INTRODUCTION TO PROGRAMMING

Department of Electronics and Communication
Indian Institute of Information Technology, Allahabad

A C PROGRAM TO FIND OUT WHETHER THE MATRIX IS UPPER TRIANGULAR AND LOWER TRIANGULAR.

TEAM MEMBERS :

1. IEC2021042 – SHUBHASHIS MALICK
2. IEC2021043 – RITEE SHARMA
3. IEC2021044 – RITAM DAS
4. IEC2021045 – ARYAN ANAND

PROFESSOR-
DR. MOHAMMED JAVED SIR

CONTENT :-

- Problem Statement
- Introduction and Logic of the program
- Algorithm
- Procedure
- Time Complexity
- Conclusion
- References
- Program

Problem Statement

After taking an input of any n **x** n matrix, write a C program to find out whether the matrix is upper triangular or lower triangular.

Introduction and Logic of the Program

- In an Upper Triangular Matrix:

$a_{ij} = 0$ for $i>j$

- In a Lower Triangular Matrix

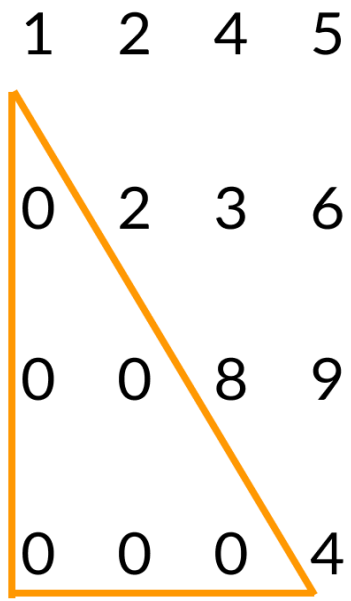
$a_{ij} = 0$ for $j>i$

- Also the diagonal elements (a_{11}, a_{22}, a_{33},...) can not be equal to zero in case of a Triangular Matrix.

Where a_{ij} represents the element i^{th} row and j^{th} column of the matrix.

Upper and Lower Triangular Matrix representation

Upper Triangular Matrix



Lower Triangular Matrix

$$A = \begin{bmatrix} a_{00} & 0 & 0 & 0 & 0 \\ a_{10} & a_{11} & 0 & 0 & 0 \\ a_{20} & a_{21} & a_{22} & 0 & 0 \\ a_{30} & a_{31} & a_{32} & a_{33} & 0 \\ a_{40} & a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

Algorithm:

- Take input from the user for the size of double dimension array (n).
- Define an integer array arr[n][n] and take input from console for the elements of the array arr[n][n].
- Calculate the minimum number of zeros that should be present in any triangular matrix of order n by the formula n(n-1)/2 and store it in any variable(number).
- Define a switch case with 2 cases for checking the upper and lower triangularity of the matrix.
- For checking the upper triangularity, set conditions to check arr[i][j] ≠ 0, for all i=j and arr[i][j]=0 for all i>j and set a flag variable (f=1) and counter (c++) variable respectively.
- For checking the lower triangularity, set conditions to check arr[i][j] ≠ 0, for all i=j and arr[i][j]=0 for all i<j and sets a flag variable (f=1) and counter (c++) variable respectively.
- If the value of counter variable (c)= number and f ≠ 1 then the given matrix is triangular else print the matrix is not triangular.

Procedure of the Code

- When the compilation begins at first, the compiler asks for the order of the 2D Array from the user.
- Enter size of the array as 3 (from console)
- Thus the array, arr[3][3] is now declared i.e. n=3
- Now the compiler asks from the console to enter the elements in array and the elements are entered row wise.

