TOBB ETÜ, BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİL 113/012 Computer Programming I

HOMEWORK 7 (40 Points)

Dec 20, 2020

1 [30 POINTS] MINESWEEPER

In this task, you are going to write the famous game Minesweeper. As you may already know, Minesweeper is played at nxm board. The board contains $n \times m$ cells. Each cell may contain a mine or not. We will refer to the cells that contain mine as mine-cells and others as free-cells. At the beginning of the game, all cells are closed. The player opens cells one by one. If he opens a mine-cell, then the game is over. The aim of the game is to detect and open all free-cells without opening any mine-cells.

We refer to the cells that are adjacent to a cell its neighbor cells. Each cell adjacent to the at most 8 cells that surround it, so each cell can have at most 8 neighbor cells. Notice that if the cell is located at the corners of the board then it has 3 neighbor cells. If the cell is located at the edges of the board but not at the corners, then it has 5 neighbors. If the cell is located anywhere else then it has 8 neighbors.

If the player opens a mine-cell game ends and the user loses. If the user opens a free-cell, it tells the user how many mine-cells are adjacent to that cell. If the player opens a cell that is not adjacent to a mine-cell, then all its neighbor cells are opened automatically. Notice that automatically cells may have 0 mine-cells neighbors, then its neighbors are opened automatically too. This can go on quite bit and one of the hardest parts of this question.

Users can flag the cells instead of opening them. Flagged cells must be unflagged before opening. The size of the game board is dynamic and there are 3 difficulty levels. Difficulty levels determine the percentage of the cells that contain mine in the board. You can find these ratios in the table below.

Difficulty	Easy	Medium	Hard
Percantage of Mine-cells	15	25	40

Table 1.1: Difficulty/mine-cell ratio table

For example, if the size of the board is $10x9$ and the difficulty is Medium, then the board should contain exactly 22 mines.

The programs should ask the sizes of the game board, then the difficulty. After that, it should scatter the mines to the cells uniformly at random. After the board is created, the program should print the board. Initially, all cells are closed. We will represent closed cells with "o". Each cell has a coordinate composed of two integers. The lower-left cell has the coordinate "1,1" and the upper right cell has the coordinate "n,m". You can find the initial state and coordinates of a $4x4$ board in the figure below. While printing the board, the program should put a space between each cell in the same row and print each row in a separate line. For instance, it should print "o o o o" for the first row for the example below.

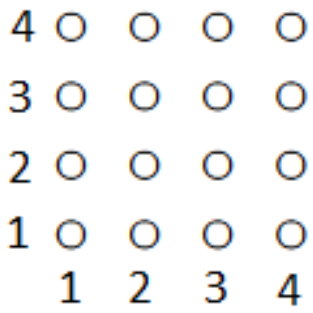

Figure 1.1: Board

The program should ask for a move and update the board according to that move until the game ends. Moves are composed of x,y ACTION. ACTION can be the following: ""(Open), "F" (Flag), and "U" (unflag). For example "2,4" is an open move, "2,4 F" is a flag move, and "2,4 U" is an unflag move. Only closed cells can be flagged and only flagged cells can be unflagged. We will represent flagged cells with "F", so flagged cells should show "F". After unflagging a cell, it should show "o" again. After a cell is opened, it should show the number of mine-cells adjacent to that cell if it is a free-cell. If the cell is not adjacent to any mine-cells, then it should show "-" instead of "0" (and open neighbor cells automatically). If the user opens a mine-cell, the game ends. The program should it should display every mine in the game (print the board again, replace all mine-cells with "X") and print the "You lost, better luck next time.". After the game ends (all free-cells are open), the program should print "Congratulations, you won.".

If the user makes an illegal move, the program needs to print an error message, then print the same board and ask for a new move. If the user tries to open a flagged cell, the program should print "Flagged cells cannot be opened". If the user tries to unflag an open or closed cell, the program should say "Only flagged cells can be unflagged". If the user tries to open an already open cell, the program should say "Cell is already open". If the user tries to flag an open cell, the program should say "Open cells cannot be flagged". If the user tries to flag an already flagged cell, the program should print "The cell is already flagged.". You can assume that the user will always make a move on an existing cell. For example, you can assume that the user will not try to make a move "5,7" on a "5x5" board.

Notes:

- **Everyone will demonstrate this homework and you will receive your grade after this presentation.** So slight variations on error messages or board printing are not that important.
- Notice that automatically opened cells may be adjacent to zero mine-cells also. This is one of the hardest parts of this question.

```
::WELCOME::
Please enter the sizes of the board (m x n):3 x 3
Please select the difficulty(E, M, H): M
o o o
o o o
o o o
Please make a move:2,2
o o o
o 2 o
o o o
Please make a move:2,2
o o o
o 2 o
o o o
Cell is already open
Please make a move:2,2 F
Open cells cannot be flagged.
o o o
o 2 o
o o o
Please make a move:2,2 U
Only flagged cells can be unflagged.
o o o
o 2 o
o o o
```

```
Please make a move:2,3
o o o
o 2 1
o o o
Please make a move:1,3
o o o
o 2 1
o 1 -
Please make a move:2,1
o o o
2 2 1
o 1 -
Please make a move:3,1
1 o o
2 2 1
o 1 -
Please make a move:3,3
1 o 1
2 2 1
o 1 -
Congratulations, you won.
```

Second example:

```
::WELCOME::
Please enter the sizes of the board (m x n):4 x 4
Please select the difficulty(E, M, H):M
o o o o
o o o o
o o o o
o o o o
Please make a move:1,2
o o o o
o o o o
o o o o
o 1 o o
Please make a move:4,4
o o 1 -
o o 2 1
o o o o
o 1 o o
```

```
Please make a move:2,4
o o 1 –
o o 2 1
o o o 2
o 1 o o
Please make a move:2,3 F
o o 1 –
o o 2 1
o o F 2
o 1 o o
Please make a move:2,3
Flagged cells cannot be opened.
o o 1 –
o o 2 1
o o F 2
o 1 o o
Please make a move:2,3 U
o o 1 –
o o 2 1
o o o 2
o 1 o o
Please make a move:2,3
X o 1 –
o X 2 1
o o X 2
o 1 o X
You lost, better luck next time.
```

2 [10 POINTS] JUMP IT

In this question, you are given an array of integers where each element represent the maximum number of element that can be jumped from that element. You are asked to find the minimum number of jumps that you need to reach to the last element from the first element. Consider the example below.
array[] = {2,5,3,1,3,2,2,1,1}
Solution:3 (0->1->6->8)

Your method should take an integer array as parameter and return the minimum number of jumps needed to reach to the last element.
Additional examples:
array[] = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
Solution:10 (0->1->2->3->4->5->6->7->8->9->10)

Then write a main method which creates an int array and prints the minimum number of jumps needed to reach to the last element. Since you are going to receive your grade after your the demonstration, output format is not important.