TESTING REPORT

Ishaan Gupta
ixg97@case.edu

Test methods of Game (JavaFx)
Method name : start()

Test: Passing of Input Parameter from main method
When you run the game, the game launches with default value (row = 19, column = 19 and number for winning = 5). A printed message indicating this also appears and a window containing a button appears.

If you write java Gomoku 6 34 12, the game launches with the values of row = 34, column = 12 and number for winning = 6. A printed message indicating this also appears and a window containing a button appears.

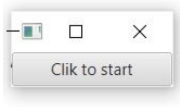
If you write java Gomoku 34 12, the game launches with the values of row = 34, column = 12 and number for winning = 5. A printed message indicating this also appears and a window containing a button appears.

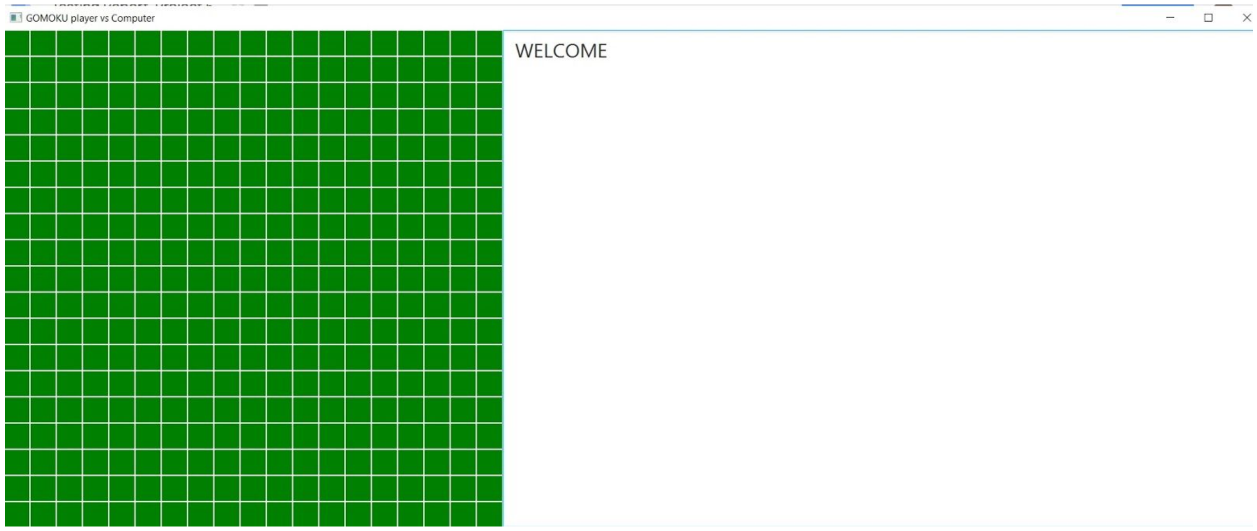
These tests can further be tested after clicking the button and playing the game.

Test: Try/ catch for valid input values
If you write java Gomoku t w r, the game launches default value (row = 19, column = 19 and number for winning = 5). A printed message indicating this also appears and a window containing a button appears.

If you write java Gomoku 4 12 23 (since number for winning 4 < 5), it is considered as an invalid input value and the game launches with the default values. A printed message indicating this also appears and a window containing a button appears.

Test: Correct Graphics of launched Game
If you pass the correct values as input parameters, a window with a button appears. On clicking, the game with grid of number of input rows and columns along with a text box saying “WELCOME” appears as shown below.

