

Lab Report#10



Course Code:

CSC141

Submitted By:

Haider Ali

Registration No:

FA21-BEE-053

Submitted To:

Dr. Muhammad Kaleem

09/06/2022

Lab # 10 Working with 2D Array in C

In-Lab Task 2: Implementing the algorithm for the Magic Square

In this task you have to make a magic square and display it on the screen. You are given a Starter Code (Annex I), that does the following:

- Asks the user to enter the order '**n**' of the magic square (odd numbers only).
- Declares a 2D array of size **n x n** and initializes it with zeros.
- Prints the magic square on the screen.

Your job is to complete this code by implement the algorithm discussed in the **Reading Task 2**.

References

1. https://en.wikipedia.org/wiki/Magic_square#A_method_for_constructing_a_magic_square_of_odd_order

Practice for Magic Square Algorithm:

You can fill in the following 5 x 5 grid to see if you have grasped the working of the algorithm. You can test your work by checking the sum of rows, columns and the main diagonal. They should all be the same.

ANNEX I: Code Listings for In-Lab Task 2

```
#include <stdio.h>
#include <stdlib.h>

int magic_square(int n);

int main()
{
    int n;
    printf("Enter the order of Magic Square (positive Odd Nums only): \n");
    scanf("%d", &n);
    printf("\n\n");
    if(magic_square(n) != 0)
        printf("\nPlease enter correct value for n!!");
    return 0;
}
```

