Array

Definition: “An array is a collection of variables of the same type that are referenced by a common name.”

- Arrays are a way to group a number of items into a larger unit. In C, array elements are stored in contiguous (consecutive) memory locations. The lowest address corresponds to the first element and the highest address to the last element. Arrays can have data items of simple types like **int** or **float** or even of user-defined types like structure or objects.

Types of Arrays

Types of Arrays:

Arrays are of different types:

- **Single-dimensional arrays**, comprised of finite homogenous(same type) elements.

For e.g.int marks[10];

- **Multi-dimensional arrays**, comprised of elements, each of which is itself an array. A two dimensional array is the simplest of the multi-dimensional arrays. However, C programming allows arrays of more than two dimensions. The exact limit (of dimensions), if any, is determined by the compiler you use.

For e.g.int A[4][3]….[n];

Implementation of Two-dimensional Array in Memory

- While storing the elements of a 2-D array in memory, these are allocated contiguous memory locations. A two-dimensional array can be implemented in a programming language in two ways :

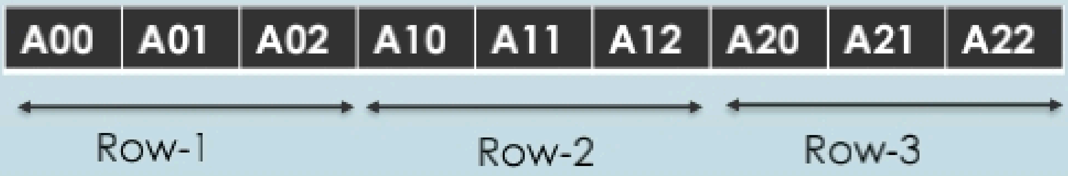
- Row-major implementation

- Column-major implementation

Row-major implementation

- Row-major implementation is a linearization technique in which elements of array are readed from the keyboard row-wise i.e. the complete first row is stored, then the complete second row is stored and so on. For example, an array **A [3] [3]** is stored in the memory as shown in Fig.(1) below :

$$\begin{vmatrix} A00 & A01 & A02 \\ A10 & A11 & A12 \\ A20 & A21 & A22 \end{vmatrix}$$

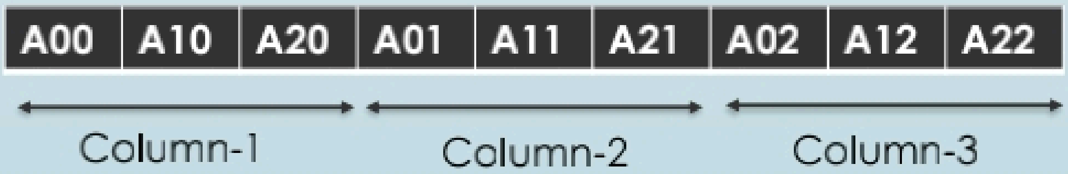


A00	A01	A02	A10	A11	A12	A20	A21	A22

Column-major implementation

- Column-major implementation is a linearization technique in which elements of array are reader from the keyboard column-wise i.e.the complete first column is stored, then the completesecond column is stored and so on. For example, an array **a[3][3]** is stored in the memory as shown in Fig.(2) below :

$$\begin{vmatrix} A00 & A01 & A02 \\ A10 & A11 & A12 \\ A20 & A21 & A22 \end{vmatrix}$$



A00	A10	A20	A01	A11	A21	A02	A12	A22

Sparse Matrix

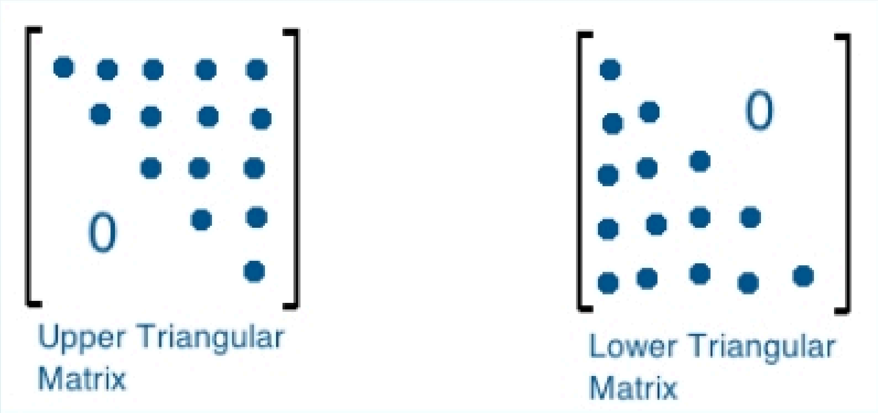
- If many of elements from a m x n matrix have a value 0 then the matrix is known as **sparse matrix**. A matrix that is not sparse is known as a **dense matrix**. There is no precise definition of when a matrix is sparse and when it is not, but it is a concept, which we can all recognize naturally. If the matrix is sparse, we must consider an alternative way of representing it rather than a normal row major or column major arrangement. This is because if majority of elements of the matrix are 0 then the alternative through which we can store only the non-zero elements and keep intact the functionality of the matrix can save a lot of memory space. Fig. (3) shows sparse matrix 5 * 5 with 5 non zero elements.

