Two Dimensional Array in C

The two-dimensional array can be defined as an array of arrays. The 2D array is organized as matrices which can be represented as the collection of rows and columns. However, 2D arrays are created to implement a relational database lookalike data structure. It provides ease of holding the bulk of data at once which can be passed to any number of functions wherever required.

Declaration of two dimensional Array in C

The syntax to declare the 2D array is given below.

```
data_type array_name[rows][columns];
```

Consider the following example.

```
int twodimen[4][3];
```

Here, 4 is the number of rows, and 3 is the number of columns.

Initialization of 2D Array in C

In the 1D array, we don't need to specify the size of the array if the declaration and initialization are being done simultaneously. However, this will not work with 2D arrays. We will have to define at least the second dimension of the array. The two-dimensional array can be declared and defined in the following way.

```
int arr[4][3]={{1,2,3},{2,3,4},{3,4,5},{4,5,6}};
```

Two-dimensional array example in C

```c
#include<stdio.h>
int main(){
int i=0,j=0;
int arr[4][3]={{1,2,3},{2,3,4},{3,4,5},{4,5,6}};
//traversing 2D array
for(i=0;i<4;i++){
 for(j=0;j<3;j++){
   printf("arr[%d] [%d] = %d \n",i,j,arr[i][j]);
 }//end of j
}//end of i
return 0;
}
```

Output

```
arr[0][0] = 1
arr[0][1] = 2
arr[0][2] = 3
arr[1][0] = 2
arr[1][1] = 3
arr[1][2] = 4
arr[2][0] = 3
arr[2][1] = 4
arr[2][2] = 5
arr[3][0] = 4
arr[3][1] = 5
arr[3][2] = 6
```

MATRIX MULTIPLICATION

```
        #include <stdio.h>
intmain()
{
int m, n, p, q, i, j, k;
int matrix1[10][10], matrix2[10][10], multiplication[10][10];

printf(“enter number of rows and columns of matrix1\n”);
scanf(“%d%d”, &m, &n);

printf(“enter number of rows and columns of matrix2\n”);
scanf(“%d%d”, &p, &q);

if (n != p)
  {
printf(“since number of columns of first matrix\n”);
printf(“is not equal to number of rows of second matrix,\n”);
printf(“hence both matrices are not multiplication compatible,\n”);
printf(“and we cannot multiply these matrices.\n”);
return0;
  }
printf(“enter %d elements of matrix1 of %dx%d\n”,m*n,m,n);

for (i =0; i< m; i++)
for (j =0; j < n; j++)
scanf(“%d”, &matrix1[i][j]);

printf(“enter %d elements of matrix2 of %dx%d\n”,p*q,p,q);

for (i =0; i< p; i++)
for (j =0; j < q; j++)
scanf(“%d”, &matrix2[i][j]);
for (i =0; i< m; i++)
for (j =0; j < q; j++)
          {
      multiplication[i][j]=0;
                  for (k =0; k < p; k++)
        multiplication[i][j] = multiplication[i][j] + matrix1[i][k]*matrix2[k][j];
     }
printf(“product / multiplication of the two matrices is as follows:\n”);
for (i =0; i< m; i++)
            {
for (j =0; j < q; j++)
printf(“%d\t”, multiplication[i][j]);
printf(“\n”);
     }
return0;
}
```

OUTPUT:

```
 Enter number of rows and columns of matrix1
2
2
Enter number of rows and columns of matrix2
2
2
Enter 4 elements of matrix1 of 2×2
-1 4 2 3
Enter 4 elements of matrix2 of 2×2
9 -3 6 1
Product / Multiplication of the two matrices is as follows:
15      7
36      -3
```