Programming Refresher Workshop

Session 4 Exercises

Learning objective:

- Problem-solving on 2-dimensional arrays

Exercise 12 (ex12): Minesweeper

[Past CS1010 lab exercise.]

Minesweeper is a computer game whose objective for the player is to clear a minefield without detonating a mine. The player is presented with a grid of squares (Figure 4) under which some contain mines, and some do not.

For a square that is **safe** (mine-free), it contains an integer value (0 to 8) indicating the number of mines surrounding it.

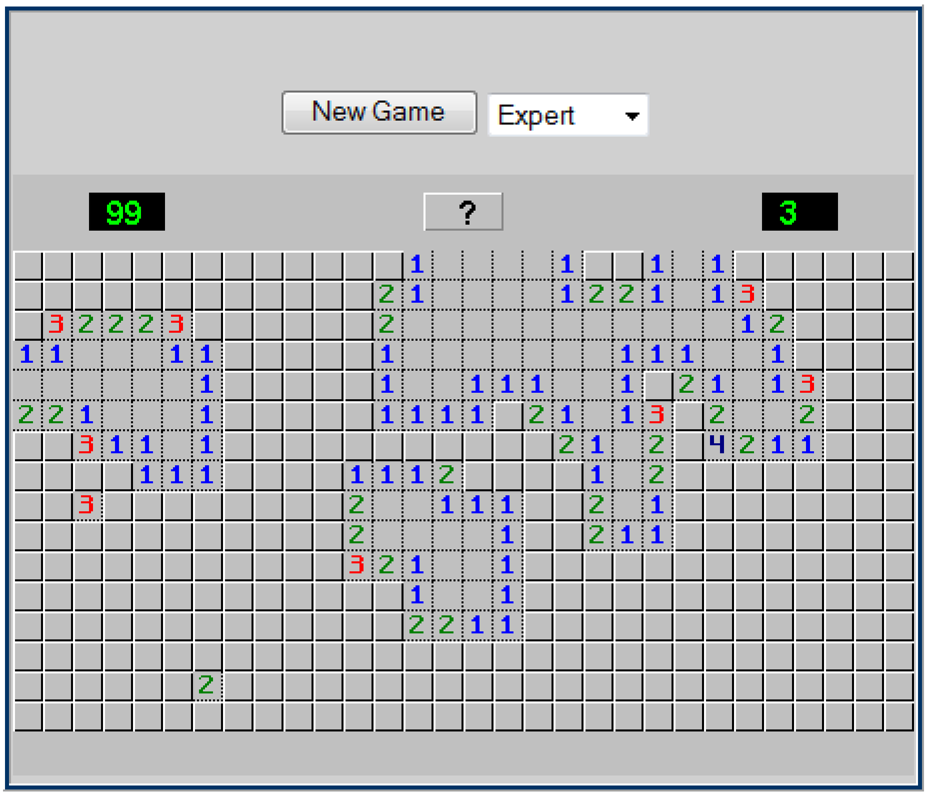


Figure 4. A Minesweeper game in progress.

A player clicks on a square to turn it over. If it contains a mine, the game ends with all the mines revealed (Figure 5). The game also ends when the player turned over all safe squares. Usually, a square that contains 0 (no mine around it) is simply displayed as a blank square.

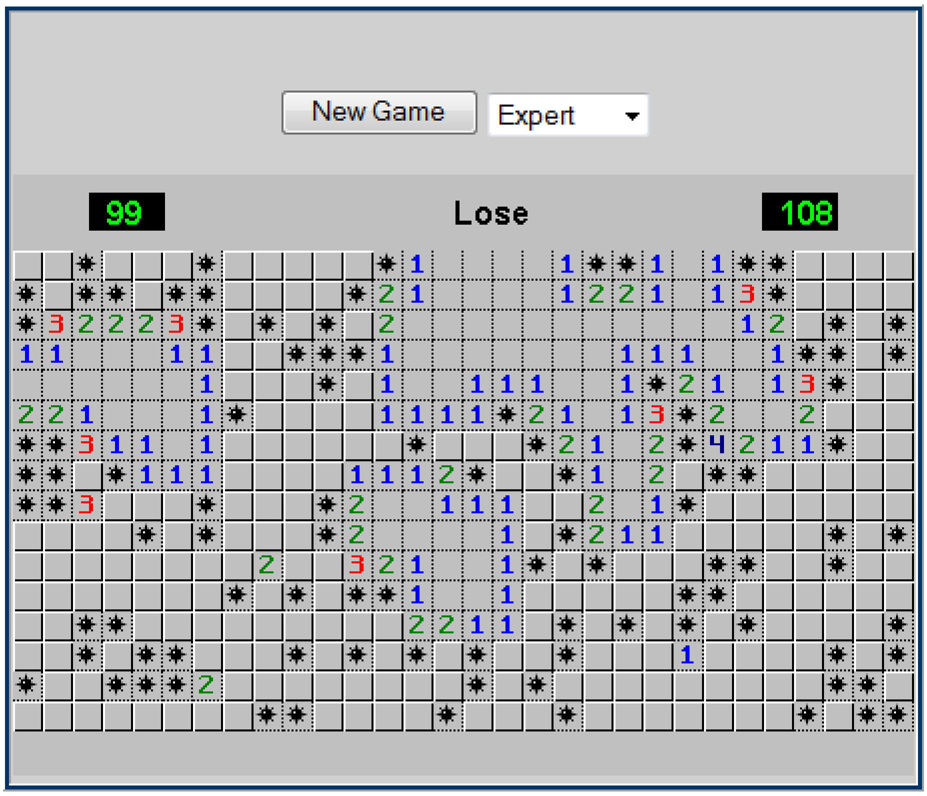


Figure 5. The Minesweeper game ends when player clicks on a square containing a mine.

