CMPE 150 Introduction to Computing, Fall 2017 - Project 3 Project Report

Problem Description:

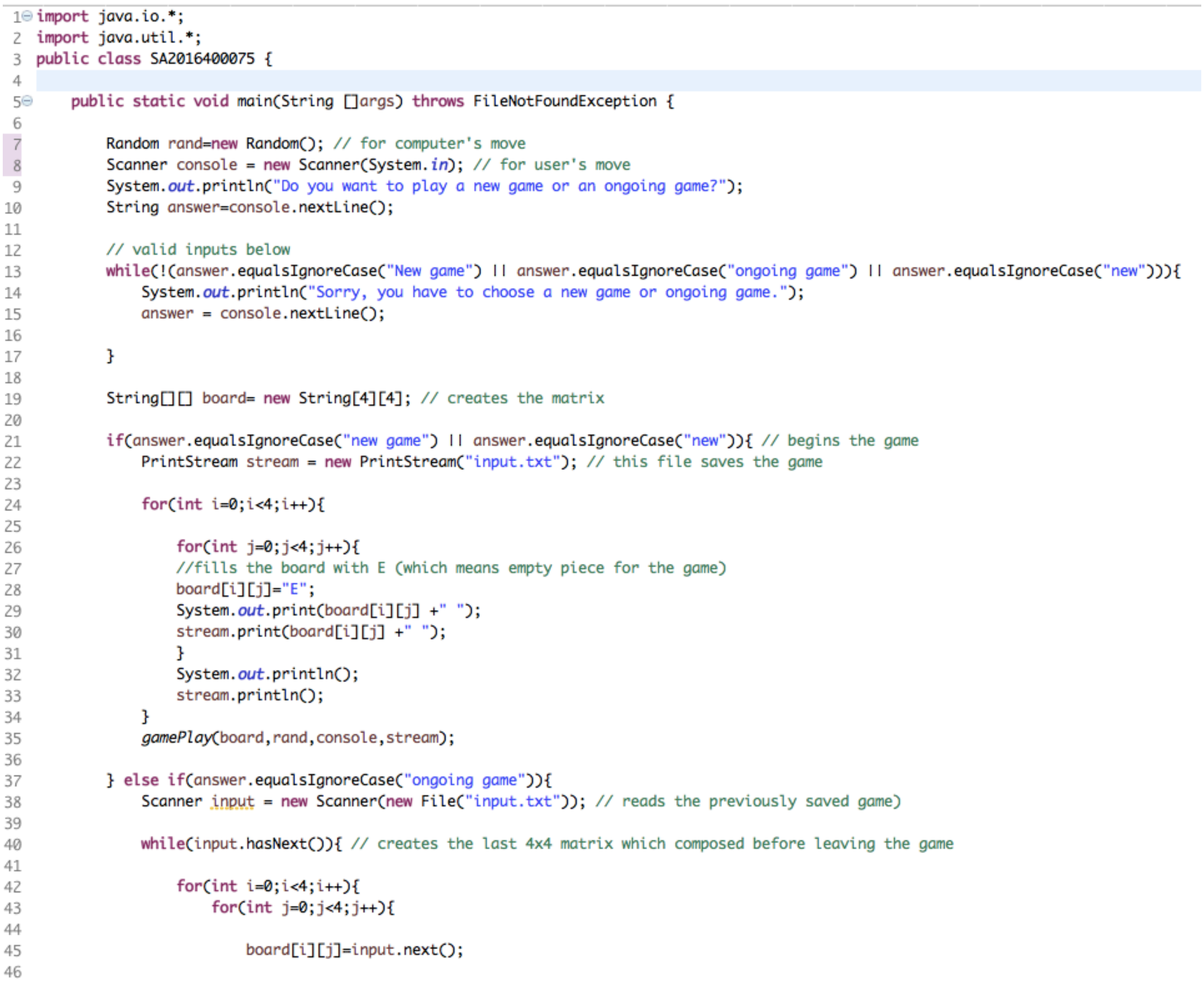
The problem is writing a java program which plays quarto game with the user. The program should ask the user if he wants to start a new game or if he wants to continue an ongoing game. If the answer is a new game, the program should load the board. Otherwise, it should read the board configuration from an input file. In both cases, it should displays it on the screen. After each move, it re-displays the current configuration. Additionally, the program should ask the user to pick a piece and if the piece is already on the board, it should ask the user to pick another piece. And then, ask the user to place it on the board by entering coordinates. At the end of the game print who has won.

