in a course.

(b) Other than one-dimensional array, can the two-dimensional or three-dimensional arrays be employed? How?

Last Class Exercise

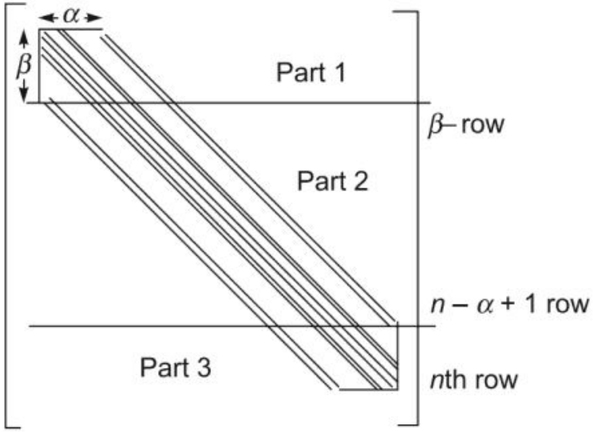
What will be the indexing formula for the **αβ**-band matrix using both the schemes?

<u>Hint:</u> there will be (**α** - 1) and (**β** - 1) sub-diagonals above and below the main diagonal respectively. As there will be three possible arrangements so the elements in each arrangement can be referred by three indexing formulas corresponding to the three parts shown below.



Solution of the Exercise

Considering the **row-major ordering** of the memory allocation, the indexing formula is explained below.



Case 1: $1 \le i \le \beta$

Address (a_{ij}) = Number of elements in the first $(i-1)$th rows + number of elements in the ith row up to the jth column

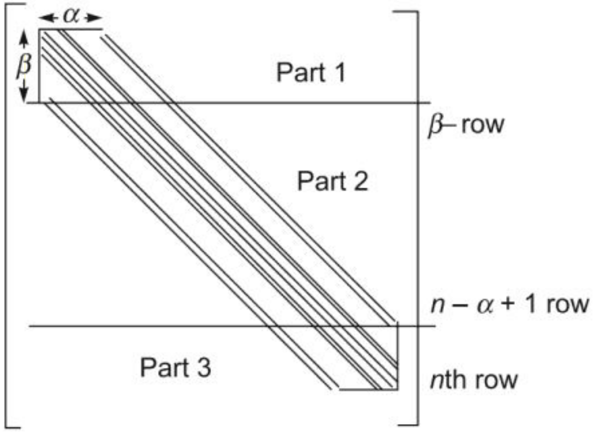
$$= \alpha + (\alpha + 1) + (\alpha + 2) + \dots + (\alpha + i - 2) + j$$

$$= \alpha \times (i-1) + [1 + 2 + 3 + \cdots + (i-2)] + j$$

$$= \alpha \times (i-1) + \frac{(i-1) \times (i-2)}{2} + j$$

Solution of the Exercise

Considering the **row-major ordering** of the memory allocation, the indexing formula is explained below.

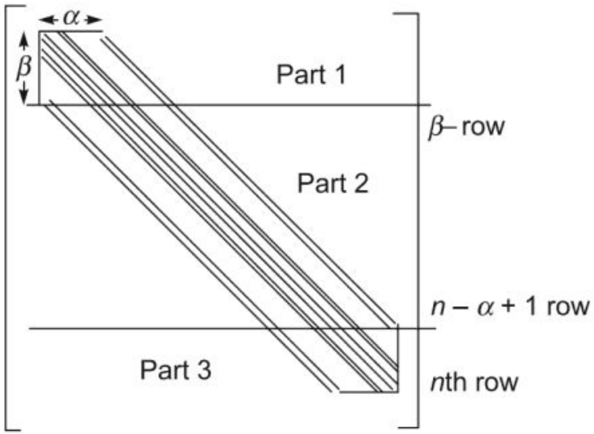


Case 2: $\beta < i \leq n - \alpha + 1$

$$\begin{aligned}
\text{Address } (a_{ij}) &= \text{Number of elements in the first } \beta \text{ rows} \\
&\quad + \text{number of elements between } (\beta + 1)\text{th row and the } (i - 1)\text{th row} \\
&\quad + \text{number of elements in } i\text{th row} \\
&= \alpha + (\alpha + 1) + (\alpha + 2) + \cdots + (\alpha + \beta - 1) \\
&\quad + (\alpha + \beta - 1) \times (i - \beta - 1) + j - i + \beta \\
&= \alpha\beta + \frac{\beta(\beta - 1)}{2} + (\alpha + \beta - 1)(i - \beta - 1) + j - i + \beta
\end{aligned}$$

Solution of the Exercise

Considering the **row-major ordering** of the memory allocation, the indexing formula is explained below.



Case 3: $n - \alpha + 1 < i$

Address (a_{ij}) = Number of elements in the first $(n - \alpha + 1)$ rows
+ number of elements after the $(n - \alpha + 1)$th row and up to the $(i - 1)$th row
+ number of elements in ith row and up to jth column

$$= \alpha\beta + \frac{\beta(\beta-1)}{2} + (\alpha+\beta-1)(n-\alpha-\beta+1) + (\alpha+\beta-2)$$
$$+ (\alpha + \beta - 3) + \cdots + \{\alpha + \beta - [(i-1) - (n - \alpha + 1)]\} + j - i + \alpha$$

$$= \alpha\beta + \frac{\beta(\beta-1)}{2} + (\alpha+\beta-1)(n-\alpha-\beta+1) + (\alpha+\beta)(i-n+\alpha-1)$$
$$- \{1 + 2 + 3 + \dots + [(i-1) - (n - \alpha + 1)]\} + 1$$

$$= \alpha\beta + \frac{\beta(\beta-1)}{2} + (\alpha+\beta-1)(n-\alpha-\beta+1) + (\alpha+\beta)(i-n+\alpha-1)$$
$$- \frac{(i-n+\alpha-1)\times(i-n\times\alpha-2)}{2} + 1$$

Next Lecture

- Linked Lists