University of Dhaka

Department of Computer Science and Engineering

2nd Year, 1st Semester

Session – (2014 – 15)

Code: CSE – 2112

Subject: Object Oriented Programming

Project: Tic Tac Toe

Project type: An object oriented game

Game Partners:

Azwad Anjum Islam Dipto (Roll – 1)

Redwan Ahmed Rizvee (Roll – 9)

Submitted to:

Md. Mustafizur Rahman

Shiplu Hawlader

Code of Tic Tac Toe

**package tic\_tac\_toe;**

public class Tic\_Tac\_Toe {

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

Beginning begining = new Beginning();

begining.createThePicture();

}

}

**//Class Beginning**

package tic\_tac\_toe;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.logging.Level;

import java.util.logging.Logger;

public class Beginning {

protected String selection;

protected boolean show;

int flag = 0;

void createThePicture() throws Exception

{

show = false;

JFrame picFrame = new JFrame("Tic Tac Toe");

JPanel picPanel = new JPanel(new GridBagLayout());

GridBagConstraints c1 = new GridBagConstraints();

GridBagConstraints c2 = new GridBagConstraints();

ButtonGroup group = new ButtonGroup();

ImageIcon image = new ImageIcon(getClass().getResource("Tic\_tac\_toe.png"));

JLabel label = new JLabel(image);

JRadioButton rbuttonOne,rbuttonTwo;

JButton buttonOne,buttonTwo;

picFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

picFrame.setLayout(new FlowLayout());

picFrame.setSize(800,500);

buttonOne = new JButton("Play Game");

buttonTwo = new JButton("Quit Game");

rbuttonOne = new JRadioButton();

rbuttonTwo = new JRadioButton();

group.add(rbuttonOne);

group.add(rbuttonTwo);

picPanel.add(label);

c1.insets = new Insets(10,10,10,10);

c1.gridx = 0;

c1.gridy = 1;

picPanel.add(rbuttonOne,c1);

c1.gridx = 1;

c1.gridy = 1;

picPanel.add(buttonOne,c1);

c2.gridx = 0;

c2.gridy = 2;

picPanel.add(rbuttonTwo,c2);

c2.gridx = 1;

c2.gridy = 2;

picPanel.add(buttonTwo,c2);

picFrame.add(picPanel);

picFrame.setVisible(true);

selection = null;

buttonOne.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(rbuttonOne.isSelected() == true) {

selection = "new game";

try {

picFrame.setVisible(false);

new MenuClass();

} catch (Exception ex) {

Logger.getLogger(Beginning.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

buttonTwo.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(rbuttonTwo.isSelected() == true) {

selection = "Quit Game";

picFrame.setVisible(false);

show = true;

int value = 0;

System.exit(value);

}

}

}

);

}

**//StartGame**

package tic\_tac\_toe;

public class StartGame {

protected int counter;

protected FirstPlayerServer firstplayerserver;

protected SecondPlayerServer secondplayerserver;

protected PcVsHuman pc;

int choose;

StartGame(int choose) throws Exception

{

counter = 0;

if(choose == 1) serverOn();

if(choose == 2)secondServer();

if(choose == 3)eightbyeight();

}

void serverOn() throws Exception {

firstplayerserver = new FirstPlayerServer();

}

void secondServer() throws Exception

{

secondplayerserver = new SecondPlayerServer();

}

void eightbyeight() {

pc = new PcVsHuman();

}

}

**//ClassFirstPlayerServer**

package tic\_tac\_toe;

import java.awt.Color;

import java.awt.GridLayout;

import java.net.\*;

import java.io.\*;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.imageio.ImageIO;

public class FirstPlayerServer {

protected String bitString;

ServerSocket serversocket;

Socket socket;

DataInputStream in;

DataOutputStream out;

protected int counter;

JFrame serverFrame;

JButton grid[][];

JPanel panel;

protected int situationarr[][];

boolean playerOneWin;

boolean playerTwoWin;

public FirstPlayerServer() throws Exception

{

counter = 0;

playerOneWin = false;

playerTwoWin = false;

situationarr = new int [10][10];

char a = (char)(counter + '0');

bitString = "";

bitString = bitString + a;

bitString = bitString + "000000000";

ServerSocket serversocket = new ServerSocket(786);

Socket socket = serversocket.accept();

in = new DataInputStream(socket.getInputStream());

out = new DataOutputStream(socket.getOutputStream());

this.run();

}

public void run() throws Exception

{

if(counter == 0) {

counter++;

constructionOfServerWindow();

}

else {

counter = bitString.charAt(0) - '0';

counter++;

work();

}

}

public void situationCheck()

{

int row = 0,col = 0;

for(int i = 1; i <bitString.length(); i++) {

char a = bitString.charAt(i);

int b = a - '0';

if(i <= 3) row = 0;

else if(i <= 6) row = 1;

else if(i <= 9) row = 2;

situationarr[row][col] = b;

col++;

if(col > 2) col = 0;

}

}

public void riImageLoading()

{

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int b = situationarr[i][j];

int row = i,col = j;

switch (b) {

case 0:

{

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j].setIcon(image);

break;

}

case 1:

{

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[i][j].setIcon(image);

break;

}

case 2:

{

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[row][col].setIcon(image);

break;

}

default:

break;

}

}

}

}

public void constructionOfServerWindow()

{

situationCheck();

serverFrame = new JFrame("Server window");

serverFrame.setSize(700,700);

serverFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

serverFrame.setLayout(new GridLayout(3,3));

serverFrame.setBackground(Color.YELLOW);

grid = new JButton[10][10];

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(situationarr[i][j] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j] = new JButton(image);

}

if(situationarr[i][j] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[i][j] = new JButton(image);

}

if(situationarr[i][j] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[i][j] = new JButton(image);

}

serverFrame.add(grid[i][j]);

}

}

serverFrame.setVisible(true);

try {

work();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

public void work() throws Exception

{

situationCheck();

riImageLoading();

Intelligence m = new Intelligence(situationarr,serverFrame);

WinnerDeterminator p = new WinnerDeterminator(situationarr);

playerOneWin = p.playerOne;

playerTwoWin = p.playerTwo;

if(playerOneWin == true) {

showForWinnerOne();

}

if(playerTwoWin == true) {

showForWinnerTwo();

}

int flag = 0;

if(counter >= 9) {

if(playerOneWin == false && playerTwoWin == false)serverFrame.setTitle("Draw !!!");

flag = 1;

}

if(counter <= 9) {

serverFrame.setTitle("Server");

}

serverFrame.setVisible(true);

grid[0][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][0].isEnabled() && situationarr[0][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[0][0].setIcon(image);

situationarr[0][0] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][1].isEnabled() && situationarr[0][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[0][1].setIcon(image);

situationarr[0][1] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][2].isEnabled() && situationarr[0][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[0][2].setIcon(image);

situationarr[0][2] = 1;

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][0].isEnabled() && situationarr[1][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[1][0].setIcon(image);

situationarr[1][0] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][1].isEnabled() && situationarr[1][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[1][1].setIcon(image);

situationarr[1][1] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][2].isEnabled() && situationarr[1][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[1][2].setIcon(image);

situationarr[1][2] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][0].isEnabled() && situationarr[2][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[2][0].setIcon(image);

situationarr[2][0] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][1].isEnabled() && situationarr[2][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[2][1].setIcon(image);

situationarr[2][1] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][2].isEnabled() && situationarr[2][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[2][2].setIcon(image);

situationarr[2][2] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

}

public void bitStringFormatAndSend() throws Exception

{

bitString = "";

if(counter <= 9)

{

char a = (char) (counter + '0');

bitString = bitString + a;

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int v = situationarr[i][j];

a = (char) (situationarr[i][j] + '0');

bitString = bitString + a;

}

}

out.writeUTF(bitString);

if(playerOneWin == true) {

showForWinnerOne();

}

if(playerTwoWin == true) {

showForWinnerTwo();

}

bitString = in.readUTF();

run();

}

}

void showForWinnerOne()

{

JFrame finalFrame = new JFrame("Winner : Server");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = situationarr;

group = new JButton[10][10];

if(situationarr[0][0] == 1 && situationarr[1][1] == 1 && situationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][2] == 1 && situationarr[1][1] == 1 && situationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][0] == 1 && situationarr[1][0] == 1 && situationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][1] == 1 && situationarr[1][1] == 1 && situationarr[2][1] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(situationarr[0][2] == 1 && situationarr[1][2] == 1 && situationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][0] == 1 && situationarr[0][1] == 1 && situationarr[0][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(situationarr[1][0] == 1 && situationarr[1][1] == 1 && situationarr[1][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(situationarr[2][0] == 1 && situationarr[2][1] == 1 && situationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

finalFrame.add(group[i][j]);

continue;

}

else if(situationarr[i][j] == 0 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 1 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 2 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

void showForWinnerTwo()

{

JFrame finalFrame = new JFrame("Winner : Client");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = situationarr;

group = new JButton[10][10];

if(situationarr[0][0] == 2 && situationarr[1][1] == 2 && situationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][2] == 2 && situationarr[1][1] == 2 && situationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][0] == 2 && situationarr[1][0] == 2 && situationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][1] == 2 && situationarr[1][1] == 2 && situationarr[2][1] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(situationarr[0][2] == 2 && situationarr[1][2] == 2 && situationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][0] == 2 && situationarr[0][1] == 2 && situationarr[0][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(situationarr[1][0] == 2 && situationarr[1][1] == 2 && situationarr[1][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(situationarr[2][0] == 2 && situationarr[2][1] == 2 && situationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

finalFrame.add(group[i][j]);

continue;

}

else if(situationarr[i][j] == 0 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 1 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 2 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

}

**//Class FourByTwo**

package tic\_tac\_toe;

public class FourByTwo {

protected int simulationarr[][];

protected int row,col;

protected int maximum;

public FourByTwo(int arr[][],int i,int j)

{

simulationarr = arr;

row = i;

col = j;

maximum = 0;

downMove();

}

boolean validity(int r)

{

if(r >= 0 && r <= 7) return true;

else return false;

}

void downMove()

{

boolean go = validity(row+1);

if(go == true) {

downMoveOne();

downMoveTwo();

downMoveThree();

downMoveFour();

}

boolean run = validity(row-1);

if(run == true) {

upMoveOne();

upMoveTwo();

upMoveThree();

upMoveFour();

}

}

void downMoveOne()

{

int sum = 1;

if(simulationarr[row+1][col] == 1) {

sum++;

for(int i = col+1; i <= col+3; i++) {

boolean one = validity(i);

if(one == true) {

if(simulationarr[row][i] == 1 && simulationarr[row+1][i] == 1) {

sum = sum+2;

}

else break;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void downMoveTwo()

{

int sum = 1;

int left = 1;

int right = 2;

if(simulationarr[row+1][col] == 1) {

sum++;

for(int i = col-1,counter = 1; i >= 0 && counter <= left;i--,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row+1][i] == 1) {

sum = sum+2;

}

else break;

}

for(int i = col + 1,counter = 1; i <= 7 && counter <= right; i++,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row+1][i] == 1) {

sum = sum+2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void downMoveThree()

{

int sum = 1;

int left = 2;

int right = 1;

if(simulationarr[row+1][col] == 1) {

sum++;

for(int i = col-1,counter = 1; i >= 0 && counter <= left;i--,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row+1][i] == 1) {

sum = sum+2;

}

else break;

}

for(int i = col + 1,counter = 1; i <= 7 && counter <= right; i++,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row+1][i] == 1) {

sum = sum+2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void downMoveFour()

{

int sum = 1;

int left = 3;

int right = 0;

if(simulationarr[row+1][col] == 1) {

sum++;

for(int i = col-1,counter = 1; i >= 0 && counter <= left;i--,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row+1][i] == 1) {

sum = sum+2;

}

else break;

}

for(int i = col + 1,counter = 1; i <= 7 && counter <= right; i++,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row+1][i] == 1) {

sum = sum+2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void upMoveOne()

{

int sum = 1;

int left = 0;

int right = 3;

if(simulationarr[row-1][col] == 1) {

sum++;

for(int i = col-1,counter = 1; i >= 0 && counter <= left;i--,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

for(int i = col + 1,counter = 1; i <= 7 && counter <= right; i++,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void upMoveTwo()

{

int sum = 1;

int left = 1;

int right = 2;

if(simulationarr[row-1][col] == 1) {

sum++;

for(int i = col-1,counter = 1; i >= 0 && counter <= left;i--,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

for(int i = col + 1,counter = 1; i <= 7 && counter <= right; i++,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void upMoveThree()

{

int sum = 1;

int left = 2;

int right = 1;

if(simulationarr[row-1][col] == 1) {

sum++;

for(int i = col-1,counter = 1; i >= 0 && counter <= left;i--,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

for(int i = col + 1,counter = 1; i <= 7 && counter <= right; i++,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void upMoveFour()

{

int sum = 1;

int left = 3;

int right = 0;

if(simulationarr[row-1][col] == 1) {

sum++;

for(int i = col-1,counter = 1; i >= 0 && counter <= left;i--,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

for(int i = col + 1,counter = 1; i <= 7 && counter <= right; i++,counter++) {

if(simulationarr[row][i] == 1 && simulationarr[row-1][i] == 1) {

sum = sum+2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

}

**// Class Intelligence**

package tic\_tac\_toe;

import javax.swing.\*;

public class Intelligence implements Runnable{

int simulationarr[][];