Problem Solution:

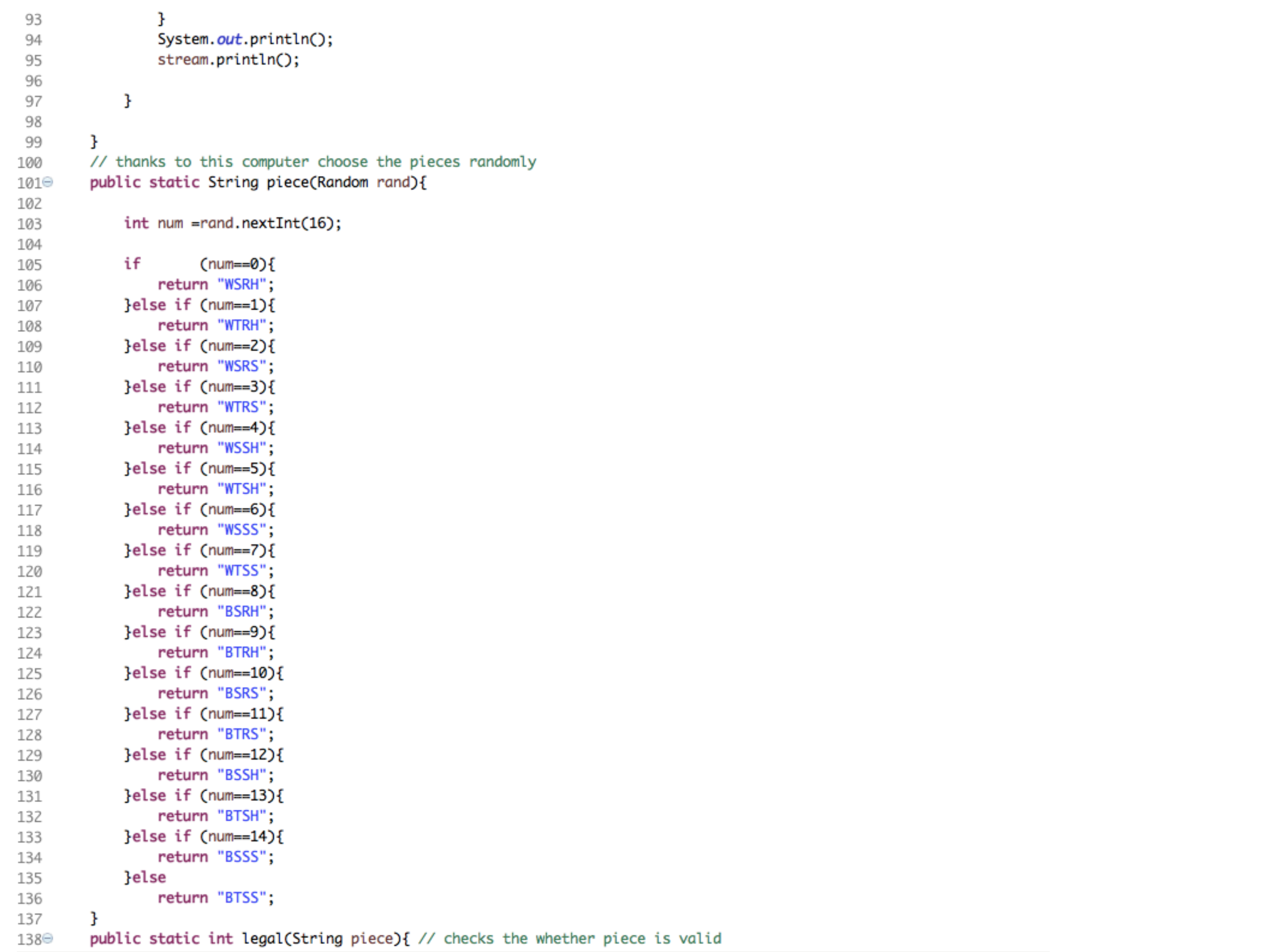
I handle the assignment by creating file, scanner, while loops, if/else statements, arrays, boolean methods and a method which returns an integer back to its caller method. Firstly, I define a scanner and random for user’ s and computer’ s move. I produced a 4x4 matrix for the base of the game board. I added while loops in the code for getting an acceptable answer. In that way, the question repeats until the program gets an appropriate answer. One of them is the first while loop which repeats until the answer is new game or ongoing game. If the answer is new game or new, the for loops fill the matrix with “E” which means empty place before the beginning of the game. I created a file called “input.txt” into the same if statement to load the game into the “input.txt”. Before the gamePlay method I generate 9 methods to facilitate my work. “printer” method updates the board after each move by adding pieces with for loops. At the same time this method wrote the updated board in to the file thanks to its stream parameter. I used “piece” method for computer’s piece choice. I created “legal” method for checking whether the selected piece from the user or computer is valid. It turns 1 for the correct piece. Then, I created 3 methods about winning configurations (vertical, horizontal, diagonal). If one of the characters of the piece is the same for each pieces in vertical, horizontal or diagonal the game finish. I created a if statement for preventing the misunderstanding of empty places (“E”). This if statement searched for a string whose length is not equal to 1. I created the “empty” method which checks the place is empty or full for taking the valid coordinates from the user and computer. I created the “sameAs” method which checks whether the piece is already taken for taking the valid piece from the user and computer. In gamePlay method, I write the game mechanics by above mentioned methods. I created an integer called “count”, which counts the number of the filled parts of the game board, when it reaches 16 , this means all the places on the board are full. I created a while loop which continues till any of the winning conditions occurred. User began the game in every new game option. Firstly, the user selected a valid piece. Then, the computer placed the chosen piece on the valid place of the game board and selected a valid piece. Then the user placed the piece ,which is chosen by computer, on the valid place of the game board and the will continue until either all the positions are filled or that there is winning configuration. Moreover, “gamePlay” method updates the current board and writes into the “input.txt”. When the user closed the game the updated board is already written in the input.txt. If the user types “ongoing game”, the program read the board configuration from “input.txt”. The input file includes each subsequent board configurations. Thus, I create a while loop which repeats till getting the last 4x4 matrix which is the board configuration just before closing the game. Then I call “save “

method which helps me to generating the last view of the game board. I created if/else statements to determine the who makes the next move by taking the mod(%2) of the full places. The “reverseGamePlay” method is basically the opposite of the gamePlay method. In “reverseGamePlay” the computer begin the ongoing game.

