1: Problem Description:

This project is a game where a player and a computer play tic tac toe on a 4x4 board until one of them wins the game or draw. Pc playes are all randomized. At the end of the game player can start a whole new game or end the program where it also shows the scoreboard. Also, a save file can be loaded at the beginning of the game to continue possible saved games.

2: Problem Solution:

First of all, the board can be printed via the Line 9 board() method. It prints 4 lines, each carrying the current rows on the board.

Line 24-70 The isItDraw() and winCondition() methods check if the game should end or not, respectively by draw or win. isItDraw() checks if there are any empty “E” coordinates left or not, winCondition() checks if there are 3 “X” s or “O” s in a row, column or diagonal.

Line93 play() method simply plays a given coordinate to the board, returns true if succeeds. Possible false returns include things like a wrong coordinate or an already played coordinate.

Line 130 pcPlay() chooses a random coordinate and plays it with play(), if play() returns false it repeats until true is returned

Line 59 howManyTimes() checks how many times a Symbol was played on the board, this method helps to check any corrupted save files.

Line 32 isItCorrupted() checks if the save file is corrupted or not by every possible corruption.

Possible corruptions include lines having more or less than 4 symbols or symbols other than “X” and “O” etc... loading an already won game is also considered as a possible corruption.

The main method:

First in the main method, we see the Scanner and Random as well as variables we’re gonna need for the game, each explained. The first loop we see it the pregame do-while loop that asks to create a game or load. Choosing to create a game asks us for our symbol, and the game starts. Choosing to load a game ask us the file’s name. if file is corrupted of unavailable, we can choose to create a new game or to load another file. Loop makes sure the inputs are right as any wrong input gives a warning and lets us write another input. End of the loop, game is ready to be played.

The second loop we see is the while(playAgain.equals(“Y”) where the actual game happens. Since playAgain is “Y” by default, we enter the loop. If a file was loaded, we get the turn to play automatically since if we were also asked to save, we could've saved only in our turn. If a new game was created, the starter is chosen by random.

The third loop is the turns' loop while(!gameOver). Respectively it:

- Prints the board.
- Lets us choose a coordinate.
- If the input is wrong, warns us and lets us choose another coordinate.
- If the input is available, it plays it.
- Checks if it is draw.
- Checks if we won.
- Lets pc to play.
- Checks if it is draw.
- Checks if pc has won.

Only was out of the loop is to win or draw. Each checking has its own gameOver=true and break to get out of the loop.

At the end of the game, we can get out of the actual game loop while(playAgain.equals("Y") by writing "N". if "Y" is written, system resets the game and loop continues.

After writing "N", the game shows us the last scoreboard and thanks us for playing!

Note that save file includes 4 lines, player's symbol, player's scoreboard and pc's scoreboard. Each is being read by .nextLine() method but the last 2. Last 2 is being read by nextInt().

An example for the save file is:

```
EXEE
XEEE
OXOE
EEEO
X
4 2
```

3: Implementation:

