1.Problem Description

We are asked to create a two-player board game. The board contains 36 pieces that are organized in a 6x6 square. Each piece can be blank ("B"), an asterisk ("X") or a zero ("O"). In the default case, the game starts with each piece showing blank. One of the players is called an "X" player and cannot play with "O" pieces and the other player is called an "O" and cannot play with "X" pieces. "B" pieces can be played by both players. A player can play with a piece on the outer square of the board. The game proceeds in a turn-taking fashion.
At each round, a player can do one of the following actions:
Change: If a piece is blank, then the user can change it to "X" if she is the "X" user and change it to "O" if she is the "O" user. A piece that shows an "X" or an "O" cannot be switched back to another label.

Move: The "X" user can take an "X" piece; the "O" user can take an "O" piece, and relocate that piece all the way to the other end of the row or the column. When that is done, the remaining pieces are pushed to the other way.

The end of the game: The "X" player wins the game if she has a row or a column of "X"s. The "O" player wins the game if she has a row or a column of "O"s. At the end of the game we should print who is the winner.

During solving this problem we should:
Ask the user if she wants to start a new game or if she wants to load the board from a previous game then continue. If the answer is a new game, load the board as full of "B" s. Otherwise, read the board configuration from an input file (called input.txt). In both cases, we should display it on the screen.
After each move, we should re-display the current configuration.

Game play: The user will make the moves based on a coordinate system, where 0 0 is the top left cornerand 5 5 is the right bottom corner. To make a "change" move user should say C 0 0(C for change 0 0 for coordinates) for example. By the way we should make sure that there is a space between each.
To make a "move" action user should say M 5 0 0 0 (where 5 0 denotes current, 0 0 denotes target coordinates and M for move) for example. When a user enters the action, we should first check whether this is possible. After the user plays, our program will take an action

2.Problem Solution

First of all I asked user to decide whether she wants to use a new or old board by an if statement. Then according to the answer, I create our board's configuration by **2 for loops**; by reading from an input or filling everywhere with "B"s into an array. Then I created **a method "printBoard"** in order to print our board.To do so it used **2 for loops** and printed each character of our array and created a table.After creating our board, we should have let the user can make moves. In order to do that I created an huge **while loop** where user makes moves untill game is over. In order to check this I created **a method named "checkedFinished"** which horizontally and vertically checks the board when we want. It does so by **2 for loops** for columns and **2 for loops** for rows. When it finds 6 same characters(except "B") consecutively, it returns 1 for "X" 2 for "O".If it never find such 6 characters, it returns 3 which means game still continues.

