EECS 293
Software Craftsmanship
2019 Fall Semester

Programming Assignment 6

Due at the beginning of your discussion session on

February 25-March 1, 2019

Reading

Read Sections 9.2, 9.3, 19.6 in Code Complete and the Quick Reference on Routine Names on canvas.

Grading Guidelines

Starting with Programming Assignment 6, an automatic C (or less) will be triggered by improperly named routines.

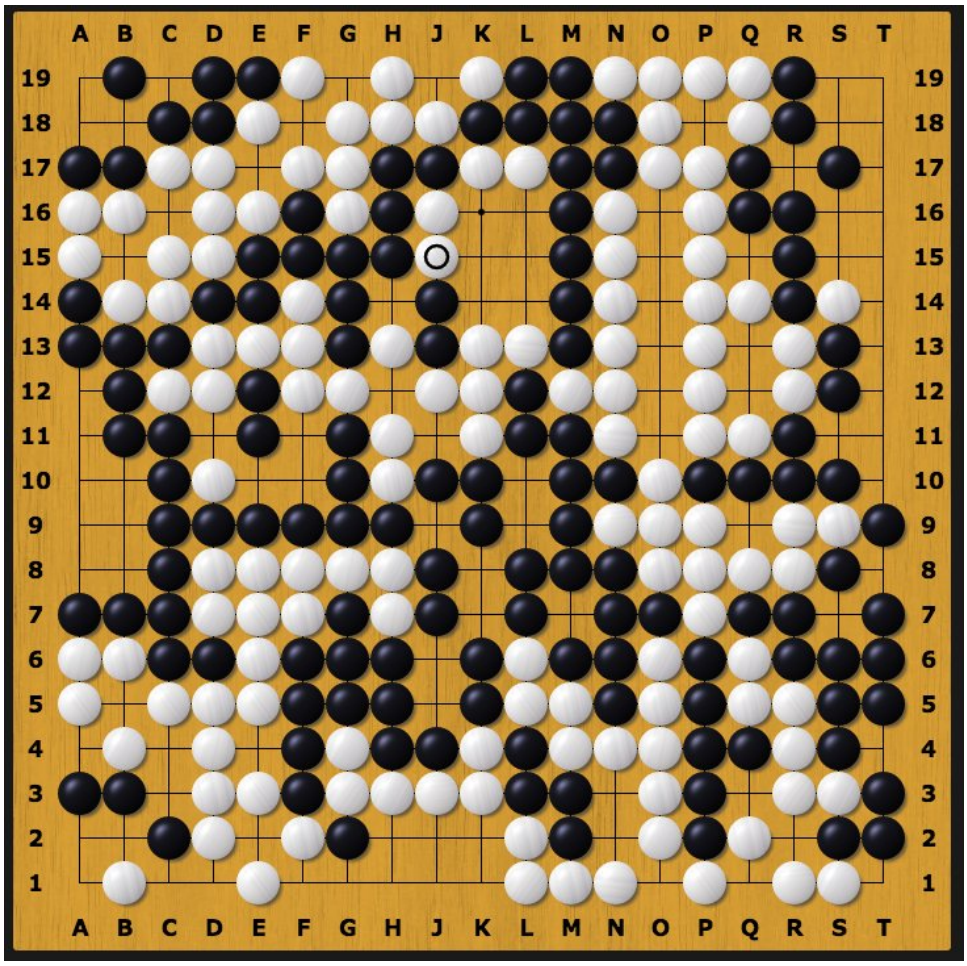
An automatic C (or less) is triggered by:

- Any routine with complexity greater than 4, or by
- Any piece of code that is essentially repeated.

Assignment

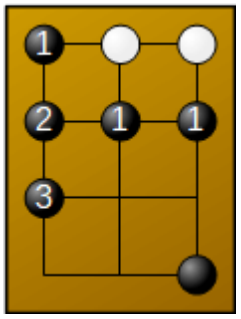
In this assignment, you will learn about the game of Gone and design an algorithm to compute the number of rounds in the game.

A game of Gone is played on a rectangular board. One player can place a white pebble at the intersection of a vertical or horizontal line, and similarly the other player can place a black pebble. No more than one pebble can be placed at any one spot, and not all legal spots are occupied (see figure).



Once the players have done setting up the board, the first player (white) starts replacing black pebbles with white pebbles according to the following rules:

- All black pebbles that are immediately adjacent to a white pebble are replaced. Adjacency matters only along horizontal and vertical lines (no diagonals).
- Repeat until no more pebbles can be replaced.



The picture shows an initial board configuration, and the round at which a black pebble has been replaced (if any).

Your algorithm takes as input a board configuration of arbitrary width and height, and it outputs:

- The number of iterations the replacement rules are applied, and

- A Boolean denoting whether any black pebble is left on the board.

In the small figure, the answer is that the algorithm takes three iterations and that there is a leftover.

Submit the pseudo-code of your algorithm. You should state the classes to which each method belongs and explain the abstraction that your classes capture. In the class discussion, you could be asked to justify the correctness of your algorithms or to analyze their running time. It is to your advantage to write down a justification for the algorithm's correctness, an analysis of its running time, and to sketch a few examples of the algorithm operation. It is your job to specify the format of the input and of the output.

No implementation is required: you will implement your pseudo-code in the next programming assignments. After this assignment, you will not be allowed to make major changes to the pseudo-code.

Canvas Resources

A module (Pseudocode Examples) contains a cheat sheet and five examples of pseudo-code from various textbook.

Group Assignment

Your discussion leader will randomly group the section into two student teams, most likely different from the previous teams. The team will be jointly responsible for Programming Assignments 6 through 8.

Discussion Guidelines

The class discussion will focus on the pseudo-code. The pseudo-code must be of sufficiently good quality that you can easily generate code from it in the next programming assignment. You may also be required to walk through your pseudo-code on some examples. It is better to present correct pseudo-code for a clearly specified subset of requirements rather than incorrect pseudo-code, or to address an unclearly stated subset of requirements.

Submission

Submit an electronic copy of your pseudo-code and any other ancillary documents on Canvas.