**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.

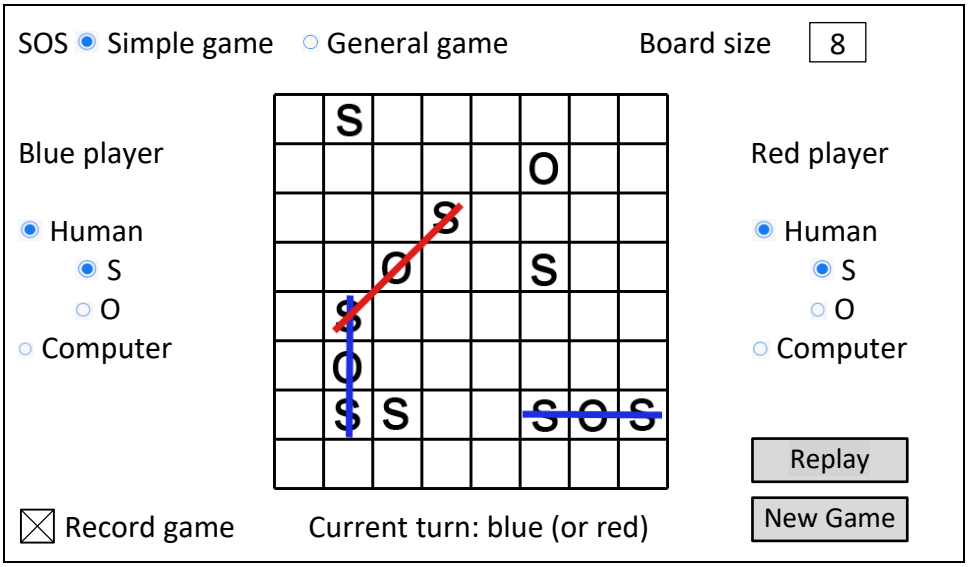


Figure 1. Sample GUI layout of the final product

GITHUB REPO: https://github.com/Tim-DeGraffenreid/CS449-Software-Engineering.git

**Total points: 16**

## 1. Demonstration (6 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 (1 points)

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

Add or delete rows as needed.

| ID | User Story Name | User Story Description | Priority | Estimated effort (hours) |
|---|---|---|---|---|
| 10 | Record Game | As an SOS player, I want to record my game, so that I can keep a history of my game. | 1 | 2 |
| 11 | Replay Game | As an SOS player, I want to replay my game, so that I can see a game replayed. | 3 | 5 |

| User Story ID and Name | AC ID | Description of Acceptance Criterion | Status (completed, toDo, inPprogress) |
|---|---|---|---|
| 10. Record Game | 10.1 | AC 10.1 A simple game between two human players<br>Given A simple game mode and valid board size are chosen<br>When A simple player selects 'Record Game'<br>Then The simple game is recorded to a txt file | completed |
| | 10.2 | AC 10.2 A general game between two human players<br>Given A general game mode and valid board size are chosen<br>When A general player selects 'Record Game'<br>Then The general game recorded to a txt file | completed |
| | 10.3 | AC 10.3 A simple game between a human and computer<br>Given A simple game mode and valid board size are chosen<br>When A simple player selects 'Record Game'<br>Then The simple game between human and computer is recorded to a txt file | completed |
| | 10.4 | AC 10.4 A general game between a human and computer<br>Given A general game mode and valid board size are chosen<br>When A general player selects 'Record Game'<br>Then The general game between human and computer is recorded to a txt file | completed |
| | 10.5 | AC 10.5 A simple game between a computer and computer<br>Given A simple game mode and valid board size are chosen<br>When A simple player selects 'Record Game'<br>Then The simple game between computer and computer is recorded to a txt file | completed |
| | 10.6 | AC 10.6 A general game between a computer and computer<br>Given A general game mode and valid board size are chosen<br>When A general player selects 'Record Game'<br>Then The general game between computer and computer is recorded to a txt file | completed |
| 11. Replay Game | 11.1 | AC 11.1 Replaying a simple SOS game<br>Given A simple SOS game has been recorded.<br>When A simple player clicks 'Replay Game'<br>Then A simple game is replayed | toDo |
| | 11.2 | AC 11.2 Replaying a general SOS game<br>Given A general SOS game has been recorded.<br>When A general player clicks 'Replay Game'<br>Then A general game is replayed | toDo |

