# TIK-TAK-TOE DOCUMENTATION

Prepared By:

| | |
|---|---|
| Yousef Ashraf Hassan Kotp | 7140 |
| Yousef Ahmed AboEid | 6883 |

## Main Idea

Our main idea is to make 4 classes which we will explain later, also we made a static global variable which is stored in class ‘CheckWinner’ as an attribute named ‘state’, this attribute is set to 0 until at least one winning condition happen, then the ‘state’ global variable is changed to 1.

## Input / Output

All inputs are taken from the console, also the output is displayed on console (GUI is not used)

## Class ‘Board’

This class does not have any attributes, this class only contains methods which are:

1. printBoard()

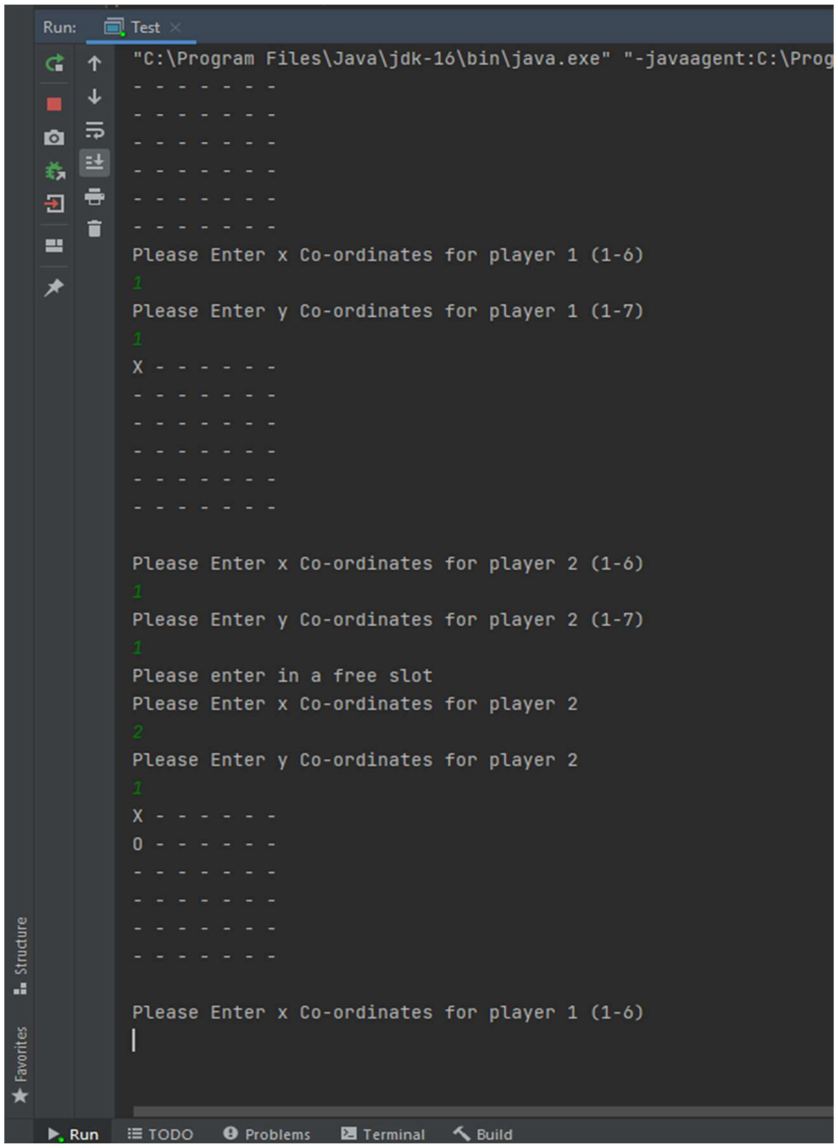
This method iterates through the whole 2D array of 6 rows and 7 columns and print them on screen.

2. fillBoard()

When we declare this 2D array, it is set to null (aka garbage), so we use this method to fill each slot in the tic-tak-toe board with ‘-’ character.

