Project Sprint #5

The main tasks of this assignment are:

(1) Adding the feature of recording a game into a text file. The user story and acceptance criteria of both record and replay are required, but the implementation of replay is for extra credit (up to 2 points in the weighted total).
(2) Conducting a code review exercise.
(3) Summarizing the lessons learned from Sprint 0 through Sprint 5.

The following is a sample GUI layout of the final product, where “Replay” is optional.

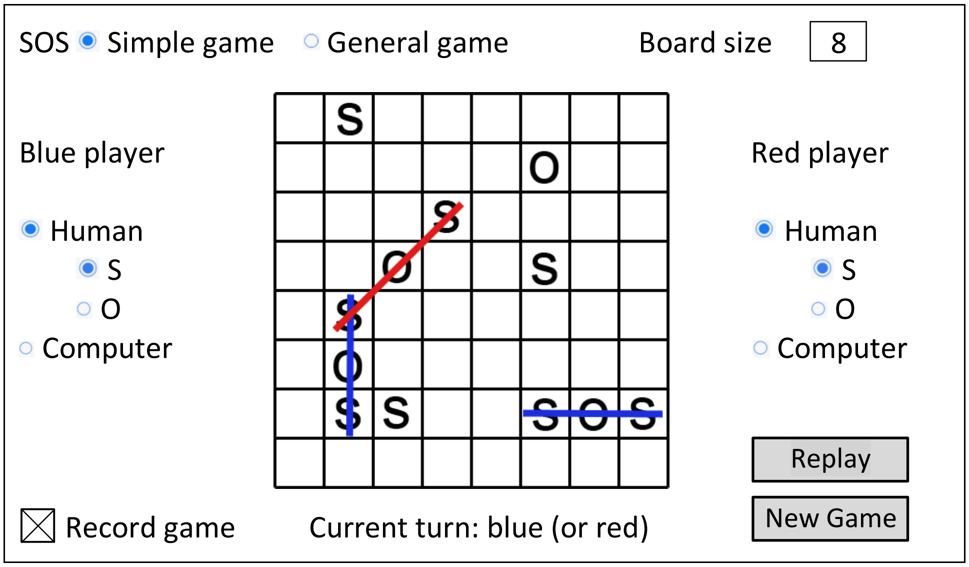

Figure 1. Sample GUI layout of the final product

1. Demonstration (9 points)

Submit a video of no more than 8 minutes, clearly demonstrating that you have implemented all the features in the following table. In the video, you must explain what is being demonstrated.

	Feature
1	A complete simple game of two human players is recorded
2	A complete general game of two human players is recorded
3	A complete simple game of human-computer players is recorded
4	A complete general game of human-computer players is recorded
5	A complete simple game of computer-computer players is recorded
6	A complete general game of computer-computer players is recorded

If you have implemented the “replay” feature for extra credit, you should include its demonstration in the video.

2. User Stories and Acceptance Criteria for the Record/Replay Requirements (3 points)

- **User Story Template**: As a <role>, I want <goal> [so that <benefit>]

ID	User Story Name	User Story Description	Priority	Estimated effort (hours)

1	Human-human recorded game	As a human player I want to play a simple and general game with another human and wants the game to be recorded so I can see the moves	Medium	5
2	Human-computer recorded game	As a human player I want to play a simple and general game with computer and wants the game to be recorded so I can see the moves	Medium	7
3	Computer-computer recorded game	As a blue computer player, I want to play a simple and general game with red computer player and wants the game to be recorded so I can see the moves	Medium	7

User Story ID and Name	**AC ID**	**Description of Acceptance Criterion**	**Status (completed, toDo, inPprogress)**
1 story one	1.1	AC 1.1 <human-human player recorded> Given As a player I chose human player to play with When I choose another player and we make a move Then it gets recorded into a text file	Completed
2 story two	2.1	AC 2.1 <human-computer player recorded> Given As a player I chose another player as computer When I choose another player and we make a move Then it gets recorded into a text file	Completed
3 story three	3.1	AC 3.1 <computer-computer player recorded> Given As a player chooses computer on both blue and red side When chooses the player type and make a move Then it gets recorded into a text file	Completed

3. Code Review (2 points)

Apply source code review to one or two most important classes (and other classes if time permits) and report the findings. In addition to looking for bugs, the review should check: (1) whether the entire project has followed the coding standard in a consistent manner, (2) whether the project has followed the design principles introduced in class, and (3) whether there are code smells that indicate the need for refactoring. The following checklists provide basic guidelines. You may add new items to each of the checklists. Make sure your answers resulted from the code review exercise. If there is no finding for an entry, you should provide an explanation. For example, if your answer to "Are the naming conventions violated?" is no, you should describe a naming convention and present an example. You will receive no credit for this part if your answers are simply yes or no without additional information.

