Team 127 Code Review
Candidate A

The design of the code was intense, in the short time constraint it would have been difficult to completely understand the role of each of the classes in the file. It had several layers in the gui which made it different from the rest. The java docs were concise and did not say much about the code except for which part of the board correspond with the class. It did not express much about how the code worked step by step. Since there were so many classes it was difficult to read and follow. However, it had a fairly large amount of tests, so the code worked pretty well. It seemed to work more efficiently than ours, but the code was too unfamiliar to use.

Candidate B

We couldn't even figure out how to play the game. The gui didn't seem too interactive and the classes didn't say much about the code. The java docs were fairly weak. The code was hard to follow due to the fact that they put the bulk of their code in their keylistener. The java tests were also fairly weak. Overall the game was too complicated and seemed to buggy and difficult to use without proper instructions. The design was very difficult to follow to the point where we just jumped past it due to time constraints.

Candidate C

We decided to use this code due to the fact that it was fairly similar to our own code in the way the player movement is implemented. All the other codes seemed like it would take a long time to follow completely. There were not as many classes and the interface was simple. It had enough tests to convince us. There were a few bugs that would take no more than 15 minutes

to fix up like how the turns end, how the players can't pick up the token he or she is standing on at the start of the turn and how the code forced the players to require four players to play. Most of the code was hard coded so there were very few loops to understand. The java docs were very descriptive and helped us understand where the code is going piece by piece. The code organized itself to complete stage 3 already like blank methods and classes to give us a starting point. Overall the code was just close to ours so we decided to use it for ease of understanding.

Candidate D

When we tried the game it shifted the tiles seemingly fine, but gave us an error instantly and we could not move the players. The classes seemed to concise and we could not find the action listeners quickly. It seemed they mashed it into the buttons as they were making them in the gui class. With all their code so cramped up it was difficult to follow. However, they had a set up ready to implement the stage 3 components, so we did consider trying to read it. But, the difficulty of reading the GUI code was too bothersome. It had enough tests and the Java docs were fairly descriptive for the classes from stage one, but candidate c was still easier to follow than this one.

Candidate E

This code seemed the most promising, after we figured out the shift tiles were not the triangle shift keys but the tiles next to them. It had a ridiculous amount of test to complement the code and it seemed fully functional. It had a lot of java docs to explain the code. It seemed like the code with the least amount of problems and implementing the stage 3 requirements would be

a breeze. However, the code style was fairly unfamiliar and the code overall seemed massive for the project it did. Candidate 3 seemed similar to our code and simpler, so even though this candidate seemed really impressive, it was also very overwhelming.