3. isFull()

The function of this method is clear from its name, this function takes 3 parameters: 2D array, index X & index Y, so this function takes the location of the slot in the board and check whether it is already filled with either ‘X’ or ‘O’.



```
"C:\Program Files\Java\jdk-16\bin\java.exe" "-javaagent:C:\Prog
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
Please Enter x Co-ordinates for player 1 (1-6)
1
Please Enter y Co-ordinates for player 1 (1-7)
1
X - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -


Please Enter x Co-ordinates for player 2 (1-6)
1
Please Enter y Co-ordinates for player 2 (1-7)
1
Please enter in a free slot
Please Enter x Co-ordinates for player 2
2
Please Enter y Co-ordinates for player 2
1
X - - - - - -
O - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -


Please Enter x Co-ordinates for player 1 (1-6)
|
```

## Class ‘CheckWinner’

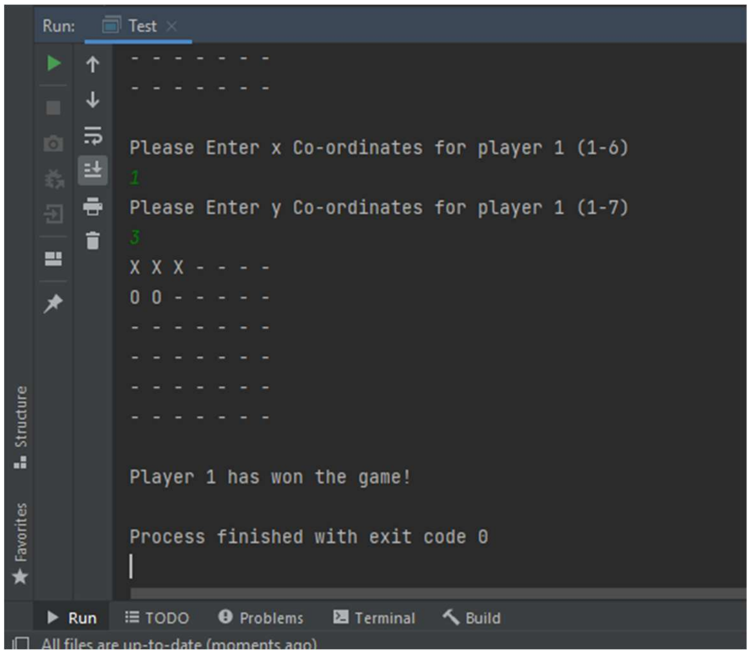
We all know that the winning condition in tik-tak-toe is to get three consecutive ‘X’ or ‘O’ in a row or column, or even diagonally. If all the slots have been taken and there is no winner, then it is a draw, using simple 4 methods, we can check each winning condition and draw condition each time the player plays.

This class has one attribute which is global variable named ‘state’ which is set to zero until at least one winning condition happens.

We made 4 methods to check winning and draw conditions, those methods are set to static so that we don’t have to create object every time we want to use them, each method takes two parameters: 2D array and character ‘type’ which can be either ‘X’ or ‘O’.

1. checkRow()

This method iterates through each row from the 6 rows, if it find 3 consecutive ‘X’ or ‘O’ it will change the ‘state’ global variable to 1.



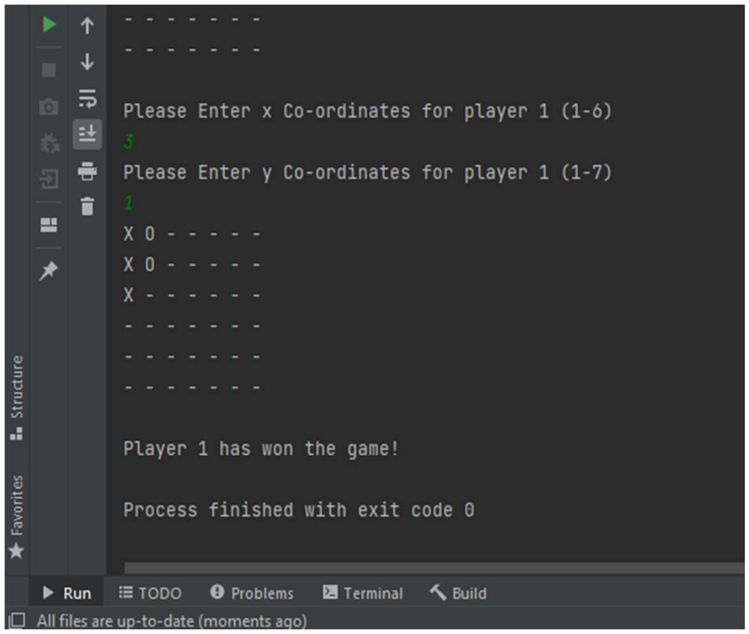
```
- - - - - - -
- - - - - - -

Please Enter x Co-ordinates for player 1 (1-6)
1
Please Enter y Co-ordinates for player 1 (1-7)
3
X X X - - - -
0 0 - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -


Player 1 has won the game!

Process finished with exit code 0
```

All files are up-to-date (moments ago)

2. checkCol()

This method iterates through each column in the same way checkRow

works but for columns.

