CL-1002 Programming Fundamentals	LAB 07 Two Dimensional Arrays and Nested Loops IN C

Learning Objectives

- Nested Loops
- Two Dimensional Arrays

NESTED LOOPS

C supports the nesting of loops in C. Nesting of loops is the feature in C that allows the looping of statements inside another loop. Let's observe an example of nesting loops in C.

Any number of loops can be defined inside another loop, i.e., there is no restriction for defining any number of loops. The nesting level can be defined at n times. You can define any loop inside another loop; for example, you can define a 'while' loop inside a 'for' loop.

Syntax:

```
Outer_loop
{
  Inner_loop
  {
     // inner loop statements.
  }
    // outer loop statements.
}
```

Note: Outer_loop and Inner_loop are valid loops that can be a 'for' loop, 'while' loop or 'do-while' loop.

Example 1: for loop

```c
#include <stdio.h>
int main()
{
   int n=4;
   int i,j;
   for(i=1;i<=n;i++)  // outer loop
   {
       printf("%d\n",i);
       for(j=1;j<=3;j++)  // inner loop
       {
           printf("\t%d\n",j); // printing the value.
       }
   }
}
```

Example 2: while loop

```c
#include <stdio.h>
int main()
{
   int k=1; // variable initialization
   char c='a';
   int i=1;
   while(i<=3) // outer loop
   {
      printf("%d\n",i);
      int j=1;
      while(j<=3)  // inner loop
       {
           printf("\t%c",c);  // printing the value of c.
           j++;
       }
       i++;
       printf("\n\n");
   }
}
```

Example 3: Do While Loop

```c
#include <stdio.h>
int main()
{    int i=1;
    do            // outer loop
    {
        printf("%d\n",i);
        int j=1;
        do        // inner loop
        {
            printf("\t%d",j);
            j++;
        }   while(j<=3);
      printf("\n");
      i++;
    }while(i<=3);
}
```

Simple Example: Suppose we have an array A of m rows and n columns. We can store the user input data into our array A and can Print Array A in the following way:

```c
#include<stdio.h>
int main()
{
    int rows=3;int cols=3;
    int A[rows][cols];
    int i,j;
    //Suppose we have an array A of m rows and n columns.
    //We can store the user input data into our array A in the following way:
    printf("Give Input Array:");
    for(i =0; i<rows;i++){
        for(j=0;j<cols;j++){
            scanf("%d", &A[i][j]);
        }
    }
    // printing 2D input Array
    printf("\n");
    for(i =0; i<rows;i++){
        printf("Row %d: ",i);
        for(j=0;j<cols;j++){
            printf(" %d\t", A[i][j]);
        }
        printf("\n");
    }
}
```

OUTPUT:

```
Give Input Array:
1 2 3
4 5 6
7 8 9

Row 0:  1       2       3
Row 1:  4       5       6
Row 2:  7       8       9
```

TWO-DIMENSIONAL ARRAYS

A two-dimensional array is a collection of a fixed number of components arranged in rows and columns (that is, in two dimensions), wherein all components are of the same type.
Two-dimensional arrays are used to represent tables of data, matrices, and other two-dimensional objects.

SYNTAX:

element-type aname [$size_1$] [$size_2$]; /* uninitialized */

INTERPRETETION

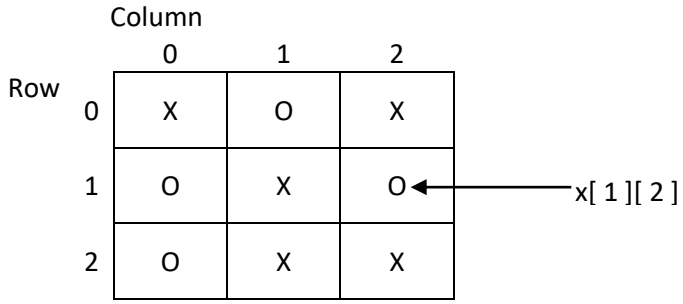
- Allocates storage for a two-dimensional array (aname) with size1 rows and size2 columns.
- This array has size1*size2 elements, each of which must be referenced by specifying a row subscript (0 , 1 ,… size1-1) and a column subscript (0 , 1 ,…size2-1).
- Each array element contains a character value.

