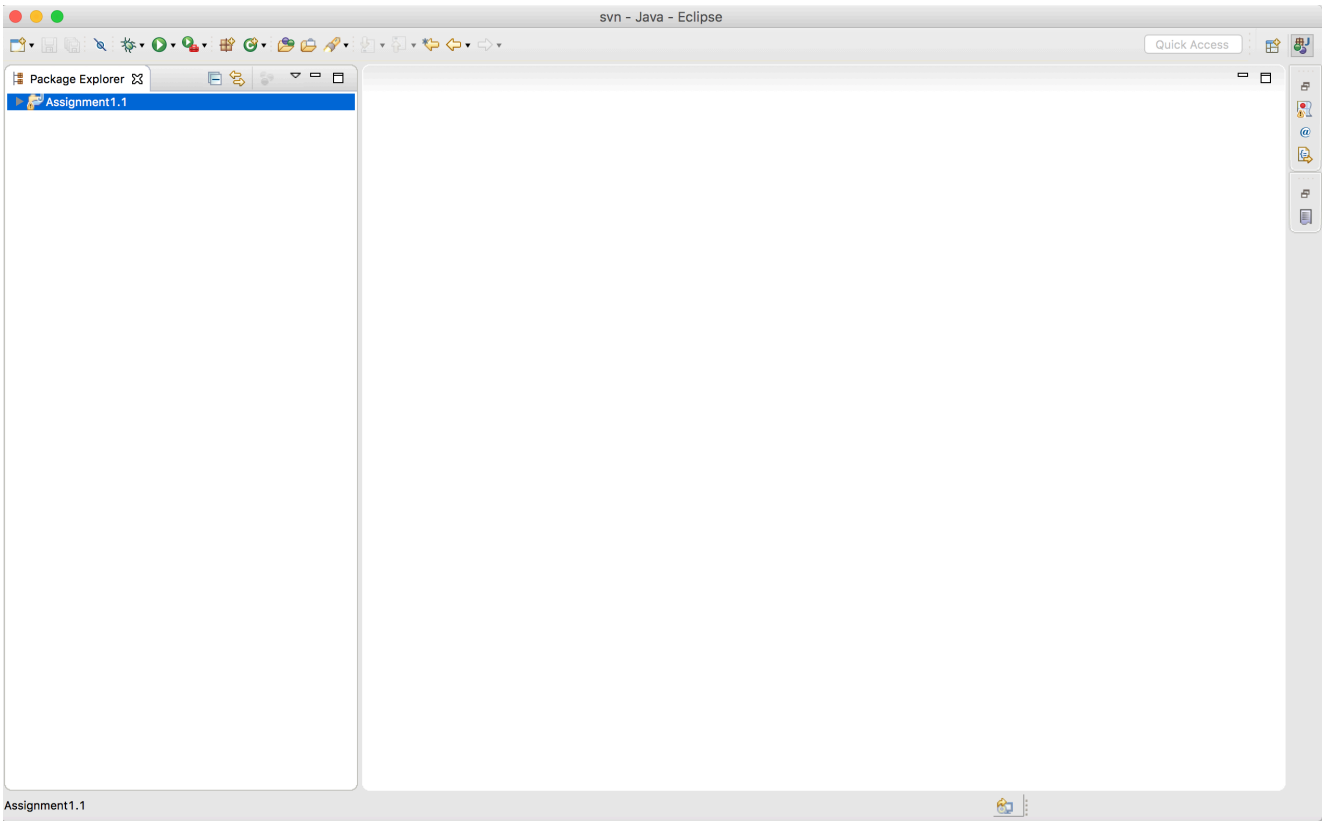
Arken Ibrahim

Manual Test Plan for Assignment1.2 (GUI)

Testing the Launch

1.) Begin by opening up the project, called “assignment1.1” in eclipse or any Java IDE would also work. After opening up the project in Eclipse, a screen should appear as such.

