# Design Summary

## WinState

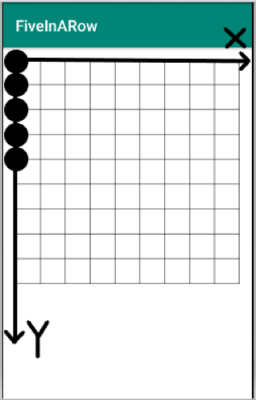
This class is made to collect all the possible win states, five pieces with the same color in one of the four orientations ( vertical, horizontal, right diagonal and left diagonal).

Generally, we use three variables, int i , j and k , to form a triple nested for loop to find all the possible states(WinState) in each orientation and add them to a list(wList). And in every WinState, we include the coordinate, the number of that WinState and the value of the WinState. A number will be allocated to five WinStates of which the pieces form a line in one of the four orientations.

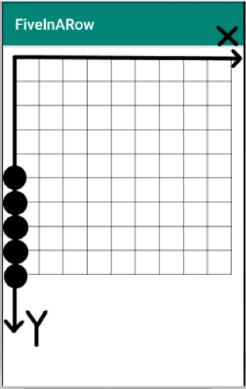
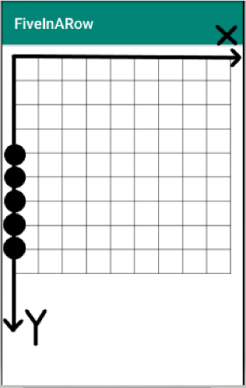
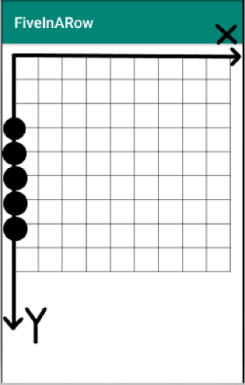
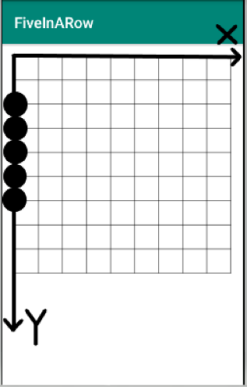
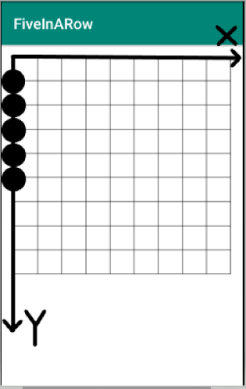
1. Vertical

We fix the horizontal ordinate and find all the 6 groups of five pieces in a vertical line for that column and we do the same for all the 10 columns. In the triple nested for loop, we use 1st for loop’s variable, int i, as horizontal ordinate so that the 2 for loop inside it can be used to find the vertical ordinates for the points in that column. The 2nd for loop’s variable, int j, is made to fix the start vertical ordinate for that group and the 3rd for loop’s variable, int k, is to fix the increment of the vertical ordinate and we use j+k to fix the vertical ordinates.

As for i=0, j=0 and k growing from 0 to 4 we can find the ordinates of the 1st group of five pieces which are (0,0),(0,1),(0,2),(0,3),(0,4). And the number 0 and the value 1 will be allocated to the five Winstates.



And as int j growing from 0 to 5, we can get all the 6 groups of five pieces in a vertical line for this column(int i=0).

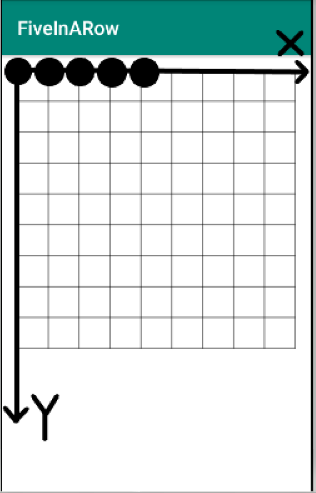


And with int i growing from 0 to 9, we can do the same operation as above to the all 10 columns so that we can find all the WinStates in vertical direction.

1. Horizontal

Similar to the Vertical one, we form a triple nested for loop but we use j+k as horizontal ordinate and i as vertical ordinate for this time(just swap the horizontal ordinate and the vertical ordinate in the vertical triple for loop). We fix the vertical ordinate and find all the 6 groups of five pieces in a horizontal line for that row and we do the same for all the 10 rows.

As for i=0, j=0 and k growing from 0 to 4 we can find the ordinates of the 1st group of five pieces in the horizontal direction which are (0,0),(1,0),(2,0),(3,0),(4,0). And a number and the value 1 will be allocated to those five Winstates.