## 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:
Date/time duration of the code review exercise:

| Checklist | Checklist Item | Findings | |
|---|---|---|---|
| Coding Standards | Are the naming conventions violated? | Yes, class names are UpperCamelCase, and method names and variable names are in lowerCamelCase | |
| | Is the ordering convention of method arguments violated? | All methods follow the ordering convention | |
| | Any comments meaningless or inconsistent with the code? | All comments have meaning and are consistent with what the code is doing | |
| | Any code block has an inconsistent formatting style? | All codes of block have a consistent formatting style | |
| | Any indentations inconsistent? | All indenting is consistent | |
| | … | | |
| Design Principles | Any class/method not well-modularized? | To my knowledge everything is well modularized | |
| | Any class with poor abstraction? | I could work on this skill. I feel GeneralBoard, GeneralComputerGame and SimpleComputerGame could be better in this respect. | |
| | Is the visibility of any variable, method, and class inappropriate? | Visibility of methods and class variables seem appropriate | |
| | Is design by contract (pre/post-condition) violated? | No contracts are violated | |
| | Is the Open-Closed Principle violated? | Classes are opened to be extended, but closed for modification | |
| | Is the Single Responsibility Principle violated? | It doesn't seem like it to me. | |
| Code Smells | Are there magic numbers? | Yes, an embarrassing amount of magic number. Sorry | |
| | Are there unnecessary global / class variable? | No, there isn't any unnecessary globals. | |
| | Is there duplicate code? | Unfortunately, yes, the board class and all of its subclasses duplicate the code in the makeMove() method. Additional code to handle the different requirements of the subclass are added after the duplicated code. | |
| | Are there long methods? | The isSOS() method is a little long, but I think its necessary to do what it needs to do. It could be refactored. | |
| | Is there any long parameter list? | Three is the longest parameter list. | |
| | Is there over-complex expression? | No expressions are overly complex | |
| | Is there switch or if-then-else that needs to be replaced with polymorphism | I think the if-then-else in the makeMove() could be replaced with polymorphism. | |
| | Any variable or method name whose intent is unclear? | No, they are all well named and clear as to what they are for. | |
| | Any similar methods in different classes? | Outside of the makeMove() method, there are no other similar methods. | |
| | | | |
| **Bugs** | **Buggy code snippet** | **What is the bug?** | **Why is it a bug?** |
| | public int isSOS(int x,int y){<br>int bound = this.getBoardSize() - 1;<br>if(y >= 2){//Straight up SOS<br>if(this.getCell(x, y)== 'S' && this.getCell(x, y-1) == 'O' && this.getCell(x, y-2)=='S'){<br>return 1;<br>}<br>} | Not all of the time, but sometimes I will get an out of bounds error because the index is out of range. | I wish I knew the exact reason. I spent hours and hours trying to fix it and it plagued me ever since it was implemented. I know it is related somehow to the fact that I had to transpose the x and y coordinates from the screen to indices of the array. |

```
        if((this.boardSize - y -1) >=
2){ //Straight down SOS
            if(this.getCell(x, y)== 'S' &&
this.getCell(x, y + 1) == 'O' &&
this.getCell(x, y + 2)=='S'){
                return 1;
            }
        }

        if(x >= 2){//Straight left SOS
            if(this.getCell(x, y)== 'S' &&
this.getCell(x - 1, y) == 'O' &&
this.getCell(x - 2, y)=='S'){
                return 1;
            }
        }

        if((this.boardSize - x - 1) >=
2){ //Straight right SOS
            if(this.getCell(x, y)== 'S' &&
this.getCell(x + 1, y) == 'O' &&
this.getCell(x + 2, y)=='S'){
                return 1;
            }
        }

        if(y >= 1 &&((this.boardSize - y -
1) >= 1)){//Straight up & down SOS O
            if(this.getCell(x, y)== 'O' &&
this.getCell(x, y-1) == 'S' &&
this.getCell(x, y+1)=='S'){
                return 1;
            }
        }


        if(x >= 1 && ((this.boardSize - x -
1) >= 1)){//Straight left and right SOS
O
            if(this.getCell(x, y)== 'O' &&
this.getCell(x - 1, y) == 'S' &&
this.getCell(x +1, y)=='S'){
                return 1;
            }
        }


        if(x >= 2 && y >= 2){//diagonal
left upper
            if(this.getCell(x, y)== 'S' &&
this.getCell(x - 1, y - 1) == 'O' &&
this.getCell(x - 2, y - 2)=='S'){
                return 1;
            }
        }

        if(x >= 2 && (this.boardSize - y -
1) >= 2){//diagonal left lower
            if(this.getCell(x, y)== 'S' &&
this.getCell(x - 1, y + 1) == 'O' &&
this.getCell(x - 2, y + 2)=='S'){
```

```
                return 1;
            }
        }

        if((this.boardSize - x -1) >= 2 &&
y >= 2){//diagonal right upper
            if(this.getCell(x, y)== 'S' &&
this.getCell(x + 1, y - 1) == 'O' &&
this.getCell(x + 2, y - 2)=='S'){
                return 1;
            }
        }

        if((this.boardSize - x -1) >= 2 &&
(this.boardSize - y -1) >= 2){//diagonal
right lower
            if(this.getCell(x, y)== 'S' &&
this.getCell(x + 1, y + 1) == 'O' &&
this.getCell(x + 2, y + 2)=='S'){
                return 1;
            }
        }

        if(x >= 1 && y >= 1){//diagonal
minor O
            if(this.getCell(x, y)== 'O' &&
this.getCell(x - 1, y - 1) == 'S' &&
this.getCell(x + 1, y + 1)=='S'){
                return 1;
            }
        }

        if((y >= 1) && (x <= 2) &&
(x >=1)){//diagonal  major O
            if(this.getCell(x, y)== 'O' &&
this.getCell(x + 1, y - 1) == 'S' &&
this.getCell(x - 1, y + 1)=='S'){
                return 1;
            }
        }

        return 0;
    }
```