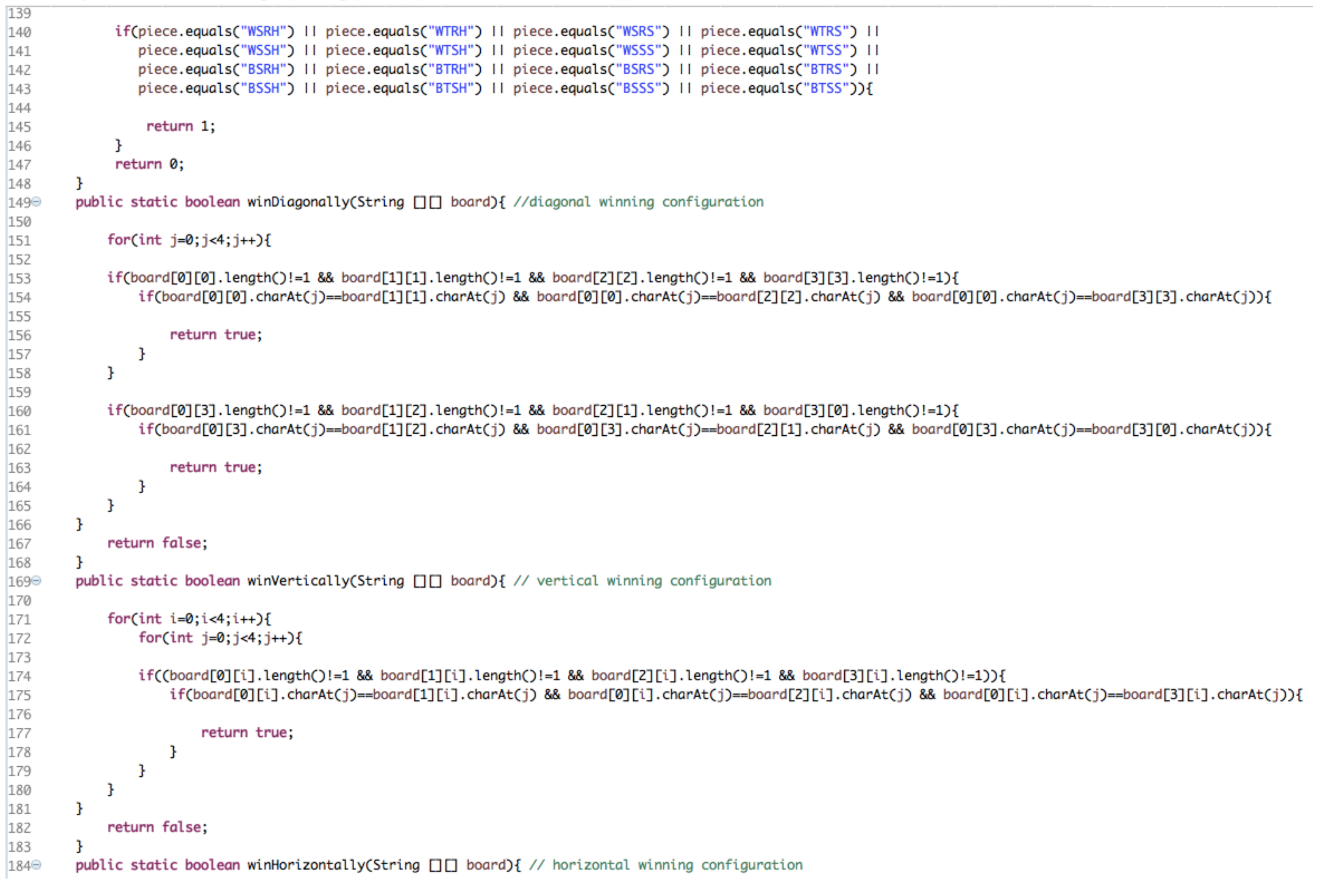
Implementation:



```java
import java.io.*;
import java.util.*;
public class SA2016400075 {

    public static void main(String []args) throws FileNotFoundException {

        Random rand=new Random(); // for computer's move
        Scanner console = new Scanner(System.in); // for user's move
        System.out.println("Do you want to play a new game or an ongoing game?");
        String answer=console.nextLine();

        // valid inputs below
        while(!(answer.equalsIgnoreCase("New game") || answer.equalsIgnoreCase("ongoing game") || answer.equalsIgnoreCase("new"))){
            System.out.println("Sorry, you have to choose a new game or ongoing game.");
            answer = console.nextLine();

        }

        String[][] board= new String[4][4]; // creates the matrix

        if(answer.equalsIgnoreCase("new game") || answer.equalsIgnoreCase("new")){ // begins the game
            PrintStream stream = new PrintStream("input.txt"); // this file saves the game

            for(int i=0;i<4;i++){

                for(int j=0;j<4;j++){
                //fills the board with E (which means empty piece for the game)
                board[i][j]="E";
                System.out.print(board[i][j] +" ");
                stream.print(board[i][j] +" ");
                }
                System.out.println();
                stream.println();
            }
            gamePlay(board,rand,console,stream);

        } else if(answer.equalsIgnoreCase("ongoing game")){
            Scanner input = new Scanner(new File("input.txt")); // reads the previously saved game)

            while(input.hasNext()){ // creates the last 4x4 matrix which composed before leaving the game

                for(int i=0;i<4;i++){
                    for(int j=0;j<4;j++){

                        board[i][j]=input.next();

                }
                input.nextLine();
            }

        }
        save(board,rand,console);
        }
    }
    // It generates the last view of the game board
    public static void save(String[][] board,Random rand,Scanner console) throws FileNotFoundException{

        PrintStream stream=new PrintStream(new File("input.txt"));
        int count = 0;
        for(int i=0;i<4;i++){
            for(int j=0;j<4;j++){

                if(board[i][j].equals("E")){
                    count++;
                }
            }
        }
        for(int i=0;i<4;i++){
            for(int j=0;j<4;j++){

                System.out.print(board[i][j] + " ");
            }
            System.out.println();

        }
        // Determines who makes the next move
            if(count%2==1){
                reverseGamePlay(board,rand,console,stream);

        } else{
                gamePlay(board,rand,console,stream);

        }
    }