Classes that have been reviewed:	Game.java , GUI.java
Date/time duration of the code review exercise:	12/9/2021 2:28PM

Checklist	**Checklist Item**	**Findings**
Coding Standards	Are the naming conventions violated?	Yes at some places, there were time when two variables were quite similar to one another and mistakenly got implemented in wrong places because of similarity. They were'nt as clear as TOTALROWS vs. getTotalRows()
	Is the ordering convention of method arguments violated?	No, he method arguments have been consistent with the ordering convention
	Any comments meaningless or inconsistent with the code?	No, I tried my best to make comments that are easy to follow for whoever is reading to refactoring the code
	Any code block has an inconsistent formatting style?	Most of the code is well formatted , there are some blocks in gui class where either spaces or a tabs are uneven
	Any indentations inconsistent?	For the vertical alignment of the code, consistent indentations are applied for the reader's comfort

Design Principles	Any class/method not well-modularized?	Yes, a method with randomMove() is not independent and the functionality isn’t emphasized separately
	Any class with poor abstraction?	No, classes are well abstracted
	Is the visibility of any variable, method, and class inappropriate?	Yes, there are some variables and rare methods which are static and private which could’ve been used in different class and help lessen the code length because of their public visibility
	Is design by contract (pre/post-condition) violated?	No, the design is well aligned with pre-post conditions and not violated
	Is the Open-Closed Principle violated?	Considering all the features that are required for the game, the open-closed principle isn’t violated but incase we decide to insert a new feature, it wouldn’t take long to violate open-closed principle
	Is the Single Responsibility Principle violated?	No, the classes that are used for the purpose initially defined are only used for that one reason, and can’t be altered
Code Smells	Are there magic numbers?	Yes, but the main integer variable n to get the board size can be changed by user input
	Are there unnecessary global / class variable?	No, most variables that are public are necessarily public on intent to use in other classes as well
	Is there duplicate code?	Yes, game and gui has some code that are duplicated so that code can run well and to avoid where inheritance could cause complication
	Are there long methods?	Yes, gui has long methods like actionListerner() , Gui(), etc
	Is there any long parameter list?	Yes, Game.java has search() where there are five parameters
	Is there over-complex expression?	No, the expressions used throughout the code are well expressed and easy to read
	Is there switch or if-then-else that needs to be replaced with polymorphism	There might be, it is not conspicuous to me which one can be replaced with polymorphism
	Any variable or method name whose intent is unclear?	No. there is a method getCell() – doesn’t really tell completely what is meant by the method name
	Any similar methods in different classes?	Yes there are some methods that are used in both classes which are similar – where inheritance could’ve been worked but not being able to make the code worked and to meet deadlines I decided to go with similar methods for example JRadioButtons.

4. **Summary of All Source Code (1 points)**

Source code file name	Production code or test code?	# lines of code
GUI.java	Production	542
Game.java	Production	200
Auto.java	Production	133
GameFileManager.java	Production	40
GameFileWriter.java	Production	17
TestBlueMoves.java	Test	63
TestGameGUI.java	Test	52
TestRandomMoves.java	Test	92
TestEmptyBoard.java	Test	43
TestCompleteGame.java	Test	76
	Total lines of code	1258

You will receive no credit for this assignment unless your complete source code is submitted.

5. Summarize the lessons learned from the entire project by answering the following questions from the perspectives of development processes, coding, design, refactoring, and testing **(5 points)**:

Minimum requirement: One full page single spaced, font size no bigger than 12 points.

- What did you personally gain from the project?

There were many things I learned and realized when working on the project – Personally I gained the two main insights are as I would call them – **Character qualities** and **Technical skill qualities**. In my opinion, the combination of both is quite necessary to be effective.

The character qualities I'm talking about here are Consistency - when working on the project, instead of doing everything at once, doing a little bit but doing it daily. Discipline – to show up to do the work to meet the deadlines. Persistency – when the project doesn't seem like it's going in right direction or not working, being persistent through the difficulty and not giving up. Humble – when the code isn't working, asking for help can seem like you don't know enough or unable to solve something simple but being humble and asking for help can ease a lot of unnecessary stress and aid you in your movement towards the goal. The project required patience, being persistent when things don't work, and there were times where I couldn't deliver the sprint on time which taught me not let one defeat define the whole process.

The technical skills - To begin with, this has been a great exercise and self-evaluation of my competence as a programmer. This project was the first thing that I have interacted with that showed what it means to be working in outside world. Throughout the sprints, even though we had given the details initially what we are trying to implement (user stories and acceptance criteria) , there were times when new requirements were given like – n cannot be a fixed number –adapting to the new requirements required the refactoring of code sometime not as much and sometimes half of the total code. I learned how the sprint process works, how sometimes requirements can cause delay in sprint deadline, how you may need refactoring of code, Project has been helpful in many technical skills that I will need in career and also taught me character skills that I need to embody to be a good programmer.

- What does your project do well, and what could your project do better?

The project did well when to comes to meeting and implementing the requirements – user stories and acceptance criteria. The program came out with an attractive interface and easy to select and use the features needed to play the game – like selecting game mode, payer options, S or O, the colors, etc. There maybe be things that could have been included in the project to make it better but so far as my knowledge goes, whatever I thought could make the project better – I implemented, maybe through practice I'll be introduced to new ideas and new ways of doing things and then I can think of ways by which the project could do better.

- How could you improve your development process if you develop a similar game from scratch?

Since from beginning - working on this project I made a lot of mistakes and realized many of them which I learned and fixed later. I think it wouldn't be difficult to do a better job next time I'm working on such project. Obviously, experience and practice that I gained from this project and practicing different ways to implementing features made me better logical thinker. The next time I'm developing a similar game or a similar project – I would keep a distant vision in my mind as in what the project will need in the future sprints, so in practice that will look like writing foundational code which can be reliable for the upcoming requirements or so that if the requirements change, I don't have to refactor the whole code (which can be time-costly and cause delays). One hurdle I faced when working on this project was that I had to refactor my code in-between the sprints so for a week I devoted the complete time on just writing code (skipping daily routine like working out, sleeping well) – I thought I was working hard on the project and making progress but since I was getting no exercise and no rest, after working on it for a week I was burnt out completely. Even though it is not a technical developmental process adjustment, I think marinating some balance between work-life will help the developmental process tremendously.