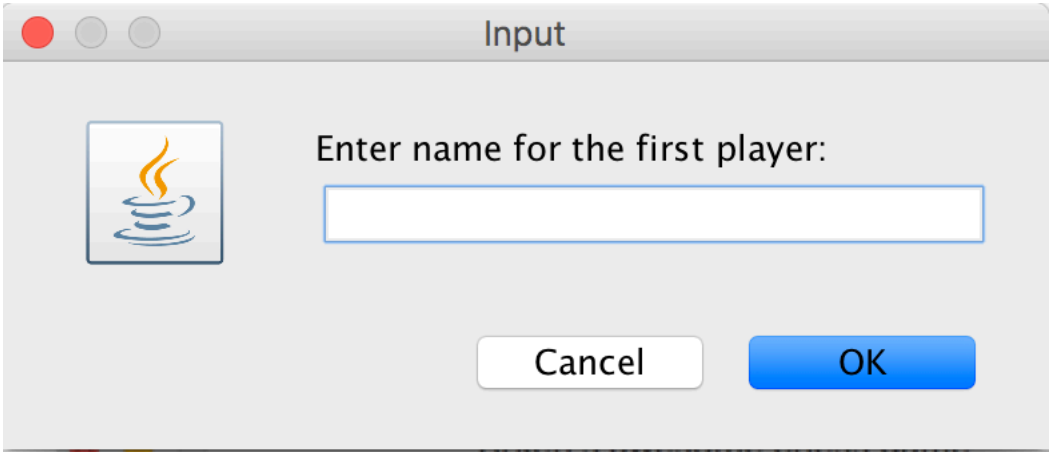


If a screen like this does not appear, than we know that there is some problem with eclipse launching up the project.

2.) Next step is to run the code to make sure that all the GUI components are working as expected. Therefore, in then top left hand side of the menu bar click the green “run” button as shown below:

This button should be present at all times. If the button is not in its proper location, than contact the Eclipse troubleshooting guide to ensure that your IDE is set up properly.

3.) After clicking the button shown in step 2, you should see two consecutive pop ups that ask for the players name. It should look like this.



4.) After you enter the names of the two players you should see a beautifully designed chess board appear in the center of your screen. Note that there might be slight differences between how the board looks on your desktop vs the screenshot due to different operating systems, etc. The board should look like this:

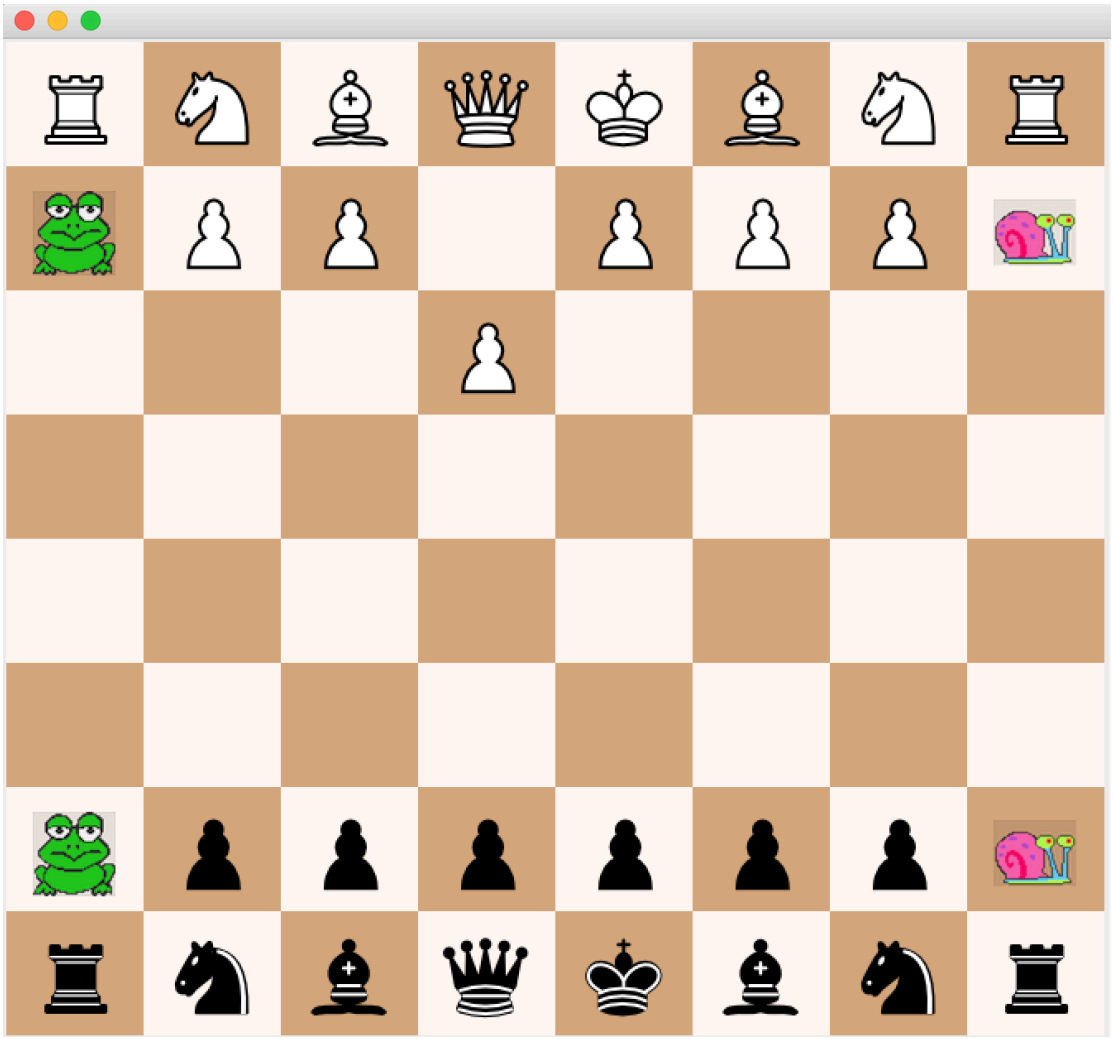
This chess board should be resizable. If you do not see this window, than there may have been some sort of an exception/error in the main 'Driver' class located in 'chess/Driver. The console should also display the relevant error messages in a red font in order to help you understand what may have gone wrong with the chess board launch.

