Purpose: Create a Solitaire Game Using a Two-Dimensional Array

Details:

THE GAME
Given a square array of “lights”, each one controlled by a push-button switch, we want to turn out all the lights by pressing the buttons. However, the buttons are connected to each other so that every switch controls up to five different lights — in particular, each light switch is connected to the one above, below, to the left, and to the right of its position (unless it is at the end of a row or column). When we press the button associated with a particular light, it also causes the buttons connected to it to be pushed. For instance, suppose the lights are initially all on, and we press the button in row 1, column 3. Then five lights are turned out. If we then click on row 0, column 4, one light is turned off, namely the one in row 0, column 4, but two more are turned back on. Here are the results of three consecutive button pushes:
THE ASSIGNMENT
Write a Java program that lets a user play this game for any size board from $3 \times 3$ to $10 \times 10$. The number $n$ of rows and columns should be specified as an args parameter, not through Scanner input. The program should reject illegal values of $n$ and halt.

Your program should use two symbols to represent “light on” and “light off” and save these in an $n \times n$ array. The array should be initialized with all lights on, and the array should be printed out before every move by the user.

If the user enters an illegal row or column number, the program should ask if the user really wants to quit; if the user enters “y” then halt the program (one way to do this is to break out of the loop that controls the game). But if the user enters “n” then the program should continue. (This way the user has the ability to stop the program, but accidentally entering a wrong value won’t terminate the program.)

The program should print the number of valid moves made by the user at the conclusion of the program.

OTHER REQUIREMENTS
Use methods to do well-defined subtasks. The “skeleton program” provided for you shows an example of a breakdown into methods. You are not required to imitate this, however. Some of the methods listed in the skeleton program are

- a char-valued method that takes one of the two symbols and returns its opposite. For instance, it might be used like this:
  
  \[ \text{board[r][c] = opposite(board[r][c]); // on to off or off to on} \]

- a void method that prints out the contents of the board

- a boolean method that checks to see if the lights have all been turned out (see “grading” section, below)

You might be able to think of more methods.

GRADING
If you get just the basic game working as described above, well documented and with good use of methods, you will receive a passing grade of no more than 9 (depending on the usual grading criteria). However, if you want to receive the full ten points, your program should also include a way to automatically halt the program whenever the user manages to turn off all of the lights. When this happens the user should be informed that the game has been won and told the number of moves. (This is not hard to do; I encourage you to attempt it!)

Anything else you can do to improve the program may also get you more points. For instance, can you figure out how to display the board in a nicer format, like this?

\[
\begin{array}{cccc}
 0 & 1 & 2 & 3 & 4 \\
\hline
 0 | # | # | | | # |
\end{array}
\]
However, please follow the basic design guidelines given above — this should be text-based, the user should be able to enter moves by row and column number, etc. (Later this semester we will talk about making this into an applet.)

Hand In
Hand in a copy of the program and a printout of a sample run that shows off all the features — correct manipulation of the lights, correct handling of user input, etc. Your output should look at least as good as the example below.

[rroos@aldenv123]$ java Lab10
Usage:

java Lab10 n

where n is between 3 and 10, e.g., java Lab10 5
[rroos@aldenv123]$ java Lab10 2
Usage:

java Lab10 n

where n is between 3 and 10, e.g., java Lab10 5
[rroos@aldenv123]$ java Lab10 20
Usage:

java Lab10 n

where n is between 3 and 10, e.g., java Lab10 5
[rroos@aldenv123]$ java Lab10 5
Bob Roos Sun Apr 09 08:41:20 EDT 2006
#####
#####
#####
#####
Enter row and column (enter illegal values to quit): 0 0
..###
.####
#####
Enter row and column (enter illegal values to quit): 1 2
...##
....#
##.##
#####
Enter row and column (enter illegal values to quit): 5 5
Really quit (y/n)? n
Enter row and column (enter illegal values to quit): 4 0
...##
....#
##.##
.####
..###
Enter row and column (enter illegal values to quit): 4 4
...##
....#
##.##
.###.
...#
Enter row and column (enter illegal values to quit): -1 -1
Really quit (y/n)? y
You made 4 moves.
Thanks for playing.

Checklist:

☐ All requirements met as described above
☐ Good use of methods
☐ Javadoc-style header comments
☐ Your name and the date printed as the first thing
☐ Correctly indented code; no wrapped lines
☐ Hard copy (printed using a2ps) of program
☐ Hard copy (printed using a2ps) of a sample run