Overview and goal

The goal of this assignment is to code review, redesign, and reimplement a Tic Tac Toe game, according to the model-view-controller (MVC) architecture pattern.

This quick-and-dirty implementation satisfies some best practices but violates other best practices. It needs a major architecture overhaul. In contrast to the current version, your implementation should try to improve understandability, extensibility, and testability.

How to get started

1. Clone the repository with the following command: git clone -b hw1 https://github.com/LASER-UMASS/cs520
2. Read the provided README in the tictactoe folder.
3. Use the commands to document, compile, test, and run the application from that folder.
4. Familiarize yourself with the original application source code contained in the `src` folder: `src/TicTacToeGame.java` and `src/TicTacToeBlock.java`.

**Code review [32 points]**

You are expected to code review the original version of the application focusing on the expected behavior (e.g., extensibility, testability) instead of coding style (e.g., amount of whitespace, if vs switch statement).

In particular, you need to identify 4 cases where best practices are satisfied. Your code review should use the following pattern **for each identified satisfaction of best practices:**

- Brief summary of the design principle or best programming practice (a few keywords)
- An illustrative example from the code

Additionally, you need to identify 4 cases where best practices are violated. Your code review should then use the following pattern **for each identified violation of best practices:**

- Brief summary of this issue (a few keywords)
- Explanation of the issue (could refer to general principles or poor design choices with respect to the desired encapsulation, extensibility, and testability)
- How to fix it (a few sentences)

In the second homework, 4 identified violations WILL need to be implemented.

**MVC architecture pattern [12 points]**

![MVC architecture diagram]

*Figure 2: Main components of the ‘TicTacToe’ UI*

From the game’s perspective, identify the following in its UI:
- Component A: View, Controller, or both? Briefly explain what is being visualized and/or what user interaction is being provided.

- Component B: View, Controller, or both? Briefly explain what is being visualized and/or what user interaction is being provided.

- Component C: View, Controller, or both? Briefly explain what is being visualized and/or what user interaction is being provided.

Identify the original application source code (including all classes, fields, and methods) corresponding to the:

- Model

- One view (from the 3 components listed above)

- One controller (from the 3 components listed above)

**Proposed extension [8 points]**

You should provide a description of how to extend the game board model to parameterize its size. In other words, the game board should be generalized from 3X3 to nXm where n and m are both greater than or equal to 3 but n and m do not need to be equal to each other. For simplicity, you can assume the same winning condition, meaning the r in a row where r is currently set to 3, is being used even though that winning condition highly favors the first player.

Your description should identify the set of fields and methods that need to be changed to support the extension. For each identified field or method, you should briefly describe the necessary changes to support the extension.

**Deliverables [2 points]**

Your submission, via [Gradescope](#), must be a single document (PDF) named *hw1.pdf*, containing:

1. Your full name and any collaborators (both at the top)
2. Code review comments
3. Identification of the MVC architecture pattern in the UI and the application code
4. Proposed extension