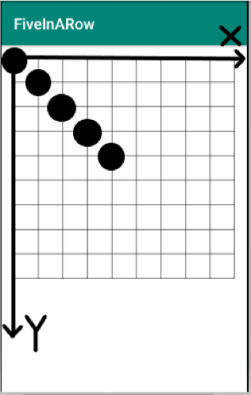
And as int j grow from 0 to 5, we can get all the 6 groups of five pieces in a horizontal line for this row(int i=0).

And with int i grow from 0 to 9, we can do the same operation as above to the all 10 rows so that we can find all the WinStates in horizontal direction.

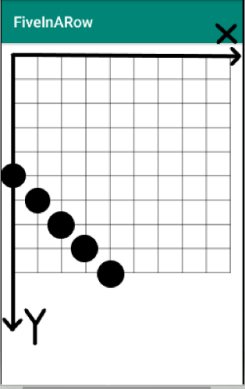
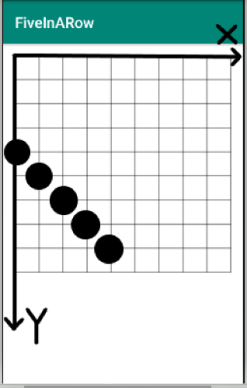
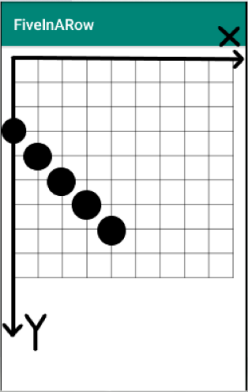
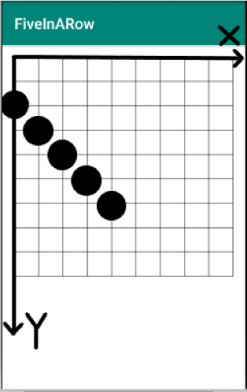
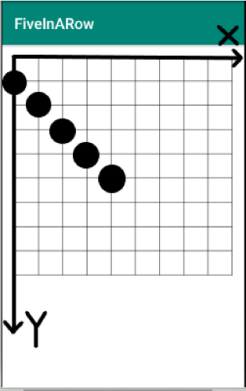
1. right diagonal

In this part, we want to find all the WinStates in right diagonal direction. We fix the start piece for each group and find the other 4 pieces in that group by adding the same number to the horizontal ordinate and the vertical ordinate of the start piece. And we use the 1st for loop’s variable, int i, as the original horizontal ordinate and the 2nd for loop’s variable, int j, as the original vertical ordinate so that we can find all the 6 original ordinates(the start point for a group) in one column, and we can do the same for from column 1 to column 6(as j grow from 0 to 5).There are 6 start pieces in every column from column 1 to column 6 and we call 6 groups of 5 pieces like that as a big group.

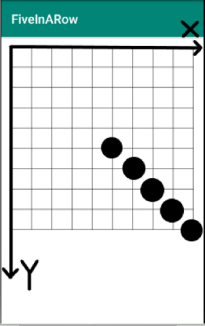
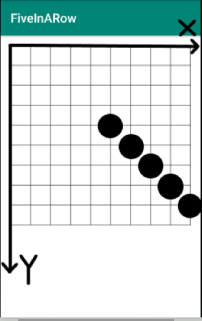
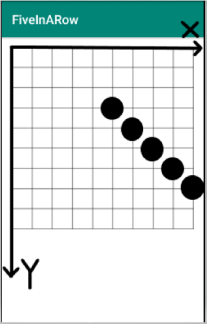
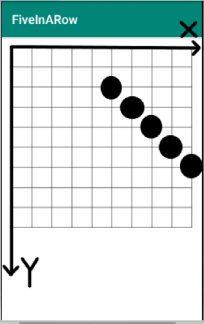
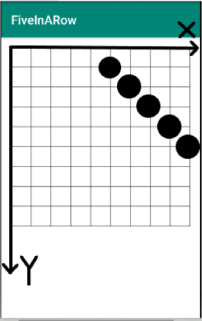
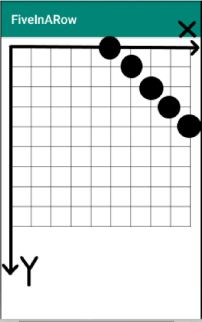
As for i=0, j=0 and k growing from 0 to 4 we can find the ordinates of the 1st group of five pieces in the right diagonal direction which are (0,0),(1,1),(2,2),(3,3),(4,4). And a number and the value 1 will be allocated to those five Winstates.



And as int j growing from 0 to 5, we can get all the 6 groups of five pieces in a right diagonal line in big group 1 (the start pieces for all the 6 groups are in column 1(int i=0)).