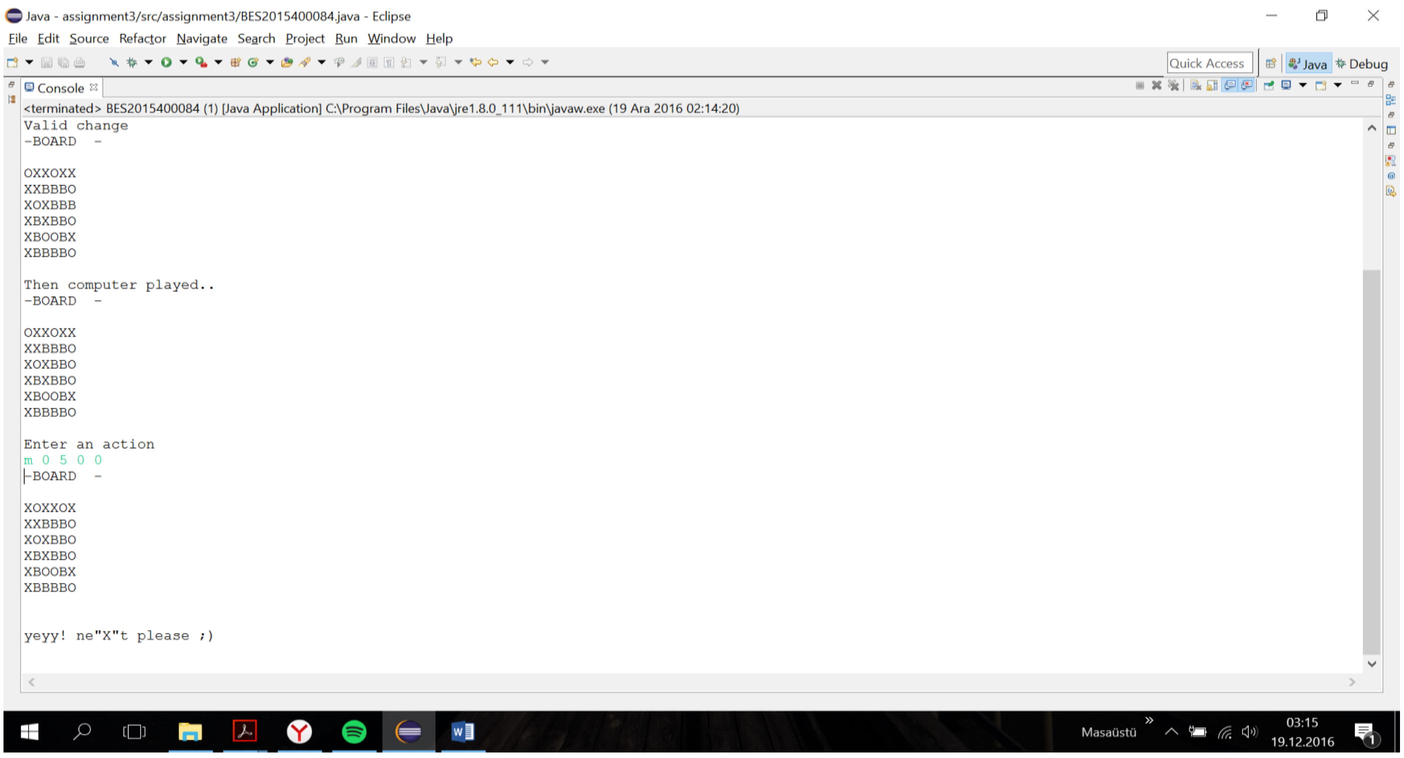
Inside that while loop a created a **boolean "playabilty"** which changes value by upcoming move tryings. I created an if statement which defines whose turn by a counter.Inside that statement, I created a **while loop** to let my user does her moves until she makes a valid move. Inside that loop I took my users instructions and controlled each then made them. I defined my user's wants by a nested if statement then took action according to it.

When user wanted to make a change move I got into **a method called "change"(is a boolean)** where I first controled user's input's coordinates by an another **method called "outerCheck"(boolean)** (which includes an another **method called "isInside(boolean)** which checks whether coordinates are inside the borders of our board)which checks whether our coordinates are on the outer shell of the board. That "change" method changes a "B" to "X" or "O"(which one is picked).

When the user wanted to make a "move" action I got into **a method named "move"(is a boolean)** where I first contolled user's input's coordinates by outerCheck method then **notMovable method(is a boolean)** which is an another method that controles the coordinates availability to move by a nested if statement. The main purpose of the "move" method is making the "move" action defined above(in description part). When coordinates are proper method does that action by totally **4 for loops** inside checker if statemens. Two for loops for shifting in a row, the other two are for shifting in a column. After the user makes her move, the turn passes to the computer thanks to the playability method. After each move printBoard method prints the board and checkFinished method checks wheter game is finished or not. When computer makes a move everything, except defining the coordinates and type of the action, is the same. When defining the type of the action I used a random; then took the coordinates by using the random either.

In total, I used 15 for loops during these project.

3.Output of the program:


Java - assignment3/src/assignment3/BES2015400084.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java
Debug
Console
Masaüstü
03:15
19.12.2016

```
<terminated> BES2015400084 (1) [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (19 Ara 2016 02:14:20)
Valid change
-BOARD  -

OXXOXX
XXBBBO
XOXBBB
XBXBBO
XBOOBX
XBBBBO

Then computer played..
-BOARD  -

OXXOXX
XXBBBO
XOXBBO
XBXBBO
XBOOBX
XBBBBO

Enter an action
m 0 5 0 0
-BOARD  -

XOXXOX
XXBBBO
XOXBBO
XBXBBO
XBOOBX
XBBBBO


yeyy! ne"X"t please ;)
```