Thread T;

JFrame nameChange;

public Intelligence(int arr[][],JFrame givenFrame) throws Exception

{

nameChange = givenFrame;

simulationarr = arr;

T = new Thread(this);

T.start();

}

public void run(){

simpleMoveChecking();

simpleMoveCheckingDistantVersion();

simpleMoveCheckingWithMultipleWaysOfWin();

adjecentVersionWithMultipleWaysOfWin();

adjacentAndDistant();

}

void simpleMoveChecking()

{

int posx[] = {-1,1,0,0,-1,1,-1,1};

int posy[] = {0,0, 1,-1,1,-1,-1,1};

int easyFrontx[] = {-2,2,0,0,-2,2,-2,2};

int easyFronty[] = {0,0,2,-2,2,-2,-2,2};

int easyBackx[] = {1,-1,0,0,1,-1,1,-1};

int easyBacky[] = {0,0,-1,1,-1,1,1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

int a = simulationarr[i][j];

for(int k = 0; k <= 7; k++) {

int closex = i + posx[k];

int closey = j + posy[k];

if(closex >= 0 && closex <= 2 && closey >= 0 && closey <= 2) {

int b = simulationarr[closex][closey];

if(a == b && a != 0) {

int nextx = i + easyFrontx[k];

int nexty = j + easyFronty[k];

if(nextx >= 0 && nextx <= 2 && nexty >= 0 && nexty <= 2) {

int c = simulationarr[nextx][nexty];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a good move");

}

}

}

int anothernextx = i + easyBackx[k];

int anothernexty = j + easyBacky[k];

if(anothernextx >= 0 && anothernextx <= 2 && anothernexty >= 0 && anothernexty <= 2) {

int c = simulationarr[anothernextx][anothernexty];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a good move");

}

}

}

}

}

}

}

}

}

void simpleMoveCheckingDistantVersion()

{

int newPosx[] = {0,0,-2,2,-2,2,-2,2};

int newPosy[] = {2,-2,0,0,2,2,-2,-2};

int midNewPosx[] = {0,0,-1,1,-1,1,-1,1};

int midNewPosy[] = {1,-1,0,0,1,1,-1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

int a = simulationarr[i][j];

for(int k = 0; k <= 7; k++) {

int durerx = i + newPosx[k];

int durery = j + newPosy[k];

if(durerx >= 0 && durerx <= 2 && durery >= 0 && durery <= 2) {

int b = simulationarr[durerx][durery];

if(a == b && a != 0) {

int midx = i + midNewPosx[k];

int midy = j + midNewPosy[k];

if(midx >= 0 && midx <= 2 && midy >= 0 && midy <= 2) {

int c = simulationarr[midx][midy];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a good move");

}

}

}

}

}

}

}

}

}

void simpleMoveCheckingWithMultipleWaysOfWin()

{

int dxforSingle[] = {0,0,-2,2,-2,2,-2,2};

int dyforSingle[] = {2,-2,0,0,2,2,-2,-2};

int dxForMid[] = {0,0,-1,1,-1,1,-1,1};

int dyForMid[] = {1,-1,0,0,1,1,-1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

for(int k = 0; k <= 7; k++) {

int firstPointx = i;

int firstPointy = j;

int lastPointx = i + dxforSingle[k];

int lastPointy = j + dyforSingle[k];

int midPointx = i + dxForMid[k];

int midPointy = j + dyForMid[k];

if(lastPointx >= 0 && lastPointy >= 0 && lastPointx <= 2 && lastPointy <= 2) {

//System.out.println("ase re pagla + " + i + " " + j);

if(midPointx >= 0 && midPointy >= 0 && midPointx <= 2 && midPointy <= 2) {

// System.out.println("ase re pagla 2+ " + i + " " + j);

for(int l = k + 1; l <= 7; l++) {

int templastPointx = i + dxforSingle[l];

int templastPointy = j + dyforSingle[l];

int tempmidPointx = i + dxForMid[l];

int tempmidPointy = j + dyForMid[l];

if(templastPointx >= 0 && templastPointy >= 0 && templastPointx <= 2 && templastPointy <= 2) {

if(tempmidPointx >= 0 && tempmidPointy >= 0 && tempmidPointx <= 2 && tempmidPointy <= 2) {

int a = simulationarr[firstPointx][firstPointy];

int b = simulationarr[lastPointx][lastPointy];

int c = simulationarr[templastPointx][templastPointy];

int midOne = simulationarr[midPointx][midPointy];

int midTwo = simulationarr[tempmidPointx][tempmidPointy];

//System.out.println("ase re pagla 4+ " + i + " " + j);

if(a == b && b == c && midOne == 0 && midTwo == 0) {

// System.out.println("hoi re ");

if(a == 1) {

nameChange.setTitle("Server is winning");

}

if(a == 2) {

nameChange.setTitle("Client is winning");

}

}

}

}

}

}

}

}

}

}

}

void adjecentVersionWithMultipleWaysOfWin()

{

int movex[] = {-1, 1, 0, 0, -1, 1, -1, 1};

int movey[] = {0, 0, 1, -1, 1, -1, -1, 1};

int frontx[]= {-2, 2, 0, 0, -2, 2, -2, 2};

int fronty[]= {0, 0, 2, -2, 2, -2, -2, 2};

int backx[] = {1, -1, 0, 0, 1, -1, 1, -1};

int backy[] = {0, 0, -1, 1, -1, 1, 1, -1};

int playerOneWin = 0,playerTwoWin = 0;

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

for(int k = 0; k <= 7; k++) {

int adjOnex = x + movex[k];

int adjOney = y + movey[k];

playerOneWin = 0;

playerTwoWin = 0;

for(int l = k + 1; l <= 7; l++) {

int adjTwox = x + movex[l];

int adjTwoy = y + movey[l];

int frontadjOnex = x + frontx[k];

int frontadjOney = y + fronty[k];

int backadjOnex = x + backx[k];

int backadjOney = y + backy[k];

int frontadjTwox = x + frontx[l];

int frontadjTwoy = y + fronty[l];

int backadjTwox = x + backx[l];

int backadjTwoy = y + backy[l];

int mainGhor = simulationarr[i][j];

int a,b;

playerOneWin = 0;

playerTwoWin = 0;

if(adjOnex >= 0 && adjOnex <= 2 && adjOney >= 0 && adjOney <= 2 ) {

a = simulationarr[adjOnex][adjOney];

if(adjTwox >= 0 && adjTwox <= 2 && adjTwoy >= 0 && adjTwoy <= 2) {

b = simulationarr[adjTwox][adjTwoy];

if(a == b && a == mainGhor && a != 0) {

if(frontadjOnex >= 0 && frontadjOnex <= 2 && frontadjOney >= 0 && frontadjOney <= 2) {

int c = simulationarr[frontadjOnex][frontadjOney];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(backadjOnex >= 0 && backadjOnex <= 2 && backadjOney >= 0 && backadjOney <= 2) {

int c = simulationarr[backadjOnex][backadjOney];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(frontadjTwox >= 0 && frontadjTwox <= 2 && frontadjTwoy >= 0 && frontadjTwoy <= 2) {

int c = simulationarr[frontadjTwox][frontadjTwoy];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(backadjTwox >= 0 && backadjTwox <= 2 && backadjTwoy >= 0 && backadjTwoy <= 2) {

int c = simulationarr[backadjTwox][backadjTwoy];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

}

}

}

if(playerOneWin > playerTwoWin && playerOneWin > 1) {

nameChange.setTitle("Server can win");

}

if(playerOneWin < playerTwoWin && playerTwoWin > 1) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

void adjacentAndDistant()

{

int mx[] = {-1, 1, 0, 0, -1, 1, -1, 1};

int my[] = {0, 0, 1, -1, 1, -1, -1, 1};

int fx[] = {-2, 2, 0, 0, -2, 2, -2, 2};

int fy[] = {0, 0, 2, -2, 2, -2, -2, 2};

int bx[] = {1, -1, 0, 0, 1, -1, 1, -1};

int by[] = {0, 0, -1, 1, -1, 1, 1, -1};

int durerx[] = {0,0,-2,2,-2,2,-2,2};

int durery[] = {2,-2,0,0,2,-2,-2,2};

int majkhanex[]= {0,0,-1,1,-1,1,-1,1};

int majkhaney[] = {1,-1,0,0,1,-1,-1,1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int mainx = i;

int mainy = j;

int a = simulationarr[mainx][mainy];

for(int k = 0; k <= 7; k++) {

int adjx = i + mx[k];

int adjy = j + my[k];

if(adjx >= 0 && adjx <= 2 && adjy >= 0 && adjy <= 2) {

int b = simulationarr[adjx][adjy];

if(a == b && a != 0) {

int samnerx = i + fx[k];

int samnery = j + fy[k];

int pichonerx = i + bx[k];

int pichonery = j + by[k];

if(samnerx >= 0 && samnerx <= 2 && samnery >= 0 && samnery <= 2) {

int c = simulationarr[samnerx][samnery];

if(c == 0) {

for(int l = 0; l <= 7; l++) {

int distx = i + durerx[l];

int disty = j + durery[l];

int moddhox = i + majkhanex[l];

int moddhoy = j + majkhaney[l];

if(distx >= 0 && distx <= 2 && disty >= 0 && disty <= 2 && moddhox >= 0 && moddhoy <= 2) {

int d = simulationarr[distx][disty];

int e = simulationarr[moddhox][moddhoy];

if(a == d && a != 0 && e == 0) {

if(a == 1) {

nameChange.setTitle("Server can win");

}

if(a == 2) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

if(pichonerx >= 0 && pichonerx <= 2 && pichonery >= 0 && pichonery <= 2) {

int c = simulationarr[pichonerx][pichonery];

if(c == 0) {

for(int l = 0; l <= 7; l++) {

int distx = i + durerx[l];

int disty = j + durery[l];

int moddhox = i + majkhanex[l];

int moddhoy = j + majkhaney[l];

if(distx >= 0 && distx <= 2 && disty >= 0 && disty <= 2 && moddhox >= 0 && moddhoy <= 2) {

int d = simulationarr[distx][disty];

int e = simulationarr[moddhox][moddhoy];

if(a == d && a != 0 && e == 0) {

if(a == 1) {

nameChange.setTitle("Server can win");

}

if(a == 2) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

}

}

}

}

}

}

}

package tic\_tac\_toe;

public class LongestRectangle {

protected int simulationarr[][];

protected int row,col,length,horizontalLength,verticalLength,diagonalLength;

boolean diagonalPointIndicator;

int diagonalGroup;

public LongestRectangle(int arr[][],int i, int j)

{

simulationarr = arr;

row = i;

col = j;

length = 0;

horizontalLength = 0;

verticalLength = 0;

diagonalLength = 0;

diagonalPointIndicator = false;

diagonalGroup = 0;

horizontalHighestLength();

verticalHighestLength();

diagonalHighestLength();

}

void horizontalHighestLength()

{

horizontalLength = 1;

for(int i = col-1; i >= 0; i--) {

if(simulationarr[row][i] == 1) {

horizontalLength++;

}

else break;

}

for(int i = col + 1; i <= 7; i++) {

if(simulationarr[row][i] == 1) {

horizontalLength++;

}

else break;

}

}

void verticalHighestLength()

{

verticalLength = 1;

for(int i = row - 1; i >= 0; i--) {

if(simulationarr[i][col] == 1) {

verticalLength++;

}

else break;

}

for(int i = row + 1; i <= 7; i++) {

if(simulationarr[i][col] == 1) {

verticalLength++;

}

else break;

}

}

void diagonalValidity()

{

if(row == 0) {

if(col == 0 || col == 7) {

diagonalPointIndicator = true;

if(col == 0) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

if(row == 1) {

if(col == 1 || col == 6) {

diagonalPointIndicator = true;

if(col == 1) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

if(row == 2) {

if(col == 2 || col == 5) {

diagonalPointIndicator = true;

if(col == 2) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

if(row == 3) {

if(col == 3 || col == 4) {

diagonalPointIndicator = true;

if(col == 3) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

if(row == 4) {

if(col == 4 || col == 3) {

diagonalPointIndicator = true;

if(col == 4) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

if(row == 5) {

if(col == 5 || col == 2) {

diagonalPointIndicator = true;

if(col == 5) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

if(row == 6) {

if(col == 6 || col == 1) {

diagonalPointIndicator = true;

if(col == 6) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

if(row == 7) {

if(col == 7 || col == 0) {

diagonalPointIndicator = true;

if(col == 7) diagonalGroup = 1;

else diagonalGroup = 2;

}

}

}

void diagonalHighestLength()

{

diagonalValidity();

if(diagonalPointIndicator == true) {

diagonalLength = 1;

if(diagonalGroup == 1) {

for(int i = row + 1; i <= 7; i++) {

if(simulationarr[i][i] == 1) {

diagonalLength++;

}

else break;

}

for(int i = row - 1; i >= 0; i--) {

if(simulationarr[i][i] == 1) {

diagonalLength++;

