Softwareschneiderei

Minesweeper rules

Your task will be to develop the classic game of minesweeper.

Version 1:

You have a rectangular game field with cells. Each cell may contain a mine. All cells are initially "covered". The task of the player is to uncover ("reveal") each cell that doesn't contain a mine. If the player has revealed all cells without mine, he wins the game. If the player reveals a cell that contains a mine, he loses immediately.
Everytime the player reveals a cell without a mine, the cell displays the amount of mines of directly adjacent cells. The minimum amount is 0, the maximum amount is 8 for every neighbouring cell. The amount is printed as a number in the cell, except for 0. In this case, the cell is printed empty and all adjacent cells that don't contain a mine are automatically revealed, too.
In this version of the game, there is no functionality to "mark" or "tag" a cell as a reminder or for reference. The only operation a player can perform is to reveal a covered cell.

A starting point for field size and mine count would be 10x10 cells and 10 randomly placed mines.

Example for amounts and auto-reveal:
The 3x3 field in our example contains two mines at the positions (0, 1) and (2, 0):

```
.*.
...
*..
```

At start, everything is covered:

```
###
###
###
```

The player chooses to reveal the topright position (0, 2). It is revealed as 1 because of the mine directly left to it:

```
##1
###
###
```

Then, the players reveals the bottom-right position at (2, 2). Because its amount of adjacent mines is 0, it is revealed empty and the neighbouring cells are automatically revealed, too:

```
##1
#21
#1.
```

Version 2 (optional):

The player can now not only reveal, but also mark a cell. If a cell is marked, it isn't revealed, but assumed to contain a mine. There is a "mine countdown" displaying the amount of remaining unmarked mines.

Version 3 (for very advanced developers only):

Not that you have an implementation of minesweeper that can reveal and mark cells and calculate the winning and losing conditions, your next task is to implement an algorithm that plays (and hopefully wins) your minesweeper automatically.

Object Calisthenics - Die Regeln

1. Pro Methode nur eine Einrücktiefe
2. Kein else
3. Kapsele alle primitiven Datentypen und Strings
4. Nur ein Punkt (Methodenaufruf) pro Zeile
5. Keine Abkürzungen und trotzdem kurze Namen
6. Kleine Entitäten (50/10-Regel)
7. Maximal zwei Instanzvariablen pro Klasse
8. Collections nur alleinstehend verwenden
9. Keine Getter, Setter und verwandte Konstrukte (z.B. public *Fields*)