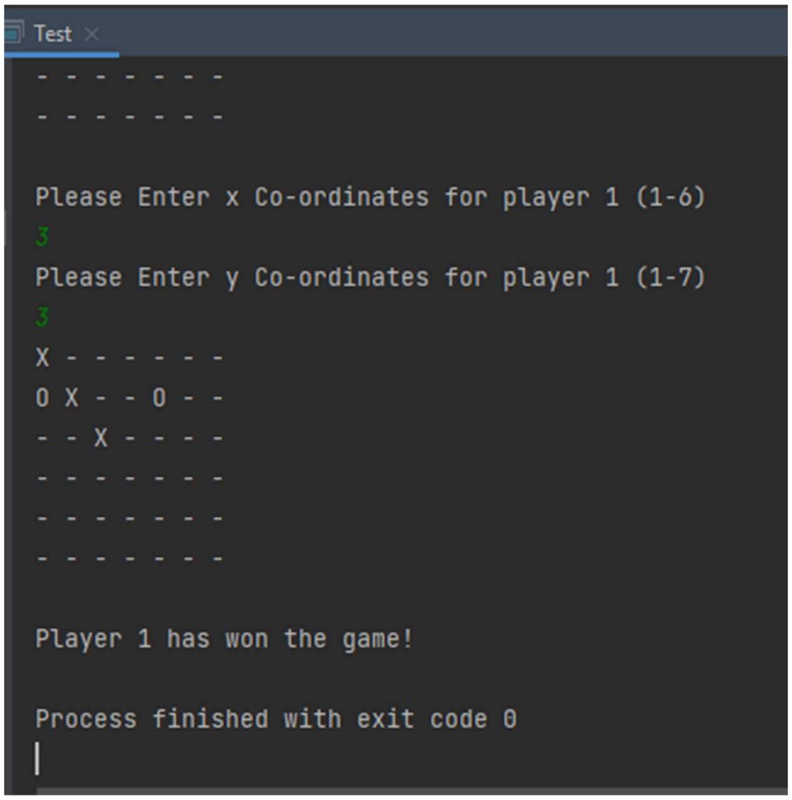


```
- - - - - - -
- - - - - - -

Please Enter x Co-ordinates for player 1 (1-6)
3
Please Enter y Co-ordinates for player 1 (1-7)
1
X 0 - - - - -
X 0 - - - - -
X - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -


Player 1 has won the game!

Process finished with exit code 0
```

All files are up-to-date (moments ago)

3. checkDiagonal()

This method is so tricky, we used our previous knowledge on matrices to check each diagonal, there are 2 nested loops, the first loop start checking from top left corner while the other one checks from the bottom left corner, these two nested loops cover the whole 6*7 board diagonals.



```
- - - - - - -
- - - - - - -

Please Enter x Co-ordinates for player 1 (1-6)
3
Please Enter y Co-ordinates for player 1 (1-7)
3
X - - - - - -
0 X - - 0 - -
- - X - - - -
- - - - - - -
- - - - - - -
- - - - - - -

Player 1 has won the game!

Process finished with exit code 0
```

4. checkDraw

The method returns integer, where it will return 1 if it is a draw, otherwise it will always return 0, whenever the function sees the '-' character it knows that the board is not full yet.

## Class 'Player'

This class is the biggest one among them, this class represent the turn of the player, it has 3 attributes which are:

1-type ('X' or 'O')

2-indexX (the desired row location for the player)

3-indexY (the desired column location for the player)

The class has 3 setters for each attribute which we will need later, it has the biggest method among all the methods we made ( startPlaying() ), this method is where all the magic happens.

## startPlaying()

We will start by declaring a 2D array which indicates our board for tik-tak-toe, we will use fillBoard() method we made earlier in Board class to fill every element of the board with '-', we will declare a scanner to take the desired x & y location of every player, we will also declare integer x which represent player number (player 1 & player 2).

We will declare a new Player object and initialize its type to 'X' which will be swapped with 'O' when player 2 has to play.

