Unit -2

Structured Data Objects

1. Types of representation
2. Multi-Dimensional Array
3. Associative Arrays
4. Record

Types of Representation:

There are two ways:

1. Packed Storage Representation:

The representation in which components of a vectors are packed in to storage sequentially without regard for placing in each component at the beginning of an address word of storage. This method saves the physical memory but it causes the several disadvantages like:

- Access to a component is much more complex
- Implementation of a accessing formula for packed causes the overall cost

Types of Representation:

2. Unpacked Storage Representation:

Each component is stored beginning at the boundary of an addressable unit of storage.

Multi-Dimensional Array:

Multi-Dimensional Arrays:

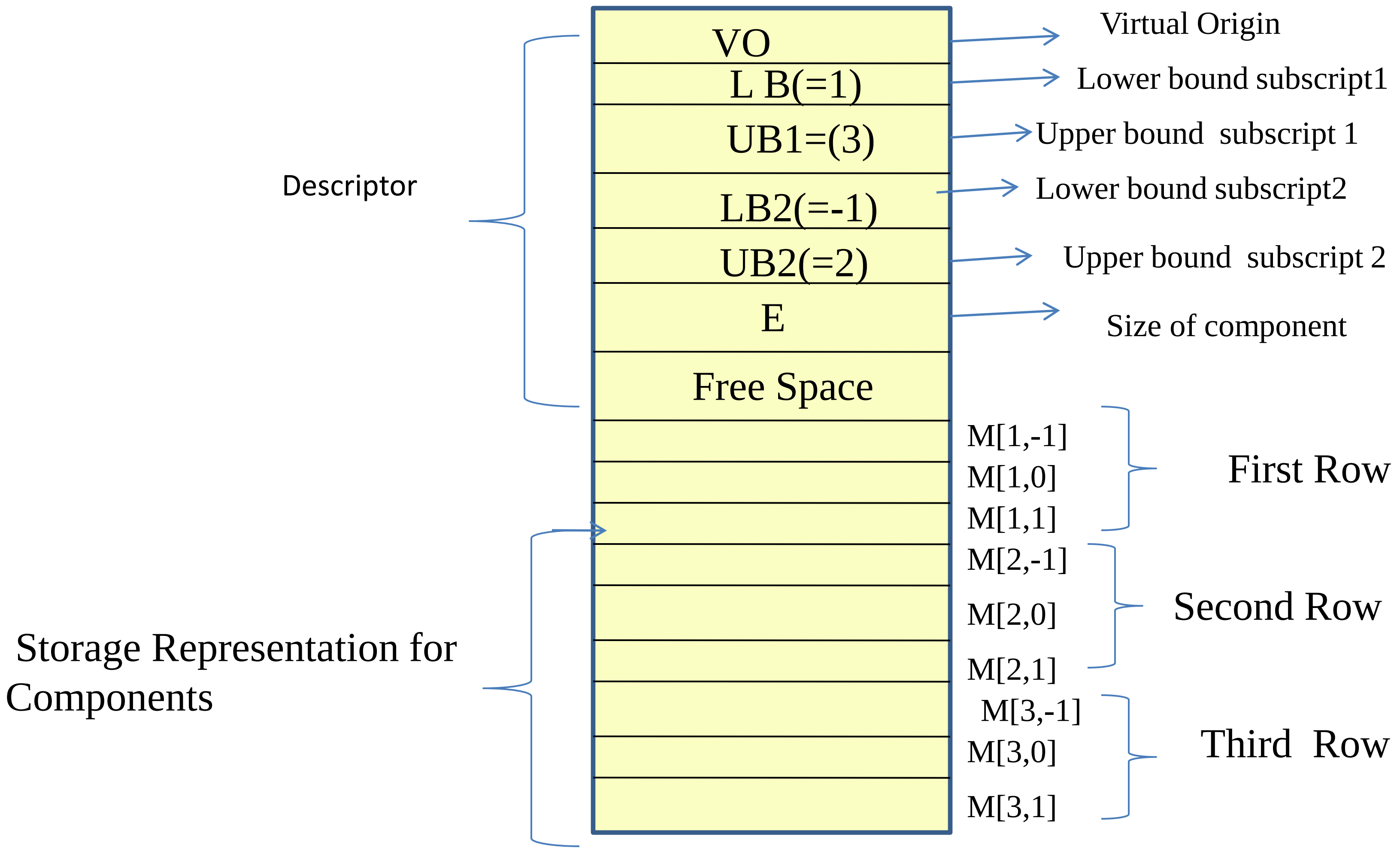
A vector is a 1-D array; a matrix is composed of rows & columns of components is a two dimensional array. A three dimensional array is composed of planes of rows and columns.

Specifications and syntax:

A multidimensional array differs from a vector in its attributes only in that a subscript range for each dimension is required as in the Pascal declaration

B: array [1…10,-5…5] of real;

M-D Array Representation:



Multi-Dimensional Array:

Location of element a[I,J] can be obtained by this :

l-value(A[I,J])= x + (I-LB1)* S + (J-LB2) * E

Where

x = base address

S = length of row = (UB2-LB2+1)*E

LB1= Lower bound on first subscript

LB2, UB2 =Lower and Upper bounds on second subscript or

l-value(A[I,J])= VO + I * S + J * E

Associative Arrays

An **associative array** (also **associative container**, **map**, **mapping**, **hash**, **dictionary**, **finite map**, and in query-processing an **index** or **index file**) is an **abstract data type** composed of a **collection** of unique keys and a collection of values, where each key is associated with one value (or set of values). The operation of finding the value associated with a key is called a *lookup* or indexing, and this is the most important operation supported by an associative array. The relationship between a key and its value is sometimes called a **mapping**.

Associative Arrays

The operations that are usually defined for an associative array are:

- **Add**: Bind a new key to a new value
- **Reassign**: Bind an old key to a new value
- **Remove**: Unbind a key from a value and remove the key from the key set
- **Lookup**: Find the value (if any) that is bound to a key

These entries can be thought of as two records in a database table:

Name	Telephone
XYZ	01-1234-56
ABC	02-4321-56

Example of Associative Arrays

One can think of a telephone book as an example of an associative array, where names are the keys and phone numbers are the values. Using the usual array-like notation, we might write

```
telephone['ada'] = '01-1234-56'
telephone['charles'] = '02-4321-56'
```

and so on. These entries can be thought of as two records in a database table

To retrieve the element from the associative array, we use a similar notation i.e.

```
x = telephone['ada']
y = telephone['charles']
```