}

else break;

}

}

else if(diagonalGroup == 2) {

for(int i = row - 1, j = col + 1; i >= 0; i--,j++) {

if(simulationarr[i][j] == 1) {

diagonalLength++;

}

else break;

}

for(int i = row + 1, j = col - 1; i <= 7; i++,j--) {

if(simulationarr[i][j] == 1) {

diagonalLength++;

}

else break;

}

}

}

else {

diagonalLength = 0;

}

}

}

**//MenuClass**

package tic\_tac\_toe;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.\*;

public class MenuClass {

JFrame menuFrame;

JButton grid[][];

public MenuClass()

{

menuFrame = new JFrame("Select");

menuFrame.setSize(540,500);

menuFrame.setLayout(new GridLayout(6,1));

menuFrame.setVisible(true);

menuFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

grid = new JButton[8][3];

for(int i = 0; i < 6; i++) {

for(int j = 0; j < 1; j++) {

grid[i][j] = new JButton();

menuFrame.add(grid[i][j]);

}

}

ImageIcon image = new ImageIcon(getClass().getResource("menuimage1.png"));

grid[0][0].setIcon(image);

image = new ImageIcon(getClass().getResource("menuimage2.png"));

grid[1][0].setIcon(image);

image = new ImageIcon(getClass().getResource("menuimage4.png"));

grid[2][0].setIcon(image);

image = new ImageIcon(getClass().getResource("menuimage5.png"));

grid[3][0].setIcon(image);

image = new ImageIcon(getClass().getResource("menuimage6.png"));

grid[4][0].setIcon(image);

image = new ImageIcon(getClass().getResource("menuimage7.png"));

grid[5][0].setIcon(image);

selection();

}

void selection()

{

grid[0][0].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(grid[0][0].isEnabled()) {

try {

menuFrame.setVisible(false);

new StartGame(1);

} catch (Exception ex) {

Logger.getLogger(MenuClass.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][0].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(grid[1][0].isEnabled()) {

try {

menuFrame.setVisible(false);

new StartGame(2);

} catch (Exception ex) {

Logger.getLogger(MenuClass.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][0].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(grid[2][0].isEnabled()) {

try {

menuFrame.setVisible(false);

new StartGame(3);

} catch (Exception ex) {

Logger.getLogger(MenuClass.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[3][0].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(grid[3][0].isEnabled()) {

try {

menuFrame.setVisible(false);

Instruction();

} catch (Exception ex) {

Logger.getLogger(MenuClass.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[4][0].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(grid[4][0].isEnabled()) {

try {

menuFrame.setVisible(false);

instructionTwo();

} catch (Exception ex) {

Logger.getLogger(MenuClass.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[5][0].addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(grid[5][0].isEnabled()) {

try {

menuFrame.setVisible(false);

instructionTwo();

} catch (Exception ex) {

Logger.getLogger(MenuClass.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

}

void Instruction()

{

JFrame f = new JFrame("3 by 3 Rules");

f.setSize(450,450);

f.setVisible(true);

f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JPanel p = new JPanel();

JLabel lineOne = new JLabel("Any Horizontal or Vertical or diagonal matched will define the winner.");

JLabel lineTwo = new JLabel("To play , first open the main tic\_tac\_toe file");

JLabel lineThree = new JLabel("Then open the client");

JLabel lineFour = new JLabel("To play server as first player open ClientTicTacToe file");

JLabel lineFive = new JLabel("To play client as first player open ClientTicTacToeFirst");

p.add(lineOne);

p.add(lineTwo);

p.add(lineThree);

p.add(lineFour);

p.add(lineFive);

JButton b = new JButton("Back");

p.add(b);

f.add(p);

b.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(b.isEnabled()== true) {

// menuFrame.setVisible(false);

f.setVisible(false);

new MenuClass();

}

}

});

}

void instructionTwo()

{

JFrame f = new JFrame("8 X 8");

f.setVisible(true);

f.setSize(400,400);

f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JPanel p = new JPanel();

JLabel lineOne = new JLabel("Any Consecutive eight square will give you the winning");

JLabel lineOnePrime = new JLabel("Horizontal,vertical,diagonal,two row four col or four row two col");

JLabel lineTwo = new JLabel("Player will give the first move");

JLabel lineThree = new JLabel("Don't give more than one move in a turn.It is unethical");

JButton b = new JButton("Back");

p.add(lineOne);

p.add(lineOnePrime);

p.add(lineTwo);

p.add(lineThree);

p.add(b);

f.add(p);

b.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(b.isEnabled()== true) {

// menuFrame.setVisible(false);

f.setVisible(false);

new MenuClass();

}

}

});

}

void lastInstruction()

{

JFrame f = new JFrame("Socket connection");

f.setVisible(true);

f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JPanel p = new JPanel();

JLabel l = new JLabel("The 3 by 3 is a online game.And so it can take local hosts.To play first always run the tic\_tac\_toe file");

JLabel l2 = new JLabel("To play server as first player open ClientTicTacToe file");

JLabel l3 =new JLabel("To play client as the first player oen clientTicTacToe File");

JButton b = new JButton("Back");

b.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(b.isEnabled()== true) {

// menuFrame.setVisible(false);

f.setVisible(false);

new MenuClass();

}

}

});

p.add(l);

p.add(l2);

p.add(l3);

p.add(b);

f.add(p);

}

}

**// Class PcEightByEight**

package tic\_tac\_toe;

public class PcEightByEightAlgorithm {

protected int selectedRow;

protected int selectedCol;

protected boolean blockedPriority;

protected int simulationarr[][];

public PcEightByEightAlgorithm(int arr[][])

{

simulationarr = arr;

blockedPriority = false;

selectedRow = 6;

selectedCol = 0;

if(blockedPriority == false) {

horizontalEightPlayerCheck();

}

if(blockedPriority == false) {

verticalEightPlayerCheck();

}

if(blockedPriority == false) {

diagonalLeftEightPlayerCheck();

}

if(blockedPriority == false) {

diagonalRightEightPlayerCheck();

}

if(blockedPriority == false) {

fourByTwoCheck();

}

if(blockedPriority == false) {

twoByFourCheck();

}

if(blockedPriority == false) {

highestRectangle();

}

}

public void horizontalEightPlayerCheck()

{

// System.out.println("ase 1");

for(int i = 0; i < 8; i++) {

int playerMoveCounter = 0;

int pcMove = 0;

for(int j = 0; j < 8; j++) {

if(simulationarr[i][j] == 2) {

playerMoveCounter++;

}

if(simulationarr[i][j] == 1) {

pcMove++;

}

}

if(playerMoveCounter == 7 && pcMove == 0) {

blockedPriority = true;

selectionFromHorizontalRow(i);

}

}

}

public void selectionFromHorizontalRow(int row)

{

if(blockedPriority == true) {

for(int i = 0; i < 8; i++) {

if(simulationarr[row][i] == 0) {

simulationarr[row][i] = 1;

selectedRow = row;

selectedCol = i;

break;

}

}

}

}

public void verticalEightPlayerCheck()

{

if(blockedPriority == false) {

// System.out.println("ase 2");

for(int i = 0; i < 8; i++) {

int playerMoveCounter = 0;

int pcMove = 0;

for(int j = 0; j < 8; j++) {

if(simulationarr[j][i] == 2) {

playerMoveCounter++;

}

if(simulationarr[j][i] == 1) {

pcMove++;

}

}

if(playerMoveCounter == 7 && pcMove == 0) {

blockedPriority = true;

selectionFromVerticalCol(i);

}

}

}

}

public void selectionFromVerticalCol(int col)

{

if(blockedPriority == true) {

for(int i = 0; i < 8; i++) {

if(simulationarr[i][col] == 0) {

selectedRow = i;

selectedCol = col;

simulationarr[i][col] = 1;

break;

}

}

}

}

public void diagonalLeftEightPlayerCheck()

{

int playerMoveCounter = 0;

int pcMove = 0;

if(blockedPriority == false) {

// System.out.println("ase 3");

for(int i = 0; i < 8; i++) {

// System.out.println( i + " " + simulationarr[i][i]);

if(simulationarr[i][i] == 2) {

playerMoveCounter++;

}

else if(simulationarr[i][i] == 1) {

pcMove++;

}

}

// System.out.println(pcMove + " " + playerMoveCounter);

if(playerMoveCounter == 7 && pcMove == 0) {

blockedPriority = true;

selectionLeftFromDiagonal();

}

}

}

public void selectionLeftFromDiagonal()

{

for(int i = 0; i < 8; i++) {

if(simulationarr[i][i] == 0) {

simulationarr[i][i] = 1;

selectedRow= i;

selectedCol = i;

break;

}

}

}

public void diagonalRightEightPlayerCheck()

{

if(blockedPriority == false)

{

// System.out.println("ase 4");

int playerMoveCounter = 0;

int pcMove = 0;

for(int i = 0,j = 7; i <= 7; i++,j--) {

if(simulationarr[i][j] == 2) {

playerMoveCounter++;

}

if(simulationarr[i][j] == 1) {

pcMove++;

}

}

// System.out.println(playerMoveCounter + " " + pcMove + " iss");

if(playerMoveCounter == 7 && pcMove == 0) {

blockedPriority = true;

selectionRightFromDiagonal();

}

}

}

public void selectionRightFromDiagonal()

{

int j = 7;

for(int i = 0; i <= 7; i++,j--) {

if(simulationarr[i][j] == 0) {

simulationarr[i][j] = 1;

selectedRow= i;

selectedCol = j;

break;

}

}

}

public void fourByTwoCheck()

{

if(blockedPriority == false) {

//System.out.println("ase 5");

for(int i = 0; i <= 6;i++) {

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

int playerMoveCounter = 0;

int pcMove = 0;

int savex = 0,savey = 0;

for(int l = i; l <= i + 1; l++) {

for(int k = j; k <= j + 3; k++) {

if(simulationarr[l][k] == 2) {

playerMoveCounter++;

}

if(simulationarr[l][k] == 1) {

pcMove++;

}

if(simulationarr[l][k] == 0) {

savex = l;

savey = k;

}

}

}

if(playerMoveCounter == 7 && pcMove == 0) {

blockedPriority = true;

selectionForFourByTwo(savex,savey);

}

}

}

}

}

public void selectionForFourByTwo(int savex,int savey)

{

simulationarr[savex][savey] = 1;

selectedRow = savex;

selectedCol = savey;

}

public void twoByFourCheck()

{

if(blockedPriority == false) {

// System.out.println("ase 6");

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

for(int j = 0; j <= 6; j++) {

int playerMoveCounter = 0;

int pcMove = 0;

int savex = 0,savey = 0;

for(int l = i; l <= i + 3; l++) {

for(int k = j; k <= j+1;k++) {

if(simulationarr[l][k] == 2) {

playerMoveCounter++;

}

if(simulationarr[l][k] == 1) {

pcMove++;

}

if(simulationarr[l][k] == 0) {

savex = l;

savey = k;

}

}

}

if(playerMoveCounter == 7 && pcMove == 0) {

blockedPriority = true;

selectionForTwoByFour(savex,savey);

}

}

}

}

}

public void selectionForTwoByFour(int savex, int savey) {

// System.out.println("hoise = " + savex + " " + savey);

simulationarr[savex][savey] = 1;

selectedRow =savex;

selectedCol = savey;

}

public void highestRectangle()

{

if(blockedPriority == false) {

int maximum = 0,savex = 0,savey = 0,save;

for(int i = 0; i < 8; i++) {

for(int j = 0; j < 8; j++) {

save = 0;

if(simulationarr[i][j] == 0) {

LongestRectangle rectangle = new LongestRectangle(simulationarr,i,j);

//System.out.println("i = " + i + " j = " + j + " " + rectangle.horizontalLength + " " + rectangle.verticalLength + " " +rectangle.diagonalLength );

int horizontalLength = rectangle.horizontalLength;

int verticalLength = rectangle.verticalLength;

int diagonalLength = rectangle.diagonalLength;

FourByTwo object = new FourByTwo(simulationarr,i,j);

int highestboxlength = object.maximum;

TwoByFour problem = new TwoByFour(simulationarr,i,j);

int twoBoxProblem = problem.maximum;

//System.out.println(twoBoxProblem + " shesh");

if(save <= horizontalLength) {

save = horizontalLength;

}

if(save <= verticalLength) {

save = verticalLength;

}

if(save <= diagonalLength) {

save = diagonalLength;

}

if(save <= highestboxlength) {

save = highestboxlength;

}

if(save <= twoBoxProblem) {

save = twoBoxProblem;

}

if(maximum < save) {

maximum = save;

savex= i;

savey = j;

blockedPriority = true;

}

}

}

}

selectedRow = savex;

selectedCol = savey;

simulationarr[savex][savey] = 1;

}

}

}

// Class PCvsHuman

package tic\_tac\_toe;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.\*;

public class PcVsHuman {

JFrame frame;

JButton grid[][];

protected int simulationarr[][];

protected int moveCounter;

boolean playerHasGivenMove;

protected int guno;

public PcVsHuman()

{

moveCounter = 0;

guno = 0;

simulationarr = new int [15][15];

creation();

}

/\* public void gamePlay()

{

}\*/

public void creation()

