# Go

Your names:

This programming project is to be done by a pair of students. (If your class has an odd number of students, you may have been assigned to a three-person “pair”.) Work together, discussing your current problem and how to solve it. Regularly trade off who is driving (actually editing the code) and who is navigating (making suggestions and consulting documentation).

By the time you are done with this activity, you and your pair should be able to:

- use classes and interfaces from the Java Collections Framework, including Stack and Set.
- use a stack to represent state history for undoing.
- debug using unit tests.

After you complete this activity, please fill out the short survey at

http://goo.gl/forms/HXjyuUb2ou

to improve this project for future users.

## Playing the game

Included in the Go project is a file Go.jar. This is a compiled version of the working game. To play it, use a terminal to navigate to the directory containing the file and type this on the command line:

```
java -jar Go.jar
```

*Go*, arguably the oldest strategy game in the world, was invented in China 3,000-5,000 years ago. It is known a *Weiqi* isn 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.

This is a two-player game. Play a couple of games within your pair. Board width 5 is fine for learning the rules. Larger boards make for more interesting but longer games; playing on the standard width 19 board can take over an hour.

Note to experience *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!

## 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.

Take notes as you work. (This is good work for the navigator.) How well are you working as a pair? What bugs and conceptual difficulties did you encounter? How did you overcome them? What did you learn?

After you're done, please fill out the survey at http://goo.gl/forms/HXjyuUb2ou.