5.) The top name on the side panel, in the image above, should be blue since he/she will always be the first player to go. (the blue means that it is your turn)

Arken Ibrahim

Testing Chess Piece Moves

1.) First of all, when you try to move one of YOUR pieces, you should see it move to the desired location. For instance If I choose to move one of my pawns forward, I the board should look like this.



2.) However, If you try to move a piece that does not belong to you, you should not be able to click that piece. If for some reason, you are able to move your opponent's pieces, than please check "chess/controller/controller.java" for any bugs.

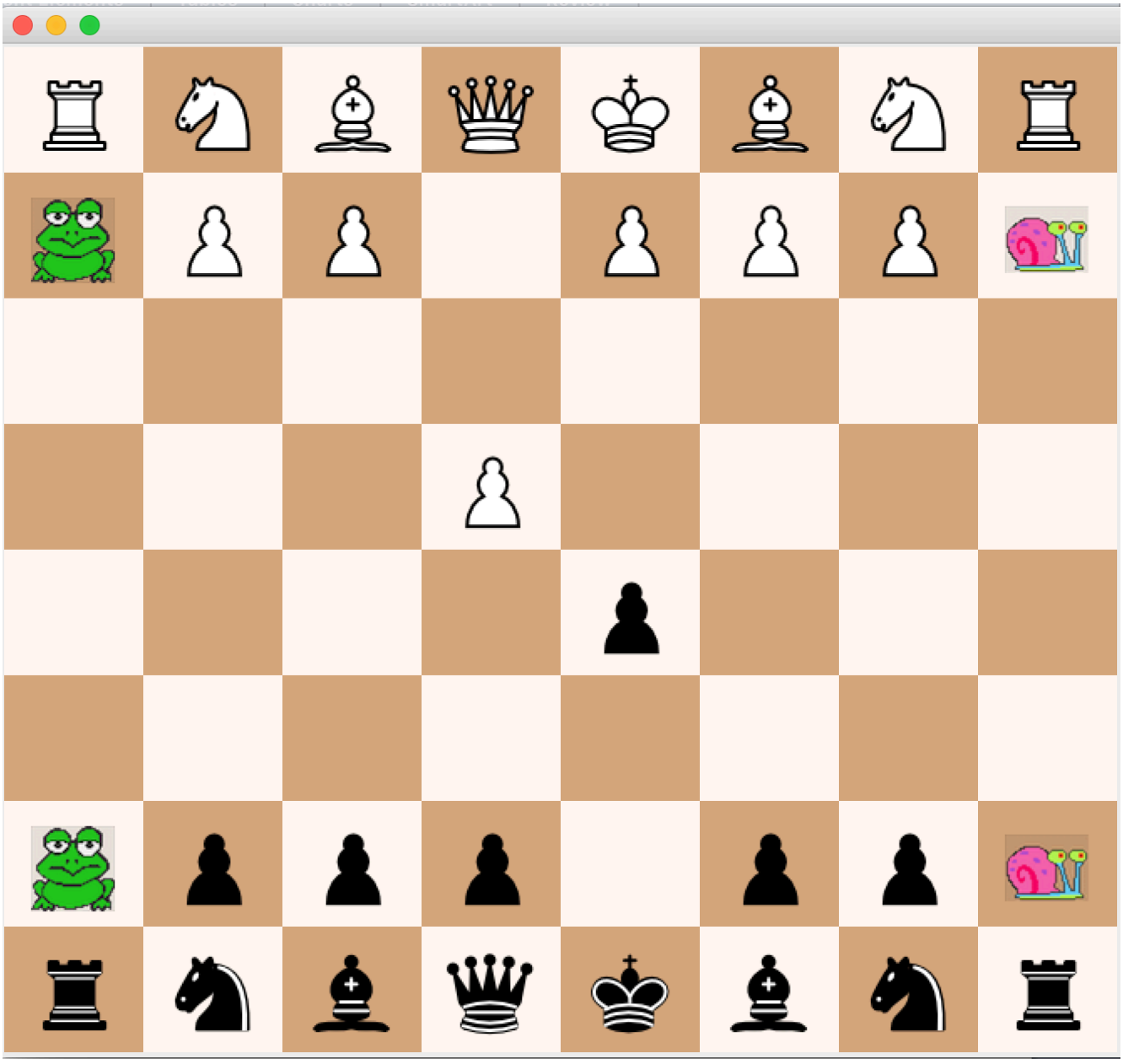
3.) When you try to move a piece to a spot that it should not be able to move to , you should receive an error warning from the system, explaining that you were not allowed to move to that spot. However, your name should still be highlighted in blue, because it should still be your turn after you attempt an illegal move. The screen should look