{

frame = new JFrame("Tic Tac Toe - Special");

frame.setSize(1200,1200);

frame.setVisible(true);

frame.setLayout(new GridLayout(8,8));

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

buttonCreation();

}

public void buttonCreation()

{

grid = new JButton[15][15];

for(int i = 0; i < 8; i++) {

for(int j = 0; j < 8; j++) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

simulationarr[i][j] = 0;

grid[i][j] = new JButton();

grid[i][j].setIcon(image);

frame.add(grid[i][j]);

}

}

player();

}

public void changeImage(int row, int col,int choice)

{

if(choice == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("x for pc.png"));

grid[row][col].setIcon(image);

guno++;

WinnerComputerVsPC w = new WinnerComputerVsPC(simulationarr);

if(w.computer == true || w.player == true) {

WinnerFrame(w);

}

player();

}

if(choice == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage1.png"));

grid[row][col].setIcon(image);

WinnerComputerVsPC w = new WinnerComputerVsPC(simulationarr);

guno++;

if(w.computer == true || w.player == true) {

WinnerFrame(w);

}

pc();

}

}

public void player()

{

while(moveCounter <= 64) {

moveCounter++;

playerHasGivenMove = false;

System.out.println(moveCounter + " move");

playerMove();

/\*if(playerHasGivenMove == true) {

pc();

}\*/

playerHasGivenMove = false;

}

}

public void pc()

{

PcEightByEightAlgorithm computer = new PcEightByEightAlgorithm(simulationarr);

int row = computer.selectedRow;

int col = computer.selectedCol;

System.out.println(row + " " + col);

changeImage(row,col,1);

}

public void playerMove()

{

grid[0][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][0].isEnabled() && simulationarr[0][0] == 0) {

simulationarr[0][0] = 2;

playerHasGivenMove = true;

changeImage(0,0,2);

// pc();

}

}

});

grid[0][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][1].isEnabled() && simulationarr[0][1] == 0) {

simulationarr[0][1] = 2;

playerHasGivenMove = true;

changeImage(0,1,2);

// pc();

}

}

});

grid[0][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][2].isEnabled() && simulationarr[0][2] == 0) {

simulationarr[0][2] = 2;

playerHasGivenMove = true;

changeImage(0,2,2);

// pc();

}

}

});

grid[0][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][3].isEnabled() && simulationarr[0][3] == 0) {

simulationarr[0][3] = 2;

playerHasGivenMove = true;

changeImage(0,3,2);

// pc();

}

}

});

grid[0][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][4].isEnabled() && simulationarr[0][4] == 0) {

simulationarr[0][4] = 2;

playerHasGivenMove = true;

changeImage(0,4,2);

// pc();

}

}

});

grid[0][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][5].isEnabled() && simulationarr[0][5] == 0) {

simulationarr[0][5] = 2;

playerHasGivenMove = true;

changeImage(0,5,2);

// pc();

}

}

});

grid[0][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][6].isEnabled() && simulationarr[0][6] == 0) {

simulationarr[0][6] = 2;

playerHasGivenMove = true;

changeImage(0,6,2);

// pc();

}

}

});

grid[0][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][7].isEnabled() && simulationarr[0][7] == 0) {

simulationarr[0][7] = 2;

System.out.println("hoi" + simulationarr[0][7]);

playerHasGivenMove = true;

changeImage(0,7,2);

// pc();

}

}

});

grid[1][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][0].isEnabled() && simulationarr[1][0] == 0) {

simulationarr[1][0] = 2;

playerHasGivenMove = true;

changeImage(1,0,2);

// pc();

}

}

});

grid[1][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][1].isEnabled() && simulationarr[1][1] == 0) {

simulationarr[1][1] = 2;

playerHasGivenMove = true;

changeImage(1,1,2);

// pc();

}

}

});

grid[1][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][2].isEnabled() && simulationarr[1][2] == 0) {

simulationarr[1][2] = 2;

playerHasGivenMove = true;

changeImage(1,2,2);

// pc();

}

}

});

grid[1][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][3].isEnabled() && simulationarr[1][3] == 0) {

simulationarr[1][3] = 2;

playerHasGivenMove = true;

changeImage(1,3,2);

// pc();

}

}

});

grid[1][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][4].isEnabled() && simulationarr[1][4] == 0) {

simulationarr[1][4] = 2;

playerHasGivenMove = true;

changeImage(1,4,2);

// pc();

}

}

});

grid[1][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][5].isEnabled() && simulationarr[1][5] == 0) {

simulationarr[1][5] = 2;

playerHasGivenMove = true;

changeImage(1,5,2);

// pc();

}

}

});

grid[1][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][6].isEnabled() && simulationarr[1][6] == 0) {

simulationarr[1][6] = 2;

playerHasGivenMove = true;

changeImage(1,6,2);

// pc();

}

}

});

grid[1][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][7].isEnabled() && simulationarr[1][7] == 0) {

simulationarr[1][7] = 2;

playerHasGivenMove = true;

changeImage(1,7,2);

// pc();

}

}

});

grid[2][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][0].isEnabled() && simulationarr[2][0] == 0) {

simulationarr[2][0] = 2;

playerHasGivenMove = true;

changeImage(2,0,2);

// pc();

}

}

});

grid[2][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][1].isEnabled() && simulationarr[2][1] == 0) {

simulationarr[2][1] = 2;

playerHasGivenMove = true;

changeImage(2,1,2);

// pc();

}

}

});

grid[2][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][2].isEnabled() && simulationarr[2][2] == 0) {

simulationarr[2][2] = 2;

playerHasGivenMove = true;

changeImage(2,2,2);

// pc();

}

}

});

grid[2][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][3].isEnabled() && simulationarr[2][3] == 0) {

simulationarr[2][3] = 2;

playerHasGivenMove = true;

changeImage(2,3,2);

// pc();

}

}

});

grid[2][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][4].isEnabled() && simulationarr[2][4] == 0) {

simulationarr[2][4] = 2;

playerHasGivenMove = true;

changeImage(2,4,2);

// pc();

}

}

});

grid[2][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][5].isEnabled() && simulationarr[2][5] == 0) {

simulationarr[2][5] = 2;

playerHasGivenMove = true;

changeImage(2,5,2);

// pc();

}

}

});

grid[2][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][6].isEnabled() && simulationarr[2][6] == 0) {

simulationarr[2][6] = 2;

playerHasGivenMove = true;

changeImage(2,6,2);

// pc();

}

}

});

grid[2][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][7].isEnabled() && simulationarr[2][7] == 0) {

simulationarr[2][7] = 2;

playerHasGivenMove = true;

changeImage(2,7,2);

// pc();

}

}

});

grid[3][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][0].isEnabled() && simulationarr[3][0] == 0) {

simulationarr[3][0] = 2;

playerHasGivenMove = true;

changeImage(3,0,2);

// pc();

}

}

});

grid[3][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][1].isEnabled() && simulationarr[3][1] == 0) {

simulationarr[3][1] = 2;

playerHasGivenMove = true;

changeImage(3,1,2);

// pc();

}

}

});

grid[3][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][2].isEnabled() && simulationarr[3][2] == 0) {

simulationarr[3][2] = 2;

playerHasGivenMove = true;

changeImage(3,2,2);

// pc();

}

}

});

grid[3][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][3].isEnabled() && simulationarr[3][3] == 0) {

simulationarr[3][3] = 2;

playerHasGivenMove = true;

changeImage(3,3,2);

// pc();

}

}

});

grid[3][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][4].isEnabled() && simulationarr[3][4] == 0) {

simulationarr[3][4] = 2;

playerHasGivenMove = true;

changeImage(3,4,2);

// pc();

}

}

});

grid[3][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][5].isEnabled() && simulationarr[3][5] == 0) {

simulationarr[3][5] = 2;

playerHasGivenMove = true;

changeImage(3,5,2);

// pc();

}

}

});

grid[3][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][6].isEnabled() && simulationarr[3][6] == 0) {

simulationarr[3][6] = 2;

playerHasGivenMove = true;

changeImage(3,6,2);

// pc();

}

}

});

grid[3][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[3][7].isEnabled() && simulationarr[3][7] == 0) {

simulationarr[3][7] = 2;

playerHasGivenMove = true;

changeImage(3,7,2);

// pc();

}

}

});

grid[4][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][0].isEnabled() && simulationarr[4][0] == 0) {

simulationarr[4][0] = 2;

playerHasGivenMove = true;

changeImage(4,0,2);

// pc();

}

}

});

grid[4][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][1].isEnabled() && simulationarr[4][1] == 0) {

simulationarr[4][1] = 2;

playerHasGivenMove = true;

changeImage(4,1,2);

// pc();

}

}

});

grid[4][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][2].isEnabled() && simulationarr[4][2] == 0) {

simulationarr[4][2] = 2;

playerHasGivenMove = true;

changeImage(4,2,2);

// pc();

}

}

});

grid[4][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][3].isEnabled() && simulationarr[4][3] == 0) {

simulationarr[4][3] = 2;

playerHasGivenMove = true;

changeImage(4,3,2);

// pc();

}

}

});

grid[4][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][4].isEnabled() && simulationarr[4][4] == 0) {

simulationarr[4][4] = 2;

playerHasGivenMove = true;

changeImage(4,4,2);

//pc();

}

}

});

grid[4][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][5].isEnabled() && simulationarr[4][5] == 0) {

simulationarr[4][5] = 2;

playerHasGivenMove = true;

changeImage(4,5,2);

// pc();

}

}

});

grid[4][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][6].isEnabled() && simulationarr[4][6] == 0) {

simulationarr[4][6] = 2;

playerHasGivenMove = true;

changeImage(4,6,2);

// pc();

}

}

});

grid[4][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[4][7].isEnabled() && simulationarr[4][7] == 0) {

simulationarr[4][7] = 2;

playerHasGivenMove = true;

changeImage(4,7,2);

//pc();

}

}

});

grid[5][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][0].isEnabled() && simulationarr[5][0] == 0) {

simulationarr[5][0] = 2;

playerHasGivenMove = true;

changeImage(5,0,2);

// pc();

}

}

});

grid[5][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][1].isEnabled() && simulationarr[5][1] == 0) {

simulationarr[5][1] = 2;

playerHasGivenMove = true;

changeImage(5,1,2);

// pc();

}

}

});

grid[5][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][2].isEnabled() && simulationarr[5][2] == 0) {

simulationarr[5][2] = 2;

playerHasGivenMove = true;

changeImage(5,2,2);

// pc();

}

}

});

grid[5][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][3].isEnabled() && simulationarr[5][3] == 0) {

simulationarr[5][3] = 2;

playerHasGivenMove = true;

changeImage(5,3,2);

//pc();

}

}

});

grid[5][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][4].isEnabled() && simulationarr[5][4] == 0) {

simulationarr[5][4] = 2;

playerHasGivenMove = true;

changeImage(5,4,2);

//pc();

}

}

});

grid[5][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][5].isEnabled() && simulationarr[5][5] == 0) {

simulationarr[5][5] = 2;

playerHasGivenMove = true;

changeImage(5,5,2);

// pc();

}

}

});

grid[5][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][6].isEnabled() && simulationarr[5][6] == 0) {

simulationarr[5][6] = 2;

playerHasGivenMove = true;

changeImage(5,6,2);

// pc();

}

}

});

grid[5][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[5][7].isEnabled() && simulationarr[5][7] == 0) {

simulationarr[5][7] = 2;

playerHasGivenMove = true;

changeImage(5,7,2);

//pc();

}

}

});

grid[6][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][0].isEnabled() && simulationarr[6][0] == 0) {

simulationarr[6][0] = 2;

playerHasGivenMove = true;

changeImage(6,0,2);

// pc();

}

}

});

grid[6][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][1].isEnabled() && simulationarr[6][1] == 0) {

simulationarr[6][1] = 2;

playerHasGivenMove = true;

changeImage(6,1,2);

// pc();

}

}

});

grid[6][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][2].isEnabled() && simulationarr[6][2] == 0) {

simulationarr[6][2] = 2;

playerHasGivenMove = true;

changeImage(6,2,2);

// pc();

}

}

});

grid[6][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][3].isEnabled() && simulationarr[6][3] == 0) {

simulationarr[6][3] = 2;

playerHasGivenMove = true;

changeImage(6,3,2);

// pc();

}

}

});

grid[6][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][4].isEnabled() && simulationarr[6][4] == 0) {

simulationarr[6][4] = 2;

playerHasGivenMove = true;

changeImage(6,4,2);

// pc();

}

}

});

grid[6][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][5].isEnabled() && simulationarr[6][5] == 0) {

simulationarr[6][5] = 2;

playerHasGivenMove = true;

changeImage(6,5,2);

// pc();

}

}

});

grid[6][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][6].isEnabled() && simulationarr[6][6] == 0) {

simulationarr[6][6] = 2;

playerHasGivenMove = true;

changeImage(6,6,2);

// pc();

}

}

});

grid[6][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[6][7].isEnabled() && simulationarr[6][7] == 0) {

simulationarr[6][7] = 2;

playerHasGivenMove = true;

changeImage(6,7,2);

// pc();