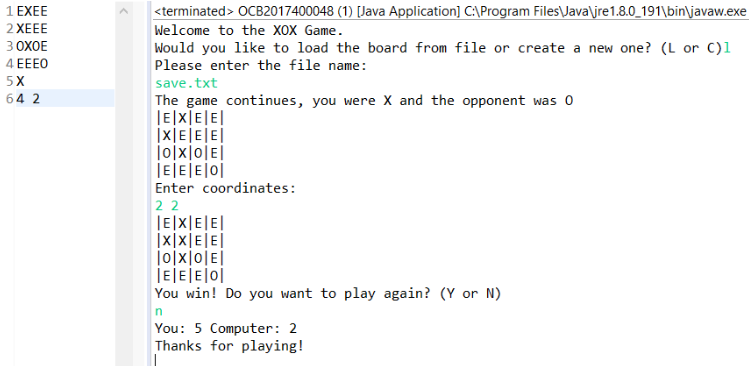
```
import java.util.Scanner;
import java.io.*;
import java.util.Random;
public class OCB2017400048 {
    static String line1 = "EEEE";
    static String line2 = "EEEE";
    static String line3 = "EEEE";
    static String line4 = "EEEE";
    public static void board(String line1, String line2, String line3, String line4) {
        //it prints the board with the help of boardLinePrinter
        boardLinePrinter(line1);
        boardLinePrinter(line2);
        boardLinePrinter(line3);
        boardLinePrinter(line4);


    }
    public static void boardLinePrinter(String line) {
        //board method was going to be too repetative so i made this method
        for(int i=0;i<4;i++) System.out.print("|"+line.charAt(i));
                System.out.println("|");



    }
    public static boolean isItDraw(String line1, String line2, String line3, String line4) {
        //this method simply checks if the game is draw, if no E left on the board returns true
        boolean noE=false;
        for (int i=0;i<4;i++) {
            noE = noE&&(line1.charAt(i)!='E'&&line2.charAt(i)!='E'&&line3.charAt(i)!='E'&&line4.charAt(i)!='E');
        }
        return noE;
    }
    public static boolean isItCorrupted (String line1,String line2,String line3,String line4,String PlayerSymbol,String pcSymbol, int youCounter, int
pcCounter) {


        // this part is to detect any possible corrupted save files, each returns true if finds a problem
        for (int i=0;i<4;i++){
            if (line1.charAt(i)!='E'&&line1.charAt(i)!='X'&&line1.charAt(i)!='O'||
                line2.charAt(i)!='E'&&line2.charAt(i)!='X'&&line2.charAt(i)!='O'||
                line3.charAt(i)!='E'&&line3.charAt(i)!='X'&&line3.charAt(i)!='O'||
                line4.charAt(i)!='E'&&line4.charAt(i)!='X'&&line4.charAt(i)!='O')
                return true;
        }
        if (winCondition(line1,line2,line3,line4)) return true;
        if (line1.length()!=4||line2.length()!=4||line3.length()!=4||line4.length()!=4) return true;
        if (youCounter<0||pcCounter<0) return true;

        //This one makes sure the player or the pc doesnt have illegal amounts of Symbols on the board
        if((howManyTimes(line1,line2,line3,line4,PlayerSymbol)>howManyTimes(line1,line2,line3,line4,pcSymbol)||
            howManyTimes(line1,line2,line3,line4,PlayerSymbol)+1<howManyTimes(line1,line2,line3,line4,pcSymbol))) {
            return true;
            //boolean part of if means !(if PlayerSymbol is equal to OpponentSymbol or PlayerSymbol is one smaller than OpponentSymbol)
        }




        return false;
    }
    public static int howManyTimes(String line1, String line2, String line3, String line4, String Symbol) {
        //this method counts how many "Symbol" 's are on the board
        int howMany=0;
        for (int i=0;i<4;i++) {
            if (line1.charAt(i)==Symbol.charAt(0)) howMany++;
            if (line2.charAt(i)==Symbol.charAt(0)) howMany++;
            if (line3.charAt(i)==Symbol.charAt(0)) howMany++;
            if (line4.charAt(i)==Symbol.charAt(0)) howMany++;
        }
        return howMany;
    }
    public static boolean winCondition(String line1,String line2,String line3,String line4) {
        //this method checks if the is won by someone, returns true if three symbols are in a row
        for (int i=0;i<4;i++) {//rows
            if(line1.charAt(i)==line2.charAt(i)&&line2.charAt(i)==line3.charAt(i)&&line1.charAt(i)!='E') return true;
            if(line2.charAt(i)==line3.charAt(i)&&line3.charAt(i)==line4.charAt(i)&&line2.charAt(i)!='E') return true;
        }
        //lines
        if (line1.contains("XXX")||line1.contains("OOO")) return true;
        if (line2.contains("XXX")||line2.contains("OOO")) return true;
        if (line3.contains("XXX")||line3.contains("OOO")) return true;
        if (line4.contains("XXX")||line4.contains("OOO")) return true;

        //cross lines
        for (int i=0;i<2;i++) {
            if(line1.charAt(i)==line2.charAt(i+1)&&line2.charAt(i+1)==line3.charAt(i+2)&&line1.charAt(i)!='E') return true;
            if(line2.charAt(i)==line3.charAt(i+1)&&line3.charAt(i+1)==line4.charAt(i+2)&&line2.charAt(i)!='E') return true;
        }
        for (int i=3;i>1;i--) {
            if(line1.charAt(i)==line2.charAt(i-1)&&line2.charAt(i-1)==line3.charAt(i-2)&&line1.charAt(i)!='E') return true;