like this, after an attempted move.

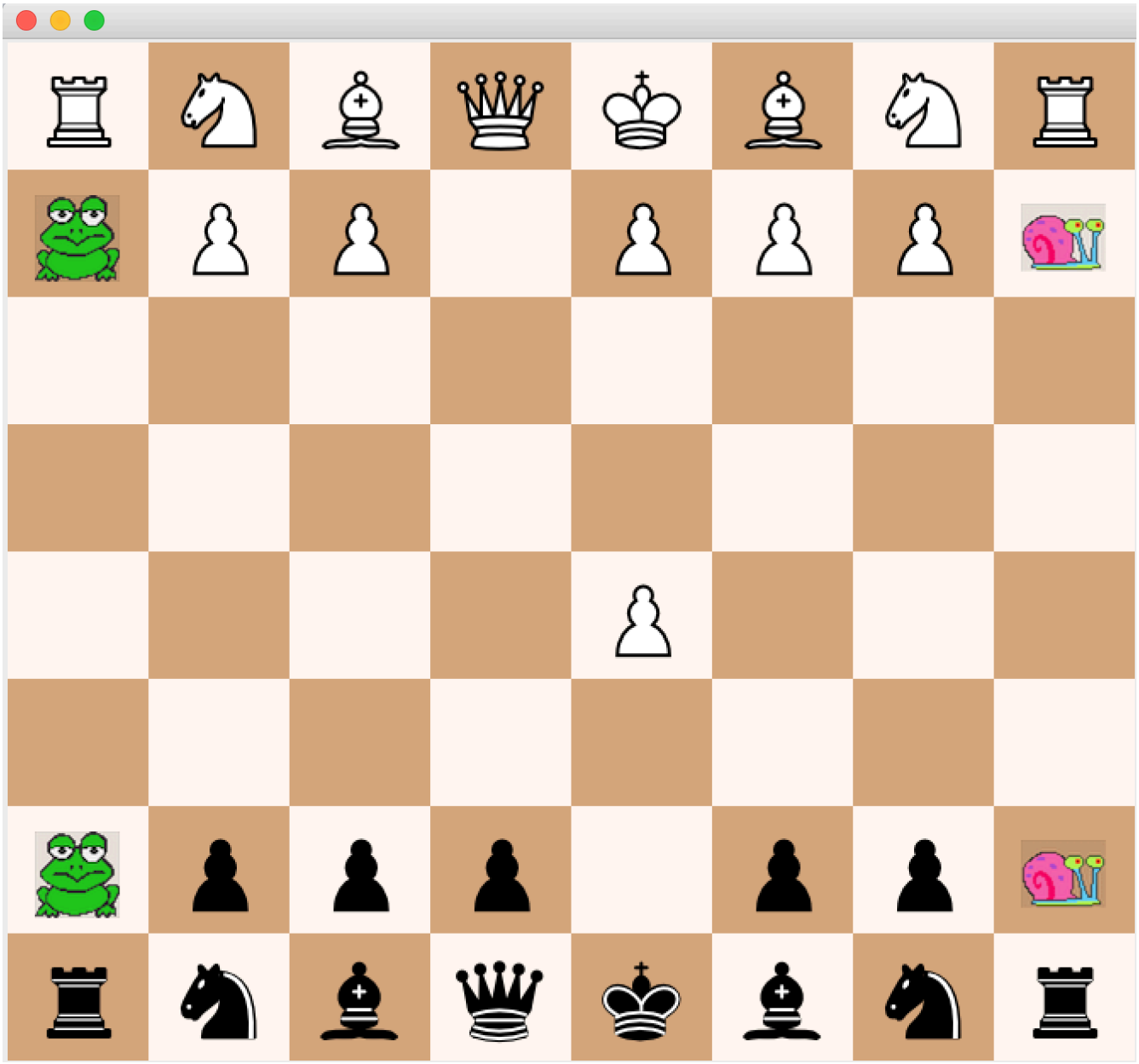
You should be able to close the dialog box and keep trying to move your desired piece until you move it to a legal spot!

