Go

Instructions: Unless it has been done for you, print a copy of this document for your pair. As you work through it, write answers to any questions or prompts that are in **boldface**.

Your names:

Learning Objectives

Content Objectives

Parentheses below correspond to part of the knowledge units in the ACM's *Computer Science Curricula 2013*.

After completing this activity, students should be able to:

- Use classes and interfaces from the Java Collections Framework, including Stack and Set (SDF/Fundamental data structures).
- Use a stack to represent state history for undoing (SDF/Fundamental data structures).
- Debug using unit tests (SDF/Development methods).

Process Objectives

After completing this activity, students should have improved their ability to:

- Program in a pair. [Management]

Model 1: Playing the Game

Open the Go project in Eclipse. Run Go.jar to play the game. It's up to you whether you want to read the rules or play the game first, but do both. You should play a 5x5 game before moving up to larger board sizes.

Go, arguably the oldest strategy game in the world, was invented in China 3,000-5,000 years ago. It is known as *Weiqi* in China, *Igo* in Japan, and *Baduk* in Korea. While the rules are simpler than those of *Chess*, nobody has yet been able to write a computer program that can beat top human *Go* players.

Rules

Players: 2

Equipment: The board is a grid of intersecting lines. The standard board is 19x19, but no rules change for a smaller board. Each player has a collection of stones in their color.

Object: Control (occupy or surround) the most territory at the end of the game.

Setup: The board is initially empty.

Play: Black plays first, then play alternates.

On a turn, a player may either place a stone on a vacant intersection or pass.

A contiguous group of stones of one color are captured and taken off the board if there are no vacant intersections adjacent to any of them.

It is illegal to make a directly suicidal move (capturing one's own stones). If both your group and one or more of your opponent's groups would be captured by your move, only your opponent's groups are removed.

It is illegal to play a stone to make the board look exactly as it did on some previous turn, with the same color to play. (This is called the "ko" rule.)

The game ends when both players pass consecutively.

Each stone left on the board scores one point for its owner. Each vacant intersection that can only trace paths (through other vacant points) to the stones of one player is also worth a point for that player. White gets an extra 7.5 points to make up for the disadvantage of having to move second. The high score wins.

Note to experienced *Go* players: To keep the program simple, this program uses Chinese scoring. No attempt is made at the end of the game to remove "dead" stones; if you think you can capture something, do it!

1. **What does the program do if you try to make an illegal move?**

2. **Does the program detect the end of the game?**

3. **Does the program detect the winner?**

4. **What strategy advice would you give?**

Model 2: Debugging with JUnit

You have been given a complete program, but it contains some bugs. Specifically, some of the methods in GoModel.java do not quite work correctly.

You have been given a JUnit test class GoModelTestHighLevel.java. This test currently fails because of the bugs.

Write a new class, GoModelTest.java, testing each of the methods in GoModel.java. You may end up testing more than one method with a single test, or creating multiple tests for one method.

Find and eliminate the bugs so that GoModelTestHighLevel.java passes.

The number of bugs is more than one and less than ten. Each can be fixed by modifying a single line of code.

5. **How many bugs did you find?**

6. **Explain what each bug was (i.e., why the code you've been given caused a problem) and how you fixed it.**