You may try the game on GamesWizard.com and many other websites on the Internet.

For this exercise, you are only required to prepare the grid before the start of the game. Given the positions of all the mines in the grid, you are to fill in the numbers 0 to 9 in each of the safe squares.

For example, Figure 6 shows the positions of the mines in a minefield. You are to compute the values of the safe squares, as shown in Figure 7.

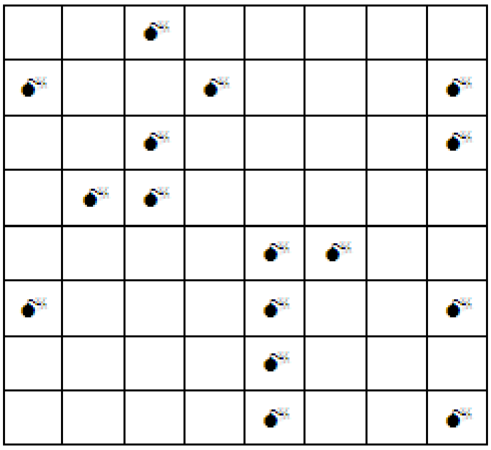
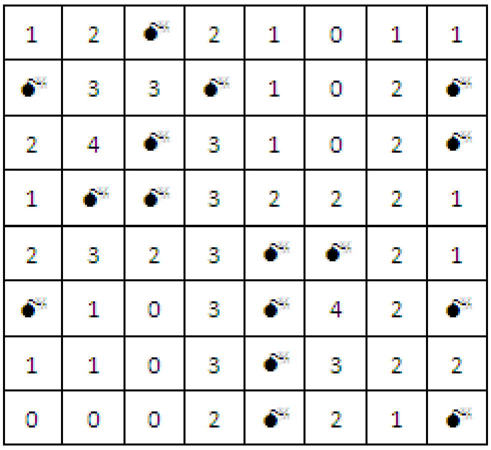


Figure 6. Position of mines.



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

Figure 7. Values in safe squares.

Write a program to read in a minefield containing the mines into a 2-dimensional character array (or a 1-dimensional array of strings), and compute the values of safe squares in a numeric 2-dimensional array.

There are 3 game levels, with each level having the following grid dimension:

- Level 1: 8 × 8 grid
- Level 2: 12 × 16 grid
- Level 3: 16 × 30 grid

Your program is to read the game level, and then the grid containing characters comprising either – (mine-free) or * (a mine). Your program then outputs a 2-dimensional integer array showing the number of mines surrounding each square. If the square in the grid contains a mine, then its corresponding value in the integer array is 9.

Note that in the output, there is a space preceding each digit.

Sample runs

```
1
--*-----
*--*---*
--*----*
-**-----
----**--
*---*--*
----*---
----*--*
 1 2 9 2 1 0 1 1
 9 3 3 9 1 0 2 9
 2 4 9 3 1 0 2 9
 1 9 9 3 2 2 2 1
 2 3 2 3 9 9 2 1
 9 1 0 3 9 4 2 9
 1 1 0 3 9 3 2 2
 0 0 0 2 9 2 1 9
```

```
2
--*-----*--*---*
--*----*-**-----
----**--*---*--*
----*-------*--*
**-----*---***--
*----*----*---*-
*--**---------**
--*--*-*------**
------***-------
---*-*-----***--
*---*----------*
---*-----*-----*
 0 2 9 2 0 0 1 2 9 3 3 9 1 0 1 9
 0 2 9 3 2 2 2 9 4 9 9 3 2 1 2 2
 0 1 1 3 9 9 2 2 9 3 2 3 9 2 2 9
 2 2 1 2 9 3 2 2 2 1 1 4 9 4 3 9
 9 9 1 1 2 2 2 9 1 1 2 9 9 9 3 2
 9 4 2 2 3 9 2 1 1 1 9 3 3 4 9 3
 9 3 2 9 9 3 3 1 1 1 1 1 0 3 9 9
 1 2 9 3 3 9 4 9 3 1 0 0 0 2 9 9
 0 1 2 2 3 3 9 9 9 1 1 2 3 3 3 2
 1 1 1 9 3 9 3 3 2 1 1 9 9 9 2 1
 9 1 2 3 9 2 1 0 1 1 2 2 3 2 3 9
 1 1 1 9 2 1 0 0 1 9 1 0 0 0 2 9
```