4.) When you go to kill an opponents piece, there will not be any alerts or dialogs that come up, however you will visually see that your piece will have taken the place of the piece that you have killed, furthermore, the opposing team will no longer have access to the piece that you have killed. Before killing an opposing team's piece the

board should look like this. (this example shows a pawn killing another pawn):



and after killing an opposing team's piece, the board should look like this:



as you can see, the white piece killed the opposing black piece, it simply took its spot, then removed the killed piece off the board. You win the game by checkmating the king. We will get to testing these conditions later on in this document.

Arken Ibrahim

Testing Game Ending Conditions

1.) There are many intricacies to test for when testing the game ending conditions of chess. My implementation makes it very obvious to the user when there is a checkmate against his own king, when there is a check against his own king, and it also forces the user to move this king to a location that gets him out of the checking situation. We will outline a sequence of steps in order to test these game ending conditions.

2.) First let us test for a simply check of the king. First move around your chess pieces such that your king is in Check, but not in a checkmate. An example board should look like this:

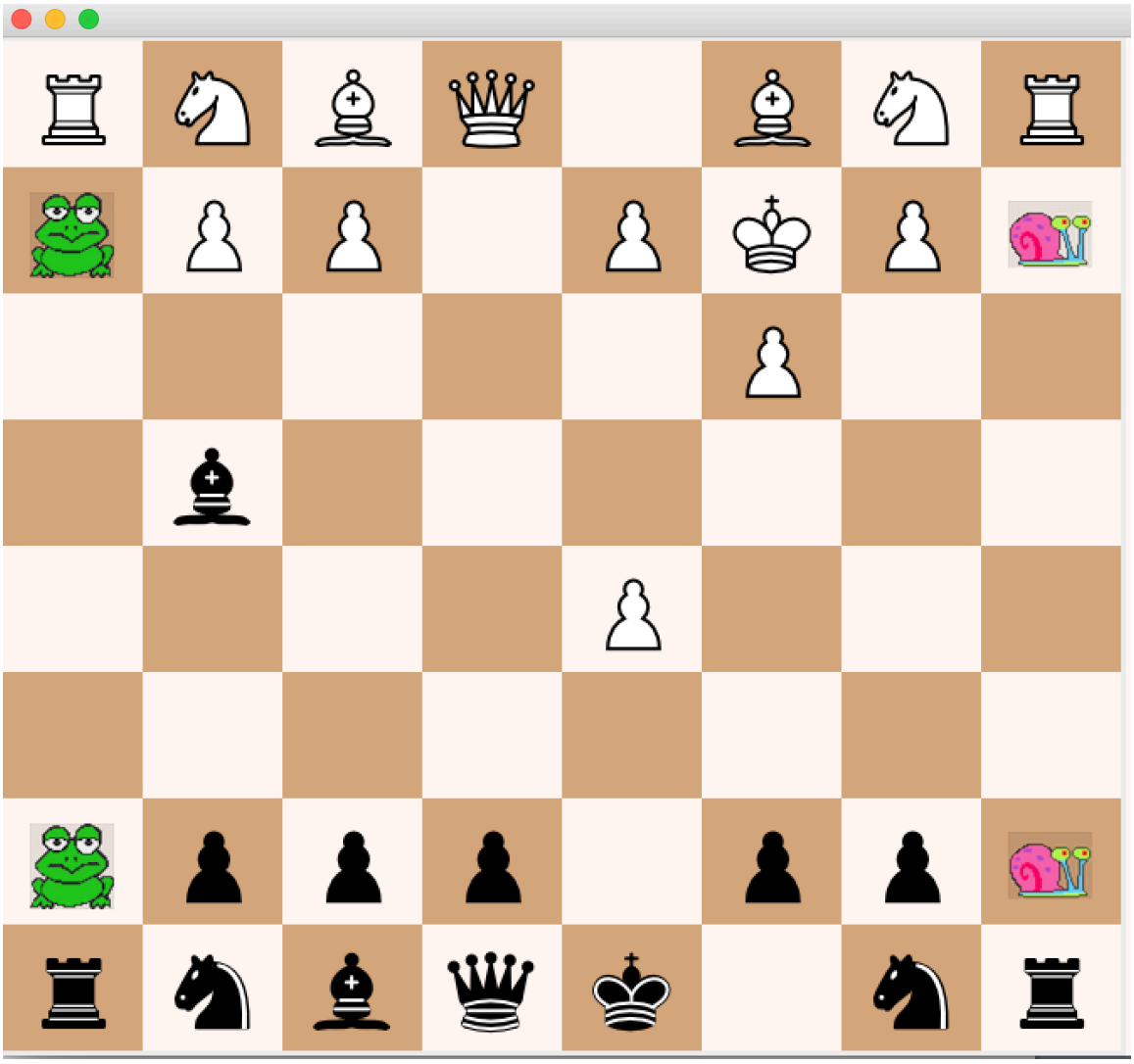
You will notice that the king is in check by the bishop, but has a way out by moving diagonally the opposite direction. The moment your opponent moves to such a position a pop up dialog such as the one in the image above should present itself.