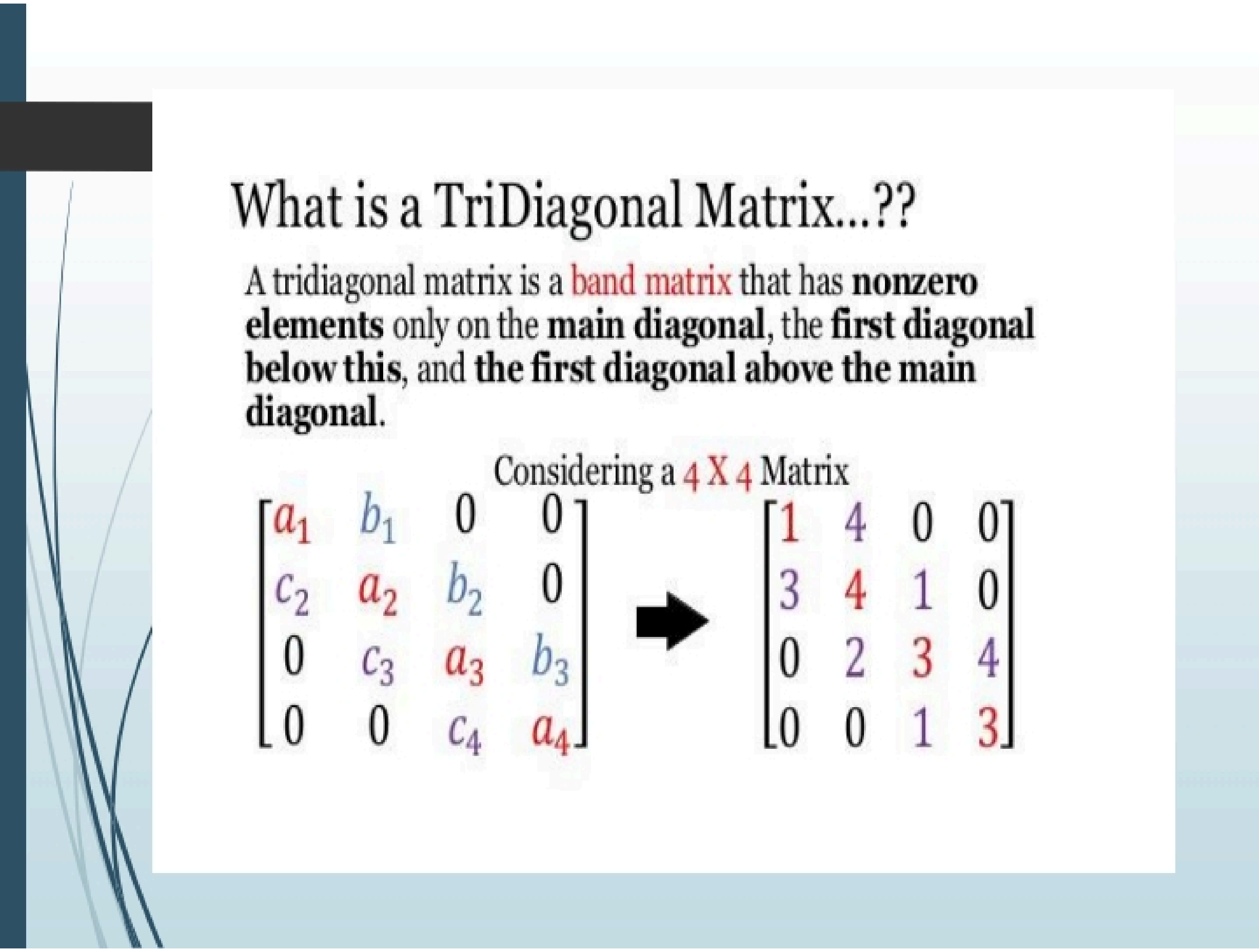
$$\begin{vmatrix} 0 & 2 & 0 & 6 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 9 \end{vmatrix}$$

Memory Allocation

- There are two types of memory allocations possible in C:
  - Static memory allocation (Compile-time allocation using arrays)
  - Dynamic memory allocation (Run-time allocation using pointers)

Triangular Matrix





What is a TriDiagonal Matrix...??

A tridiagonal matrix is a band matrix that has **nonzero elements** only on the **main diagonal**, the **first diagonal below this**, and **the first diagonal above the main diagonal**.

Considering a 4 X 4 Matrix

$$\begin{bmatrix} a_1 & b_1 & 0 & 0 \\ c_2 & a_2 & b_2 & 0 \\ 0 & c_3 & a_3 & b_3 \\ 0 & 0 & c_4 & a_4 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 4 & 0 & 0 \\ 3 & 4 & 1 & 0 \\ 0 & 2 & 3 & 4 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$