Lab 09 2D Arrays

Objectives:

- Learn the basic operations on 2D arrays.
- Printing 2D array contents to console screen.
- Accessing 2D array elements via pointers.
- Practicing 2D array operations by implementing a Magic Square algorithm.

Reading Task 1: Working with Arrays

Chapter 08 Arrays (pages 269 to 303) from the book: “Let us C” by Yashavant Kanetkar

Reading Task 2: The Magic Square

The **magic square** is a square matrix, whose **order is odd** and where the sum of the elements for each row or each column or each diagonal is same. The sum of each row or each column or each diagonal can be found using this formula. n(n2+ 1)/2 where **‘n’** is the order of the matrix.

8	1	6
3	5	7
4	9	2

Figure 1: *Magic Square of order 3*

A method for constructing magic squares of odd order was published by the French diplomat de la Loubère in his book, *“A new historical relation of the kingdom of Siam”* (Du Royaume de Siam, 1693), in the chapter entitled *The problem of the magical square according to the Indians*. The method operates as follows:

1. The method prescribes starting in the central column of the first row with the number 1.
2. After that, the fundamental movement for filling the squares is diagonally up and right, one step at a time.
3. If a filled square is encountered, one moves vertically down one square instead, then continues as before.
4. When an "up and to the right" move would leave the matrix, it is wrapped around to the last row or first column, respectively.

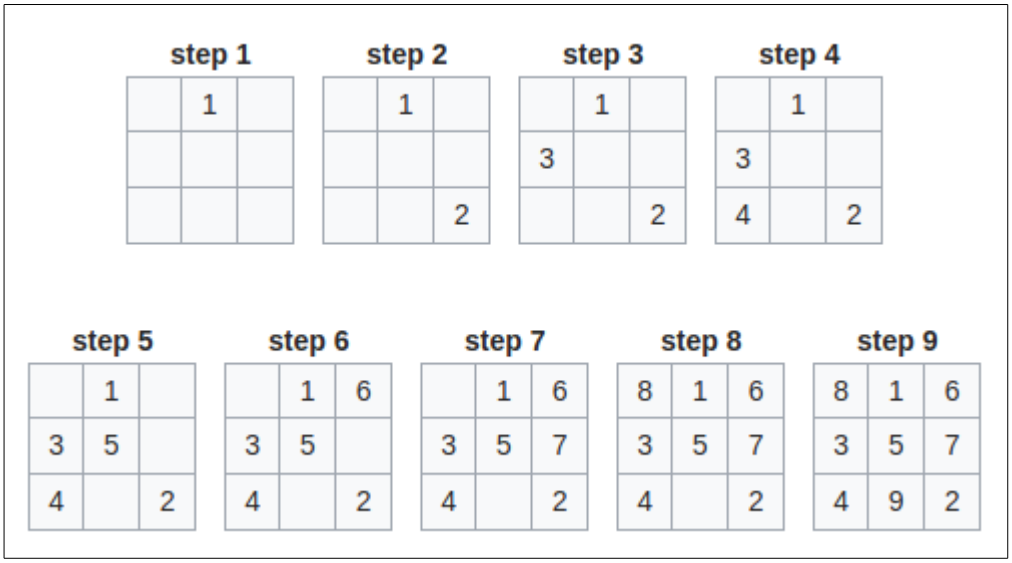


Figure 2. *Algorithm for filling the Magic Square*

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.

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.

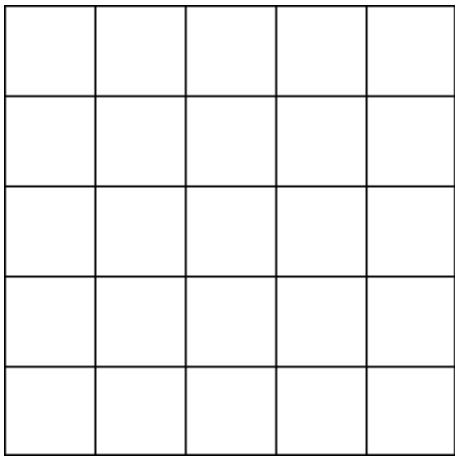
Your job is to complete this code by implement the algorithm discussed in the **Reading Task 2**.

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 Listing 1

```
int magic_square(int n)
{
   /// to rule out zero, negative and even inputs
   if((n<1) || (n%2 == 0))
   return(-1);

   /// Declare an n x n array using C VLAs (Variable Length Arrays)
   int mSquare[n][n];

   /// Initialize the matrix with zeros
   for(int i=0; i<n; i++)
   {
      for(int j=0; j<n; j++)
      {
         mSquare[i][j] = 0;
      }
   }

   int count = 1;  /// The first entry in the Magic Square

   ///start from middle col in the first row.
   int col = (n-1)/2;
   int row = 0;

   /// Step1: Fill 1 in the middle column first row
      mSquare[row][col] = count;

   int row_t;  /// for temporary storage
   int col_t;

  /************* INSERT YOUR CODE HERE ********/

/// Print the Magic Square
   for(int i=0; i<n; i++)
   {
      for(int j=0; j<n; j++)
      {
         printf("%d\t", mSquare[i][j]);
      }
      printf("\n");
   }

   return(0);
}
```

Code Listing 2