        // Updates the board after each move by adding pieces
    public static void printer(String [][] board,PrintStream stream){

        for(int i=0;i<4;i++){
            for(int j=0;j<4;j++){

                System.out.print(board[i][j] + " ");
                stream.print(board[i][j] + " ");
```

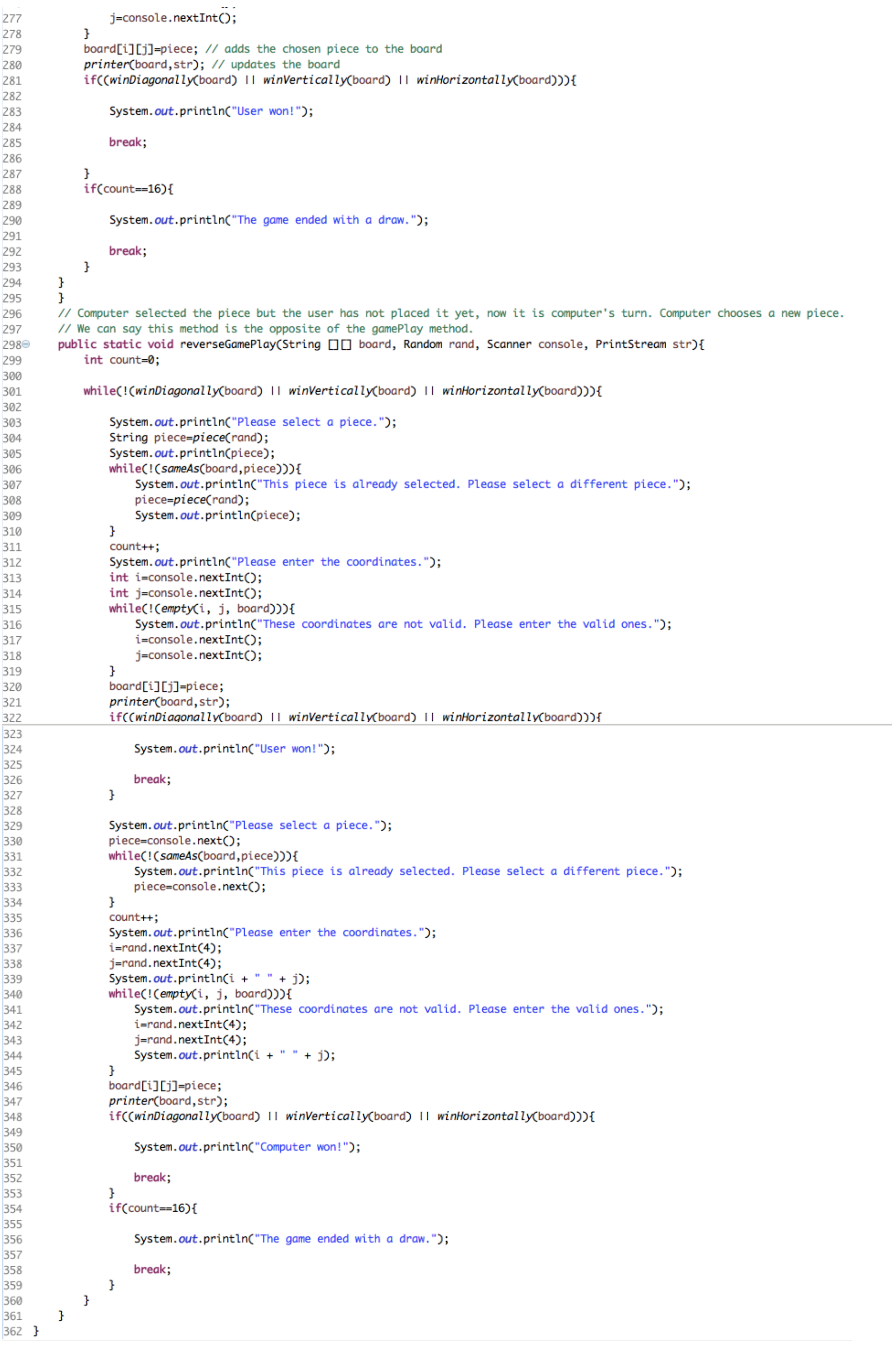


```java
            }
            System.out.println();
            stream.println();

        }

    }
    // thanks to this computer choose the pieces randomly
    public static String piece(Random rand){

        int num =rand.nextInt(16);

        if       (num==0){
            return "WSRH";
        }else if (num==1){
            return "WTRH";
        }else if (num==2){
            return "WSRS";
        }else if (num==3){
            return "WTRS";
        }else if (num==4){
            return "WSSH";
        }else if (num==5){
            return "WTSH";
        }else if (num==6){
            return "WSSS";
        }else if (num==7){
            return "WTSS";
        }else if (num==8){
            return "BSRH";
        }else if (num==9){
            return "BTRH";
        }else if (num==10){
            return "BSRS";
        }else if (num==11){
            return "BTRS";
        }else if (num==12){
            return "BSSH";
        }else if (num==13){
            return "BTSH";
        }else if (num==14){
            return "BSSS";
        }else
            return "BTSS";
    }
    public static int legal(String piece){ // checks the whether piece is valid
```



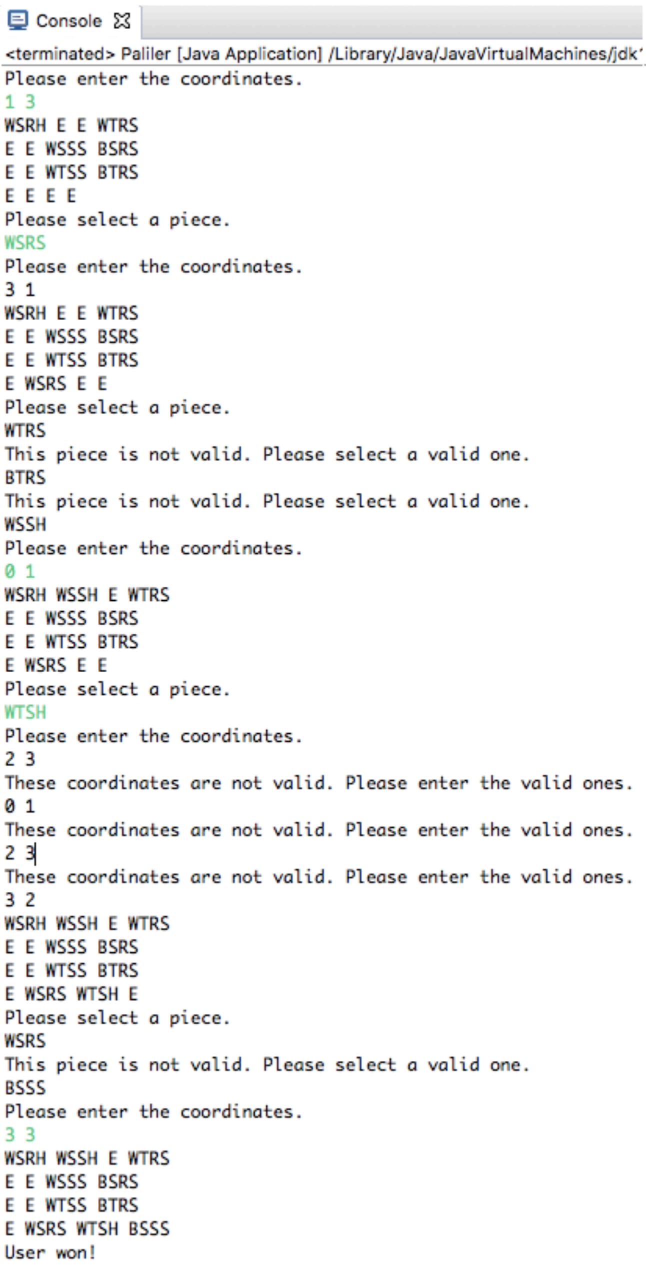
```java

         if(piece.equals("WSRH") || piece.equals("WTRH") || piece.equals("WSRS") || piece.equals("WTRS") ||
            piece.equals("WSSH") || piece.equals("WTSH") || piece.equals("WSSS") || piece.equals("WTSS") ||
            piece.equals("BSRH") || piece.equals("BTRH") || piece.equals("BSRS") || piece.equals("BTRS") ||
            piece.equals("BSSH") || piece.equals("BTSH") || piece.equals("BSSS") || piece.equals("BTSS")){

             return 1;
         }
         return 0;
    }
    public static boolean winDiagonally(String [][] board){ //diagonal winning configuration

        for(int j=0;j<4;j++){

        if(board[0][0].length()!=1 && board[1][1].length()!=1 && board[2][2].length()!=1 && board[3][3].length()!=1){
            if(board[0][0].charAt(j)==board[1][1].charAt(j) && board[0][0].charAt(j)==board[2][2].charAt(j) && board[0][0].charAt(j)==board[3][3].charAt(j)){

                return true;
            }
        }

        if(board[0][3].length()!=1 && board[1][2].length()!=1 && board[2][1].length()!=1 && board[3][0].length()!=1){
            if(board[0][3].charAt(j)==board[1][2].charAt(j) && board[0][3].charAt(j)==board[2][1].charAt(j) && board[0][3].charAt(j)==board[3][0].charAt(j)){

                return true;