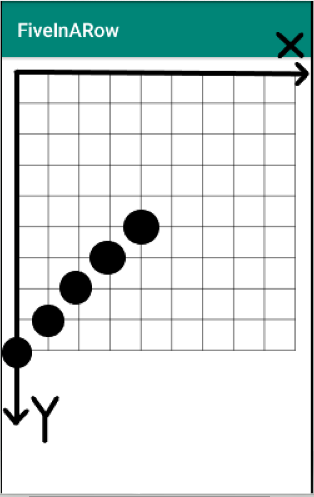
And with int i grow from 1 to 5, we can do the same operation as above to the other 5 big groups of which the start pieces are in from column 2 to column 6. The last big group should be that:



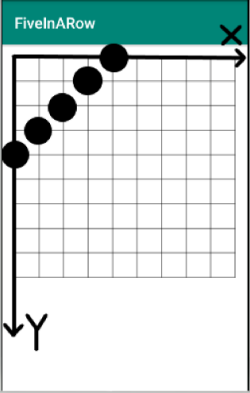
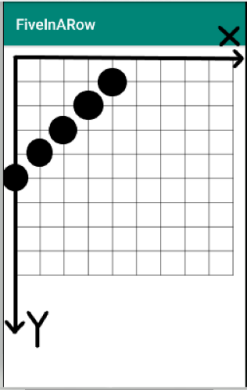
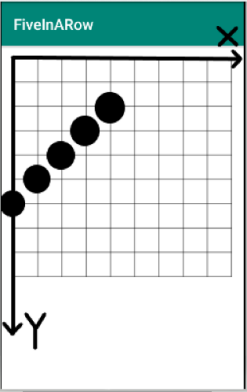
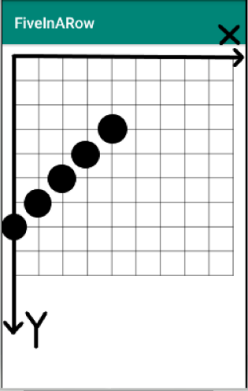
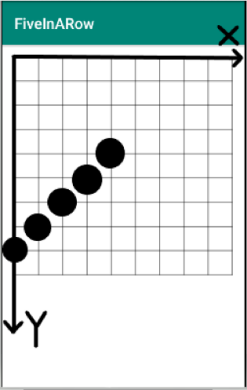
1. left diagonal

In this part, we want to find all the WinStates in left diagonal direction. Similar to the right diagonal part, we use 1st for loop’s variable int i as original horizontal ordinate and 2nd for loop’s variable int j as original vertical ordinate. But this time we start from the left bottom of the board as we made the 2nd for loop starts from j=9. And to get each group of five pieces, we should add the corresponding number, int k, to the original horizontal ordinate and subtract the same number from the original vertical ordinate. There are 6 start pieces in every column from column 1 to column 6 and we call 6 groups of 5 pieces like that as a big group.

As for i=0, j=9 and k growing from 0 to 4 we can find the ordinates of the 1st group of five pieces in the left diagonal direction which are (0,9),(1,8),(2,7),(3,6),(4,5). And a number and the value 1 will be allocated to those five Winstates.



And as int j reducing from 9 to 4, we can get all the 6 groups of five pieces in a left diagonal line in the big group 1 (the start pieces for all the 6 groups are in column 1(int i=0)).



And with int i grow from 1 to 5, we can do the same operation as above to the other 5 big groups of which the start pieces are in from column 2 to column 6. So that we can get all the WinStates in left diagonal direction.

And using those 4 for loops, we can find all the WinStates.

And we also made 2 variables, IWin and CWin, for each group of five pieces in a line indexed by the number allocated to the Winstate to count the sores for the player and the computer.

And at the end of the class, we made a function which can be used to get the value as we knowing the coordinate and the number of a piece.

## Socket

We use Java Socket to realize the two-way real-time communication between client and server. In the java.net package, two classes, “Client” and “ServerS”, are defined, which respectively realize the client and the server side of the two-way connection.

Implementation process on the server:

1.Instantiate a ServerSocket object with the specified port through which the server can listen for connection requests from client;

2.Call the accept() method to listen for requests from the connection port, which is blocked.

3.Do the read and write operations (IO) using the client socket object returned by this method ;

4.Close the Socket object.

Implementation process on the client:

1.Do the instantiating with the IP address and port provided by the server side;

2.Call the connect() method to connect to the server;

3.Obtain the flow on the Socket and encapsulate that into the BufferedReader/PrintWriter instance, for reading and writing;

4.Realize the interacting with the server using getInputStream and getOutputStream provided by Socket;

5.Close the Socket object.