By making a loop where the global variable 'state' is 0, we will never get out of the loop until someone wins the game or it ends with a draw.

The player is then asked to enter the desired row and column he wants to fill, if the user entered a slot which was already filled it will ask him to enter in a free slot using the method 'isFull()', when the user enter the x & y co-ordinates the slot is filled with either 'X' or 'O'.

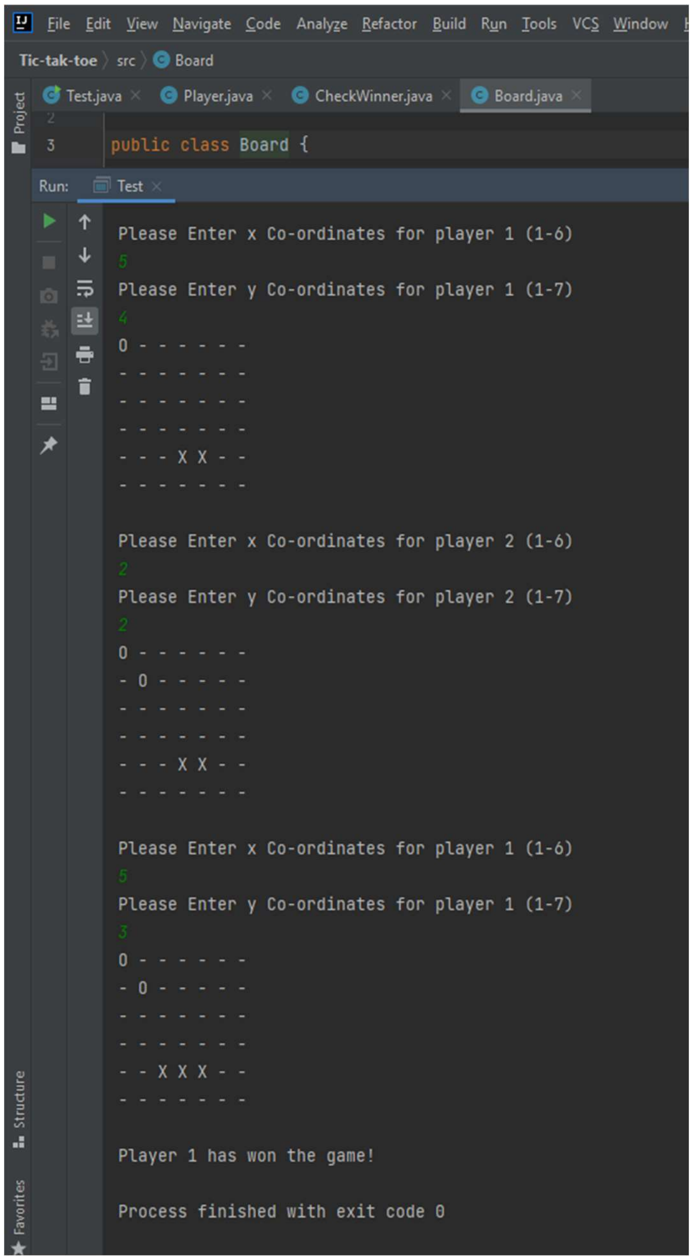
After playing, we check if there is any three consecutive 'X' or 'O' in row or column or even diagonally using the methods we already made in CheckWinner class, if any condition is true, the game will tell the which player has won, if it is a draw, it will say so too.

Before re-looping, we change the player number and type, if the player's type is 'X' then it is converted to 'O'.

