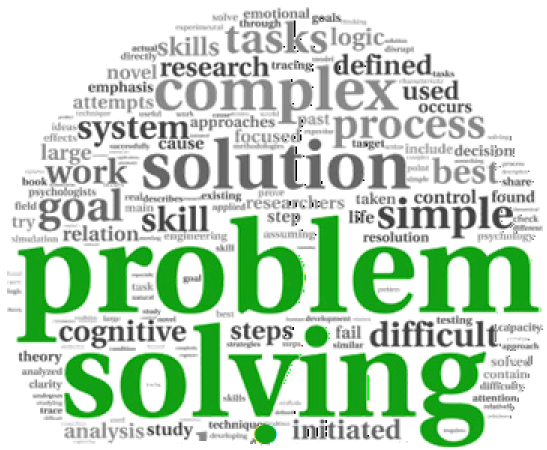
MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)



problem solving using computers

CSE 1051

2 D ARRAYS

S14_1

Objectives

To learn and appreciate the following concepts

- 2D Array declaration, initialization
- Simple Programs using 2D arrays

Session outcome

At the end of session student will be able to

→ Declare, initialize and access 2D array

→ Write simple programs using 2D array

2 Dimensional Array

- It is an ordered table of homogeneous elements.
- It can be imagined as a two dimensional table made of elements, all of them of a same uniform data type.
- It is generally referred to as matrix, of rows and columns.
- It is also called as a two-subscripted variable.

2 Dimensional Arrays

For example

```
int marks[5][3];
float matrix[3][3];
char page[25][80];
```

- ✓The first example tells that marks is a 2-D array of 5 rows and 3 columns.
- ✓The second example tells that matrix is a 2-D array of 3 rows and 3 columns.
- ✓Similarly, the third example tells that page is a 2-D array of 25 rows and 80 columns.

2 dimensional Arrays

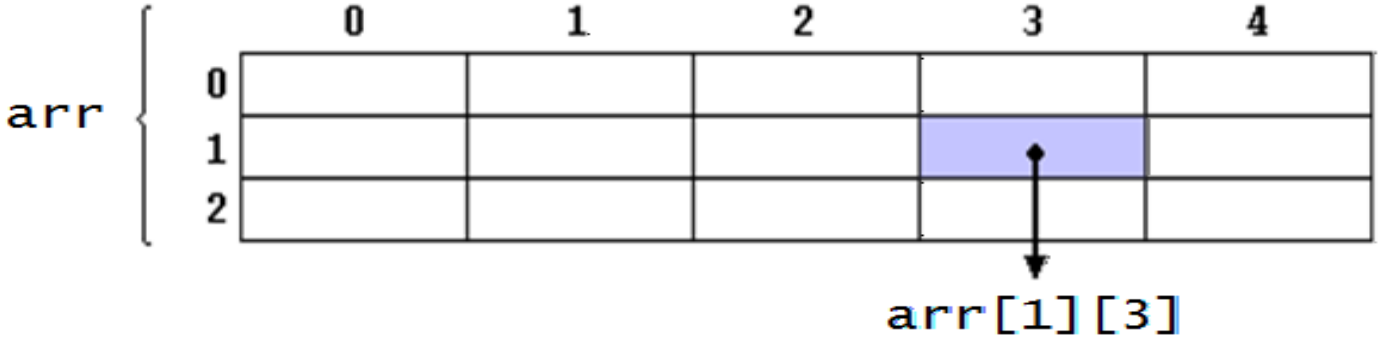
Declaration

type array_name[*row_size*][*column_size*];

For example,

int arr [3][5];

✓arr represents a two dimensional array or table having 3 rows and 5 columns and it can store 15 integer values.



2 Dimensional Arrays

Initialization of two dimensional arrays

type array-name [*row size*] [*col size*] ={list of values};

```
int table [2][3]={0,0,0,1,1,1};
```

initializes the elements of the first row to zero and the second row to 1.

Initialization is always done row by row

The above statement can be equivalently written as

```
int table [2][3]={{0,0,0},{1,1,1}};
```

OR in matrix form it can be written as

```
int table [2][3]=     {     {0,0,0},
                            {1,1,1}     };
```

2 Dimensional Arrays

When array is completely initialized with all values, need not necessarily specify the first dimension.

```
int table [][3]=    {       {0,0,0},
                            {1, 1, 1 }
                    };
```

If the values are missing in an initializer, they are set to zero

```
int table [2][3]=       {       {1,1},
                                {2}
                        };
```

will initialize the first two elements of the first row to 1, the first element of the second row to two, and all other elements to zero.

To set all elements to zero

```
int table [3][3]={{0},{0},{0}};
```

Go to posts/chat box for the link to the question

submit your solution in next 2 minutes

The session will resume in 3 minutes

Read a matrix and display it

```
int main()
{
int i, j, m, n, a[10][10];

printf("enter dimension for a:“);
scanf(“%d %d”, &m, &n);

printf("\n enter elements\n“);
for(i=0; i<m; i++)
{
    for(j=0; j<n; j++)
        scanf(“%d”, &a[i][j]);
}
for(i=0; i<m; i++)
{
   for(j=0; j<n; j++)
      printf(“%d\t”, a[i][j]);
   printf(“\n”);
}

return 0;
}
```

```
enter dimension for a: 2 3

 enter elements
 1 2 3 4 5 6
1        2        3
4        5        6
```

Addition of two Matrices

```
#include<stdlib.h>
int main( ){
int i, j, m, n, p, q, a[10][10],
b[10][10], c[10][10];
printf("enter dimension for a \n“);
scanf(“%d  %d”, &m, &n);
printf("enter dimension for b\n“);
scanf(“%d  %d”, &p, &q);
if (m!=p||n!=q)
{
printf("cannot add \n“);
 exit(0);          }
//Reading the elements
printf("enter elements for a \n“);
for (i=0; i<m; i++)
        for(j=0;j<n;j++)
            scanf(“%d”, &a[i][j]);
```

Matrix Addition

```
printf("\n enter elements for b\n”);
for(i=0;i<p;i++)
   for(j=0;j<q;j++)
      scanf(“%d”, &b[i][j]);
  //Addition
   for(i=0;i<m;i++)
     for(j=0;j<n;j++)
        c[i][j]=a[i][j]+b[i][j];
```

```
//Display
 printf("\n final matrix is \n“);
 for(i=0;i<m;i++) {
 for(j=0;j<n;j++)
    printf(“%d”,c[i][j]);
  printf(“\n”);
 }
return 0;
}
```

```
enter dimension for a
 2 3
enter dimension for b
 2 3
enter elements for a
 1 2 3 4 5 6

 enter elements for b
 1 2 3 4 5 6

 final matrix is
        2       4       6
        8       10      12
```

```
enter dimension for a
 2 3
enter dimension for b
 3 4
cannot add
```

Syntax Recap

Declaration

data-type array_name[*row_size*][*column_size*];

Initialization of two dimensional arrays:

type array-name [*row size*] [*col size*] ={list of values};

Reading a Matrix

```
int a[10][100];
for(i=0; i<m; i++)
{
   for(j=0; j<n; j++)
      scanf("%d", &a[i][j]);
}
```

Display a Matrix

```
int a[10][10];
for(i=0; i<m; i++)
{
   for(j=0; j<n; j++)
      printf("%d\t", a[i][j]);
 printf("\n");
}
```

Summary

- Declare, initialize and access 2D array
- Write simple programs using 2D array