Code:

```
#include <stdio.h> // preprocessor directives

int main() // Starting of the main body of function

{

    int a[3][3],i,j,sumd1=0,sumd2=0,f=0,sumr=0,sumc=0; //Deceleration and initialization

    printf("Enter Matrix Elements...\n"); // Giving Message to the User

    for(i=0;i<3;i++)

    {

        for(j=0;j<3;j++)

        {

            scanf("%d",&a[i][j]); // Taking Input from the user

        }

    }

    printf("The Matrix is...\n"); // Giving Message to the User

    for(i=0;i<3;i++)

    {

        for(j=0;j<3;j++)

        {

            printf("%3d",a[i][j]);

        }

        printf("\n");

    }

    for(i=0;i<3;i++)

    {

        for(j=0;j<3;j++)

        {
```

```
    if(i==j)

    {

        sumd1=sumd1+a[i][j];

    }

    if(i+j==3-1)

    {

        sumd2=sumd2+a[i][j];

    }

}

}

if(sumd1!=sumd2)

{

    f=1;

}

else

{

    for(i=0;i<3;i++)

    {

        sumc=0;

        sumr=0;

        for(j=0;j<3;j++)

        {

            sumc=sumc+a[j][i];

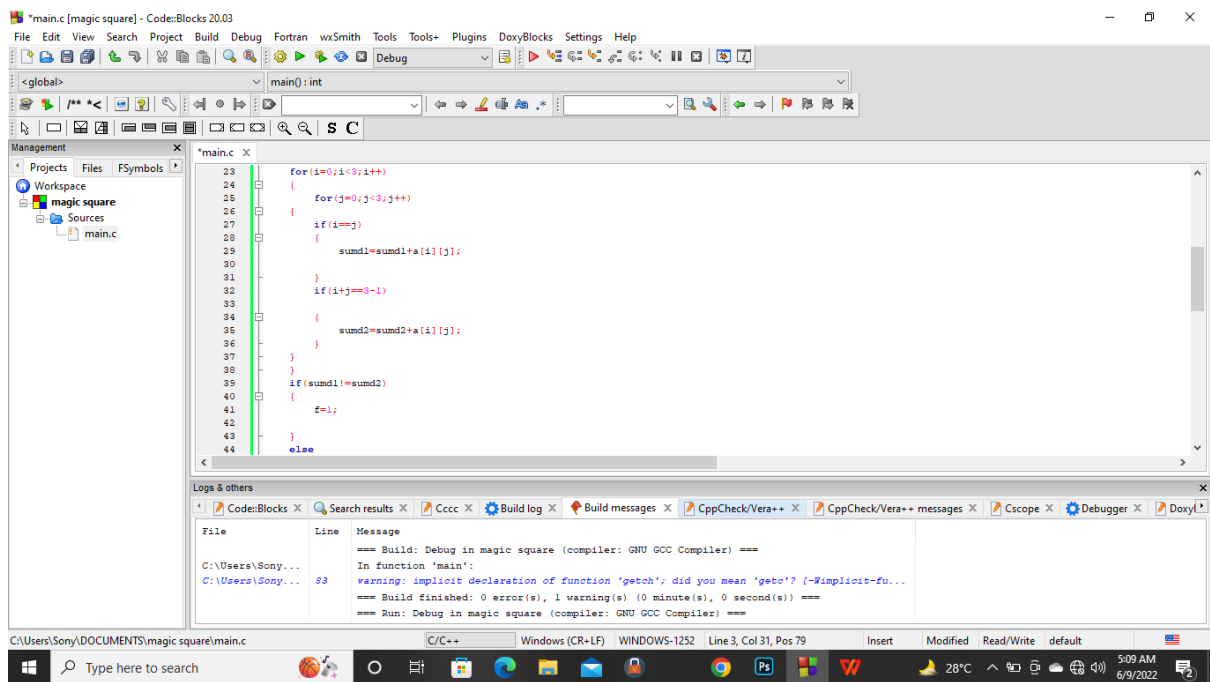