            if(line2.charAt(i)==line3.charAt(i-1)&&line3.charAt(i-1)==line4.charAt(i-2)&&line2.charAt(i)!='E') return true;
        }
        return false;
    }
    public static boolean play(String l1, String l2, String l3, String l4, int x_coordinate, int vertical, String Symbol) {
        //this method plays the players or the computers coordinates to the board, returns true if succeeds
        if (x_coordinate>=4||x_coordinate<0) return false;

            switch (vertical) {

                case 0 :
                    if (l1.charAt(x_coordinate)=='E') {
                        line1=l1.substring(0, x_coordinate)+Symbol+l1.substring(x_coordinate+1);
                        return true;
                    }

                    break;
                case 1 :
                    if (l2.charAt(x_coordinate)=='E') {
                        line2=l2.substring(0, x_coordinate)+Symbol+l2.substring(x_coordinate+1);
                        return true;
                    }
                    break;
                case 2 :
                    if (l3.charAt(x_coordinate)=='E') {
                        line3=l3.substring(0, x_coordinate)+Symbol+l3.substring(x_coordinate+1);
                        return true;
                    }
                break;
            case 3 :
                if (l4.charAt(x_coordinate)=='E') {
                    line4=l4.substring(0, x_coordinate)+Symbol+l4.substring(x_coordinate+1);
                    return true;
                }
                break;

            }
        return false;
    }


public static void pcPlay(String line1,String line2,String line3,String line4,Random random, String Symbol) {
    //this method randomly plays the pc coordinates untill it randoms a spot that has not been played
    while(true) {
        int vertical =random.nextInt(4);
        int horizontal = random.nextInt(4);
        play(line1,line2,line3,line4,horizontal,vertical,Symbol);
        if (play(line1,line2,line3,line4,horizontal,vertical,Symbol)) break;
    }

}
public static void main(String[]args) throws FileNotFoundException {
    int youCounter = 0;//counts how many games you won.
    int pcCounter = 0;//counts how many games pc has won.
    String playAgain="Y";//Y by default, helps for the game loop at the very end (Do you want to play again? (Y or N)).
    String PlayerSymbol="";//Player's symbol, X or O.
    String OpponentSymbol="";//pc's symbol, the opposite of playerSymbol.
    boolean youWon=false;//at the very end, if you win this becomes true. If pc wins, stays false.
    boolean isItDraw=false;//checks if it is draw or not.
    boolean gameOver=false;//stops the game.
    boolean corruptedFile=false;//in the load section, true if file is corrupted.
    boolean fileNotAvailable=false;//true if system cant find the file.
    boolean loop=false;//true if player enters stuff other than (L,C), resets to false at the start of 156-227 do while loop
    Random random = new Random();
    Scanner console= new Scanner(System.in);
    System.out.print("Welcome to the XOX Game.\nWould you like to load the board from file or create a new one? (L or C)");
    String saveLoad = console.next();
```

```
        do{
            corruptedFile=false;
            fileNotAvailable=false;
            loop=false;
            //those 3 booleans have to reset every loop
            if  (saveLoad.equalsIgnoreCase("L")) {
                //if input is L
                System.out.println("Please enter the file name: ") ;
                String saveInput = console.next();
                File saveFile = new File(saveInput);
                if(!saveFile.exists()) {
                    fileNotAvailable=true;
                }
                else {
                    Scanner save = new Scanner(saveFile);
                    line1= save.nextLine();
                    line2=save.nextLine();
                    line3=save.nextLine();
                    line4=save.nextLine();
                    PlayerSymbol=save.nextLine();
                    youCounter=save.nextInt();
                    pcCounter=save.nextInt();
                    if (PlayerSymbol.equals("X")) OpponentSymbol="O";
                    else if (PlayerSymbol.equals("O")) OpponentSymbol="X";
                    else corruptedFile=true;
                    save.close();
                }
