Test Cases:

Module: Piece

public void testIsValidTrue() -- tests isValid method with move to open space in a rook's path.

public void testIsValidFalse() -- tests isValid method with move to open space not in a rook's path.

public void testMove() -- tests move method to ensure move took place

public void testCaptureReplacement() -- tests move method to ensure captured piece is replaced.

public void testCaptureTaken() -- tests move method to ensure captured piece's isTaken flag is set to true.

public void testCaptureOffBoard() -- tests to ensure captured piece is off the board with its position field being [-1, -1].

public void testBlackUpgrade() -- tests to ensure a move by a black rook into the white castle results in an upgrade to a black queen.

public void testBlackNoUpgrade() -- tests to ensure a move by a black rook into the black castle does not result in an upgrade to a black queen.

public void testWhiteUpgrade() -- tests to ensure a move by a white rook into the black castle results in an upgrade to a white queen.

public void testWhiteUpgrade() -- tests to ensure a move by a white rook into the white castle does not result in an upgrade to a white queen.

public void testCausesCheckTrue() -- tests the causesCheck method to ensure the method returns true when a move causes check

Module: Game

public void testCheckmate1() -- tests isCheckMate method with simple checkmate case.

public void testCheckmate2() -- tests isCheckMate method when King can capture one queen but that queen is protected by another piece.

public void testCheckmate3() -- test isCheckMate method when opponent pieces are on wall and can't be captured by the king.

public void testCheckmate4() -- test isCheckMate method with another simple checkmate case where king has nowhere to go.

public void testCheckmate5() -- test isCheckMate method when there is another piece on the board as the checked king, but the piece cannot capture the checking piece or block check.

public void testNotCheckmate() -- test isCheckMate method for simple case when King is not in checkmate.

public void testNotCheckmate2() -- test isCheckMate method for another simple case where the king is not in checkmate.

public void testNotCheckmate3() -- test isCheckMate method to ensure that king is not in checkmate when it's not even in check.

public void testNotCheckmate4() -- test isCheckMate method when king is really in stalemate, therefore not checkmate.

public void testNotCheckmate5() -- test isCheckMate method when king can capture the piece that is putting it into check.

public void testNotCheckmate6() -- test isCheckMate method when another piece can block the check check.

public void testNotCheckmate7() -- test isCheckMate method when another piece can capture the piece putting the king into check.

public void testStalemate() -- test isStaleMate method with a simple case when the king is stalemate.

public void testNotStalemate1() -- test isStaleMate method when king is not the last piece on the board so king is not in stalemate.

public void testNotStalemate2() -- test isStaleMate method when king can capture a piece.

public void testNotStalemate3() -- test isStaleMate method when the king is actually in checkmate.

public void testNotStalemate4() -- test isStaleMate method when the king is in check.

Module: Invite

void testAcceptReturnsTrueWithPendingInvitation() -- test accept method to ensure pending invitation can be accepted.

void testAcceptReturnsFalseWithCancelledInvitation() -- test accept method to ensure cancelled invitation cannot be accepted.

void testAcceptReturnsFalseWithAcceptedInvitation() -- test accept method to ensure accepted invitation cannot be accepted twice.

void testCancelReturnsTrueWithPendingInvitation() -- test cancel method to ensure a pending invitation can be cancelled.

void testCancelReturnsFalseWithAcceptedInvitation() -- test cancel method to ensure an accepted invitation cannot be cancelled.

void testCancelReturnsFalseWithCancelledInvitation() -- test cancel method to ensure a cancelled invitation cannot be cancelled again.

void testInviteUsersReturnsTrueWithPendingInvitation() -- test inviteUsers method to ensure inviting users to a game is valid for a pending invitation.

void testInviteUsersReturnsFalseWithAcceptedInvitation() -- test inviteUsers method to ensure inviting users to a game is invalid for an accepted invitation.