            }
        }
    }
        return false;
    }
    public static boolean winVertically(String [][] board){ // vertical winning configuration

        for(int i=0;i<4;i++){
            for(int j=0;j<4;j++){

            if((board[0][i].length()!=1 && board[1][i].length()!=1 && board[2][i].length()!=1 && board[3][i].length()!=1)){
                if(board[0][i].charAt(j)==board[1][i].charAt(j) && board[0][i].charAt(j)==board[2][i].charAt(j) && board[0][i].charAt(j)==board[3][i].charAt(j)){

                    return true;
                }
            }
        }
    }
        return false;
    }
    public static boolean winHorizontally(String [][] board){ // horizontal winning configuration
```

```java

        for(int i=0;i<4;i++){
            for(int j=0;j<4;j++){

            if(board[i][0].length()!=1 && board[i][1].length()!=1 && board[i][2].length()!=1 && board[i][3].length()!=1)    {
                if(board[i][0].charAt(j)==board[i][1].charAt(j) && board[i][0].charAt(j)==board[i][2].charAt(j) && board[i][0].charAt(j)==board[i][3].charAt(j)){

                    return true;
                }
            }
        }
    }
        return false;
    }

    // checks the place is empty or full
    public static boolean empty(int i, int j, String □□ board){
        if(i<4 && j<4){
        if(board[i][j].equalsIgnoreCase("E")){

            return true;
            }
        }
        return false;

    }
    // checks whether the piece is already taken
    public static boolean sameAs(String □□ board,String piece){

        for(int i=0;i<4;i++){
            for(int j=0;j<4;j++){

                if(board[i][j].equals(piece)){
                    return false;
                }
            }
        }
        return true;
    }

    // the process of the gamePlay
    public static void gamePlay(String □□ board,Random rand,Scanner console,PrintStream str){

        int count=0; // increases with the filled parts of the game board, if it reaches 16 there is no empty places on the board
        while(!(winDiagonally(board) || winVertically(board) || winHorizontally(board))){ // continues till the any of the winning configurations is consisted

        System.out.println("Please select a piece."); // user begins the game
        String piece=console.next();
        while(legal(piece)==0){
        System.out.println("This piece is not valid. Please select a valid one.");
            piece=console.next();
        }
        while(!(sameAs(board,piece))){
            System.out.println("This piece is already selected. Please select a different piece.");
            piece=console.next();
        }
        count++; // increase with filled parts of the game board
        System.out.println("Please enter the coordinates."); // computer places the piece which is chosen by user by entering the coordinates
