Programming (2)

Tic-tac-toe

Done By:

1] Name: Rowan Samy Gamal El-din Aly El-Rabeaa.

ID:6733

Group 4, section (2).

2] Name: Youmna Ahmed Ali Ibrahim Mohamed Omar.

ID:7161

Group 4, section (1).

Description:

This is tic-tac-toe game in which a grid board is displayed with dimensions 6*7 where 6 is the number of rows and 7 is the number of columns. Two players play against each other one after the other. Who completes connected 3 is the winner whether horizontally , vertically or diagonal. If no one wins the game therefore the game was draw.

Introduction:

First of all, we divided the package into separate classes where each class contains methods that has its own role in the program. So, we are going to explain each class including its methods and its aim:

1-Test:

It contains the main function where the execution of the program starts. In main function, we called all methods needed during execution from other classes. We used while (1) to allow the players access the grid after each input and then prints it to the screen, until one of the players win or the game draws. These methods are:

- Initialization.
- Display.
- Access.

2-Grid:

We declared a 2d array (grid) of dimensions 6*7. Then we added 2 public methods so they can be accessed from anywhere:

1-public static void initialization () :

Two for loops are being constructed to initialize an empty 2d grid.

2-public static void display () :

- First, the empty grid is printed.
- Then we initialized the variable end = 1 and we will loop on the grid array if it finds and empty element '-', it will assign zero to end which indicates that there are still empty elements in the array and the players can continue accessing it.
- If end remained =1 therefore all array elements are filled either with 'X' or 'O' and since none of them won so this indicates that it is a draw game.

3-Input:

This class has 2 functions:

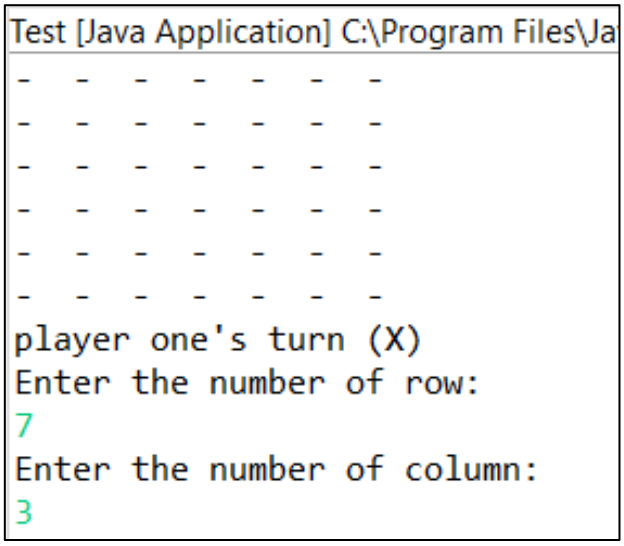
1-public static void input () :

- we initialized an object g1 of data type Grid so it can access all members of class Grid (like the struct).
- We imported a scanner so the user can enter the no of the row and the column he wants to access.
- We inserted a while loop to check the validity of all of the input where the number of the row and el column entered doesn't exceed the range of the array (6*7) and there is no negative values entered.
- If it spotted any invalid data; Invalid will be printed and will allow input to the user again.

Examples to invalid input:

1-Invalid row.

(greater than the range which is 6)



2-Invalid column.

(greater than range which is 7)

```
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player one's turn (X)
Enter the number of row:
4
Enter the number of column:
8
Invalid row/column please enter the place again
Enter the number of row:
4
Enter the number of column:
10
Invalid row/column please enter the place again
```

3-A non-positive number either in rows or columns.

(negative or zero)

```
Test [Java Application] C:\Program Files\Java\jdk-16\bin\javaw.exe
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player one's turn (X)
Enter the number of row:
-1
Enter the number of column:
3
Invalid row/column please enter the place again
Enter the number of row:
5
Enter the number of column:
0
Invalid row/column please enter the place again
Enter the number of row:
4
Enter the number of column:
-3
Invalid row/column please enter the place again
Enter the number of row:
```

Therefore, for a valid input:

