Initial thoughts

The task is to create a turn-based game, with a simple console interface no “UI”. With automated tests.

This is as a skilled demonstrator for a mobile development position.

Combining these points rase the greatest challenge with the given task.

Neave the engine or console app leaves much room for demonstrating SOLID principles beyond Single Responsibility. The lack of a UI removes any clear approach to use interfaces or dependency injection.

The best practice for mobile development is using the MVVM design patter and leans heavily on Task and event driven interactions with a heavy bios to dependency injection and interface driven interaction. None of which could easily be demonstrated in this task.

A UI would normal be created as part of a test harness for a task like this to both aid with and speed up manual testing and development.

From this I am concluding the task is to be completed with out a UI to force the developer to have to develop internally visualise the current board state during testing and level creation.

Design considerations

No information about game parameters is given beyond showing positions as chess moves.

We will default the board size to that of a chess board 8x8

To keep the game sensible but allow a good scope for implementation of the engine in different game difficulty at 50x50 seems like a good max size to accommodate what people may come up with

5 lives seems like reasonable initial default amount of lives for an 8x8

10 is a nice round number for the default number of mines for an 8x8

No starting position is specified so we will let the player chose the row in the first column to start from.

We will assume the game will end when the user first enters the last column.

The user will be able to repeat moving over the same square.

Hitting a mine in the last row will result in a win as long as it does not also cause the player to run out of lives.

Going over a square with a mine that has already cost the player a life will not result in the loss of a second life.

The console implementation will display simple data current move list, score, lives and accept only valid game inputs, the play can loop though multiple games until they choose to quit.

The game engine will need the ability to set up, save and load games of a known configuration to facilitate repeatable automated testing

An XUnit test project will be created to test the engine.

Tests will be created to show the though process for areas of coverage and not necessarily comprehensive cover all scenarios.

I will exclude full game play testing due to time constraints.

The area that requires the most testing in mobile applications is always deferent scenarios around UI interaction so with the given demonstration requirements there is way to demonstrate more than a simulation of Domain code testing so I will focus on that.