3.) We now need to make sure that the king can only move to spots that get him out of this check situation. So lets try moving him to a location that keeps him in the very same situation. Your screen should look like this:

If you try to move your checked king to a location that keeps him in check, this dialog box which states, “Illegal Move” must appear on your screen. If it does not appear please have a look at “controller.java” for bugs and more detailed specification about what is going on in the backend.

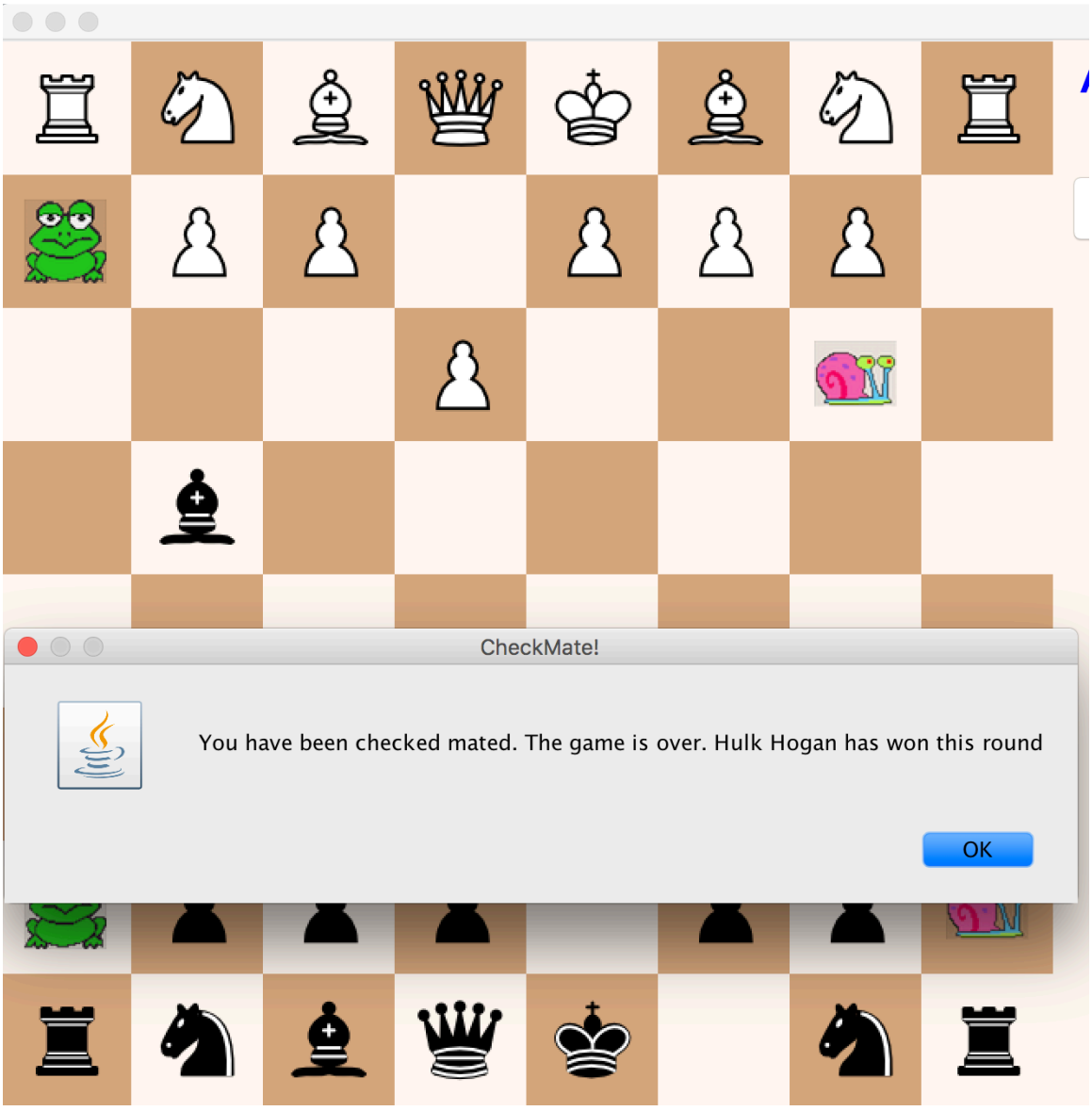
4.) Now if we try to move the king to a location that removes him from the check situation, the board should allow it and the game should