}

}

});

grid[7][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][0].isEnabled() && simulationarr[7][0] == 0) {

simulationarr[7][0] = 2;

playerHasGivenMove = true;

changeImage(7,0,2);

// pc();

}

}

});

grid[7][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][1].isEnabled() && simulationarr[7][1] == 0) {

simulationarr[7][1] = 2;

playerHasGivenMove = true;

changeImage(7,1,2);

//pc();

}

}

});

grid[7][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][2].isEnabled() && simulationarr[7][2] == 0) {

simulationarr[7][2] = 2;

playerHasGivenMove = true;

changeImage(7,2,2);

// pc();

}

}

});

grid[7][3].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][3].isEnabled() && simulationarr[7][3] == 0) {

simulationarr[7][3] = 2;

playerHasGivenMove = true;

changeImage(7,3,2);

//pc();

}

}

});

grid[7][4].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][4].isEnabled() && simulationarr[7][4] == 0) {

simulationarr[7][4] = 2;

playerHasGivenMove = true;

changeImage(7,4,2);

//pc();

}

}

});

grid[7][5].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][5].isEnabled() && simulationarr[7][5] == 0) {

simulationarr[7][5] = 2;

playerHasGivenMove = true;

changeImage(7,5,2);

//pc();

}

}

});

grid[7][6].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][6].isEnabled() && simulationarr[7][6] == 0) {

simulationarr[7][6] = 2;

playerHasGivenMove = true;

changeImage(7,6,2);

//pc();

}

}

});

grid[7][7].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[7][7].isEnabled() && simulationarr[7][7] == 0) {

simulationarr[7][7] = 2;

playerHasGivenMove = true;

changeImage(7,7,2);

//pc();

}

}

});

}

public void WinnerFrame(WinnerComputerVsPC w)

{

JFrame finalFrame = new JFrame("Match Result: Computer Vs Player");

JPanel finalPanel = new JPanel();

JLabel text = new JLabel();

finalFrame.setSize(300,200);

finalFrame.setVisible(true);

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

String winner = "",process = "";

if(w.computer == true) winner = "Computer";

if(w.player == true) winner = "Player";

if(w.computer == false && w.player == false && guno >= 64) {

text.setText("Match is Drawn");

JButton button = new JButton("Exit");

button.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(button.isEnabled()) {

int v = 0;

System.exit(v);

}

}

});

finalPanel.add(text);

finalFrame.add(finalPanel);

finalPanel.add(button);

}

else {

if(w.horizontal == true) text.setText(winner + " wins by horizontal move");

if(w.vertical == true) text.setText(winner + " wins by vertical move");

if(w.diagonalLeft == true) text.setText(winner + " wins by left Diagonal move");

if(w.diagonalRight == true) text.setText(winner + " wins by right Diagonal move");

if(w.fourByTwo == true) text.setText(winner + " wins by four by two move");

if(w.twoByFour == true) text.setText(winner + " wins by two by four move");

JButton button = new JButton("Exit");

button.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(button.isEnabled()) {

int v = 0;

System.exit(v);

}

}

});

finalPanel.add(text);

finalFrame.add(finalPanel);

finalPanel.add(button);

}

}

}

**//Class SecondPlayerServer**

package tic\_tac\_toe;

import java.awt.Color;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.\*;

import java.net.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.\*;

public class SecondPlayerServer {

protected String bitString;

ServerSocket serversocket;

Socket socket;

DataInputStream in;

DataOutputStream out;

protected int counter;

JFrame serverFrame;

JButton grid[][];

JPanel panel;

protected int situationarr[][];

boolean playerOneWin;

boolean playerTwoWin;

public SecondPlayerServer() throws Exception

{

counter = 0;

playerOneWin = false;

playerTwoWin = false;

situationarr = new int [10][10];

char a = (char)(counter + '0');

bitString = "";

bitString = bitString + a;

bitString = bitString + "000000000";

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

situationarr[i][j] = 0;

}

}

ServerSocket serversocket = new ServerSocket(786);

Socket socket = serversocket.accept();

in = new DataInputStream(socket.getInputStream());

out = new DataOutputStream(socket.getOutputStream());

this.run();

}

public void situationCheck()

{

int row = 0,col = 0;

for(int i = 1; i <bitString.length(); i++) {

char a = bitString.charAt(i);

//System.out.println("char = " + a);

int b = a - '0';

if(i <= 3) row = 0;

else if(i <= 6) row = 1;

else if(i <= 9) row = 2;

situationarr[row][col] = b;

col++;

if(col > 2) col = 0;

}

}

public void riImageLoading()

{

//System.out.println("check kori = " + bitString);

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int b = situationarr[i][j];

//System.out.println("b = " + b);

int row = i,col = j;

switch (b) {

case 0:

{

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j].setIcon(image);

break;

}

case 1:

{

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[i][j].setIcon(image);

break;

}

case 2:

{

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[row][col].setIcon(image);

break;

}

default:

break;

}

}

}

}

public void constructionOfServerWindow()

{

situationCheck();

serverFrame = new JFrame("Server window");

serverFrame.setSize(700,700);

serverFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

serverFrame.setLayout(new GridLayout(3,3));

serverFrame.setBackground(Color.YELLOW);

grid = new JButton[10][10];

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(situationarr[i][j] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j] = new JButton(image);

}

if(situationarr[i][j] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[i][j] = new JButton(image);

}

if(situationarr[i][j] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[i][j] = new JButton(image);

}

serverFrame.add(grid[i][j]);

}

}

// serverFrame.setVisible(true);

try {

// work();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

void run() throws Exception

{

if(counter == 0) {

constructionOfServerWindow();

out.writeUTF(bitString);

bitString = in.readUTF();

System.out.println("ase " + bitString);

work();

}

}

public void work() throws Exception

{

situationCheck();

riImageLoading();

serverFrame.setVisible(true);

Intelligence m = new Intelligence(situationarr,serverFrame);

WinnerDeterminator p = new WinnerDeterminator(situationarr);

playerOneWin = p.playerOne;

playerTwoWin = p.playerTwo;

if(playerOneWin == true) {

showForWinnerOne();

}

if(playerTwoWin == true) {

showForWinnerTwo();

}

int flag = 0;

if(counter > 9) {

if(playerOneWin == false && playerTwoWin == false)serverFrame.setTitle("Draw !!!");

flag = 1;

}

serverFrame.setTitle("Server");

serverFrame.setVisible(true);

grid[0][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][0].isEnabled() && situationarr[0][0] == 0) {

System.out.println("Workdone in 1");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[0][0].setIcon(image);

situationarr[0][0] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][1].isEnabled() && situationarr[0][1] == 0) {

System.out.println("Workdone");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[0][1].setIcon(image);

situationarr[0][1] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][2].isEnabled() && situationarr[0][2] == 0) {

System.out.println("Workdone 3");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[0][2].setIcon(image);

situationarr[0][2] = 1;

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][0].isEnabled() && situationarr[1][0] == 0) {

System.out.println("Workdone");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[1][0].setIcon(image);

situationarr[1][0] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][1].isEnabled() && situationarr[1][1] == 0) {

System.out.println("Workdone");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[1][1].setIcon(image);

situationarr[1][1] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][2].isEnabled() && situationarr[1][2] == 0) {

System.out.println("Workdone");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[1][2].setIcon(image);

situationarr[1][2] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][0].isEnabled() && situationarr[2][0] == 0) {

System.out.println("Workdone");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[2][0].setIcon(image);

situationarr[2][0] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][1].isEnabled() && situationarr[2][1] == 0) {

System.out.println("Workdone");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[2][1].setIcon(image);

situationarr[2][1] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][2].isEnabled() && situationarr[2][2] == 0) {

System.out.println("Workdone");

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[2][2].setIcon(image);

situationarr[2][2] = 1;

try {

Intelligence clever = new Intelligence(situationarr,serverFrame);

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringFormatAndSend();

} catch (Exception ex) {

Logger.getLogger(FirstPlayerServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

}

void showForWinnerOne()

{

JFrame finalFrame = new JFrame("Winner : Server");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = situationarr;

group = new JButton[10][10];

if(situationarr[0][0] == 1 && situationarr[1][1] == 1 && situationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][2] == 1 && situationarr[1][1] == 1 && situationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][0] == 1 && situationarr[1][0] == 1 && situationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][1] == 1 && situationarr[1][1] == 1 && situationarr[2][1] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(situationarr[0][2] == 1 && situationarr[1][2] == 1 && situationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][0] == 1 && situationarr[0][1] == 1 && situationarr[0][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(situationarr[1][0] == 1 && situationarr[1][1] == 1 && situationarr[1][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(situationarr[2][0] == 1 && situationarr[2][1] == 1 && situationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

System.out.println(" lol " + i + " " + j);

finalFrame.add(group[i][j]);

continue;

}

else if(situationarr[i][j] == 0 && changeDitector[i][j] != -1) {

System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 1 && changeDitector[i][j] != -1) {

System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 2 && changeDitector[i][j] != -1) {

System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

void showForWinnerTwo()

{

JFrame finalFrame = new JFrame("Winner : Client");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = situationarr;

group = new JButton[10][10];

if(situationarr[0][0] == 2 && situationarr[1][1] == 2 && situationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][2] == 2 && situationarr[1][1] == 2 && situationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][0] == 2 && situationarr[1][0] == 2 && situationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(situationarr[0][1] == 2 && situationarr[1][1] == 2 && situationarr[2][1] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(situationarr[0][2] == 2 && situationarr[1][2] == 2 && situationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(situationarr[0][0] == 2 && situationarr[0][1] == 2 && situationarr[0][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(situationarr[1][0] == 2 && situationarr[1][1] == 2 && situationarr[1][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(situationarr[2][0] == 2 && situationarr[2][1] == 2 && situationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

//System.out.println(" lol " + i + " " + j);

finalFrame.add(group[i][j]);

continue;

}

else if(situationarr[i][j] == 0 && changeDitector[i][j] != -1) {

//System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 1 && changeDitector[i][j] != -1) {

// System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(situationarr[i][j] == 2 && changeDitector[i][j] != -1) {

// System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

void bitStringFormatAndSend() throws Exception

{

System.out.println("chole = " + bitString);

char a = bitString.charAt(0);

bitString = "";

counter = a - '0';

counter = counter + 1;

a = (char) (counter + '0');

bitString = bitString + a;

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

a = (char) (situationarr[i][j] + '0');

bitString = bitString + a;

}

}

System.out.println(" pathabo = " + bitString);

out.writeUTF(bitString);

bitString = in.readUTF();

int v = bitString.charAt(0) - '0';

if(playerOneWin == true) {

showForWinnerOne();

}

if(playerTwoWin == true) {

showForWinnerTwo();

}

if(v <= 8) {

work();

}

if(v == 9) {

situationCheck();

riImageLoading();

WinnerDeterminator w = new WinnerDeterminator(situationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

if(playerOneWin == true) {

showForWinnerOne();

}

if(playerTwoWin == true) {

showForWinnerTwo();

}

if(playerOneWin == false && playerTwoWin == false) {

serverFrame.setTitle("Draw !!!");

out.writeUTF(bitString);

}

}

}

}

**//Class TwoByFour**

package tic\_tac\_toe;

public class TwoByFour {

protected int simulationarr[][];

protected int row,col,maximum;

public TwoByFour(int arr[][],int i, int j)

{

simulationarr = arr;

row = i;

col = j;

maximum = 0;

choice();

}

boolean validity(int r)

{

if(r >= 0 && r <= 7) return true;

else return false;

}

void choice()

{

boolean go = validity(col+1);

if(go == true) {

upZeroDownThree();

upOneDownTwo();

upTwoDownOne();

upThreeDownZero();

}

boolean run = validity(col-1);

if(run == true) {

leftUpThreeDownZero();

leftUpTwoDownOne();

leftUpOneDownTwo();

leftUpZeroDownThree();

}

}

void upZeroDownThree()

{

int sum = 1;

int up = 0;

int down = 3;

if(simulationarr[row][col+1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void upOneDownTwo()

{

int sum = 1;

int up = 1;

int down = 2;

if(simulationarr[row][col+1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void upTwoDownOne()

{

int sum = 1;

int up = 2;

int down = 1;

if(simulationarr[row][col+1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void upThreeDownZero()

{

int sum = 1;

int up = 3;

int down = 0;

if(simulationarr[row][col+1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col+1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void leftUpZeroDownThree()

{

int sum = 1;

int up = 0;

int down = 3;

if(simulationarr[row][col-1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void leftUpOneDownTwo()

{

int sum = 1;

int up = 1;

int down = 2;

if(simulationarr[row][col-1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void leftUpTwoDownOne()

{

int sum = 1;

int up = 2;

int down = 1;

if(simulationarr[row][col-1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

void leftUpThreeDownZero()

{

int sum = 1;

int up = 3;

int down = 0;

if(simulationarr[row][col-1] == 1) {

sum++;

for(int i = row - 1, counter = 1; i >= 0 && counter <= up; counter++,i--) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

for(int i = row+1,counter = 1; i <= 7 && counter <= down; counter++,i++) {

if(simulationarr[i][col] == 1 && simulationarr[i][col-1] == 1) {

sum = sum + 2;

}

else break;

}

}

if(sum > maximum) {

maximum = sum;

