Nathan Gillette

For Project 3, I chose to create Peg Solitaire as I am familiar with the game. Since this is a single player game, I only had to solve for one set of moves. The game board is represented as a 7 x 7 array of integers, with 32 pegs and 1 open space in the middle. When the program is initialized, 2 console arguments are passed to the program. These represent the number of remaining pegs at the end of the run and the other is the maximum depth. I then check to see if the depth is deep enough to solve for the specified pegs, if not the program exits. But if the depth is valid, the program generates an ArrayList of possible moves which are then given to move evaluators which randomly try different moves until the correct number of pegs are left. Depending on the flags used, the program will then display the moves leading up to the final move in the console, and then the program will display a gui showing the final board state.

There were 2 main things that tripped me up in the development of this project. The first one being that I have to copy my board whenever I'd update it, or else the board instance would be lost and I'd end up with an empty board. The other problem I faced was that if the move evaluators weren't making progress on the pegs, I'd get stuck in an infinite loop. It was fairly easy to track this down, but I was stumped a bit trying to figure it out. It occurred to me that I needed to increase the depth so that there would be enough sets of moves to continue. I also did figure out perfbar and it is included in my screencaps.

**Links:**

**Screencaps**