(После ще ги подредим)

1. Jagged array replaced by matrix, and made all code working with it.
2. Setters added in class Player for name and score
3. Validation and exception handling added in Player.cs
4. Use setters in the constructor in Player.cs due to validation
5. Validation and exception handling added in Field.cs
6. Loop in the constructor of Board.cs extracted in separate method PrepareMatrix(). Made code more testable.
7. Override Equals() for Field.cs, due to some unit testing
8. Override GetHashCode() for Field.cs, NOT IMPLEMENTED FOR NOW , because it works with Equals together.
9. Tests for PrepateMatrix() added
10. Tests for Equals() added
11. IPlayer interface improved (string Name{get;} and int Score{get;} added)
12. Made all fields in Game.cs non-static
13. Made code in Game.cs work with non-static methods (including keyword this)
14. Create constructor for Game.cs
15. Remove InitializeGameBoard method (not needed anymore)
16. Remove InitializeTopPlayers method (not needed anymore)
17. Create GameStarted abstract class – it only runs the game
18. Replace Main method from Game.cs in GameStarter.cs
19. Made Menu() method public, to be called from GameStarter.cs
20. Fields in Board.cs are no longer readonly
21. Properties Rows, Columns and MinesCount created in Board.cs.Exception handled.

VISITOR DESIGN PATTERN implementation started

1. Indexer created in Board.cs
2. IVisitor interface added (Started to implement Visitor Design Pattern)
3. Implement MineSetterVisitor class which implements IVisitor
4. Create Accept method in Board.cs
5. Delete SetMines() method from Board.cs.

VISITOR DESIGN PATTERN IMPLEMENTED

1. Menu() method renamed to Run();
2. In Board.cs fields renamed to fieldsMatrix (to be more descriptive)
3. Add IBoard interface (Board implements it)
4. IBoardManager interface created
5. IBoardScanner interface created (there is repeating logic, so it will be extracted in separate class implementing this interface)
6. BoardScanner.cs created.
7. Replace logic for ScanSurroundingFields from Board to the new BoardScanner class. It depends on the private IsMineInPosition() method, so it goes to the new class too.
8. BoardManager.cs created implemented IBoardManager
9. Renderer.cs created implementing IRenderer
10. Duplicated methods wich are extracted in separate classes are removed from Board.cs
11. Adapt Game.cs work with new classes