}

}

}

**//Class WinnerComputerVsPC**

package tic\_tac\_toe;

public class WinnerComputerVsPC {

boolean computer;

boolean player;

boolean horizontal,vertical,diagonalLeft,diagonalRight,fourByTwo,twoByFour;

int simulationarr[][];

public WinnerComputerVsPC(int arr[][])

{

computer = false;

player = false;

simulationarr = arr;

horizontal = false;

vertical = false;

diagonalLeft = false;

diagonalRight = false;

fourByTwo = false;

twoByFour = false;

if(computer == false && player == false) {

horizontalWin();

}

if(computer == false && player == false) {

verticalWin();

}

if(computer == false && player == false) {

diagonalLeftWin();

}

if(computer == false && player == false) {

diagonalRightWin();

}

if(computer == false && player == false) {

fourByTwoWin();

}

if(computer == false && player == false) {

twoByFourWin();

}

}

public void horizontalWin()

{

for(int i = 0; i < 8; i++) {

boolean going = false;

int a = 0;

for(int j = 0; j <= 6; j++) {

if(simulationarr[i][j] == simulationarr[i][j+1] && simulationarr[i][j] != 0) {

going = true;

a = simulationarr[i][j];

}

else {

going = false;

break;

}

}

if(going == true) {

horizontal = true;

if(a == 1) {

computer = true;

break;

}

else if(a == 2) {

player = true;

break;

}

}

}

}

public void verticalWin()

{

for(int i = 0; i <= 7; i++) {

boolean going = false;

int a = 0;

for(int j = 0; j <= 6; j++) {

if(simulationarr[j][i] == simulationarr[j+1][i] && simulationarr[j][i] != 0) {

going = true;

a = simulationarr[j][i];

}

else {

going = false;

break;

}

}

if(going == true) {

vertical = true;

if(a == 1) {

computer = true;

break;

}

else if(a == 2) {

player = true;

break;

}

}

}

}

public void diagonalLeftWin()

{

boolean going = false;

int a = 0;

for(int i = 0; i <= 6; i++) {

if(simulationarr[i][i] == simulationarr[i+1][i+1] && simulationarr[i][i] != 0) {

going = true;

a = simulationarr[i][i];

}

else {

going = false;

break;

}

}

if(going == true) {

diagonalLeft = true;

if(a == 1) {

computer = true;

}

else if(a == 2) {

player = true;

}

}

}

public void diagonalRightWin()

{

boolean going = false;

int a = 0;

for(int i = 0,j = 7; i <= 6; i++,j--) {

if(simulationarr[i][j] == simulationarr[i+1][j-1] && simulationarr[i][j] != 0) {

going = true;

a = simulationarr[i][j];

}

else {

going = false;

break;

}

}

if(going == true) {

diagonalRight = true;

if(a == 1) {

computer = true;

}

else if(a == 2) {

player = true;

}

}

}

public void fourByTwoWin()

{

for(int i = 0; i <= 6; i++) {

boolean going = false;

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

going = false;

int a = simulationarr[i][j];

for(int l = i; l <= i+1; l++) {

for(int k = j; k <= j + 3; k++) {

if(a == simulationarr[l][k] && a != 0) {

going = true;

}

else {

going = false;

break;

}

}

if(going== false) break;

}

if(going == true && a != 0) {

fourByTwo = true;

if(a == 1) {

computer = true;

break;

}

else if(a == 2) {

player = true;

break;

}

}

}

if(going == true) break;

}

}

public void twoByFourWin()

{

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

boolean going = false;

int a = 0;

for(int j = 0; j <= 6;j++) {

going = false;

a = simulationarr[i][j];

for(int l = i; l <= i+3; l++) {

for(int k = j; k <= j + 1;k++) {

if(a == simulationarr[l][k] && a != 0) {

going = true;

}

else {

going = false;

break;

}

}

if(going== false) break;

}

if(going == true && a != 0) {

twoByFour = true;

if(a == 1) {

computer = true;

break;

}

else if(a == 2) {

player = true;

break;

}

}

}

if(going == true) break;

}

}

}

**// Class WinnerDeterminator**

package tic\_tac\_toe;

public class WinnerDeterminator {

boolean playerOne;

boolean playerTwo;

int simulationarr[][];

public WinnerDeterminator(int arr[][])

{

playerOne = false;

playerTwo = false;

simulationarr = arr;

movesOfPlayerOne();

movesOfPlayerTwo();

}

public void movesOfPlayerOne()

{

if(simulationarr[0][0] == 1 && simulationarr[1][1] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][1] == 1 && simulationarr[2][0] == 1) {

playerOne = true;

}

else if(simulationarr[0][0] == 1 && simulationarr[1][0] == 1 && simulationarr[2][0] == 1) {

playerOne = true;

}

else if(simulationarr[0][1] == 1 && simulationarr[1][1] == 1 && simulationarr[2][1] == 1) {

playerOne = true;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][2] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else if(simulationarr[0][0] == 1 && simulationarr[0][1] == 1 && simulationarr[0][2] == 1) {

playerOne = true;

}

else if(simulationarr[1][0] == 1 && simulationarr[1][1] == 1 && simulationarr[1][2] == 1) {

playerOne = true;

}

else if(simulationarr[2][0] == 1 && simulationarr[2][1] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else {

playerOne = false;

}

}

public void movesOfPlayerTwo()

{

if(simulationarr[0][0] == 2 && simulationarr[1][1] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][1] == 2 && simulationarr[2][0] == 2) {

playerTwo = true;

}

else if(simulationarr[0][0] == 2 && simulationarr[1][0] == 2 && simulationarr[2][0] == 2) {

playerTwo = true;

}

else if(simulationarr[0][1] == 2 && simulationarr[1][1] == 2 && simulationarr[2][1] == 2) {

playerTwo = true;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][2] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else if(simulationarr[0][0] == 2 && simulationarr[0][1] == 2 && simulationarr[0][2] == 2) {

playerTwo = true;

}

else if(simulationarr[1][0] == 2 && simulationarr[1][1] == 2 && simulationarr[1][2] == 2) {

playerTwo = true;

}

else if(simulationarr[2][0] == 2 && simulationarr[2][1] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else {

playerTwo = false;

}

System.out.println("class e one = " + playerOne + " two = " + playerTwo);

}

}

**Package ClientTicTacToe**

**//ClientTicTacToe**

package clienttictactoe;

import java.awt.Color;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.net.\*;

import java.io.\*;

import java.util.Scanner;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.\*;

public class ClientTicTacToe {

JFrame clientFrame;

JButton grid[][];

String bitString;

int counter;

int simulationarr[][];

boolean playerOneWin = false;

boolean playerTwoWin = false;

protected String Sender = "localhost";

protected int portNumber = 786;

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

ClientTicTacToe clienttictactoe = new ClientTicTacToe();

clienttictactoe.takeInput(clienttictactoe);

clienttictactoe.creation(clienttictactoe);

}

public void creation(ClientTicTacToe clienttictactoe) throws Exception

{

simulationarr = new int[10][10];

clientFrame = new JFrame("Client Window");

clientFrame.setSize(700,700);

clientFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

clientFrame.setLayout(new GridLayout(3,3));

clientFrame.setBackground(Color.yellow);

grid = new JButton[10][10];

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j] = new JButton(image);

clientFrame.add(grid[i][j]);

}

}

clienttictactoe.run();

}

void perfectingTheScenario()

{

int row = 0,col = 0;

for(int i = 1; i < bitString.length(); i++) {

if(i <= 3) {

row = 0;

}

else if(i <= 6) {

row = 1;

}

else if(i <= 9) {

row = 2;

}

int d = i - 1;

simulationarr[row][d%3] = bitString.charAt(i) - '0';

//System.out.println(simulationarr[row][d%3] + " " + row + " " + d%3);

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(simulationarr[i][j] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j].setIcon(image);

}

if(simulationarr[i][j] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[i][j].setIcon(image);

}

if(simulationarr[i][j] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[i][j] = new JButton(image);

}

}

}

// System.out.println("bitString = " + bitString);

}

void bitStringChange()

{

counter = counter + 1;

char a = (char) (counter + '0');

bitString = "";

bitString = bitString + a;

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

a = (char) (simulationarr[i][j] + '0');

bitString = bitString + a;

}

}

}

public void run() throws Exception {

Socket socket;

socket = new Socket(Sender,portNumber);

boolean program = true;

counter = 0;

while(program) {

DataInputStream in = new DataInputStream(socket.getInputStream());

DataOutputStream out = new DataOutputStream(socket.getOutputStream());

bitString = "";

bitString = in.readUTF();

counter = bitString.charAt(0) - '0';

perfectingTheScenario();

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

Intelligence intelligent = new Intelligence(simulationarr,clientFrame);

if(playerOneWin == true) {

out.writeUTF(bitString);

showForWinnerOne();

}

else if(playerTwoWin == true) {

out.writeUTF(bitString);

showForWinnerTwo();

}

if(counter == 9) {

// System.out.println("dh");

out.writeUTF(bitString);

WinnerDeterminator m = new WinnerDeterminator(simulationarr);

playerOneWin = m.playerOne;

playerTwoWin = m.playerTwo;

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

if(playerOneWin == true) {

showForWinnerOne();

}

if(playerTwoWin == true) {

showForWinnerTwo();

}

else {

if(playerOneWin == false && playerTwoWin == false) {

clientFrame.setTitle("Draw !!!");

}

}

}

clientFrame.setVisible(true);

if(counter < 9) clientFrame.setTitle("Client");

grid[0][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][0].isEnabled() && simulationarr[0][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[0][0].setIcon(image);

simulationarr[0][0] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringChange();

// System.out.println(bitString + " 1");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][1].isEnabled() && simulationarr[0][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[0][1].setIcon(image);

simulationarr[0][1] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

//System.out.println(bitString + " 2");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][2].isEnabled() && simulationarr[0][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[0][2].setIcon(image);

simulationarr[0][2] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

System.out.println(bitString + " 2");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][0].isEnabled() && simulationarr[1][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[1][0].setIcon(image);

simulationarr[1][0] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringChange();

// System.out.println(bitString + " 4");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][1].isEnabled() && simulationarr[1][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[1][1].setIcon(image);

simulationarr[1][1] = 2;

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 5");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][2].isEnabled() && simulationarr[1][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[1][2].setIcon(image);

simulationarr[1][2] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 6");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][0].isEnabled() && simulationarr[2][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[2][0].setIcon(image);

simulationarr[2][0] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 7");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][1].isEnabled() && simulationarr[2][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[2][1].setIcon(image);

simulationarr[2][1] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

//System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 8");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][2].isEnabled() && simulationarr[2][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[2][2].setIcon(image);

simulationarr[2][2] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

//System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString + " 9");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToe.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

}

}

void showForWinnerOne()

{

JFrame finalFrame = new JFrame("Winner : Server");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = simulationarr;

group = new JButton[10][10];

if(simulationarr[0][0] == 1 && simulationarr[1][1] == 1 && simulationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][1] == 1 && simulationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][0] == 1 && simulationarr[1][0] == 1 && simulationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][1] == 1 && simulationarr[1][1] == 1 && simulationarr[2][1] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][2] == 1 && simulationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][0] == 1 && simulationarr[0][1] == 1 && simulationarr[0][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(simulationarr[1][0] == 1 && simulationarr[1][1] == 1 && simulationarr[1][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(simulationarr[2][0] == 1 && simulationarr[2][1] == 1 && simulationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

finalFrame.add(group[i][j]);

continue;

}

else if(simulationarr[i][j] == 0 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 1 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 2 && changeDitector[i][j] != -1) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

void showForWinnerTwo()

{

JFrame finalFrame = new JFrame("Winner : Client");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = simulationarr;

group = new JButton[10][10];

if(simulationarr[0][0] == 2 && simulationarr[1][1] == 2 && simulationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][1] == 2 && simulationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][0] == 2 && simulationarr[1][0] == 2 && simulationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][1] == 2 && simulationarr[1][1] == 2 && simulationarr[2][1] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][2] == 2 && simulationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][0] == 2 && simulationarr[0][1] == 2 && simulationarr[0][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(simulationarr[1][0] == 2 && simulationarr[1][1] == 2 && simulationarr[1][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(simulationarr[2][0] == 2 && simulationarr[2][1] == 2 && simulationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

//System.out.println(" lol " + i + " " + j);

finalFrame.add(group[i][j]);

continue;

}

else if(simulationarr[i][j] == 0 && changeDitector[i][j] != -1) {

//System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 1 && changeDitector[i][j] != -1) {

// System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 2 && changeDitector[i][j] != -1) {

// System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

void takeInput(ClientTicTacToe clienttictactoe)

{

Scanner scan = new Scanner(System.in);

System.out.println("Give the name of local host");

Sender = scan.next();

}

}

**// Class Intelligence**

package clienttictactoe;

import javax.swing.\*;

public class Intelligence implements Runnable{

int simulationarr[][];

Thread T;

JFrame nameChange;

public Intelligence(int arr[][],JFrame givenFrame) throws Exception

{

nameChange = givenFrame;

simulationarr = arr;

T = new Thread(this);

T.start();

