Dukosi Coding Challenge - Minesweeper

# Minesweeper Background and Rules of the Game

Minesweeper is a graphical game where the player has to hunt and identify all the mines on a grid made up of squares.

The computer allocates a fixed number of mines to squares within the grid at the start of a game. So each square contains 1 mine or no mines. Whether a square contains a mine is hidden from the player.

The player is told how many mines are on the grid. The player’s objective is to clear all squares on the grid that contain a mine, leaving only those squares that contain a mine un-cleared.

The player can clear a square on the grid. If there was a mine at that square then the player has lost the game and the game is over. If there was no mine at the square then the number of mines in adjacent square is displayed in the square.

The adjacent squares are the 8 squares found immediately to the left, right, above, and below a location. For square X in the diagram, the 8 adjacent squares are highlighted in blue.

A picture containing shoji, crossword puzzle, building

Description automatically generated

The player may flag a square where they believe there to be a mine. This is for the players own information only and it has no effect on whether the game is won or lost.

The player wins the game when all squares are cleared except for those squares that contain a mine.

# Example Game Outcomes

The player has lost. They cleared a square containing a mine.

Graphical user interface, application, Teams

Description automatically generated

The player has won. They cleared all squares except for those that contained a mine. (They have also flagged all squares that do not contain a mine, however that is not necessary to win with our minesweeper rules.)

A screenshot of a phone

Description automatically generated with low confidence

# The Task

Your task is to create a module in C that is used to manage the state of a game of minesweeper and produce a test application (also in C) to test your module.

Such a module could be integrated into an GUI application that displays the state of the game and allows a player to make moves. You should not implement the graphical side of the application.

Your module should

* A allow a new game to be started and set up the state of the game
* Expose functions that allow a player to take the actions permitted by the rules of the game and update the game state accordingly
* Expose the game state in a way that would allow a GUI to use this module to draw the game
* Indicate whether the game is won or lost
* Include suitable data items and data structures to achieve the above.

Most implementations of minesweeper recursively clear squares adjacent to each cleared square that contains no mine. Your implementation does not need to do this.

## Tests

Write sufficient tests to thoroughly test the module. Your test application should work through the test cases and report test results.

## Development Environment

Use a development environment and toolchain of your choice. Include short instructions that specify the environment/toolchain used and how to build, as well as expected output when the application is run.

Use of the standard C libraries is allowed, and this will be useful for outputting test results.

## Assumptions

If you find you need to make additional assumptions then document them as comments within the code.

1. The game uses a grid that has a fixed size of 8x8 squares.
2. There are 10 mines hidden on the grid, so in each game 10 squares will contain a mine
3. The locations of mines within the grid may be input to the module when a new game is started. So your module does not need to randomly allocate mines within the grid.

# Evaluation Criteria

The code and other files that make up your submission should be your best work.

It is up to your how long you wish to spend on the task before submission. We are all busy, so it is acceptable to leave out features or pieces of functionality provided that their absence is documented. Your submission will be evaluated on the parts that are present.

Criteria

1. Is the code functionally correct?
2. Is the code easy to follow and maintain?
3. Is the module easy to use? (If someone wanted to make a minesweeper GUI, would it be easy for them to integrate your module?)
4. Is the implementation efficient in use of resources? (compute power and memory)
5. Is the implementation portable?
6. Is the implementation secure?
7. Is the code well commented? (architecture and design should be clear from comments within the code)
8. Is the code consistent?
9. Is the code functionally complete?

# Submission

Collect your code plus any other files required to build your application, and any accompanying documentation in a zip file.

Then email to [fbain@dukosi.com](mailto:fbain@dukosi.com)

Good luck and happy coding :)