move on as normal. The board should look like this if I make such a life-saving move:



As you can see, the king successfully moved to the diagonal spot which saved him from being in check. There should be no problems there.

5.) We now need to test for a checkmate. Once a checkmate occurs, the game is over and we tally up a point for the winning player. Lets first get our board in a position that places one of the kings in a checkmate position. One example of such a board looks like this:



The bishop in the image above, places the king in checkmate, and since the king has no where to go, the game has ended and the black team has won.

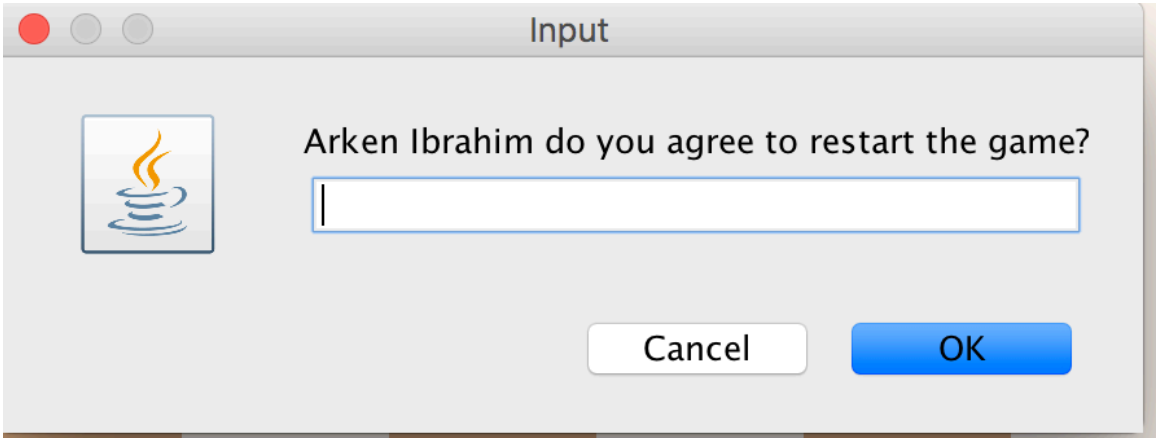
6.) When you click on the “Ok” button which tells you that you have entered a checkmate, than the board should look like this:

The board should look brand new and ready to play a new game, except you should notice that whoever won the last round will have a