}

public void run(){

simpleMoveChecking();

simpleMoveCheckingDistantVersion();

simpleMoveCheckingWithMultipleWaysOfWin();

adjecentVersionWithMultipleWaysOfWin();

adjacentAndDistant();

}

void simpleMoveChecking()

{

int posx[] = {-1,1,0,0,-1,1,-1,1};

int posy[] = {0,0, 1,-1,1,-1,-1,1};

int easyFrontx[] = {-2,2,0,0,-2,2,-2,2};

int easyFronty[] = {0,0,2,-2,2,-2,-2,2};

int easyBackx[] = {1,-1,0,0,1,-1,1,-1};

int easyBacky[] = {0,0,-1,1,-1,1,1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

int a = simulationarr[i][j];

for(int k = 0; k <= 7; k++) {

int closex = i + posx[k];

int closey = j + posy[k];

if(closex >= 0 && closex <= 2 && closey >= 0 && closey <= 2) {

int b = simulationarr[closex][closey];

if(a == b && a != 0) {

int nextx = i + easyFrontx[k];

int nexty = j + easyFronty[k];

if(nextx >= 0 && nextx <= 2 && nexty >= 0 && nexty <= 2) {

int c = simulationarr[nextx][nexty];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a winning move");

}

}

}

int anothernextx = i + easyBackx[k];

int anothernexty = j + easyBacky[k];

if(anothernextx >= 0 && anothernextx <= 2 && anothernexty >= 0 && anothernexty <= 2) {

int c = simulationarr[anothernextx][anothernexty];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a good move");

}

}

}

}

}

}

}

}

}

void simpleMoveCheckingDistantVersion()

{

int newPosx[] = {0,0,-2,2,-2,2,-2,2};

int newPosy[] = {2,-2,0,0,2,2,-2,-2};

int midNewPosx[] = {0,0,-1,1,-1,1,-1,1};

int midNewPosy[] = {1,-1,0,0,1,1,-1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

int a = simulationarr[i][j];

for(int k = 0; k <= 7; k++) {

int durerx = i + newPosx[k];

int durery = j + newPosy[k];

if(durerx >= 0 && durerx <= 2 && durery >= 0 && durery <= 2) {

int b = simulationarr[durerx][durery];

if(a == b && a != 0) {

int midx = i + midNewPosx[k];

int midy = j + midNewPosy[k];

if(midx >= 0 && midx <= 2 && midy >= 0 && midy <= 2) {

int c = simulationarr[midx][midy];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a good move");

}

}

}

}

}

}

}

}

}

void simpleMoveCheckingWithMultipleWaysOfWin()

{

int dxforSingle[] = {0,0,-2,2,-2,2,-2,2};

int dyforSingle[] = {2,-2,0,0,2,2,-2,-2};

int dxForMid[] = {0,0,-1,1,-1,1,-1,1};

int dyForMid[] = {1,-1,0,0,1,1,-1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

for(int k = 0; k <= 7; k++) {

int firstPointx = i;

int firstPointy = j;

int lastPointx = i + dxforSingle[k];

int lastPointy = j + dyforSingle[k];

int midPointx = i + dxForMid[k];

int midPointy = j + dyForMid[k];

if(lastPointx >= 0 && lastPointy >= 0 && lastPointx <= 2 && lastPointy <= 2) {

//System.out.println("ase re pagla + " + i + " " + j);

if(midPointx >= 0 && midPointy >= 0 && midPointx <= 2 && midPointy <= 2) {

// System.out.println("ase re pagla 2+ " + i + " " + j);

for(int l = k + 1; l <= 7; l++) {

int templastPointx = i + dxforSingle[l];

int templastPointy = j + dyforSingle[l];

int tempmidPointx = i + dxForMid[l];

int tempmidPointy = j + dyForMid[l];

if(templastPointx >= 0 && templastPointy >= 0 && templastPointx <= 2 && templastPointy <= 2) {

if(tempmidPointx >= 0 && tempmidPointy >= 0 && tempmidPointx <= 2 && tempmidPointy <= 2) {

int a = simulationarr[firstPointx][firstPointy];

int b = simulationarr[lastPointx][lastPointy];

int c = simulationarr[templastPointx][templastPointy];

int midOne = simulationarr[midPointx][midPointy];

int midTwo = simulationarr[tempmidPointx][tempmidPointy];

//System.out.println("ase re pagla 4+ " + i + " " + j);

if(a == b && b == c && midOne == 0 && midTwo == 0) {

// System.out.println("hoi re ");

if(a == 1) {

nameChange.setTitle("Server is winning");

}

if(a == 2) {

nameChange.setTitle("Client is winning");

}

}

}

}

}

}

}

}

}

}

}

void adjecentVersionWithMultipleWaysOfWin()

{

int movex[] = {-1, 1, 0, 0, -1, 1, -1, 1};

int movey[] = {0, 0, 1, -1, 1, -1, -1, 1};

int frontx[]= {-2, 2, 0, 0, -2, 2, -2, 2};

int fronty[]= {0, 0, 2, -2, 2, -2, -2, 2};

int backx[] = {1, -1, 0, 0, 1, -1, 1, -1};

int backy[] = {0, 0, -1, 1, -1, 1, 1, -1};

int playerOneWin = 0,playerTwoWin = 0;

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

for(int k = 0; k <= 7; k++) {

int adjOnex = x + movex[k];

int adjOney = y + movey[k];

playerOneWin = 0;

playerTwoWin = 0;

for(int l = k + 1; l <= 7; l++) {

int adjTwox = x + movex[l];

int adjTwoy = y + movey[l];

int frontadjOnex = x + frontx[k];

int frontadjOney = y + fronty[k];

int backadjOnex = x + backx[k];

int backadjOney = y + backy[k];

int frontadjTwox = x + frontx[l];

int frontadjTwoy = y + fronty[l];

int backadjTwox = x + backx[l];

int backadjTwoy = y + backy[l];

int mainGhor = simulationarr[i][j];

int a,b;

playerOneWin = 0;

playerTwoWin = 0;

if(adjOnex >= 0 && adjOnex <= 2 && adjOney >= 0 && adjOney <= 2 ) {

a = simulationarr[adjOnex][adjOney];

if(adjTwox >= 0 && adjTwox <= 2 && adjTwoy >= 0 && adjTwoy <= 2) {

b = simulationarr[adjTwox][adjTwoy];

if(a == b && a == mainGhor && a != 0) {

if(frontadjOnex >= 0 && frontadjOnex <= 2 && frontadjOney >= 0 && frontadjOney <= 2) {

int c = simulationarr[frontadjOnex][frontadjOney];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(backadjOnex >= 0 && backadjOnex <= 2 && backadjOney >= 0 && backadjOney <= 2) {

int c = simulationarr[backadjOnex][backadjOney];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(frontadjTwox >= 0 && frontadjTwox <= 2 && frontadjTwoy >= 0 && frontadjTwoy <= 2) {

int c = simulationarr[frontadjTwox][frontadjTwoy];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(backadjTwox >= 0 && backadjTwox <= 2 && backadjTwoy >= 0 && backadjTwoy <= 2) {

int c = simulationarr[backadjTwox][backadjTwoy];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

}

}

}

if(playerOneWin > playerTwoWin && playerOneWin > 1) {

nameChange.setTitle("Server can win");

}

if(playerOneWin < playerTwoWin && playerTwoWin > 1) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

void adjacentAndDistant()

{

int mx[] = {-1, 1, 0, 0, -1, 1, -1, 1};

int my[] = {0, 0, 1, -1, 1, -1, -1, 1};

int fx[] = {-2, 2, 0, 0, -2, 2, -2, 2};

int fy[] = {0, 0, 2, -2, 2, -2, -2, 2};

int bx[] = {1, -1, 0, 0, 1, -1, 1, -1};

int by[] = {0, 0, -1, 1, -1, 1, 1, -1};

int durerx[] = {0,0,-2,2,-2,2,-2,2};

int durery[] = {2,-2,0,0,2,-2,-2,2};

int majkhanex[]= {0,0,-1,1,-1,1,-1,1};

int majkhaney[] = {1,-1,0,0,1,-1,-1,1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int mainx = i;

int mainy = j;

int a = simulationarr[mainx][mainy];

for(int k = 0; k <= 7; k++) {

int adjx = i + mx[k];

int adjy = j + my[k];

if(adjx >= 0 && adjx <= 2 && adjy >= 0 && adjy <= 2) {

int b = simulationarr[adjx][adjy];

if(a == b && a != 0) {

int samnerx = i + fx[k];

int samnery = j + fy[k];

int pichonerx = i + bx[k];

int pichonery = j + by[k];

if(samnerx >= 0 && samnerx <= 2 && samnery >= 0 && samnery <= 2) {

int c = simulationarr[samnerx][samnery];

if(c == 0) {

for(int l = 0; l <= 7; l++) {

int distx = i + durerx[l];

int disty = j + durery[l];

int moddhox = i + majkhanex[l];

int moddhoy = j + majkhaney[l];

if(distx >= 0 && distx <= 2 && disty >= 0 && disty <= 2 && moddhox >= 0 && moddhoy <= 2) {

int d = simulationarr[distx][disty];

int e = simulationarr[moddhox][moddhoy];

if(a == d && a != 0 && e == 0) {

if(a == 1) {

nameChange.setTitle("Server can win");

}

if(a == 2) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

if(pichonerx >= 0 && pichonerx <= 2 && pichonery >= 0 && pichonery <= 2) {

int c = simulationarr[pichonerx][pichonery];

if(c == 0) {

for(int l = 0; l <= 7; l++) {

int distx = i + durerx[l];

int disty = j + durery[l];

int moddhox = i + majkhanex[l];

int moddhoy = j + majkhaney[l];

if(distx >= 0 && distx <= 2 && disty >= 0 && disty <= 2 && moddhox >= 0 && moddhoy <= 2) {

int d = simulationarr[distx][disty];

int e = simulationarr[moddhox][moddhoy];

if(a == d && a != 0 && e == 0) {

if(a == 1) {

nameChange.setTitle("Server can win");

}

if(a == 2) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

}

}

}

}

}

}

}

**//Class WinnerDeterminator**

package clienttictactoe;

public class WinnerDeterminator {

boolean playerOne;

boolean playerTwo;

int simulationarr[][];

public WinnerDeterminator(int arr[][])

{

playerOne = false;

playerTwo = false;

simulationarr = arr;

movesOfPlayerOne();

movesOfPlayerTwo();

}

public void movesOfPlayerOne()

{

if(simulationarr[0][0] == 1 && simulationarr[1][1] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][1] == 1 && simulationarr[2][0] == 1) {

playerOne = true;

}

else if(simulationarr[0][0] == 1 && simulationarr[1][0] == 1 && simulationarr[2][0] == 1) {

playerOne = true;

}

else if(simulationarr[0][1] == 1 && simulationarr[1][1] == 1 && simulationarr[2][1] == 1) {

playerOne = true;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][2] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else if(simulationarr[0][0] == 1 && simulationarr[0][1] == 1 && simulationarr[0][2] == 1) {

playerOne = true;

}

else if(simulationarr[1][0] == 1 && simulationarr[1][1] == 1 && simulationarr[1][2] == 1) {

playerOne = true;

}

else if(simulationarr[2][0] == 1 && simulationarr[2][1] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else {

playerOne = false;

}

}

public void movesOfPlayerTwo()

{

if(simulationarr[0][0] == 2 && simulationarr[1][1] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][1] == 2 && simulationarr[2][0] == 2) {

playerTwo = true;

}

else if(simulationarr[0][0] == 2 && simulationarr[1][0] == 2 && simulationarr[2][0] == 2) {

playerTwo = true;

}

else if(simulationarr[0][1] == 2 && simulationarr[1][1] == 2 && simulationarr[2][1] == 2) {

playerTwo = true;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][2] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else if(simulationarr[0][0] == 2 && simulationarr[0][1] == 2 && simulationarr[0][2] == 2) {

playerTwo = true;

}

else if(simulationarr[1][0] == 2 && simulationarr[1][1] == 2 && simulationarr[1][2] == 2) {

playerTwo = true;

}

else if(simulationarr[2][0] == 2 && simulationarr[2][1] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else {

playerTwo = false;

}

System.out.println("class e one = " + playerOne + " two = " + playerTwo);

}

}

**Package ClientTicTacToeFirst**

package clienttictactoefirst;

import javax.swing.\*;

import java.net.\*;

import java.io.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Scanner;

import java.util.logging.Level;

import java.util.logging.Logger;

public class ClientTicTacToeFirst {

JFrame clientFrame;

JButton grid[][];

String bitString;

int counter;

int simulationarr[][];

boolean playerOneWin = false;

boolean playerTwoWin = false;

protected String Sender = "localhost";

protected int portNumber = 786;

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

ClientTicTacToeFirst client = new ClientTicTacToeFirst();

client.takeInput(client);

client.creation(client);

}

public void creation(ClientTicTacToeFirst clienttictactoefirst) throws Exception

{

simulationarr = new int[10][10];

clientFrame = new JFrame("Client Window");

clientFrame.setSize(700,700);

clientFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

clientFrame.setLayout(new GridLayout(3,3));

clientFrame.setBackground(Color.yellow);

grid = new JButton[10][10];