## Class 'Test'

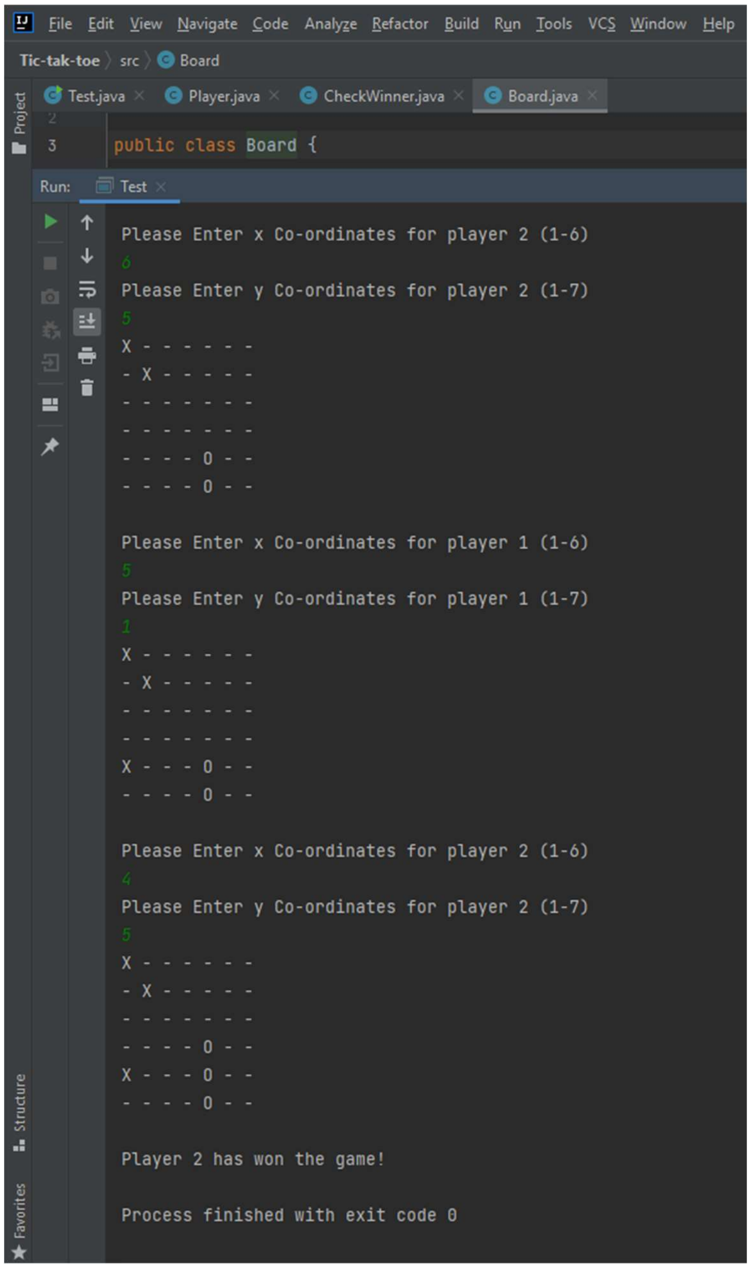
There is nothing much to be said here, we will just test our methods and classes in our main, we will call the 'startPlaying()' which will launch the game.