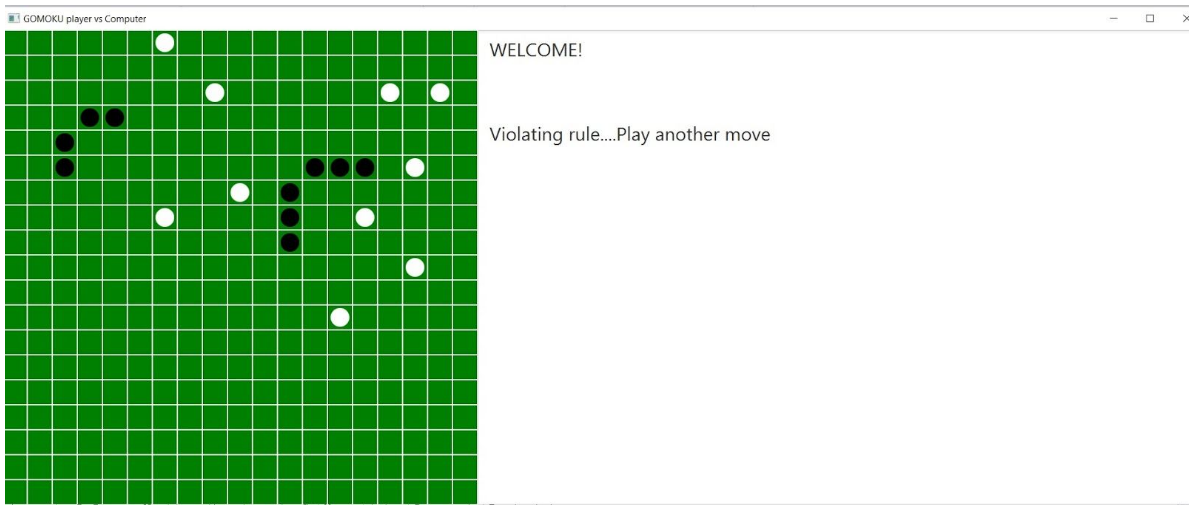




Method name : handle()
Once the game is available with above shown grid, the squares (button) are clickable. Since, the game starts with black and alternates with white, on clicking the buttons, black and white pieces will appear on the clicked squares. This process can be repeated till there is no such row containing winning number of same colored pieces. Before placing a piece, the three-three and four-four methods will be called and checked to make sure there is no violation of any rule. If there is any such violation, a message is printed and button is not clickable. And after placing a piece, the winning condition is checked to call the showWinner method.

Since while playing, all these things work perfectly well, this method works correctly.

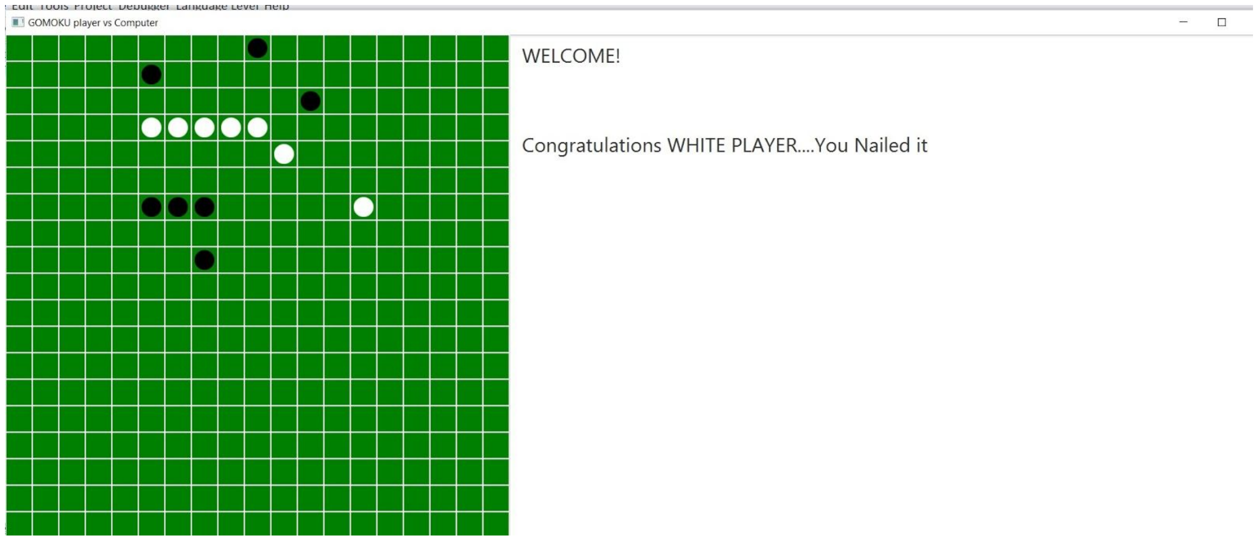
Method Name: showMove()
Once the game is available and while playing a player violates the three-three or four-four method, an appropriate message is shown in the textBox (as shown below).



WELCOME!

Violating rule....Play another move

Method name: showWinner()
Once the game is available and one of the players is successful in making a row of same pieces (number of pieces equal to strike), a message in the textBox is shown claiming the win (as shown below).


Edit Tools Project Debugger Language Level Help
GOMOKU player vs Computer

WELCOME!