                if (!fileNotAvailable) {
                    if  (isItCorrupted(line1,line2,line3,line4,PlayerSymbol,OpponentSymbol,youCounter,pcCounter))
corruptedFile=true;
                }
            }
            else if (saveLoad.equalsIgnoreCase("C")) {
                //if the input is C
                System.out.println("Enter your symbol: (X or O) ");
                PlayerSymbol = console.next().toUpperCase();
                while (true){
                    if (PlayerSymbol.equals("X")||PlayerSymbol.equals("O")) break;
                    System.out.println("Please use only \"X\" or \"O\"");
                    PlayerSymbol=console.next().toUpperCase();
        }
        if (PlayerSymbol.equals("X")) {
            OpponentSymbol="O";
        }
        else {
            OpponentSymbol="X";
        }
        System.out.println("You are player "+PlayerSymbol+" and the computer is "+OpponentSymbol+".");
        break; //this break which is the only way out of this while loop for "C"
             //makes sure player is X or O, and the pc is the exact opposite
    }
    else {
        //if the input is something other than L or C
        System.out.println("Please use only \"L\" or \"C\" !");
        saveLoad = console.next();
        loop=true;
        continue;
    }
    if (corruptedFile) {
        //false by default, true if player loads a corrupted file
        System.out.println("The load file is corrupted, please use another file (L) or create your own game (C)");
        line1="EEEE";
        line2="EEEE";
        line3="EEEE";
        line4="EEEE";
        saveLoad = console.next();
    }
    if (fileNotAvailable) {
        //false by default, true if player loads an unavailable file
        System.out.println("Unable to reach the file, please use another file (L) or create your own game (C)");
        saveLoad = console.next();
    }
} while (corruptedFile||fileNotAvailable||loop);
//loops if an input is wrong, only loops for "L" input


while (playAgain.equals("Y")) {
    //This is the start of the main game
    int rng = random.nextInt(2);
```

```
if (saveLoad.equalsIgnoreCase("L")) {
    rng=0;//Since it can only be saved while its your turn, you have to start if you load a game
    System.out.println("The game continues, you were "+PlayerSymbol+" and the opponent was "+OpponentSymbol);
}
else {
    //only if a new game is created (rng is random by default)
    if (rng%2==0) {
        System.out.println("You will start.");
    }
    else {
        System.out.println("Computer will start");
        pcPlay(line1,line2,line3,line4,random,OpponentSymbol);
    }
}
while(!gameOver) {
    //this part loops untill someone wins or draw
    board(line1,line2,line3,line4);
    System.out.println("Enter coordinates:");
    int temp1=0;
    int temp2=0;
    boolean s = true;
    do {
        while(true) {
            if (console.hasNextInt()){
                temp1=console.nextInt()-1;
                break;
            }
            console.next();
        }
        while(true) {
            if (console.hasNextInt()){
                temp2=console.nextInt()-1;
                break;
            }
            console.next();
        }//coordinate is written, makes sure only integers are inputs
        s = play(line1,line2,line3,line4,temp1,temp2,PlayerSymbol);
        if(!s) System.out.println("Please write a number that is available or not used");
        //if coordinates are unavailable
    } while (!s); //getting out of the loop means the coordinates are accepted


    if (winCondition(line1,line2,line3,line4)) {
        //if you win
        youWon=true;
        gameOver=true;
        break;
    }
    if (isItDraw(line1,line2,line3,line4)) {
        //if its draw after you played
        gameOver=true;
        isItDraw=true;
        break;
    }



    pcPlay(line1,line2,line3,line4,random,OpponentSymbol);
    if (winCondition(line1,line2,line3,line4)) {
        //if pc wins
        gameOver=true;
        break;
    }
    if (isItDraw(line1,line2,line3,line4)) {
        //if its draw after pc plays
        isItDraw=true;
        gameOver=true;
        break;

    }
        //and loop continues for another turn
}

board(line1,line2,line3,line4);

if (isItDraw)  System.out.println("It's a draw! How did you do that?\nDo you want to play again? (Y or N)");
else {

    if (youWon) {
        youCounter++;
```

