Computing the Sum of Diagonal Elements of a Matrix

Aman Raj
Department of Electronics and Communication
Indian Institute of Information Technology, Allahabad
Dhanbad, Jharkhand
iec2021050@iiita.ac.in

Nimish Rajurkar
Department of Electronics and Communication
Indian Institute of Information Technology, Allahabad
Nanded, Maharashtra
iec2021051@iiita.ac.in

Vamsi K
Department of Electronics and Communication
Indian Institute of Information Technology, Allahabad
Hyderabad, Telangana
iec2021052@iiita.ac.in

Sai Praneeth
Department of Electronics and Communication
Indian Institute of Information Technology, Allahabad
Hyderabad, Telangana
iec2021053@iiita.ac.in

***Abstract*—Given a square matrix of size n, write a C program to find the sum of major and minor diagonal elements.**

I. Introduction

We are given the size of a square matrix and elements of the same, using this we have to compute the sum of the major and minor diagonals of the matrix using the C programming language.

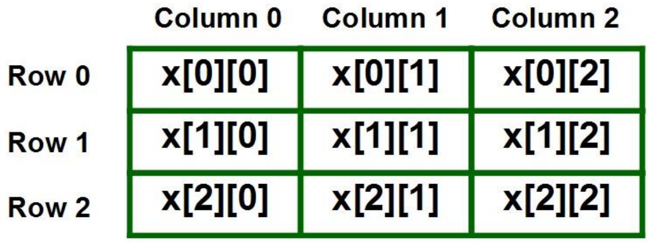
We will be using the 2-Dimensional array to store the elements of the given matrix and nested for loops to traverse through the array.

II. 2-Dimensional Arrays

An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. One of the three types of arrays is a multidimensional array. Two – dimensional array is the simplest form of a multidimensional array. We can see a two – dimensional array as an array of one – dimensional array for easier understanding. The basic form of declaring a two-dimensional array of size x, y is by the syntax:

data_type array_name[x][y];
data_type: Type of data to be stored. Valid C data type.

Elements in two-dimensional arrays are commonly referred to by x[i][j] where i is the row number and ‘j’ is the column number. A two – dimensional array can be seen as a table with ‘x’ rows and ‘y’ columns where the row number ranges from 0 to (x-1) and column number ranges from 0 to (y-1). A two – dimensional array ‘x’ with 3 rows and 3 columns is as shown in Fig.1



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

Fig.1: A sample 2-D array with 3 rows and 3 columns.

III. For Loop

A for loop is a repetition control structure which allows us to write a loop that is executed a specific number of times. The loop enables us to perform n number of steps together in one line. The basic form of declaring a for loop is by the syntax:

```
for (initialization; condition; increment;)
{
    // body of the loop
    // statements we want to execute
}
```

Nested loop means a **loop statement** inside another loop statement. That is why nested loops are also called as l**oop inside loop**. The basic form of declaring a nested for loop is by the syntax:

```
for (initialization; condition; increment) {
   for (initialization; condition; increment) {
      // statement of inside loop
   }
   // statement of outer loop
}
```

IV. Process

In this C Program to find Sum of Diagonal Elements of a Matrix, We declared single Two dimensional array using the following syntax:

```
     int n;
     printf("Enter the size of the Matrix: ");
     scanf("%d", &n);
     int matrix[n][n];
```

The above syntax ask the user to enter the Matrix size (Number of rows and columns. For instance n=3 will be a square matrix with 3 rows and 3 columns).

Next, we used C Programming for loop to iterate every cell present in the matrix using the following syntax:

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

The scanf statement inside the for loop will store the user entered values in every individual array element such as a[0][0], a[0][1], …
Now we initialize 2 variables of data type int, “sum_of_major_diagonal_elements = 0” and “sum_of_minor_diagonal_elements = 0”

Now using a for loop, we add the values of major diagonal elements to the variable sum_of_major_diagonal_elements using the following syntax:

```
    int sum_of_major_diagonal_elements = 0;
    for(int i = 0; i < n; i++){
        sum_of_major_diagonal_elements += matrix[i][i];
    }
```

Similarly, using another for loop, we add the values of minor diagonal elements to the variable sum_of_minor_diagonal_elements using the following syntax:

```
int sum_of_minor_diagonal_elements = 0;
for(int i = 0; i < n; i++){
   sum_of_minor_diagonal_elements += matrix[i][n-1-i];
   }
```

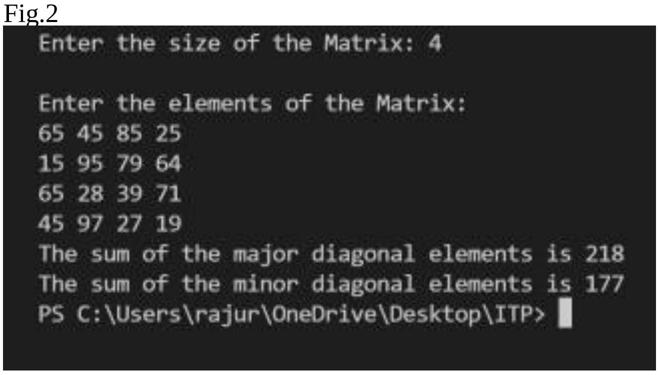
The final values of the 2 variables will be printed using the printf function by the means of following syntax:

```
printf("The sum of the major diagonal elements is %d\n", sum_of_major_diagonal_elements);
```

```
printf("The sum of the minor diagonal elements is %d", sum_of_minor_diagonal_elements);
```

