CECS 277 – Lab 3 – 2D Lists

Minesweeper Board Creator

Create a program that generates a RxC solution board for the game Minesweeper. Prompt the user to enter the size of the grid (rows and columns between 5-10), and the number of mines (also between 5-10). Randomly place the mines on the grid, then fill the remaining spaces by counting the number of mines in any adjacent squares (diagonals included). Display the filled grid. Error check all user input and document each of your functions.

Create the following functions for your program:

1. `place_mines(board, mines)` – for each mine, generate a random row and column within the bounds of the grid and check that location. If it is a 0, then place an 'X' there, otherwise repeat the loop to generate a new location (this ensures that it cannot overwrite a previously written 'X' and generate the incorrect number of mines).
2. `count_mines(board)` – for each spot in the grid (except places with a mine 'X'), count any 'X's in the surrounding eight spaces, then place the value of the counter in that spot. Use a nested set of loops to iterate through the surrounding eight spaces. Be careful not to overwrite any mines.
3. `display_board(board)` – display the contents of the grid using a set of nested loops.

The main method should prompt the user for the three inputs, then call each of the three methods in the above order.

Example Outputs (user input is in italics)**:**

```
Minesweeper Maker
Enter number of rows (5-10): 5
Enter number of columns (5-10): 5
Enter number of mines (5-10): 5

X 2 2 1 1
2 X 2 X 1
2 2 2 1 1
X 2 1 1 0
1 2 X 1 0
```

```
Minesweeper Maker
Enter number of rows (5-10): 6
Enter number of columns (5-10): 8
Enter number of mines (5-10): 10

0 0 1 1 1 0 0 0
1 1 1 X 3 2 2 1
X 3 3 4 X X 3 X
X 3 X X 3 3 4 X
1 2 2 2 1 1 X 2
0 0 0 0 0 1 1 1
```

Notes:

1. Check all user input for invalid values. You can use the check_input module provided on Canvas.
2. Use the random module to randomly generate the rows and columns for the mines.
3. Do not create any extra functions or add any extra parameters.
4. Do not create any global variables.

5. Follow the guidelines in the Coding Standards document.
6. Use docstrings to document each of your functions.
7. Place your name, the date, and a brief description of the program in a comment block at the top of your program. Place brief comments throughout your code.
8. If you like, you can create your grid 2 spaces larger than the user's inputted width and height so that your count_mines method doesn't have to do any extra bounds checking. Just make sure that your other methods account for this.
9. Thoroughly test your program before submitting/demoing.
    a. Make sure that the grid displays correctly.
    b. Make sure that you didn't mix up the rows and columns.
    c. Make sure that it always places the correct number of mines on the board.
    d. Make sure that each of the spaces has the correct number of mines counted.
    e. Error check each of the inputs (ranges 5-10).