        int i=rand.nextInt(4);
        int j=rand.nextInt(4);
        System.out.println(i + " " + j);
        while(!(empty(i,j, board))){
            System.out.println("These coordinates are not valid. Please enter the valid ones.");
            i=rand.nextInt(4);
            j=rand.nextInt(4);
            System.out.println(i + " " + j);
        }
        board[i][j]=piece;  // adds the chosen piece to the board
        printer(board,str); // updates the board
        if((winDiagonally(board) || winVertically(board) || winHorizontally(board))){

            System.out.println("Computer won!");

            break; // terminates the loop if a winning condition has occured

        }

        System.out.println("Please select a piece.");
        piece=piece(rand);
        System.out.println(piece);
        while(!(sameAs(board,piece))){
            System.out.println("This piece is not valid. Please select a valid one.");
            piece=piece(rand);
            System.out.println(piece);
        }
        count++;
        System.out.println("Please enter the coordinates."); // user places the piece which is chosen by computer by entering the coordinates
        i=console.nextInt();
        j=console.nextInt();
        while(!(empty(i, j, board))){
            System.out.println("These coordinates are not valid. Please enter the valid ones.");
            i=console.nextInt();
```

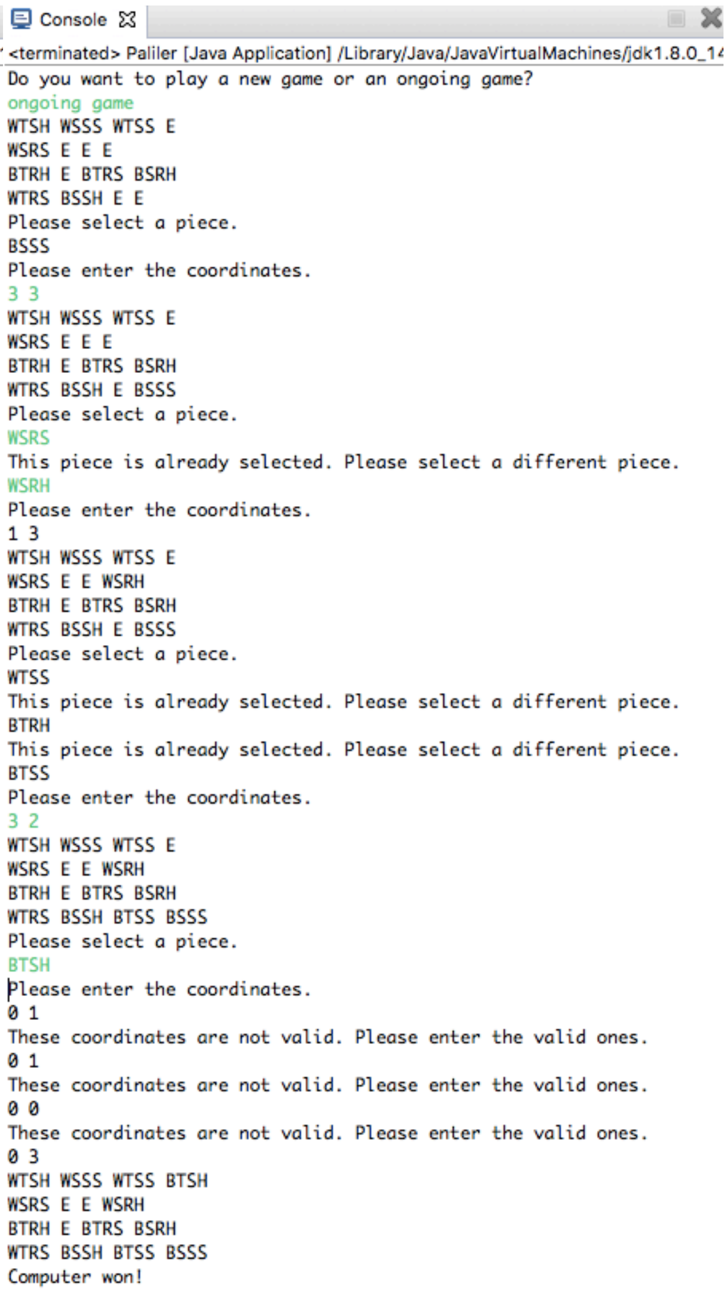


```
            j=console.nextInt();
        }
        board[i][j]=piece; // adds the chosen piece to the board
        printer(board,str); // updates the board
        if((winDiagonally(board) || winVertically(board) || winHorizontally(board))){

            System.out.println("User won!");

            break;

        }
        if(count==16){

            System.out.println("The game ended with a draw.");

            break;
        }
    }
    }
    // Computer selected the piece but the user has not placed it yet, now it is computer's turn. Computer chooses a new piece.
    // We can say this method is the opposite of the gamePlay method.
    public static void reverseGamePlay(String [][] board, Random rand, Scanner console, PrintStream str){
        int count=0;

        while(!(winDiagonally(board) || winVertically(board) || winHorizontally(board))){

            System.out.println("Please select a piece.");
            String piece=piece(rand);
            System.out.println(piece);
            while(!(sameAs(board,piece))){
                System.out.println("This piece is already selected. Please select a different piece.");
                piece=piece(rand);
                System.out.println(piece);
            }
            count++;
            System.out.println("Please enter the coordinates.");