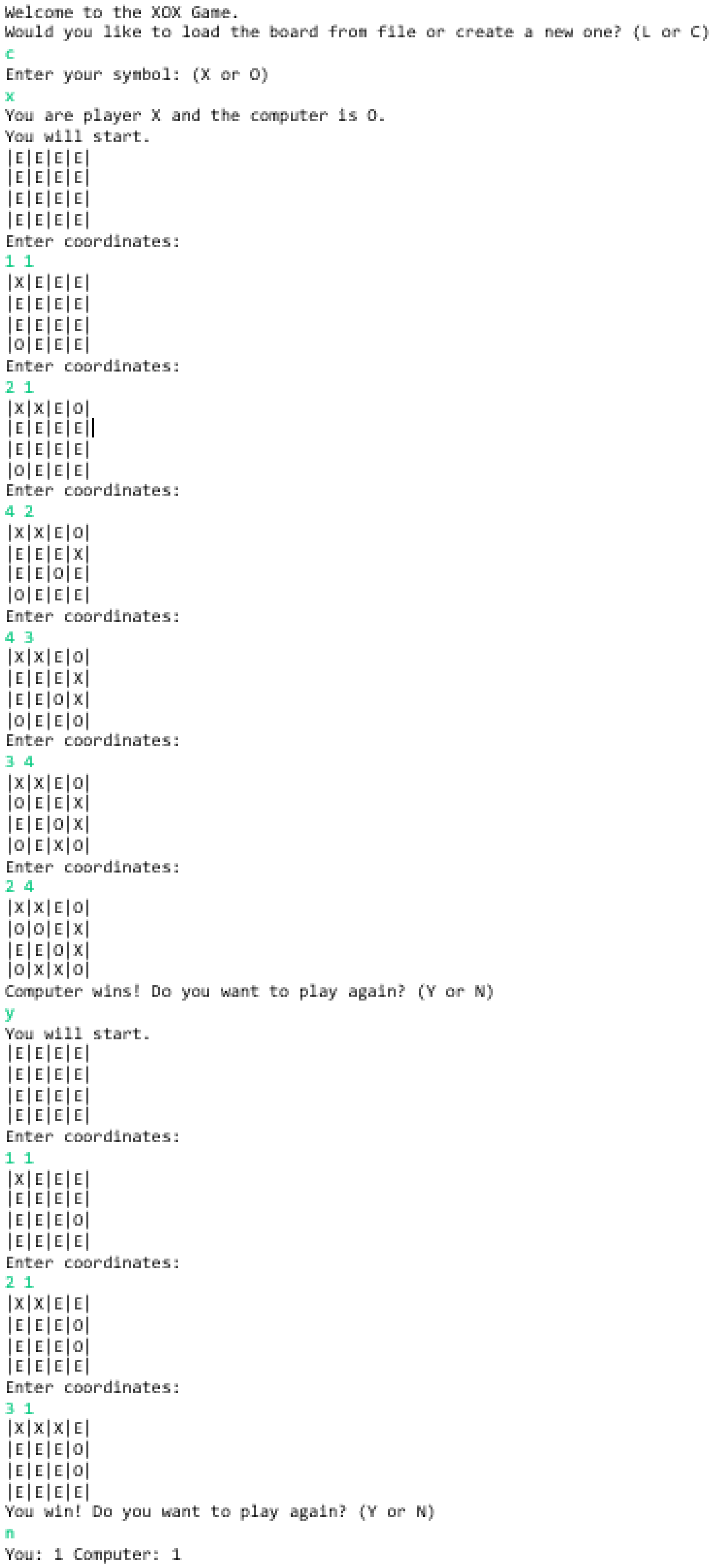
```
                    System.out.println("You win! Do you want to play again? (Y or N)");
                }
                else {
                    pcCounter++;
                    System.out.println("Computer wins! Do you want to play again? (Y or N)");
                }
            }
            //3 possibilities with a line for each
            do {
                playAgain = console.next().toUpperCase();
            }while (!playAgain.equals("Y")&&!playAgain.equals("N"));
            if (playAgain.equals("Y")){
                //if player writes Y, it resets the game
                line1="EEEE";
                line2="EEEE";
                line3="EEEE";
                line4="EEEE";
                youWon=false;
                gameOver=false;
                saveLoad="C";
            }
            //if player writes N, it gets out of the main game loop
            //it loops untill (Y or N) is written so there is no need to worry about other inputs
        }


        System.out.println("You: "+youCounter+" Computer: "+pcCounter+"\nThanks for playing!");
        //youCounter and pcCounter is also saved in the file so scores continues from where they were
        console.close();
    }
}
```