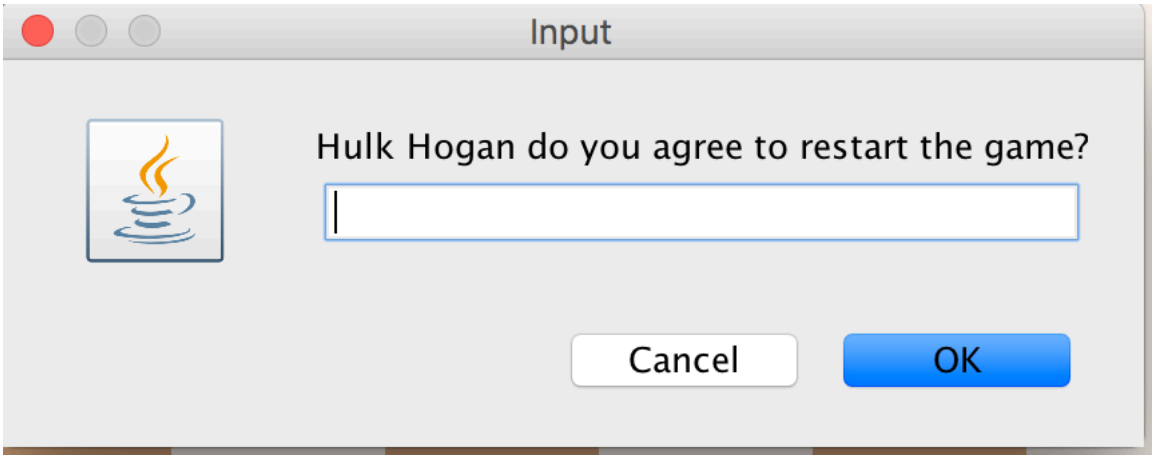
point tallied up. In this example, you can see that because of the previous checkmate, Hulk Hogan, has gained one point. They can now play another round if they wish!

Testing Restart and Forfeit

1.) When you click restart you should see two consecutive dialog boxes asking for permission from BOTH players to restart the game. The first dialog will have the name of the first player as such:



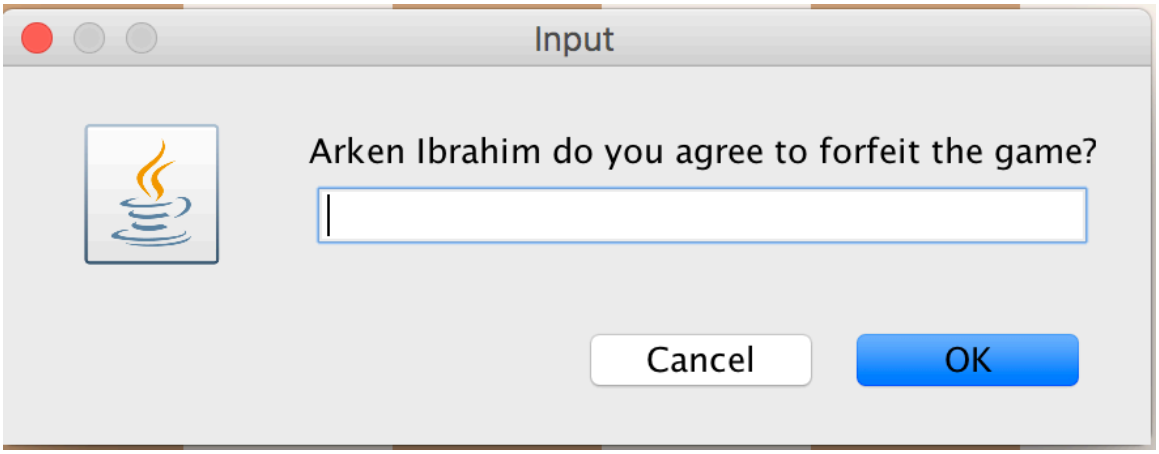
and the second dialog box will have the name of the second player. Like this:



Unless both payers write “yes” into the textbox, the game will not restart. However, if both players do write “yes” into the textbox, that means that a new game will begin, and no one will gain any points from the game that was restarted, in other words it will be counted as a draw. After both players click “yes”, the board should look like this:

You should notice that the game score has not changed. Hulk Hogan still has 1 point and Arken Ibrahim still has 0 points. Restarting the game had no effect on the score of the game. One should also notice that all the pieces on the board are in their proper places and are ready to be played in a whole, new game!

2.) Forfeit is a lot like restart. However, which ever player chooses to forfeit, that means that he will lose the round and the opposing player will gain a point. A new round of chess will be ready to be played after forfeiting. When you first click the button “forfeit”, you should see a dialog box like this.



the box simply asks the forfeiting player whether he is sure he wants to forfeit. If he writes anything other than “yes”, the game will continue. If he writes “yes”, the game board should look like this:

one should notice that since Arken Ibrahim forfeited, Hulk Hogan now has an extra point.

Testing Undo

1.) The Undo functionality is arguably the most powerful feature of this game. It was implemented in an efficient way in which we stored command objects and pushed onto the stack. And whenever an undo call was made, we simply executed the reverse of the command that was popped off the stack. If you board looks like this:

then after clicking the undo button three times the board should look like this:

Arken Ibrahim

Note that you can undo the game all the way till the beginning! Yes it is that powerful!!