Tic-tac-toe Game

Introduction:

The program is to implement a Tic-tac-toe board game with dimensions 6*7 where 6 is the number of rows and 7 is the number of columns by using java. The program uses OOP as it is divided into three classes NewAssignment.java, Player.java and Table.java.

1.Player:

Class player contains two attributes which are win and turn.

2.Table:

Class table contain a 2D array with the required dimensions 6*7 where 6 is the number of rows and 7 is the number of columns which represents the board.

2.1.printTable():

The method printable prints the 2D array using two for loops as shown in code 2.1 to print the board

```
for (i = 0; i < 6; i++) {
    for (j = 0; j < 7; j++) {
System.out.print(a[i][j] + " ");
        }
    System.out.println();
        }
```

Code 2.1

2.2 input():

This method takes the input from the user using simple GUI techniques by JOptionPane.showInputDialog and javax.swing.JOptionPane must be imported and by using do while loop to loop until there is a winner or till the game is draw. The do while loops till the variable win is equal to 1. The win variable depend on the four check which are defined as methods ,in order to determine if the last input from the user will make him win the game, which are checkRow() ,checkColumn(), checkDiagonal (int r ,int c) and checkDiagonal2(int r, int c)as shown in code 2.2. Then it converts the input from string to integer and checks if it is a valid input, within the dimensions and isn't chosen before by the user. If the input isn't valid, it asks the user to enter a valid input. The method also contains a variable called full which is incremented every time the user enters an input so when there is no winner the program prints out that the game is draw by checking if full surpassed 42.

```
win = checkRow() || checkColumn() || checkDiagonal(row, column) ||
checkDiagonal2(row, column);
```

code 2.2

2.3 edit():

The method takes two parameters row and column to modify the 2D array then it prints the table again for the user to see his choice on the board as shown in code 2.3.

```
public void edit(int row,
int column, int a[][]) {


a[row][column] = turn;


System.out.println();
 printTable();}
```

Code 2.3

The main Algorithms used in class Table

2.4 checkRow():

This method checks if there is a winner who has entered three connected horizontal same input for example three consecutive 1 and it has a Boolean return value. The method uses two for loops one loop loops through the rows and another loops through the columns. The one which loops through the columns checks if the element in the array is equal to the variable turn which is either 1 or 2 if it is true it increments the variable count if the count becomes equal to 3 it returns true and the player is announced as a winner .However, if one of the consecutive elements isn't equal to the turn the count is set to zero.

2.5 checkColumn():

This method checks if there is a winner who has entered three connected vertical same input for example three consecutive 1 and it has a Boolean return value. The method uses two for loops one loop loops through the rows and another loops through the columns. The one which loops through the rows checks if the element in the array is equal to the variable turn which is either 1 or 2 if it is true it increments the variable count if the count becomes equal to 3 it returns true and the player is announced as a winner .However, if one of the consecutive elements isn't equal to the turn the count is set to zero.

2.6 checkdDiagonal(int r, int c):

This method takes two parameter row and columns. Then it loops while the row and the column aren't equal to zero if any of them is equal to zero the loop ends

In this loop it moves to the first element in the diagonal by decrementing the row and column each by 1 in each loop. Then there is another loop which loops till the end of the diagonal by incrementing the row and column in each loop. In this loop it checks if there is 3 consecutive inputs which are equal to the same value if true the count is incremented till it is equal three then it returns true else it returns false.

2.7 checkDiagonal2(int r, int c):

This method takes two parameter row and columns. Then it loops while the row is not equal to zero and while column isn't equal to six if any of them breaks its condition the loop ends .In this loop it moves to the first element in the second diagonal by decrementing the row by one but incrementing column by 1 in each loop. There is another while loop which loops while column is greater than or equal zero and while row is less than or equal five to loop from the start of the diagonal to its end by incrementing the row and decrementing the column and check every element if it is equal to the turn if it is equal the count is incremented by 1 but if one values isn't equal to the turn the count is set to zero as it doesn't satisfy the condition as shown in code 2.4.

```
while (c >= 0 && r <= 5) {
        if (a[r][c] == turn) {
           count++;
           if (count == 3) {
              return true;

           }
        } else {
           count = 0;
        }
        r++;
        c--;
     }
```