$$\begin{vmatrix} 1 & 2 & 3 \\ 0 & 4 & 8 \\ 0 & 0 & 6 \end{vmatrix}$$

- Now p is a dummy variable which stores n-1 i.e. 3-1 =2. The variable p has been introduced to avoid the error caused by precedence of operator.
- Number = n**x**p/2 = 3**x**2/2 = 3 (minimum number of zeros in a triangular matrix)
- Further compiler will ask console to input 1 or 2 for checking upper or lower triangularity of matrix respectively using the message - “Enter 1 for checking upper Triangularity of the matrix or enter 2 for checking the lower triangularity”.
- Suppose 1 is input by the console. Now the compiler will execute case 1 of switch case.
- For the value i=0, j will take values upto j=0, j=1 and j=2 and simultaneously arr[0][0], arr [0][1], arr[0][2] will be checked for value 0 for i=j or i>j accordingly and thus value of “c” or “f” will be updated for each iteration of the inner loop.
- Now if f=1, then the compiler will print “not a triangular matrix” and the compiler will come out of the outer loop else i will be incremented to be 1 and gradually to 2 and thus the other positions like arr[1][0], arr[1][1], arr[1][2], arr[2][0], arr[2][1], arr[2][2] will be checked similarly and value of “c” and “f” will be updated.

Finally it will check that value of c = number and f ≠ 1, if found true, it will print “it is a triangular matrix” else it will print “not a triangular matrix”.

Time Complexity

There can be 6 different cases to our program. The cases are elaborated in a much better way in our IEEE document. But let's say the time to execute case 1 is t_1, to execute case 2 is t_2 and so on.

Time complexity= (t1+t2+t3+t4+t5+t6) x 0.1667

The Time complexity comes out to be 0.0056395 seconds.

Conclusion

- Our program helps to determine whether the matrix inputted by the user is an upper or lower triangular matrix. Moreover, it is also capable of identifying if the matrix is not a triangular matrix.

- It is a very user friendly and menu driven program a gives freedom to the user to check for upper and lower triangularity.

References :-

1. https://www.codechef.com/ide- For checking the run time of program
2. https://ncert.nic.in/textbook.php NCERT Class 12- For basic knowledge of matrices and determinants.
3. Let us C (Yashavant Kanetkar) - For basic C language theory.

Program :

```c
#include<stdio.h>
int main()
{
    int i,j,option;
    int p;
    int c=0;
    int f = 0;
    int k =0;
    int number; // a variable to count minimum number of zeros in any triangular matrix
    printf("enter the size of the array: ");
    int n;
    scanf("%d",&n);
    int arr[n][n];
    printf("enter the elements in the array \n ");
    for(i=0;i<n;i++)
    {   for(j=0;j<n;j++)
        {
          scanf("%d",&arr[i][j]);
        }
    } p = n-1; // using formula
    number = n*p/2;  // we all know that the minimum number of zero in any triangular matrix of order n is n(n-1)/2

printf("enter 1 for checking upper triangularity of the matrix ");
printf(" \n enter 2 for checking lower triangularity of the matrix ");
scanf("%d",&option);
switch(option)
{
    case 1 : for(i=0;i<n;i++)
             {
                for (j=0;j<n;j++)
                {   if(i==j){
                    if(arr[i][j]==0)
                        f=1;

                }   else if (i>j){
                    if(arr[i][j]==0)
                        c++;
                } else{
                  if(arr[i][j]==0)
                      k++;
                }
                if (f==1){ // to check whether there is a non zero element in main diagonal of the given matrix
                printf("not a triangular matrix");
                 break;
                 }

            } if (c>=number && k!=c){  // check that the counter variable is atleast greater than the minimum number of zeros
              printf("this is a upper triangular matrix");
            } else {
                printf("\n this is not a upper triangular matrix");
            }

        break;
    case 2 : for(i=0;i<n;i++)
             {
                for (j=0;j<n;j++)
                {   if(i==j){
                    if(arr[i][j]==0)
                        f=1;

                }   else if (i<j){
                    if(arr[i][j]==0)
                        c++;
                } else {
                  if(arr[i][j]==0)
                      k++;

                }
                }
                if (f==1){
                printf("not a triangular matrix");
                 break;
                }

            }if (c>=number && k!=c){
              printf("this is a lower triangular matrix");
            } else {
                printf(" \n this is not a lower triangular matrix");
            }
        break;
    default : printf("incorrect option choosen");
}
return 0;
}
```