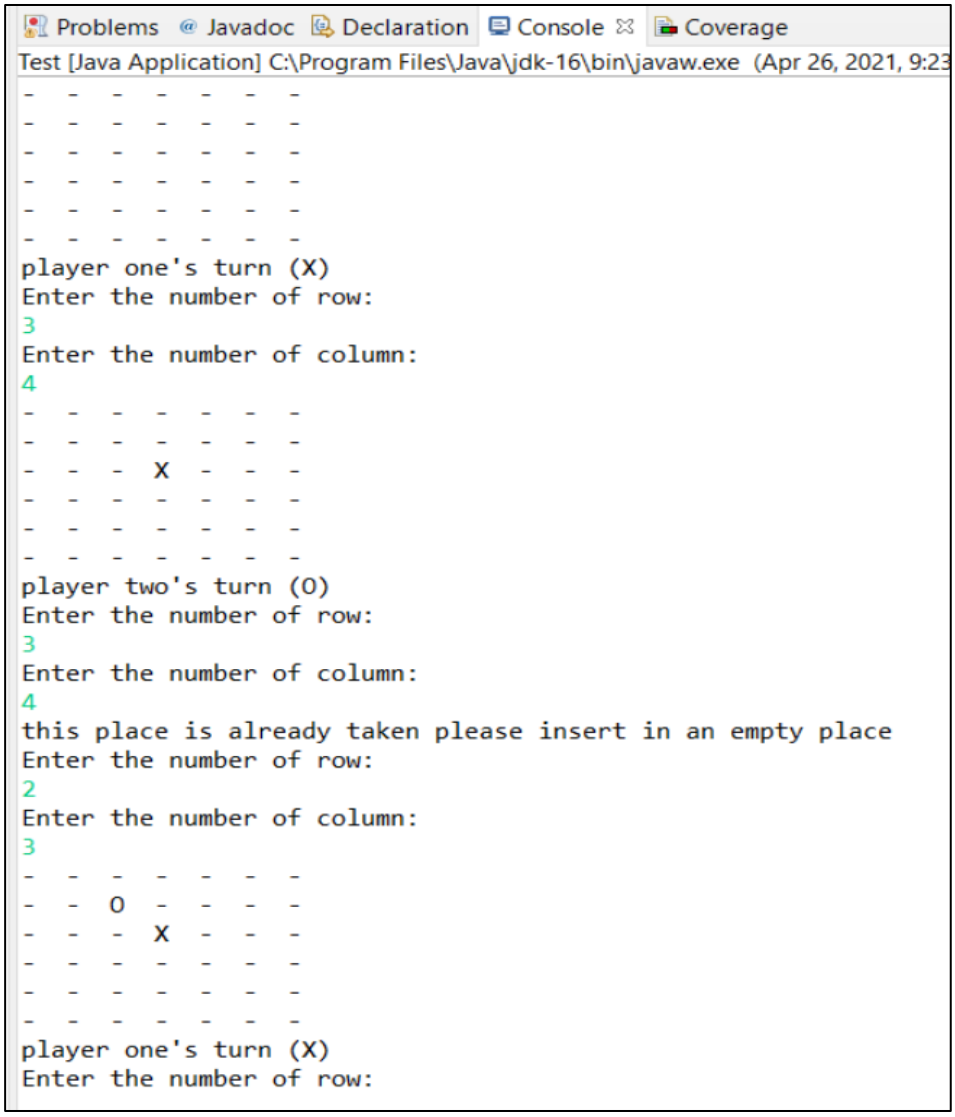
Range of rows from 1 till 6.

Range of columns from 1 till 7.

```
Test [Java Application] C:\Program Files\J
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player one's turn (X)
Enter the number of row:
2
Enter the number of column:
3
-  -  -  -  -  -  -
-  -  X  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player two's turn (O)
Enter the number of row:
4
Enter the number of column:
4
-  -  -  -  -  -  -
-  -  X  -  -  -  -
-  -  -  -  -  -  -
-  -  -  O  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player one's turn (X)
Enter the number of row:
```

2-public static void placeCheck () :

- It checks whether the place that the user enters is empty or not.
- If place is empty so no error, but if this place is already taken before then “this place is already taken” will be printed and will allow the user to input again.



4-CheckWinner:

It contains the function winner, its

Prototype: public static boolean winner (char symbol)

Description:

This is a public function of return type boolean (true or false). It receives X or O and checks after each move the player does for the rows, columns, diagonals, and cross diagonals. If it finds that any player completed any connected 3 therefore it returns true which means that this player wins else returns false.

Mechanism:

1-To check rows, it loops the grid and checks whether there are 3 consecutive same symbols “x or o only” or not. (‘-‘is not included as a symbol)

2-It keeps on repeating same mechanism for checking columns, diagonals, and cross diagonals.

3-If it finds any, it will return true which means that the player who has this repeated 3 consecutive symbol is a winner else it will return false

5-Modify:

This class includes public static void access (String player) function.

- It assigns the symbol x to the first player and assigns symbol y to the second player according to the string the function received.
- Then takes input from the user and checks the validity of its place using placeCheck() function.
- Then calls winner function if it returns true therefore player wins will be printed.

❖ User Manual:

1-When the game starts an empty grid is displayed.

2-Player one will start and will enter which row and column he would like to put his symbol “X” and the same for player two but with symbol “O”.