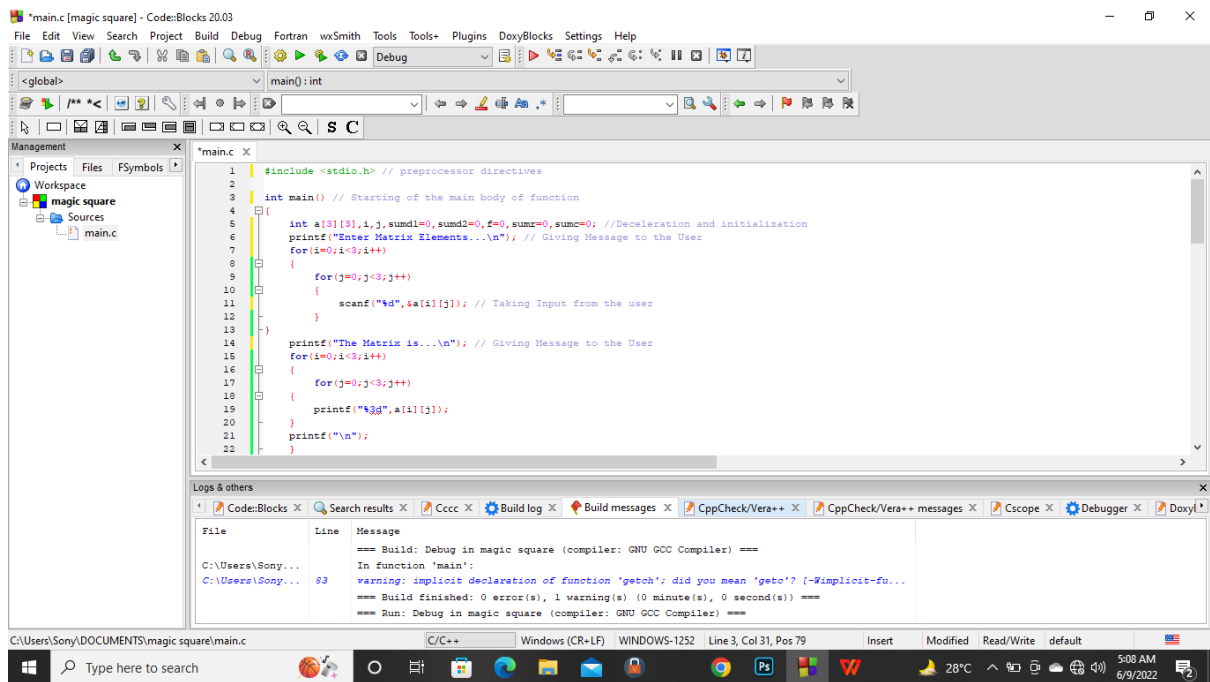
            sumr=sumr+a[i][j];

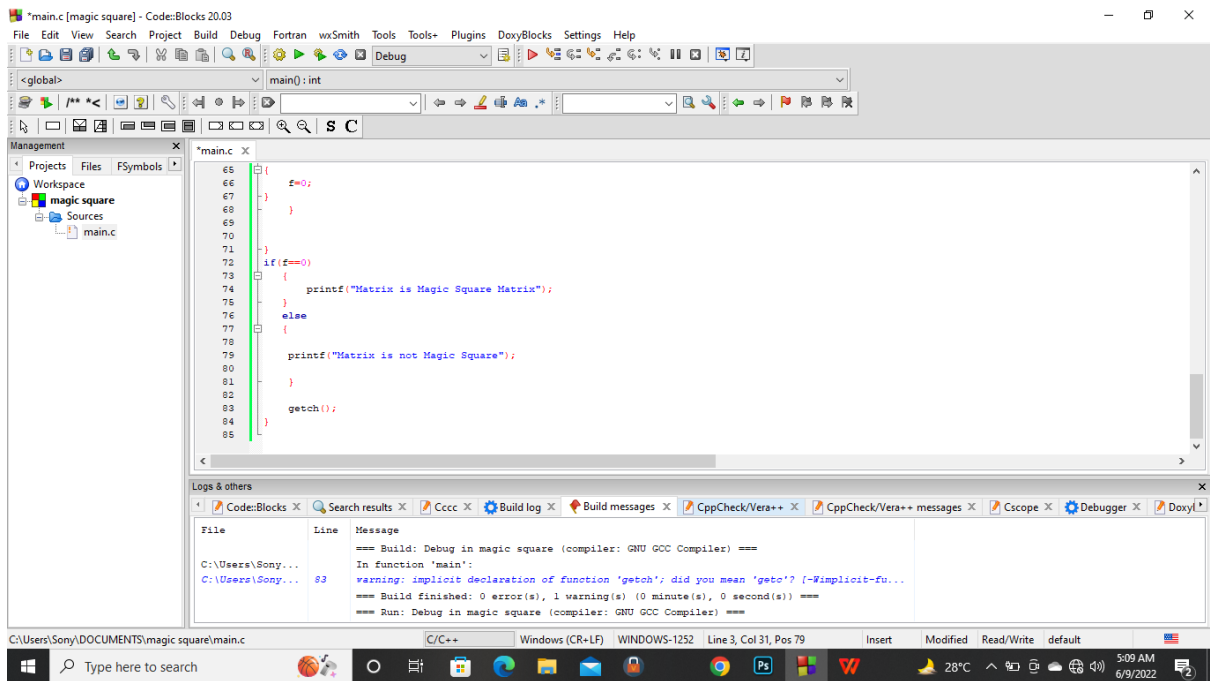
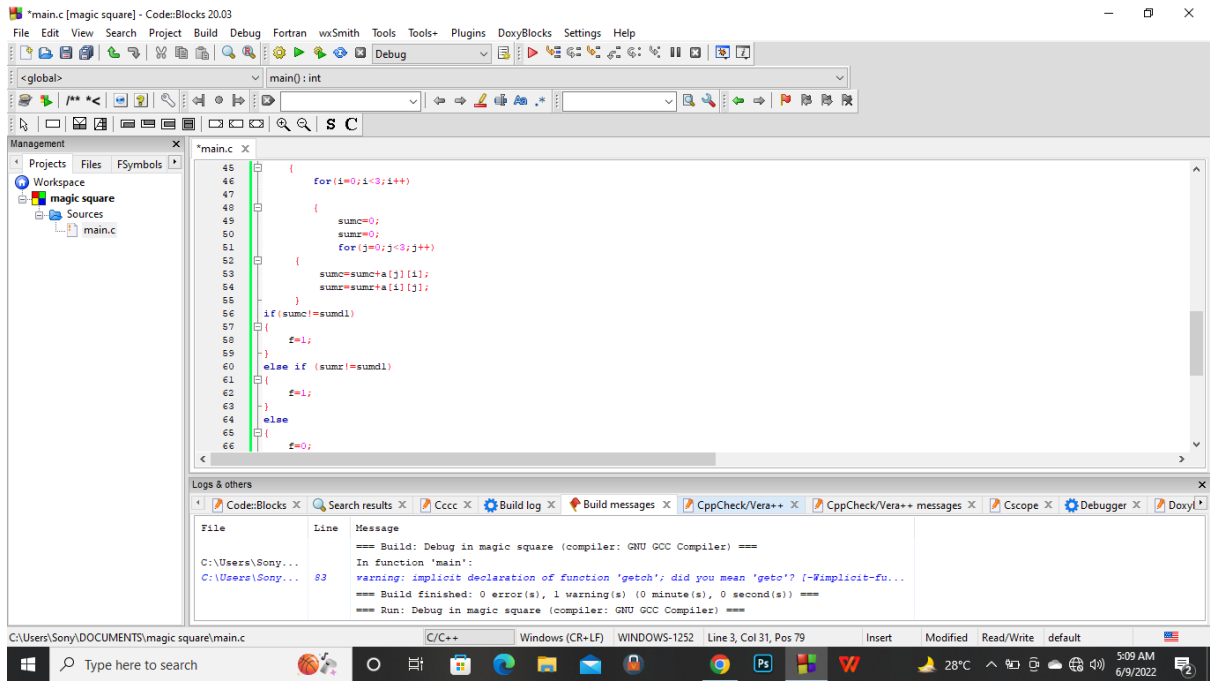
        }

    }

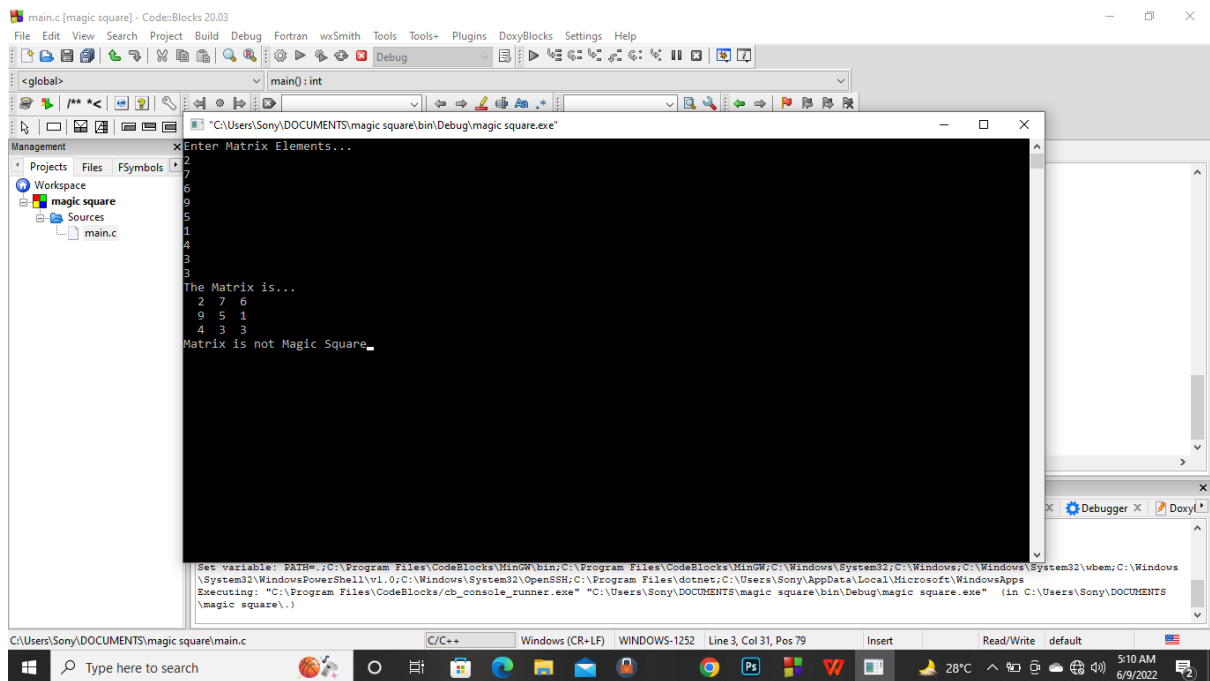
    if(sumc!=sumd1)
```

```
{  
    f=1;  
}  
else if (sumr!=sumd1)  
  
{  
    f=1;  
}  
else  
  
{  
    f=0;  
}  
}  
  
if(f==0)  
  
{  
    printf("Matrix is Magic Square Matrix");  
}  
else  
  
{  
    printf("Matrix is not Magic Square");  
}  
  
getch();  
}
```