MEMORY REPRESENTATION

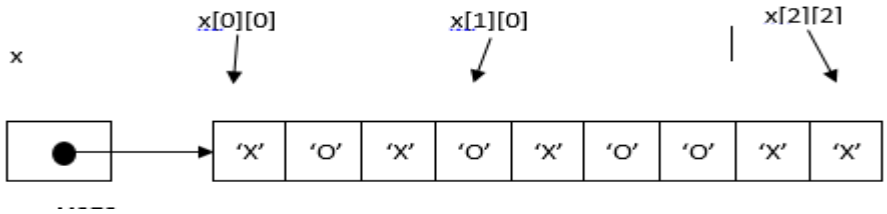
char x[3][3] = {{'X', '0', 'X'}, {'0', 'X', '0'}, {'0', 'X', 'X'}};

Array x

Row \ Column	0	1	2
0	x[0][0]	x[0][1]	x[0][2]
1	x[1][0]	x[1][1]	x[1][2]
2	x[2][0]	x[2][1]	x[2][2]



Because memory is addressed linearly, a better representation is like:



USES

- Storing a table of data (not the only way).
- Any kind of matrix processing, as a 2D array really is a matrix.

Example 01:

```c
#include<stdio.h>
int main(){
   /* 2D array declaration*/
   int array[3][3];
   /*Counter variables for the loop*/
   int i, j;
   for(i=0; i<3; i++) {
      for(j=0;j<3;j++) {
         printf("Enter value for array[%d][%d]:", i, j);
         scanf("%d", &array[i][j]);
      }
   }
   //Displaying array elements
   printf("Two Dimensional array elements:\n");
   for(i=0; i<3; i++) {
      for(j=0;j<3;j++) {
         printf("%d ", array[i][j]);
         if(j==2){
            printf("\n");
         }
      }
   }
   return 0;
}
```

Output:

```
Enter value for array[0][0]:1
Enter value for array[0][1]:0
Enter value for array[0][2]:0
Enter value for array[1][0]:0
Enter value for array[1][1]:1
Enter value for array[1][2]:0
Enter value for array[2][0]:0
Enter value for array[2][1]:0
Enter value for array[2][2]:1
Two Dimensional array elements:
1 0 0
0 1 0
0 0 1
```

MULTIDIMENSIONAL ARRAYS

Multidimensional array is a collection of a fixed number of elements (called components) arranged in n dimensions (n>=1).

SYNTAX:

element-type aname [size 1] [size 2] … [size n]; /* storage allocation */

INTERPRETATION:

- Allocates storage space for an array aname consisting of size 1 × size 2 × … × size n memory cells.
- Each memory cell can store one data item whose data type is specified by element-type . The individual array elements are referenced by the subscripted variables aname [0][0] … [0] through aname [size 1 −1][size 2 −1] …[size n −1] .
- An integer constant expression is used to specify each size i .

USES

With input data on temperatures referenced by day, city, county, and state, day would be the first dimension, city would be the second dimension, county would be the third dimension, and state would be the fourth dimension of the array. In any case, any temperature could be found as long as the day, the city, the county, and the state are known. A multidimensional array allows the programmer to use one array for all the data.

Example:

```c
#include <stdio.h>

int main(void)
{
    // initializing the 3-dimensional array
    int x[2][3][2] = { { { 0, 1 }, { 2, 3 }, { 4, 5 } },
                       { { 6, 7 }, { 8, 9 }, { 10, 11 } } };

    // output each element's value
    for (int i = 0; i < 2; ++i) {
        for (int j = 0; j < 3; ++j) {
            for (int k = 0; k < 2; ++k) {
                printf("Element at x[%i][%i][%i] = %d\n", i, j, k, x[i][j][k]);
            }
        }
    }
    return (0);
}
```

