CS 520
Homework 1
Code review and Model-View-Controller in action

Due: **Tuesday, October 16, 2018, 9:00 AM EDT** via Moodle. You may work with others on this assignment but each student must submit **their own, individual write up and code**, clearly specifying the collaborators. The write ups and code should be individual, not created jointly, and written by the student on their own. Late assignments will not be accepted without **prior** permission.

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) architectural pattern.

The following repository provides a basic implementation of the Tic Tac Toe game:

`https://github.com/LASER-UMASS/cs520`

This quick-and-dirty implementation violates many best practices and needs a major design overhaul.

In contrast to the current version, your implementation should support a possible extensions without violating the open/closed principle. Additionally, your implementation should enable better testability so that the individual components can be tested in isolation.

Code review

You are expected to code-review the current version of the application. In particular, you are expected to **identify four (4) violations of best practices**. For each violation, briefly explain what the issue is (you may refer to general principles or poor design choices with respect to the desired extensibility) and how to fix it. Your code review should follow the following pattern **for each identified violation**:

- Brief summary (a few keywords)
- What's the issue (one or two sentences)
- How to fix it (one or two sentences)

Your reimplementation should reflect and implement your code review suggestions.

Implementation

Desired extensibility

Your implementation must implement the rules of tic tac toe. It must, in that sense, be a correct implementation of the rules.

Your design needs to support the following extensions:

- Using a different visualization (View) for the game board without changing the model or the controllers.
- Using a different data representation (Model) without changing the view or the controllers.
- Reading game moves from other sources (e.g., reading from the command line).

You **do not need to implement these extensions**, but your design needs to be flexible enough to support them. Also, your design does not need to support different GUI frameworks (only supporting one GUI framework of your choice is sufficient).

Make sure you use the following directory structure to organize your application source code and test files.

```
tictactoe/
     src/
          model/
          view/
          controller/
     test/
     build.xml
     README
```

Testing

Your design must allow testing of individual components. To show testability, you are expected to submit at least one test case per component (M, V, and C) along with your implementation.

Version control

You are expected to clone the existing repository and keep your implementation under version control, using the cloned repository. You will submit your repository to us, so you should make *coherent* and *atomic* commits, and use descriptive log messages.

How to get started

1. Clone the repository `https://github.com/LASER-UMASS/cs520`
2. Read the provided `README` in the `tictactoe` folder.
3. Compile, test, and run the application.
4. Familiarize yourself with the existing code (`src/TicTacToe.java`).

Feel free to use the provided `Ant` build file (`build.xml`) to compile and test your code. Alternatively, you can import the implementation into an IDE of your choice — most IDEs can import a project from an existing build file. However, neither `Ant` nor a particular IDE is a requirement. Additionally, you can use the provided example test class (`test/TestExample.java`) as a template to write your own test cases.

Deliverables

Your submission, via Moodle, must be a single archive (`.zip`, `.tar`, or `.tar.gz`) file, containing:

1. Your code review in a plain text file named as `hw1-codereview.txt`.
2. The `cs520` folder with all the updated source files and test cases of your application residing inside the `tictactoe` folder. Make sure that `.git` directory exists in `cs520` folder in which you committed your code. You can see your commits by running the `git log` command inside `cs520` folder.

   The repository should have a set of coherent commits showing your work, not a single version of the code.

3. A `README` file describing the **command line arguments to build, test, and run** your code from within the `tictactoe` folder. You can use `Ant` or other build tools, but the `README` needs to explicitly say the commands to run for each of the three actions (refer existing `README` provided in the `tictactoe` folder).

Your application is expected to compile and run correctly for tic tac toe. Unless your application can be compiled and tested with the provided build file, please provide brief instructions for how to compile, test, and run your code within a `README` file that should exist inside the `tictactoe` folder.