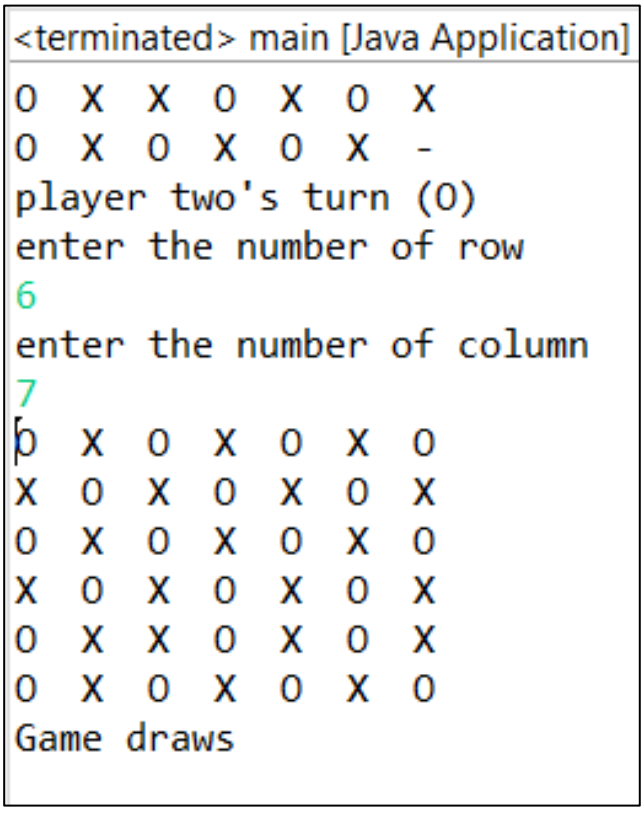
3-The game will continue exchanging turns between player 1 and player 2 till on of the following two cases happens.

First case:

One of the players connected 3 either horizontally or vertically or diagonally. (like next sample test case)

Second case:

The 2 players keep exchanging turns till all the grid is completely full with X and O and the game draws.



❖ Sample Test Case:

1-The first player inserts “X” in the second row and in the second column.

2- The second player inserts “O” in the second row and in the first column.

3- The first player inserts “X” in the first row and in the first column.

4- The second player inserts “O” in the third row and in the third column.

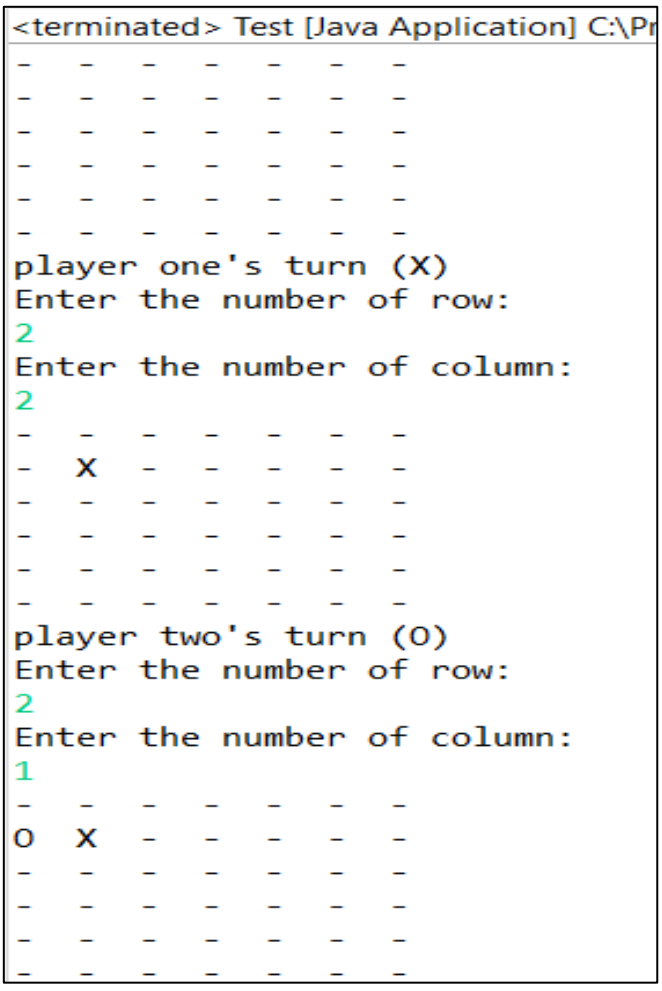
5-The first player inserts “X” in the first row and in the third column.

6- The second player inserts “O” in the third row and in the first column.

7- The first player inserts “X” in the first row and in the second column.

8- End the game and display first player is the winner

1 and 2-



3 and 4 -

```
player one's turn (X)
Enter the number of row:
1
Enter the number of column:
1
X  -  -  -  -  -  -
O  X  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player two's turn (O)
Enter the number of row:
3
Enter the number of column:
3
X  -  -  -  -  -  -
O  X  -  -  -  -  -
-  -  O  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
```

5 and 6 and 7 and 8-

```
player one's turn (X)
Enter the number of row:
1
Enter the number of column:
3
X  -  X  -  -  -  -
O  X  -  -  -  -  -
-  -  O  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player two's turn (O)
Enter the number of row:
3
Enter the number of column:
1
X  -  X  -  -  -  -
O  X  -  -  -  -  -
O  -  O  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player one's turn (X)
Enter the number of row:
1
Enter the number of column:
2
X  X  X  -  -  -  -
O  X  -  -  -  -  -
O  -  O  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
-  -  -  -  -  -  -
player one wins
```