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

| Source code file name | Production code or test code? | # lines of code |
|---|---|---|
| Board.java | Production | 299 |
| GameFrame.java | Production | 620 |
| GeneralBoard.java | Production | 73 |
| GeneralComputerGame.java | Production | 95 |
| SimpleComputerGame.java | Production | 89 |
| BoardTest.java | Test | 379 |
| | Total lines of code | |

**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 **(6 points)**:

- What did you personally gain from the project?
- What does your project do well, and what could your project do better?
- How could you improve your development process if you develop a similar game from scratch?

Minimum requirement for (5): One full page single spaced, font size no bigger than 12 points.

**What did I gain personally from this project?**

Wow, far more than I expected to gain. I gained an understanding of applying agile principles to the software development process and how important it is to building quality software. User stories and acceptance criteria were concepts I was previously unaware of, but after learning about them and implementing them in this project, I realized the importance of having clearly defined objectives and clearly defined solutions to those objectives. The value of fully understanding and defining the problem before even considering the solution was very powerful. Analyzing and systematically breaking down the problem into a set of smaller problems was something I thought I did before this project but, seeing how it is done professionally has opened my eyes to a whole new way of doing things. I gained a better understanding and appreciation for testing and how important it is to the software development process. I think what I gained more than anything was a realization of what I don't know about software development. Several times during this project and in class, I thought to myself, "wow, I don't know anything." I had setbacks and challenges that I never really had in a class before. On the one hand, that was a little discouraging and I even wondered if I even had what it took to make this my profession, but on the other it was motivational and made me want to try harder and work at making the processes and principles I learned second nature. I'm glad I bought the book instead of checking it out because I plan on referencing it a lot and implementing what I've learned in all of my personal projects to do just that.

**What does my project do well, and what could my project do better?**

Most of the projects I have done have been in Web Development and after each project, I always look at what I've done and thought I would have done just about everything better. I think I did the best in my base class for a simple game. The makeMove() method of my base class was probably my best nontrivial method. Frankly, I don't think I did much well in this project, but if I had to pick one thing that would be it. The inheritance and polymorphism, the important stuff, could have been done better. The code in the makeMove() method was repeated throughout every subclass with code added to accommodate each new subclass. I had never used Java before this class and thought I learned it fairly well in the first month, but I constantly found my knowledge lacking. I knew what I wanted to do, but there were things I just couldn't figure out how to implement in Java, so I worked around it with a bunch of smelly code. I could have, should have gained a better understanding of Java and GUI before I started the project. I think I cheated myself out of learning more about software engineering because I spent so much time playing catch up on Java code. The isSOS() method turned out to be a disaster for me and I tried to make it better on every sprint after it was first implemented, but it is not and never has been acceptable. I randomly get out of bounds errors when checking the board grid for an SOS. I think the problem stems from the fact that I transposed the x and y coordinates of the GUI board to match the 2D array so the 'S' and 'O' would be displayed correctly on the screen.

**How could I improve my development process if I developed a similar game from scratch?**

I would spend more time analyzing and understanding the problem at the beginning. Thinking and planning more carefully about how I was going to implement my classes and subclasses. Particularly how I would use inheritance and polymorphism to my advantage. I would gain more knowledge and understanding about the programming language I was using and how the things I am trying to use are implemented in that language. Also, I would not move on to the next spring until all my code was bug free and working exactly as it was supposed to. I learned the hard way moving forward when something isn't right or has a bug, only creates more

problems down the road. I would use more unit tests and I would test more broadly. I think I would also like to implement and explore Test Driven Development more thoroughly. Paired programming also seems an interesting and effective way to write better programs and I would like to see how somebody else would implement the same class or method. Having gone through the process of using Agile to develop this project would certainly allow me to improve the process in my next project.