The Magic Square - 2 Dimensional Arrays

A Magic Square is a grid of unique numbers such that each row, each column and each diagonal add up to the same number. Each number in the grid should be between 1 and s^2 and should be used once and only once.

A 4 x 4 Dürer magic square has the following additional properties:

- The sum of the 4 corners is the same as the row, column and diagonal sum
- The sum of the 4 center cells is the same as the row, column and diagonal sum
- The sum of the top and bottom of the two center columns is the same as the row, column and diagonal sum.
- The sum of the left and right of the two center rows is the same as the row, column and diagonal sum.
- If you go clockwise around the square and start in any cell other than the corners, the sum of the 4 cells in the corresponding position is the same as the row, column and diagonal sum.

See
`http://mathforum.org/alejandre/magic.square/adler/adler.whatsquare.html`

Create a class called `MagicSquare`

`MagicSquare` has a 2 dimensional array as its instance variable. There is no default constructor however, the paramertized constructor accepts the size as it's only parameter, instantiates the 2 dimensional array and prompts the user to enter values left to right, row by row.

Include the following methods:

- `public boolean isMagic()` - returns true if the 2-dim array is a magic square
- `public boolean isA4x4Durer()` – returns true if it `isMagic()` and it holds properties of a Dürer magic square. It should verify that the array is a 4 x 4 array.
- `private int cornerSum()` – returns the sum of the corners of the 2-dim array
- `private int centerSum()` – returns the sum of the 4 inner cells of the 2-dim array
- `private int colSum(int c)` – returns the sum of column c
- `private int rowSum (int r)` – returns the sum of column r
- `private int diag1()` – returns the sum of upper-left to lower-right diagonal
- `private int diag2()` – returns the sum of lower-left to upper-right diagonal
- `private boolean valuesCheck()` – each number is between 1 and s^2 and used only once
- `private int topBottomCenterColSum()` – returns the sum of the values indicated below

- `private int leftRightCenterRowSum()` – returns the sum of the values indicated below

- `private int clockwiseSum()` – compares the sum of the values of the four cells traveling clockwise around the square that start in cell b with the sum of the values of the four cells that start in cell c (see *Illustration of a Dürer Magic Square* below). Return the sum or -1 if not equal.

- `public String toString()` - displays the contents of 2-dim array in grid format

Illustration of a Dürer Magic Square is described below.

a	b	c	d
e	f	g	h
i	j	k	l
m	n	o	p

cornerSum() = a + d + p + m
centerSum() = f + g + k + j
colSum() = a + e + i + m OR b + f + j + n
OR c + g + k + o OR d + h + l + p
rowSum() = a + b + c + d OR e + f + g + h
OR i + j + k + l OR m + n + o + p
topBottomCenterColSum = b + c + o + n
leftRightCenterRowSum = e + h + i + l
clockwiseSum = b + h + o + i OR c + l + n + e

Sample tester program.

```
public static void main (String [] args) {
     MagicSquare ms = new MagicSquare(3);
     if (ms.isMagic())
          System.out.println (ms + "\nis a Magic Square");
     else
          System.out.println (ms + "\nis not a Magic Square");
     System.out.println ();
     MagicSquare durer = new MagicSquare(4);
     if (durer.isA4X4MagicSquare() )
          System.out.println (durer + "\nis a Durer Magic Square");
     else
          System.our.println (durer + "\nis not a Durer Magic Square");
```

You can use the following squares to test your program:

8	1	6
3	5	7
4	9	2

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