4: Output of the Program:



```
EXEE
XEEE
OXOE
EEEO
X
4 2
```

```
<terminated> OCB2017400048 (1) [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe
Welcome to the XOX Game.
Would you like to load the board from file or create a new one? (L or C)l
Please enter the file name:
save.txt
The game continues, you were X and the opponent was O
|E|X|E|E|
|X|E|E|E|
|O|X|O|E|
|E|E|E|O|
Enter coordinates:
2 2
|E|X|E|E|
|X|X|E|E|
|O|X|O|E|
|E|E|E|O|
You win! Do you want to play again? (Y or N)
n
You: 5 Computer: 2
Thanks for playing!
```



```
Welcome to the XOX Game.
Would you like to load the board from file or create a new one? (L or C)
c
Enter your symbol: (X or O)
x
You are player X and the computer is O.
You will start.
|E|E|E|E|
|E|E|E|E|
|E|E|E|E|
|E|E|E|E|
Enter coordinates:
1 1
|X|E|E|E|
|E|E|E|E|
|E|E|E|E|
|O|E|E|E|
Enter coordinates:
2 1
|X|X|E|O|
|E|E|E|E|
|E|E|E|E|
|O|E|E|E|
Enter coordinates:
4 2
|X|X|E|O|
|E|E|E|X|
|E|E|O|E|
|O|E|E|E|
Enter coordinates:
4 3
|X|X|E|O|
|E|E|E|X|
|E|E|O|X|
|O|E|E|O|
Enter coordinates:
3 4
|X|X|E|O|
|O|E|E|X|
|E|E|O|X|
|O|E|X|O|
Enter coordinates:
2 4
|X|X|E|O|
|O|O|E|X|
|E|E|O|X|
|O|X|X|O|
Computer wins! Do you want to play again? (Y or N)
y
You will start.
|E|E|E|E|
|E|E|E|E|
|E|E|E|E|
|E|E|E|E|
Enter coordinates:
1 1
|X|E|E|E|
|E|E|E|E|
|E|E|E|O|
|E|E|E|E|
Enter coordinates:
2 1
|X|X|E|E|
|E|E|E|O|
|E|E|E|O|
|E|E|E|E|
Enter coordinates:
3 1
|X|X|X|E|
|E|E|E|O|
|E|E|E|O|
|E|E|E|E|
You win! Do you want to play again? (Y or N)
n
You: 1 Computer: 1
```

5: Conclusion:

The project was extremely confusing, and I'm quite sure some variables are so similar that they can be reduced. There were some vague things we were asked to do so I used my imagination to make the game as efficient as possible, especially at the loaded file. The requirement might've been to only load the board but I choose to load my symbol and the scoreboards as well since if it was a real save file they were also known. I thought you can only save if its your turn so it starts as your turn if you load any file. Because of that, an already won game is considered to be corrupted.

I tried my best to find any crashes or bugs, there might be still some left but not that I'm aware of. There is the requirement of entering 2 integers after the "Enter coordinates:" part, it might look glitched but its fine and the program will continue if 2 integers are written.

Also, the C-L, X-O, Y-N stuff that needs to be written are all case insensitive.