## Sample run



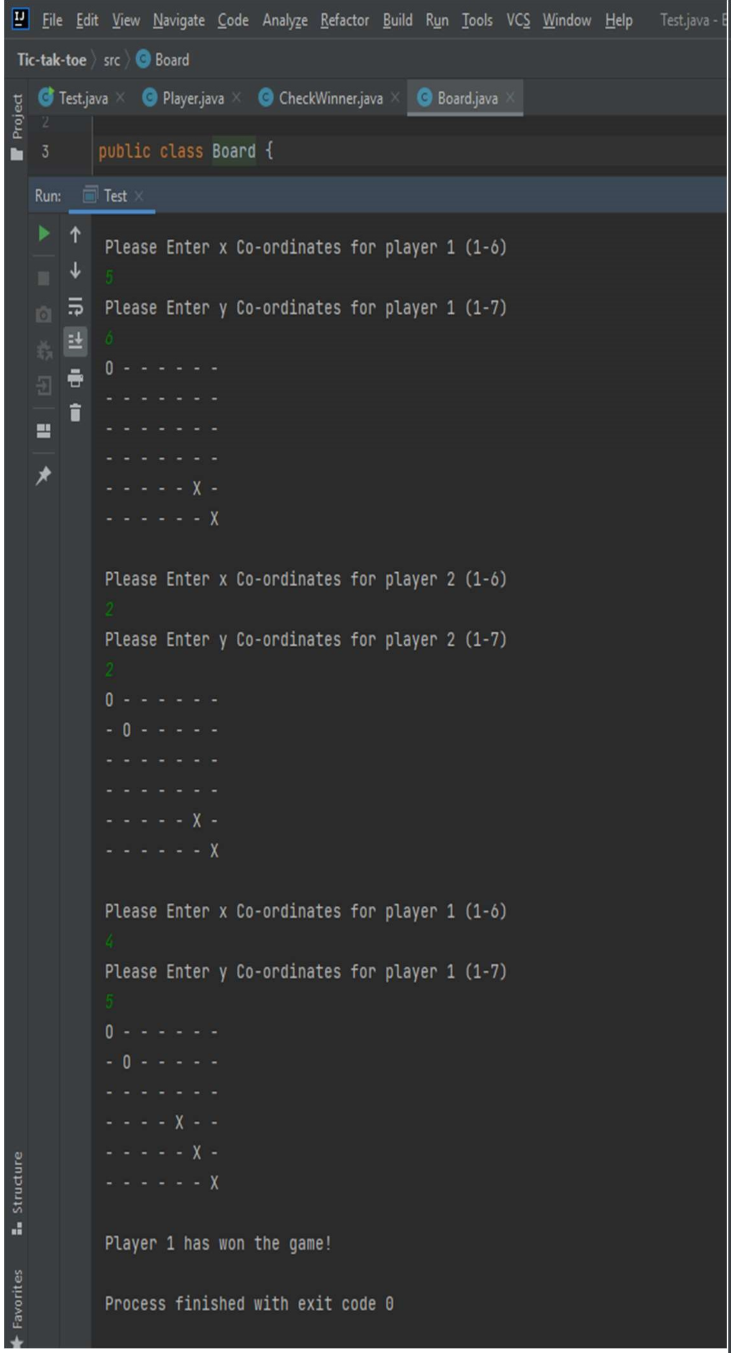
```
public class Board {
```

```
Please Enter x Co-ordinates for player 1 (1-6)
5
Please Enter y Co-ordinates for player 1 (1-7)
4
0 - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - X X - -
- - - - - - -

Please Enter x Co-ordinates for player 2 (1-6)
2
Please Enter y Co-ordinates for player 2 (1-7)
2
0 - - - - - -
- 0 - - - - -
- - - - - - -
- - - - - - -
- - - X X - -
- - - - - - -

Please Enter x Co-ordinates for player 1 (1-6)
5
Please Enter y Co-ordinates for player 1 (1-7)
3
0 - - - - - -
- 0 - - - - -
- - - - - - -
- - - - - - -
- - X X X - -
- - - - - - -

Player 1 has won the game!

Process finished with exit code 0
```