V. Conclusion

The output will be displayed in the terminal as shown in Fig.2



```
Enter the size of the Matrix: 4

Enter the elements of the Matrix:
65 45 85 25
15 95 79 64
65 28 39 71
45 97 27 19
The sum of the major diagonal elements is 218
The sum of the minor diagonal elements is 177
PS C:\Users\rajur\OneDrive\Desktop\ITP>
```

Fig.2: The output of a sample matrix

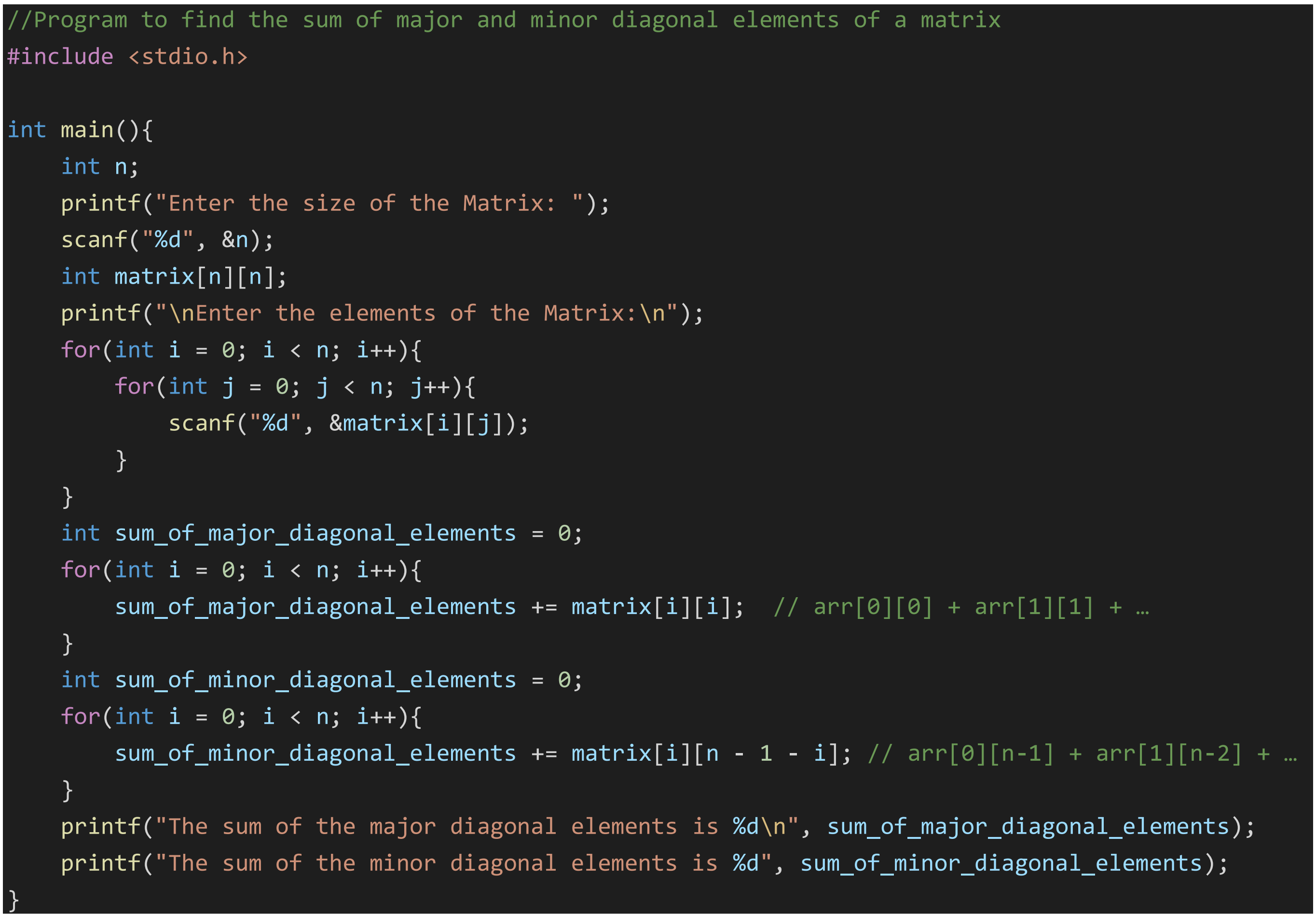
VI. References

[1] https://www.geeksforgeeks.org/multidimensional-arrays-c-cpp/

[2] https://www.geeksforgeeks.org/loops-in-c-and-cpp/?ref=gcse

[3] https://www.geeksforgeeks.org/nested-loops-in-c-with-examples/

[4] https://www.geeksforgeeks.org/array-data-structure/



```c
//Program to find the sum of major and minor diagonal elements of a matrix
#include <stdio.h>

int main(){
    int n;
    printf("Enter the size of the Matrix: ");
    scanf("%d", &n);
    int matrix[n][n];
    printf("\nEnter the elements of the Matrix:\n");
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            scanf("%d", &matrix[i][j]);
        }
    }
    int sum_of_major_diagonal_elements = 0;
    for(int i = 0; i < n; i++){
        sum_of_major_diagonal_elements += matrix[i][i];  // arr[0][0] + arr[1][1] + …
    }
    int sum_of_minor_diagonal_elements = 0;
    for(int i = 0; i < n; i++){
        sum_of_minor_diagonal_elements += matrix[i][n - 1 - i]; // arr[0][n-1] + arr[1][n-2] + …
    }
    printf("The sum of the major diagonal elements is %d\n", sum_of_major_diagonal_elements);
    printf("The sum of the minor diagonal elements is %d", sum_of_minor_diagonal_elements);
}
```