1. Consider the problem of searching in a sorted matrix. That is, you are given an n × n matrix A, where each entry is a distinct integer. Each row of the matrix is sorted in ascending order, and each column is also sorted in ascending order. Given a value x, the problem is to decide whether x is stored somewhere in the matrix (i.e., whether there is some i and j such that A[i][j] = x). Write a program to implement an efficient (O(n)) algorithm for searching in a sorted square matrix.

Input:

The given matrix is :

1 2 3

4 6 9

7 8 10

Search item : 8

Output:

The element found at the position: 2, 1

Input:

Search item : 5

Output:

The given element not found in the given matrix

Input:

The given matrix is :

11 21 31 41 51

12 22 32 42 52

13 23 33 43 53

14 24 34 44 54

15 25 35 45 55

Search item : 34

Output:

The element found at the position: 3, 2

Input:

Search item : 55

Output:

The element found at the position: 4, 4

Input:

Search item : 56

Output:

The given element not found in the given matrix

2. Write an efficient program to generate the following spiral pattern matrix for any n, where n is the order of the square matrix.

For example, n=4, following spiral pattern matrix will be generated.

7	8	9	10
6	1	2	11
5	4	3	12
16	15	14	13

Test your program for n=2,3,4,5,6,7,8,……

3. A magic square is an $n \times n$ matrix that have an arrangement of n^2 numbers, usually distinct positive integers, such that the sum of n numbers in all rows, all columns and both diagonals is the same constant called the magic sum. A normal magic square contains the integers from 1 to n^2. Normal magic squares exist for all $n > 2$.

The smallest nontrivial case, shown below, is of order 3 and whose magic sum is 15.

2	9	4
7	5	3
6	1	8

Write an efficient program to generate the above Magic Squares of odd order (3,5,7, etc.) up to order 51 using following algorithm for generating magic square when $n>2$ and n is odd.

Start with 1 in the middle cell of the bottom row, then go down and left, assigning numbers in increasing order to empty squares; if you fall off the square imagine the same square as tiling the plane and continue; if a square is occupied, move up instead and continue. Repeatedly do these till n^2.