Output:

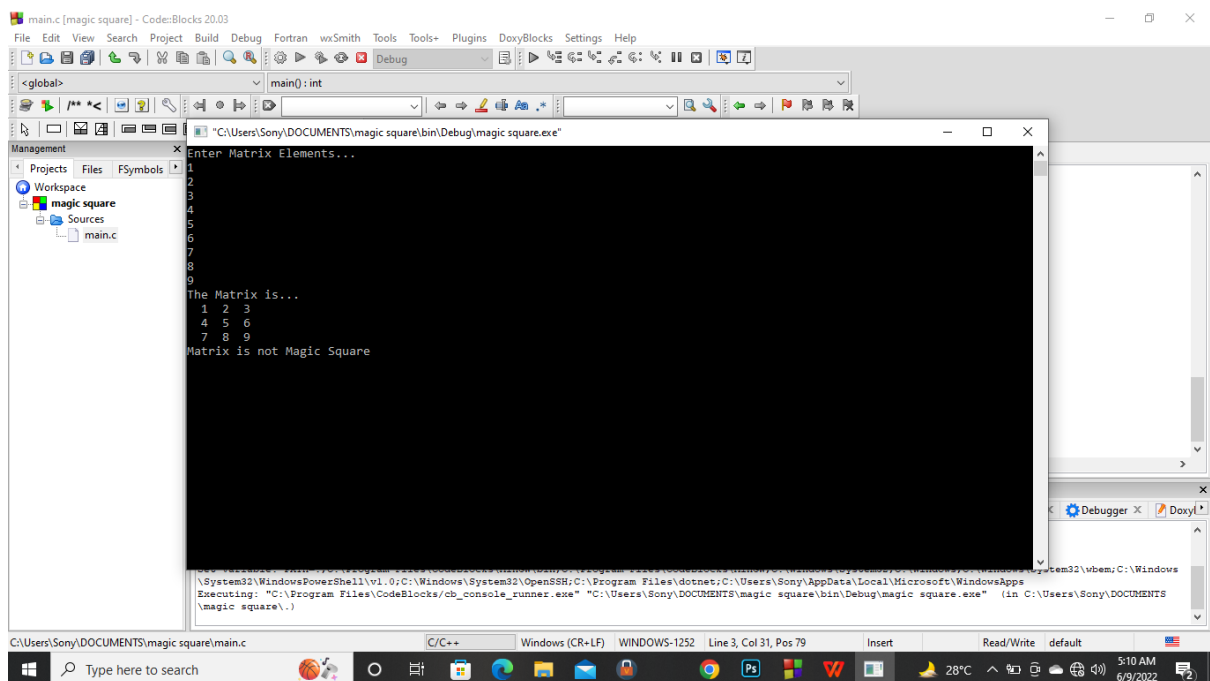


The screenshot shows the Code::Blocks IDE with a project named 'magic square'. The main.c file is open, and the program is being debugged. The console window displays the following output:

```
Enter Matrix Elements...
2
7
6
9
5
1
4
3
3

The Matrix is...
2 7 6
9 5 1
4 3 3

Matrix is not Magic Square
```



