2D ARRAYS

EXERCISES

The following code has been written to work with 2D arrays

```
public class ArrayQuestions
{
    public static void printArray(int[][] arr) {
        for (int i=0; i<arr.length; i++) {
            for (int j=0; j<arr[i].length; j++) {
                System.out.print(arr[i][j]+" ");
            }
            System.out.println();
        }
    }

    public static void main() {
        int[][] arr = new int[5][5];
        arr = question1(arr);
        printArray(arr);
    }

    public static int[][] question1(int[][] arr) {
        return (arr);
    }

    public static int[][] question2(int[][] arr) {
        return (arr);
    }

    public static int[][] question3(int[][] arr) {
        return (arr);
    }
}
```

For each question, you are to complete a function containing an algorithm that fills the array such that it matches the target output. You may use for loops, while loops, or a combination of several loops. You may also create whatever variables you want. The pattern should hold true if the size of the array changes (though it will always be a square).

For example, if the target output was, your sample code could look like (there are multiple ways it could be solved):

```
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
```

```
public static int[][] example(int[][] arr) {
    int counter = 1;
    for (int i=0; i<arr.length; i++) {
        for (int j=0; j<arr[i].length; j++) {
            arr[i][j] = counter++;
        }
    }
    return (arr);
}
```

Some questions may not be as easy as they seem… have fun!!!

(the shading is just to help you spot the pattern)

0	1	2	3	4
0	1	2	3	4
0	1	2	3	4
0	1	2	3	4
0	1	2	3	4

Problem 1

0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4

Problem 2

5	4	3	2	1
10	9	8	7	6
15	14	13	12	11
20	19	18	17	16
25	24	23	22	21

Problem 3

21	22	23	24	25
16	17	18	19	20
11	12	13	14	15
6	7	8	9	10
1	2	3	4	5

Problem 4

25	24	23	22	21
20	19	18	17	16
15	14	13	12	11
10	9	8	7	6
5	4	3	2	1

Problem 5

1	6	11	16	21
2	7	12	17	22
3	8	13	18	23
4	9	14	19	24
5	10	15	20	25

Problem 6

1	2	3	4	5
10	9	8	7	6
11	12	13	14	15
20	19	18	17	16
21	22	23	24	25

Problem 7

1	10	11	20	21
2	9	12	19	22
3	8	13	18	23
4	7	14	17	24
5	6	15	16	25

Problem 8

1	2	3	4	5
16	6	7	8	9
17	18	10	11	12
19	20	21	13	14
22	23	24	25	15

Problem 9*

1	2	3	4	5
10	6	7	8	9
14	15	11	12	13
18	19	20	16	17
22	23	24	25	21

Problem 10*

1	3	6	10	15
2	5	9	14	19
4	8	13	18	22
7	12	17	21	24
11	16	20	23	25

Problem 11*

3	3	3	3	3
3	2	2	2	3
3	2	1	2	3
3	2	2	2	3
3	3	3	3	3

Problem 12**

(note: for problem 12 you can assume the array length will always be an odd integer)

Ready for a challenge?

Problem 13 (Spiral): See if you can generate the following spiral pattern.

1	2	3	4	5
16	17	18	19	6
15	24	25	20	7
14	23	22	21	8
13	12	11	10	9

SIZE = 5

1	2	3	4	5	6
20	21	22	23	24	7
19	32	33	34	25	8
18	31	36	35	26	9
17	30	29	28	27	10
16	15	14	13	12	11

SIZE = 6

1	2	3	4	5	6	7
24	25	26	27	28	29	8
23	40	41	42	43	30	9
22	39	48	49	44	31	10
21	38	47	46	45	32	11
20	37	36	35	34	33	12
19	18	17	16	15	14	13

SIZE = 7

(Adapted for Java. Original C++ version by Brian Choi 2011)