Output:

```
Element at x[0][0][0] = 0
Element at x[0][0][1] = 1
Element at x[0][1][0] = 2
Element at x[0][1][1] = 3
Element at x[0][2][0] = 4
Element at x[0][2][1] = 5
Element at x[1][0][0] = 6
Element at x[1][0][1] = 7
Element at x[1][1][0] = 8
Element at x[1][1][1] = 9
Element at x[1][2][0] = 10
Element at x[1][2][1] = 11
```

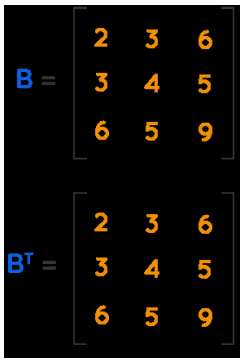
Lab Tasks

1. Take Two numbers from user and divide both numbers but do not use the division operator.

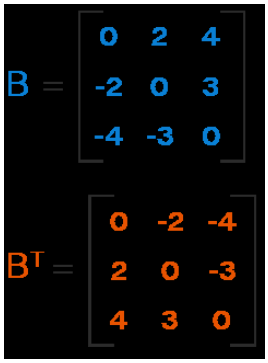
2. Given an array of integers of size N and an integer "d", the task is to rotate the array elements to the **left** by **d** positions. Note: Solve the question without declaring another array. (The input array itself must be modified).
   Input: {1, 2, 3, 4, 5, 6, 7}, d = 2
   Output: {3, 4, 5, 6, 7, 1, 2}

3. Take a number from user and count the sum of their digits.
   Input: 206
   Output: 8
   Input: 4569
   Output: 24

4. Take a 2D array(Matrix) as input from user and check that array is symmetric or not. If array is symmetric then print **"Array is Symmetric"** with array if not, then print **"array is not symmetric"** with given array.
   Note: A symmetric matrix is a matrix that is equal to its transpose. Where elements in first row are equal to elements in first column and so on.
   Formal Definition: Aij = Aji.



Symmetric



Non-Symmetric

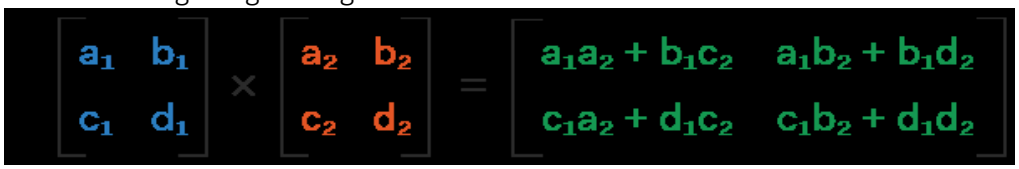
5. Print the following pattern on C console:

Note: Perform the above task using loops

6. Take an array with N elements as input, and then output the frequency of each element present in the array. Example:

Input: {2, 4, 2, 3, 5, 5, 4, 4}
Output:
Frequency of 2 = 2
Frequency of 4 = 3
Frequency of 3 = 1
Frequency of 5 = 2

7. Take 2 2x2 matrices as input in a 2D array. Then, multiply the 2 matrices and print the final output matrix. Use the following image as a guide:



8. Take an array with N elements as input, and sort the array into ascending order and then print the sorted version.

Input: {6, 3, 2, 7, 1, 5}
Output: {1, 2, 3, 5, 6, 7}

9. Junaid wants to keep track of all mobile phone bills in his X company branches. Let Y be the number of company mobile phones in each branch. Create a 2D array for bill amount, where keep track of branch ID in row subscript, mobile phone IDs in column subscript. Ask users to enter a bill for all mobile phones in all branches. Your program should print the following:

- Total bill for all branches
- Total bill for each branch
- Branch ID where maximum bill arrived
- Branch and Mobile Phone IDs where bill is highest of all mobile phones.

10. Given an unsorted array A of size N that contains only non-negative integers, find a continuous sub-array which adds to a given number S. In case of multiple subarrays, return the subarray which comes first on moving from left to right.

Input:
Enter Number S = 12
Array: {1, 2, 3, 7, 5}
Output:
The Elements from Index 1 to 3 when summed results in the output of 12.