The screenshot shows the Code::Blocks IDE with the same project 'magic square'. The main.c file is open, and the program is being debugged. The console window displays the following output:

```
Enter Matrix Elements...
1
2
3
4
5
6
7
8
9

The Matrix is...
1 2 3
4 5 6
7 8 9

Matrix is not Magic Square
```

In-Lab Task 1: 2D Arrays and user defined functions

- Your task is to declare a 2D array whose dimensions should be entered by the user of the program.
- Then you should initialize the array with ones (using nested loops).
- Next you have to write a function **array_multiply()** which takes in the array or its pointer as argument and multiplies all its entries with a user input number. (**Hint:** you will also need to pass in the dimensions of the matrix to this function).
- Similarly write a function **array_add()** that adds a constant number to all the entries in a 2D array.
- Print the results of calling these functions.

Code:

```
*main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  void array_multiply(int rows, int columns, int arr1[rows][columns]);
4  void array_add(int rows, int columns, int arr2[rows][columns]);
5
6  void array_multiply(int rows, int columns, int arr1[rows][columns])
7  {
8      int num=0,x,y;
9      printf("\nMultiplication\n\n");
10     printf("Enter a number : ");
11     scanf("%d",&num);
12     for(x=0;x<rows;x++)
13     {
14         for(y=0;y<columns;y++)
15         {
16             arr1[x][y]=arr1[x][y]*num;
17         }
18     }
19     for(x=0;x<rows;x++)
20     {
21         for(y=0;y<columns;y++)
22         {
23             printf("%d ",arr1[x][y]);
24         }
25         printf("\n");
26     }
27     void array_add(int rows, int columns, int arr2[rows][columns])
28     {
29         int num1=0,x,y;
30         printf("\nAddition\n\n");
31         printf("Enter a number : ");
32         scanf("%d",&num1);
33         for(x=0;x<rows;x++)
```

```
*main.c X
30     scanf("%d",&num1);
31     for(x=0;x<rows;x++)
32     {
33         for(y=0;y<columns;y++)
34         {
35             arr2[x][y]=arr2[x][y]+num1;
36         }
37     }
38     for(x=0;x<rows;x++)
39     {
40         for(y=0;y<columns;y++)
41         {
42             printf("%d ",arr2[x][y]);
43         }
44         printf("\n");
45     }
46 int main()
47 {
48     int x,y,rows,columns,n;
49     printf("Enter the number of rows : ");
50     scanf("%d",&rows);
51     printf("Enter the number of columns : ");
52     scanf("%d",&columns);
53     int arr[rows][columns],i;
54     printf("Enteries of matrix");
55     for(x=0;x<rows;x++)
56     {
57         for(y=0;y<columns;y++)
58         {
59             scanf("%d",&i);
60             arr[x][y]=i;
61         }
62     }
63     array_multiply(rows,columns,arr);
64     array_add(rows,columns,arr);
65 }
```

Output:

```
Enter the number of rows : 3
Enter the number of columns : 2
Enteries of matrix
7
8
9
0
2

Multiplication
Enter a number : 6
36 42
48 54
0 12

Addition
Enter a number : 3
39 45
51 57
3 15

Process returned 0 (0x0)    execution time : 18.265 s
Press any key to continue.
```

Critical Analysis / Conclusion:

In this lab we learn the basic operations on 2D arrays. We learn how to declare and use 2d arrays and learn how to implement them in functions. We print 2D array contents to console screen. We learn how to access 2D array elements via pointers. We also practicing 2D array operations by implementing a Magic Square algorithm. We learned about the magic square and how to fill it in.

Lab Assessment				
Pre Lab			/1	/10
In Lab			/5	
Post Lab	Data Analysis	/4	/4	
	Data Presentation	/4		
	Writing Style	/4		
Instructor Signature and Comments				