Congratulations WHITE PLAYER....You Nailed it

Test Helper methods of Game

Test method Name: testNumberInLine()

This test method tests the numberInLine method of Gomoku class

Tests Done:

1. **Test 0 :** (There are 0 elements except the one from which you start searching in the row in a particular direction)

2. **Test 1:** There are 1 elements except the one from which you start searching in the row in a particular direction
3. **Test Many:** There are more than 1 elements except the one from which you start searching in the row in a particular direction
4. **Test First:** Wrong (Unequal) element found on the very first step of searching in a particular direction
5. **Test Middle:** Wrong element found in the middle row while searching in a particular direction
6. **Test Last:** Wrong element found in the end of the row while searching in a particular direction
7. **Test for Each Direction:** These 8 tests test for each 8 direction

Description for each tests: I create a 2-D array of integers. Then I decide the row and column index of line in which method is to start calculating. Then I create a new instance of the class and call the method with appropriate parameters passed.

Test method Name: testIsOpen()

This test method tests the isOpen method of Gomoku class

Tests Done:

1. **Test 0 :** There are 0 open spaces from the position you start searching in the row in a particular direction
2. **Test 1:** There are 1 open spaces from the position you start searching in the row in a particular direction
3. **Test Many:** There are more than 1 open spaces from the position you start searching in the row in a particular direction
4. **Test First:** There is Open space on first step of searching in a row
5. **Test Middle:** There is an Open space in the middle of a row step while searching
6. **Test Last:** There is an Open space in the last end of a row step while searching
7. **Test for Each Direction:** These 8 tests test for each 8 direction which tests for both conditions: when there is an open space present and when not

Description for each tests: I create a 2-D array of integers. Then I decide the row and column index of line in which method is to start calculating. Then I create a new instance of the class and call the method with appropriate parameters passed.

Test method Name: testCheckLessThanStrike()

This test method tests the checkLessThanStrikemethod of Gomoku class

Tests Done:

1. **Test :** When all the conditions of the method gets false i.e there is no row with 4 same elements
2. **Test :** When one of the conditions of the method gets true i.e there is 1 row with 4 same elements
3. **Test :** When 2 of the conditions of the method gets true i.e there are 2 rows with 4 same elements
4. **Test :** When more than 2 of the conditions of the method gets true i.e there are more than 2 rows with 4 same elements

Description for each tests: I create a 2-D array of integers. Then I decide the row and column index of line in which method is to start calculating. Then I create a new instance of the class, change the integer board with the board I created and finally call the method with appropriate parameters passed.

Test method Name: testCheckTwoLessThanStrike()

This test method tests the checkTwoLessThanStrikemethod of Gomoku class

Tests Done:

1. **Test :** When all the conditions of the method gets false i.e there is no row with 3 same elements and open spaces on all 4 sides
2. **Test :** When all the conditions of the method gets false i.e there is a row with 3 same elements but no open spaces on all 4 sides
3. **Test :** When 1 of the conditions of the method gets true i.e there is one row with 3 same elements and open spaces on all 4 sides
4. **Test :** When 2 of the conditions of the method get true i.e there is 2row with 3 same elements and open spaces on all 4 sides
5. **Test :** When more than 2 of the conditions of the method gets true i.e there is more than 2 rows with 3 same elements and open spaces on all 4 sides
6. **Note: Test in which left condition gets false and right condition gets true not done. This is because when left conditions is false, right condition is not checked.**

Description for each tests: I create a 2-D array of integers. Then I decide the row and column index of line in which method is to start calculating. Then I create a new instance of the class, change the integer board with the board I created and finally call the method with appropriate parameters passed.

Test method Name: testButtonIsPressed()

This test method tests the buttonIsPressed of Gomoku class
Also, Tests the getter and setter methods of playerCounter field
Tests the getter and setter methods of board field

Tests Done:

7. **Test :** When the black player places its piece
8. **Test :** When the white player places its piece

Description for each tests: I create a 2-D array of integers. Then I decide the row and column index of line on which the piece is to be placed. Then I create a new instance of the class, change the integer board with the board I created and finally call the method with appropriate parameters passed. Then I create the expected array and compare it with array stored in the instance of the class.

Test method Name: testIsEmpty()

This test method tests the isEmpty() of Gomoku class

Tests Done:

9. **Test :** Method called When the position is empty
10. **Test :** Method called When the position is non-empty

Description for each tests: I create a 2-D array of integers. Then I decide the row and column index of line on which the piece is to be placed. Then I create a new instance of the class, change the integer board with the board I created and finally call the method with the position to be checked for emptiness.

Test method Name: testIsWinner()

This test method tests the isWinner() of Gomoku class
Also, Tests getter and setter methods of isWinner field

Tests Done:

11. **Test :** Method called When there is a winner
12. **Test :** Method called When there is a winner

Description for each tests: I create a 2-D array of integers. Then I decide the row and column index of line on which the piece is to be placed. Then I create a new instance of the class, change the integer board with the board I created and finally call the method with the appropriate parameters. Finally, I check the value of the isWinner field by the getter method for it.