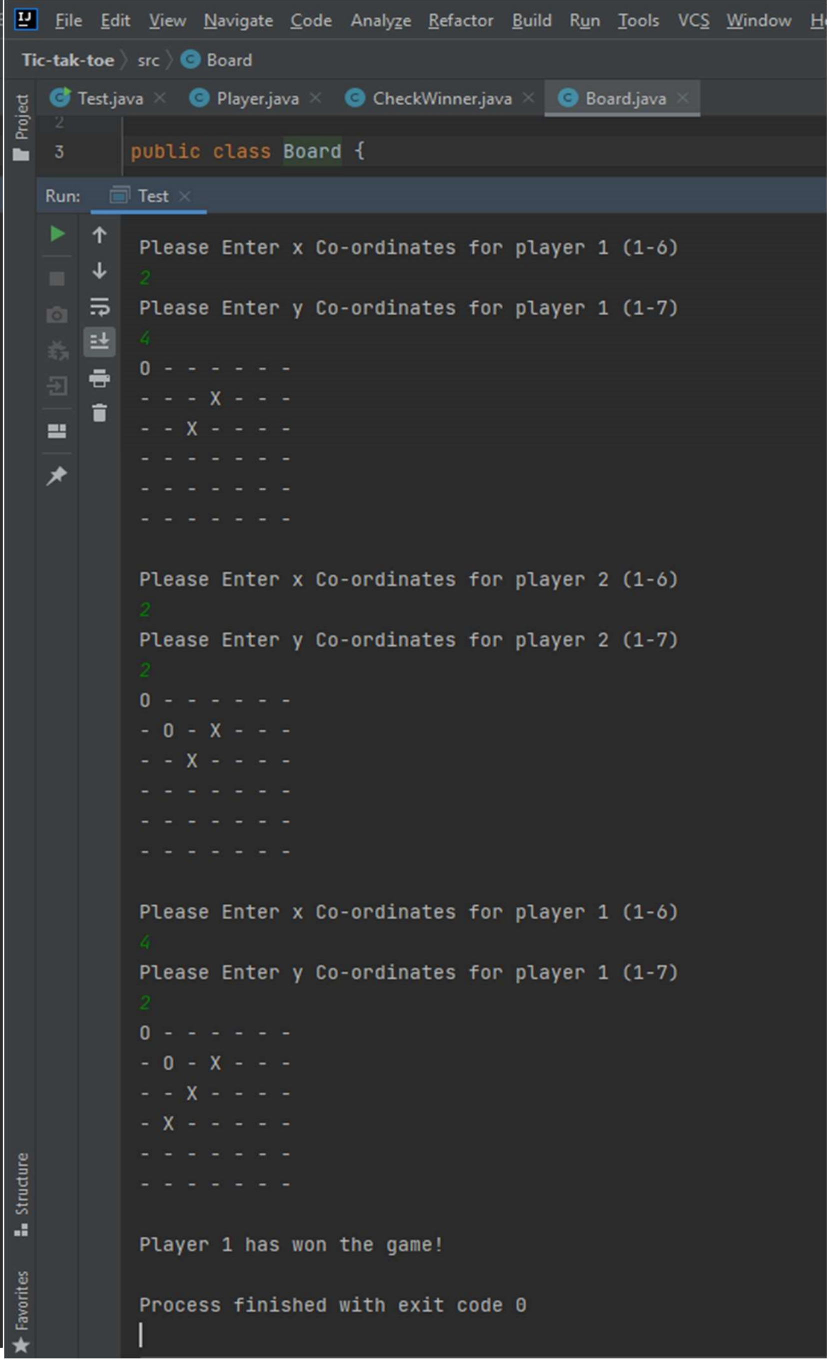
```
public class Board {
```

```
Please Enter x Co-ordinates for player 2 (1-6)
6
Please Enter y Co-ordinates for player 2 (1-7)
5
X - - - - - -
- X - - - - -
- - - - - - -
- - - - - - -
- - - - 0 - -
- - - - 0 - -

Please Enter x Co-ordinates for player 1 (1-6)
5
Please Enter y Co-ordinates for player 1 (1-7)
1
X - - - - - -
- X - - - - -
- - - - - - -
- - - - - - -
X - - - 0 - -
- - - - 0 - -

Please Enter x Co-ordinates for player 2 (1-6)
4
Please Enter y Co-ordinates for player 2 (1-7)
5
X - - - - - -
- X - - - - -
- - - - - - -
- - - - 0 - -
X - - - 0 - -
- - - - 0 - -

Player 2 has won the game!

Process finished with exit code 0
```


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Tic-tak-toe 〉 src 〉 Board
Test.java
Player.java
CheckWinner.java
Board.java
Run:
Test
Project
Structure
Favorites

```
public class Board {
```

```
Please Enter x Co-ordinates for player 1 (1-6)
5
Please Enter y Co-ordinates for player 1 (1-7)
6
0 - - - - - -
- - - - - - -
- - - - - - -
- - - - - - -
- - - - - X -
- - - - - - X

Please Enter x Co-ordinates for player 2 (1-6)
2
Please Enter y Co-ordinates for player 2 (1-7)
2
0 - - - - - -
- 0 - - - - -
- - - - - - -
- - - - - - -
- - - - - X -
- - - - - - X

Please Enter x Co-ordinates for player 1 (1-6)
4
Please Enter y Co-ordinates for player 1 (1-7)
5
0 - - - - - -
- 0 - - - - -
- - - - - - -
- - - - X - -
- - - - - X -
- - - - - - X

Player 1 has won the game!

Process finished with exit code 0
```


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window
Tic-tak-toe 〉 src 〉 Board
Test.java
Player.java
CheckWinner.java
Board.java
Run:
Test
Project
Structure
Favorites

```
public class Board {
```

```
Please Enter x Co-ordinates for player 1 (1-6)
2
Please Enter y Co-ordinates for player 1 (1-7)
4
0 - - - - - -
- - - X - - -
- - X - - - -
- - - - - - -
- - - - - - -
- - - - - - -

Please Enter x Co-ordinates for player 2 (1-6)
2
Please Enter y Co-ordinates for player 2 (1-7)
2
0 - - - - - -
- 0 - X - - -
- - X - - - -
- - - - - - -
- - - - - - -
- - - - - - -

Please Enter x Co-ordinates for player 1 (1-6)
4
Please Enter y Co-ordinates for player 1 (1-7)
2
0 - - - - - -
- 0 - X - - -
- - X - - - -
- X - - - - -
- - - - - - -
- - - - - - -

Player 1 has won the game!

Process finished with exit code 0
```