            int i=console.nextInt();
            int j=console.nextInt();
            while(!(empty(i, j, board))){
                System.out.println("These coordinates are not valid. Please enter the valid ones.");
                i=console.nextInt();
                j=console.nextInt();
            }
            board[i][j]=piece;
            printer(board,str);
            if((winDiagonally(board) || winVertically(board) || winHorizontally(board))){

                System.out.println("User won!");

                break;
            }

            System.out.println("Please select a piece.");
            piece=console.next();
            while(!(sameAs(board,piece))){
                System.out.println("This piece is already selected. Please select a different piece.");
                piece=console.next();
            }
            count++;
            System.out.println("Please enter the coordinates.");
            i=rand.nextInt(4);
            j=rand.nextInt(4);
            System.out.println(i + " " + j);
            while(!(empty(i, j, board))){
                System.out.println("These coordinates are not valid. Please enter the valid ones.");
                i=rand.nextInt(4);
                j=rand.nextInt(4);
                System.out.println(i + " " + j);
            }
            board[i][j]=piece;
            printer(board,str);
            if((winDiagonally(board) || winVertically(board) || winHorizontally(board))){

                System.out.println("Computer won!");

                break;
            }
            if(count==16){

                System.out.println("The game ended with a draw.");

                break;
            }
        }
    }
}
```

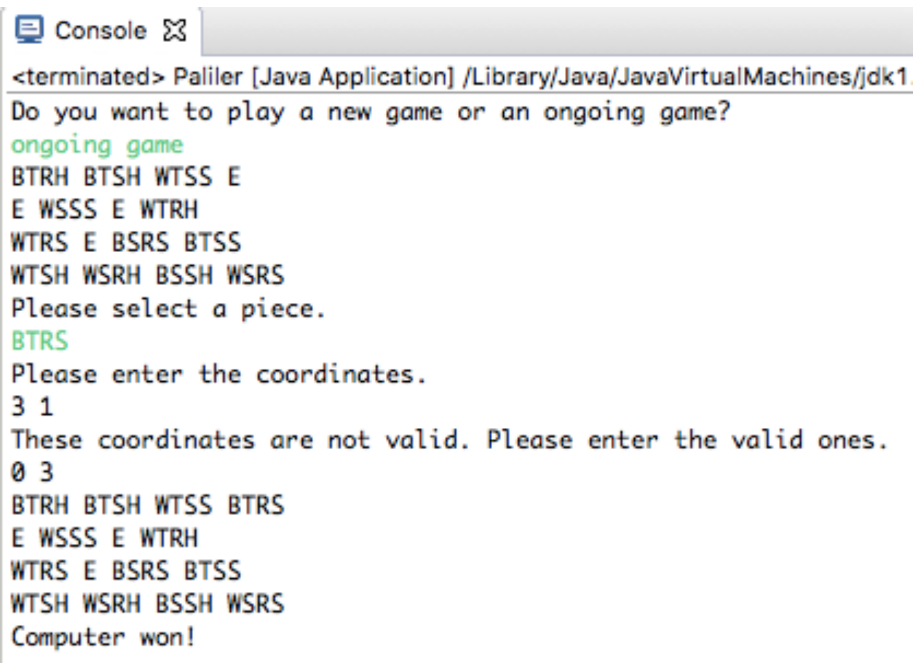
Outputs of the Program:



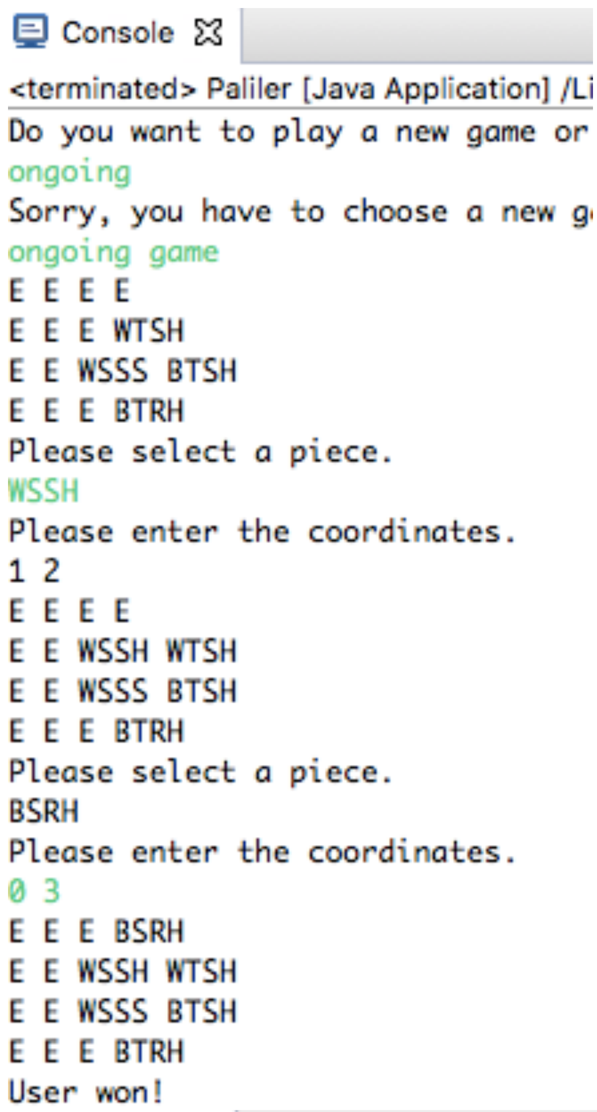
```
<terminated> Paliler [Java Application] /Library/Java/JavaVirtualMachines/jdk
Please enter the coordinates.
1 3
WSRH E E WTRS
E E WSSS BSRS
E E WTSS BTRS
E E E E
Please select a piece.
WSRS
Please enter the coordinates.
3 1
WSRH E E WTRS
E E WSSS BSRS
E E WTSS BTRS
E WSRS E E
Please select a piece.
WTRS
This piece is not valid. Please select a valid one.
BTRS
This piece is not valid. Please select a valid one.
WSSH
Please enter the coordinates.
0 1
WSRH WSSH E WTRS
E E WSSS BSRS
E E WTSS BTRS
E WSRS E E
Please select a piece.
WTSH
Please enter the coordinates.
2 3
These coordinates are not valid. Please enter the valid ones.
0 1
These coordinates are not valid. Please enter the valid ones.
2 3
These coordinates are not valid. Please enter the valid ones.
3 2
WSRH WSSH E WTRS
E E WSSS BSRS
E E WTSS BTRS
E WSRS WTSH E
Please select a piece.
WSRS
This piece is not valid. Please select a valid one.
BSSS
Please enter the coordinates.
3 3
WSRH WSSH E WTRS
E E WSSS BSRS
E E WTSS BTRS
E WSRS WTSH BSSS
User won!
