Joshua Nelson

CSCE 146

Lab 01: Mine Counter

Problem

Create a board for a game of Mine Sweeper. The board should be a 10x10 grid with 10 mines placed. Each space without a mine should show the number of adjacent mines (0 if none).

Proposed Solution

Create a two dimensional character array. While the board is not full, loop over each element. The probability that the element will be populated is `mines / (x * y)`. To avoid favoring earlier elements, the probability should be halved while populating. For each mine bordering an empty square (row – 1, row + 1, row -1 and column -1, etc.), increase the number displayed on the empty square. Numbers should be represented in ASCII, NOT in decimal. For each character in the board, print it to standard out. For each row, print a new line.

Tests and Results

```
0 | 0 | 0 | 0 | 1 | * | 2 | * | 1 | 0
0 | 0 | 0 | 0 | 1 | 1 | 2 | 1 | 1 | 0
0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0
2 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0
* | * | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1
3 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | *
* | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 3 | *
2 | 3 | * | 1 | 0 | 0 | 0 | 0 | 2 | *
1 | * | 2 | 1 | 0 | 0 | 0 | 0 | 1 | 1
1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0

Make a new board? [yes]

Enter x dimension
10 x 10 board
Enter a y dimension.
10
Enter number of mines
```

Joshua Nelson

CSCE 146

```
10
1 | * | 1 | 0 | 0 | 0 | 1 | * | 1 | 0
2 | 2 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0
* | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0
1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 2
0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | * | *
2 | 2 | 1 | 0 | 0 | 0 | 0 | 1 | 3 | *
* | * | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1
2 | 2 | 1 | 0 | 0 | 1 | * | 1 | 0 | 0
0 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 0 | 0
0 | 0 | 0 | 0 | 0 | 1 | * | 1 | 0 | 0

Make a new board? [yes]
Different size array
Enter x dimension
12
Enter a y dimension.
12
Enter number of mines
144
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *
* | * | * | * | * | * | * | * | * | * | * | *

Make a new board? [yes]
Edge case
Enter x dimension
```

Joshua Nelson

CSCE 146

```
1
Enter a y dimension.
1
Enter number of mines
0
0

Make a new board? [yes]
Edge case
Enter x dimension
1
Enter a y dimension.
1
Enter number of mines
12
Invalid argument: Mines must be less than or equal to number of
spaces
Enter x dimension
j
Invalid argument: For input string: "j"
Enter x dimension
4
Enter a y dimension.
8
Enter number of mines
12
* | 4 | 2 | 2 | * | * | 2 | 1
* | * | * | 2 | 3 | 3 | 4 | *
4 | 5 | 3 | 1 | 2 | * | 4 | *
* | * | 1 | 0 | 2 | * | 3 | 1

Make a new board? [yes]
Edge case
Enter x dimension
-1
Enter a y dimension.
```

Joshua Nelson

CSCE 146

```
-1
Enter number of mines
3
Invalid argument: Rows must be greater than 0.

^C
```

Problems Encountered

It’s hard to find mines bordering a square! The iterative solution I came up with is almost as long as a series of if-statements would have been.

Major problems:

- When taking input for x and y dimensions, it was hard to check for valid input. I ended up throwing an argument exception, which delays checking input until after x, y, and mines have all been input.
- When printing board, I wanted to print a delimeter (‘ | ‘) between each mine. However, this printed a pipe character at the end of each line. Had to address this on its own.
- Dividing by zero is not, in fact, a valid mathematical statement. Had to reason this out when determining the probability any given space would be a mine when the number of mines was set to 0. It turns out that if 0 spaces should be mines, the probability a space should be a mine is 0.
- Public classes must have the same name as their containing file, which is really annoying: `mv MineCounter.java Main.java; javac Main.java` throws an error.

Conclusions and Discussion

This was a fun lab. I generalized the board to take arbitrary x and y sizes from user input. This does get ugly if you input a y greater than the column size of your screen.

I could have checked input immediately by generalizing a static validator and running that after each input, as well as within the constructor itself. I think it is important to note that validation is separate from a setter: you can validate x and y without having to call the constructor itself.

Joshua Nelson

CSCE 146

I could certainly have added more comments. Most of this is very dense code. I did do a good job (in my opinion) of having descriptive variable and method names.

Additional Questions

1. Is it possible to change the size of a standard array in Java after it has been constructed?

No. This is only possible when using java.util.List or other abstracted classes (Linked Lists, HashMaps, etc). I believe this holds true for all compiled languages, but this is not true (for example) of Python's lists.

2. A multidimensional array can be thought of an array of arrays. In Java, does each array of another have to be the same size or does Java support ragged arrays?

Java supports ragged arrays. The only language I know of where this is not natively supported is C++ (and C), but ragged arrays can still be implemented with an array of pointers to arrays.