Code 2.4

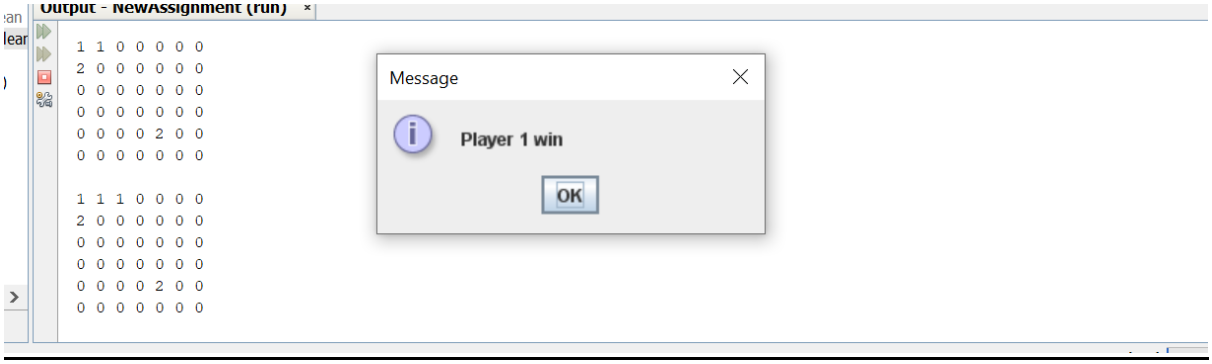
3.1 NewAssignment classs:

In the main an object from class table is created then the functions printTable()

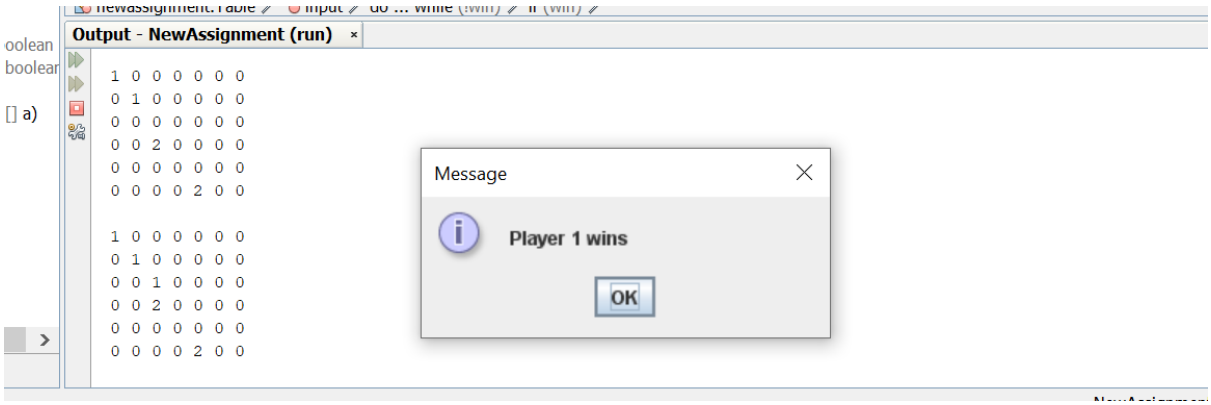
And input are called using this object.

Sample Runs:

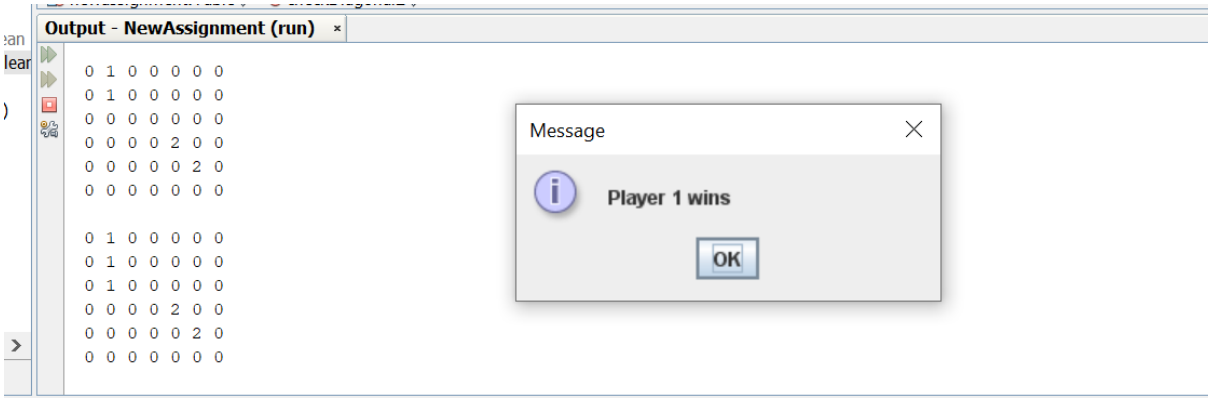
1) Row check



2) First diagonal check



3) column check



4) Second diagonal check