```



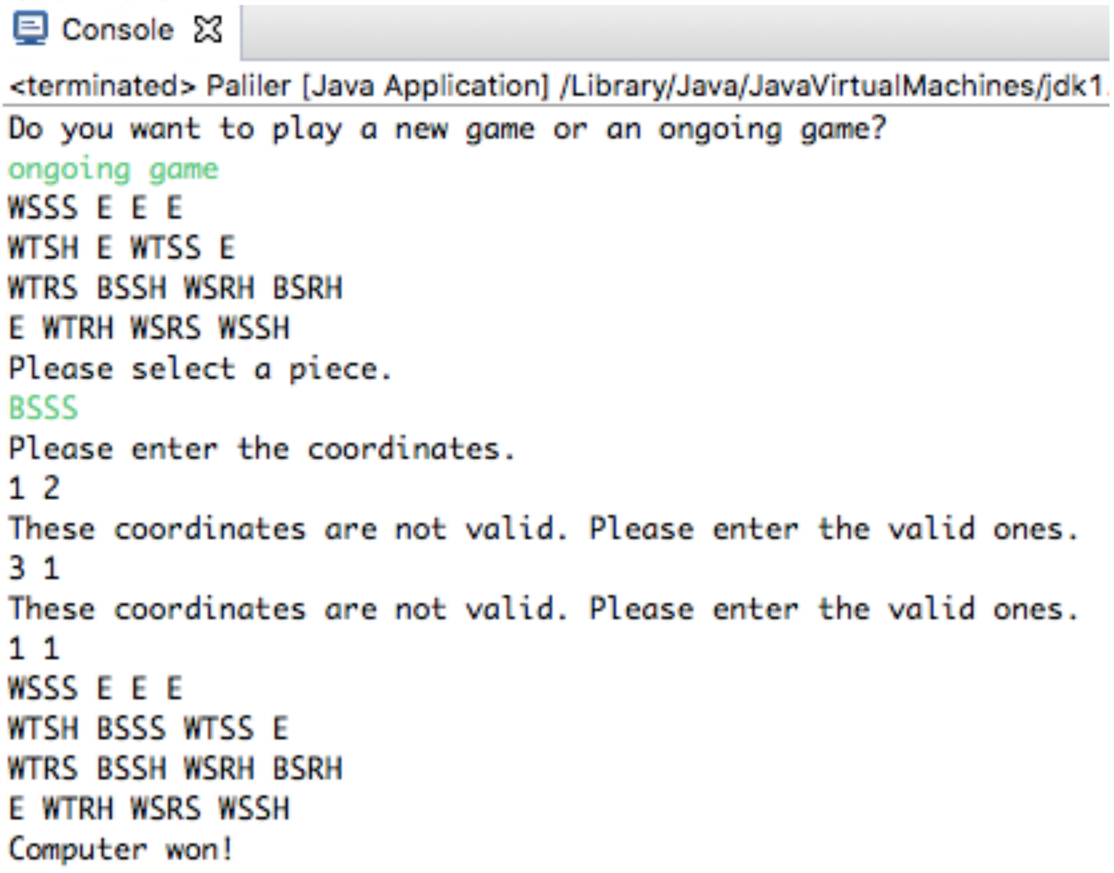
```
<terminated> Paliler [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_14
Do you want to play a new game or an ongoing game?
ongoing game
WTSH WSSS WTSS E
WSRS E E E
BTRH E BTRS BSRH
WTRS BSSH E E
Please select a piece.
BSSS
Please enter the coordinates.
3 3
WTSH WSSS WTSS E
WSRS E E E
BTRH E BTRS BSRH
WTRS BSSH E BSSS
Please select a piece.
WSRS
This piece is already selected. Please select a different piece.
WSRH
Please enter the coordinates.
1 3
WTSH WSSS WTSS E
WSRS E E WSRH
BTRH E BTRS BSRH
WTRS BSSH E BSSS
Please select a piece.
WTSS
This piece is already selected. Please select a different piece.
BTRH
This piece is already selected. Please select a different piece.
BTSS
Please enter the coordinates.
3 2
WTSH WSSS WTSS E
WSRS E E WSRH
BTRH E BTRS BSRH
WTRS BSSH BTSS BSSS
Please select a piece.
BTSH
Please enter the coordinates.
0 1
These coordinates are not valid. Please enter the valid ones.
0 1
These coordinates are not valid. Please enter the valid ones.
0 0
These coordinates are not valid. Please enter the valid ones.
0 3
WTSH WSSS WTSS BTSH
WSRS E E WSRH
BTRH E BTRS BSRH
WTRS BSSH BTSS BSSS
Computer won!
```



```
<terminated> Paliler [Java Application] /Library/Java/JavaVirtualMachines/jdk1
Do you want to play a new game or an ongoing game?
ongoing game
BTRH BTSH WTSS E
E WSSS E WTRH
WTRS E BSRS BTSS
WTSH WSRH BSSH WSRS
Please select a piece.
BTRS
Please enter the coordinates.
3 1
These coordinates are not valid. Please enter the valid ones.
0 3
BTRH BTSH WTSS BTRS
E WSSS E WTRH
WTRS E BSRS BTSS
WTSH WSRH BSSH WSRS
Computer won!
```



```
<terminated> Paliler [Java Application] /Li
Do you want to play a new game or
ongoing
Sorry, you have to choose a new g
ongoing game
E E E E
E E E WTSH
E E WSSS BTSH
E E E BTRH
Please select a piece.
WSSH
Please enter the coordinates.
1 2
E E E E
E E WSSH WTSH
E E WSSS BTSH
E E E BTRH
Please select a piece.
BSRH
Please enter the coordinates.
0 3
E E E BSRH
E E WSSH WTSH
E E WSSS BTSH
E E E BTRH
User won!
```



```
<terminated> Paliler [Java Application] /Library/Java/JavaVirtualMachines/jdk1
Do you want to play a new game or an ongoing game?
ongoing game
WSSS E E E
WTSH E WTSS E
WTRS BSSH WSRH BSRH
E WTRH WSRS WSSH
Please select a piece.
BSSS
Please enter the coordinates.
1 2
These coordinates are not valid. Please enter the valid ones.
3 1
These coordinates are not valid. Please enter the valid ones.
1 1
WSSS E E E
WTSH BSSS WTSS E
WTRS BSSH WSRH BSRH
E WTRH WSRS WSSH
Computer won!
```

Conclusion:

As seen from the outputs, the program is able to handle the requested task correctly. I don’t get any compiler errors and I get the requested outputs.