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j] = new JButton(image);

clientFrame.add(grid[i][j]);

}

}

clientFrame.setVisible(true);

clienttictactoefirst.run();

}

void perfectingTheScenario()

{

int row = 0,col = 0;

for(int i = 1; i < bitString.length(); i++) {

if(i <= 3) {

row = 0;

}

else if(i <= 6) {

row = 1;

}

else if(i <= 9) {

row = 2;

}

int d = i - 1;

simulationarr[row][d%3] = bitString.charAt(i) - '0';

//System.out.println(simulationarr[row][d%3] + " " + row + " " + d%3);

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(simulationarr[i][j] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

grid[i][j].setIcon(image);

}

if(simulationarr[i][j] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

grid[i][j].setIcon(image);

}

if(simulationarr[i][j] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[i][j] = new JButton(image);

}

}

}

}

void bitStringChange()

{

//counter = counter + 1;

char a = (char) (counter + '0');

bitString = "";

bitString = bitString + a;

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

a = (char) (simulationarr[i][j] + '0');

bitString = bitString + a;

}

}

}

public void run() throws Exception

{

Socket socket = new Socket(Sender,portNumber);

boolean program = true;

while(program == true) {

DataInputStream in = new DataInputStream(socket.getInputStream());

DataOutputStream out = new DataOutputStream(socket.getOutputStream());

bitString = "";

bitString = in.readUTF();

System.out.println(bitString + "asche");

counter = bitString.charAt(0) - '0';

perfectingTheScenario();

counter = counter + 1;

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

Intelligence intelligent = new Intelligence(simulationarr,clientFrame);

if(playerOneWin == true) {

out.writeUTF(bitString);

showForWinnerOne();

}

else if(playerTwoWin == true) {

out.writeUTF(bitString);

showForWinnerTwo();

}

clientFrame.setVisible(true);

if(counter <= 9) clientFrame.setTitle("Client");

if(counter > 9) {

if(playerOneWin == false && playerTwoWin == false) {

clientFrame.setTitle("Draw !!!");

}

}

grid[0][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][0].isEnabled() && simulationarr[0][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[0][0].setIcon(image);

simulationarr[0][0] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringChange();

System.out.println(bitString + " 1");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][1].isEnabled() && simulationarr[0][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[0][1].setIcon(image);

simulationarr[0][1] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

//System.out.println(bitString + " 2");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[0][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[0][2].isEnabled() && simulationarr[0][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[0][2].setIcon(image);

simulationarr[0][2] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

System.out.println(bitString + " 2");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][0].isEnabled() && simulationarr[1][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[1][0].setIcon(image);

simulationarr[1][0] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

bitStringChange();

// System.out.println(bitString + " 4");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][1].isEnabled() && simulationarr[1][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[1][1].setIcon(image);

simulationarr[1][1] = 2;

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 5");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[1][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[1][2].isEnabled() && simulationarr[1][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[1][2].setIcon(image);

simulationarr[1][2] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 6");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][0].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][0].isEnabled() && simulationarr[2][0] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[2][0].setIcon(image);

simulationarr[2][0] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

// System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 7");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][1].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][1].isEnabled() && simulationarr[2][1] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[2][1].setIcon(image);

simulationarr[2][1] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

//System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString+ " 8");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

grid[2][2].addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

if(grid[2][2].isEnabled() && simulationarr[2][2] == 0) {

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

grid[2][2].setIcon(image);

simulationarr[2][2] = 2;

try {

Intelligence intelligence = new Intelligence(simulationarr,clientFrame);

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

WinnerDeterminator w = new WinnerDeterminator(simulationarr);

playerOneWin = w.playerOne;

playerTwoWin = w.playerTwo;

//System.out.println("one = " + w.playerOne + " " + w.playerTwo);

try {

bitStringChange();

// System.out.println(bitString + " 9");

out.writeUTF(bitString);

if(playerOneWin == true) showForWinnerOne();

if(playerTwoWin == true) showForWinnerTwo();

} catch (Exception ex) {

Logger.getLogger(ClientTicTacToeFirst.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

});

}

}

void showForWinnerOne()

{

JFrame finalFrame = new JFrame("Winner : Server");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = simulationarr;

group = new JButton[10][10];

if(simulationarr[0][0] == 1 && simulationarr[1][1] == 1 && simulationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][1] == 1 && simulationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][0] == 1 && simulationarr[1][0] == 1 && simulationarr[2][0] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][1] == 1 && simulationarr[1][1] == 1 && simulationarr[2][1] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][2] == 1 && simulationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][0] == 1 && simulationarr[0][1] == 1 && simulationarr[0][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(simulationarr[1][0] == 1 && simulationarr[1][1] == 1 && simulationarr[1][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(simulationarr[2][0] == 1 && simulationarr[2][1] == 1 && simulationarr[2][2] == 1) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

System.out.println(" lol " + i + " " + j);

finalFrame.add(group[i][j]);

continue;

}

else if(simulationarr[i][j] == 0 && changeDitector[i][j] != -1) {

System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 1 && changeDitector[i][j] != -1) {

System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 2 && changeDitector[i][j] != -1) {

System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

void showForWinnerTwo()

{

JFrame finalFrame = new JFrame("Winner : Client");

finalFrame.setSize(700,700);

finalFrame.setLayout(new GridLayout(3,3));

finalFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton group[][];

int changeDitector[][];

changeDitector = simulationarr;

group = new JButton[10][10];

if(simulationarr[0][0] == 2 && simulationarr[1][1] == 2 && simulationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][1] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][1] == 2 && simulationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][1] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][1] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][0] == 2 && simulationarr[1][0] == 2 && simulationarr[2][0] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[1][0] = new JButton(image);

group[2][0] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[1][0] = -1;

changeDitector[2][0] = -1;

}

else if(simulationarr[0][1] == 2 && simulationarr[1][1] == 2 && simulationarr[2][1] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][1] = new JButton(image);

group[1][1] = new JButton(image);

group[2][1] = new JButton(image);

changeDitector[0][1] = -1;

changeDitector[1][1] = -1;

changeDitector[2][1] = -1;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][2] == 2 && simulationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][2] = new JButton(image);

group[1][2] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[0][2] = -1;

changeDitector[1][2] = -1;

changeDitector[2][2] = -1;

}

else if(simulationarr[0][0] == 2 && simulationarr[0][1] == 2 && simulationarr[0][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[0][0] = new JButton(image);

group[0][1] = new JButton(image);

group[0][2] = new JButton(image);

changeDitector[0][0] = -1;

changeDitector[0][1] = -1;

changeDitector[0][2] = -1;

}

else if(simulationarr[1][0] == 2 && simulationarr[1][1] == 2 && simulationarr[1][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[1][0] = new JButton(image);

group[1][1] = new JButton(image);

group[1][2] = new JButton(image);

changeDitector[1][0] = -1;

changeDitector[1][1] = -1;

changeDitector[1][2] = -1;

}

else if(simulationarr[2][0] == 2 && simulationarr[2][1] == 2 && simulationarr[2][2] == 2) {

ImageIcon image = new ImageIcon(getClass().getResource("finalimage.png"));

group[2][0] = new JButton(image);

group[2][1] = new JButton(image);

group[2][2] = new JButton(image);

changeDitector[2][0] = -1;

changeDitector[2][1] = -1;

changeDitector[2][2] = -1;

}

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

if(changeDitector[i][j] == -1){

//System.out.println(" lol " + i + " " + j);

finalFrame.add(group[i][j]);

continue;

}

else if(simulationarr[i][j] == 0 && changeDitector[i][j] != -1) {

//System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("firstlook.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 1 && changeDitector[i][j] != -1) {

// System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("atlast.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

else if(simulationarr[i][j] == 2 && changeDitector[i][j] != -1) {

// System.out.println("hai = " + i + " " + j);

ImageIcon image = new ImageIcon(getClass().getResource("circleimage.jpg"));

group[i][j] = new JButton(image);

finalFrame.add(group[i][j]);

}

}

}

finalFrame.setVisible(true);

}

void takeInput(ClientTicTacToeFirst client) {

Scanner scan = new Scanner(System.in);

System.out.println("Give the name of your local host");

Sender = scan.next();

}

}package clienttictactoefirst;

import javax.swing.\*;

public class Intelligence implements Runnable{

int simulationarr[][];

Thread T;

JFrame nameChange;

public Intelligence(int arr[][],JFrame givenFrame) throws Exception

{

nameChange = givenFrame;

simulationarr = arr;

T = new Thread(this);

T.start();

}

public void run(){

simpleMoveChecking();

simpleMoveCheckingDistantVersion();

simpleMoveCheckingWithMultipleWaysOfWin();

adjecentVersionWithMultipleWaysOfWin();

adjacentAndDistant();

}

void simpleMoveChecking()

{

int posx[] = {-1,1,0,0,-1,1,-1,1};

int posy[] = {0,0, 1,-1,1,-1,-1,1};

int easyFrontx[] = {-2,2,0,0,-2,2,-2,2};

int easyFronty[] = {0,0,2,-2,2,-2,-2,2};

int easyBackx[] = {1,-1,0,0,1,-1,1,-1};

int easyBacky[] = {0,0,-1,1,-1,1,1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

int a = simulationarr[i][j];

for(int k = 0; k <= 7; k++) {

int closex = i + posx[k];

int closey = j + posy[k];

if(closex >= 0 && closex <= 2 && closey >= 0 && closey <= 2) {

int b = simulationarr[closex][closey];

if(a == b && a != 0) {

int nextx = i + easyFrontx[k];

int nexty = j + easyFronty[k];

if(nextx >= 0 && nextx <= 2 && nexty >= 0 && nexty <= 2) {

int c = simulationarr[nextx][nexty];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a winning move");

}

}

}

int anothernextx = i + easyBackx[k];

int anothernexty = j + easyBacky[k];

if(anothernextx >= 0 && anothernextx <= 2 && anothernexty >= 0 && anothernexty <= 2) {

int c = simulationarr[anothernextx][anothernexty];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a good move");

}

}

}

}

}

}

}

}

}

void simpleMoveCheckingDistantVersion()

{

int newPosx[] = {0,0,-2,2,-2,2,-2,2};

int newPosy[] = {2,-2,0,0,2,2,-2,-2};

int midNewPosx[] = {0,0,-1,1,-1,1,-1,1};

int midNewPosy[] = {1,-1,0,0,1,1,-1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

int a = simulationarr[i][j];

for(int k = 0; k <= 7; k++) {

int durerx = i + newPosx[k];

int durery = j + newPosy[k];

if(durerx >= 0 && durerx <= 2 && durery >= 0 && durery <= 2) {

int b = simulationarr[durerx][durery];

if(a == b && a != 0) {

int midx = i + midNewPosx[k];

int midy = j + midNewPosy[k];

if(midx >= 0 && midx <= 2 && midy >= 0 && midy <= 2) {

int c = simulationarr[midx][midy];

if(c == 0) {

if(a == 1) {

nameChange.setTitle("Server has given a good move");

}

if(a == 2) {

nameChange.setTitle("Client has given a good move");

}

}

}

}

}

}

}

}

}

void simpleMoveCheckingWithMultipleWaysOfWin()

{

int dxforSingle[] = {0,0,-2,2,-2,2,-2,2};

int dyforSingle[] = {2,-2,0,0,2,2,-2,-2};

int dxForMid[] = {0,0,-1,1,-1,1,-1,1};

int dyForMid[] = {1,-1,0,0,1,1,-1,-1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

for(int k = 0; k <= 7; k++) {

int firstPointx = i;

int firstPointy = j;

int lastPointx = i + dxforSingle[k];

int lastPointy = j + dyforSingle[k];

int midPointx = i + dxForMid[k];

int midPointy = j + dyForMid[k];

if(lastPointx >= 0 && lastPointy >= 0 && lastPointx <= 2 && lastPointy <= 2) {

//System.out.println("ase re pagla + " + i + " " + j);

if(midPointx >= 0 && midPointy >= 0 && midPointx <= 2 && midPointy <= 2) {

// System.out.println("ase re pagla 2+ " + i + " " + j);

for(int l = k + 1; l <= 7; l++) {

int templastPointx = i + dxforSingle[l];

int templastPointy = j + dyforSingle[l];

int tempmidPointx = i + dxForMid[l];

int tempmidPointy = j + dyForMid[l];

if(templastPointx >= 0 && templastPointy >= 0 && templastPointx <= 2 && templastPointy <= 2) {

if(tempmidPointx >= 0 && tempmidPointy >= 0 && tempmidPointx <= 2 && tempmidPointy <= 2) {

int a = simulationarr[firstPointx][firstPointy];

int b = simulationarr[lastPointx][lastPointy];

int c = simulationarr[templastPointx][templastPointy];

int midOne = simulationarr[midPointx][midPointy];

int midTwo = simulationarr[tempmidPointx][tempmidPointy];

//System.out.println("ase re pagla 4+ " + i + " " + j);

if(a == b && b == c && midOne == 0 && midTwo == 0) {

// System.out.println("hoi re ");

if(a == 1) {

nameChange.setTitle("Server is winning");

}

if(a == 2) {

nameChange.setTitle("Client is winning");

}

}

}

}

}

}

}

}

}

}

}

void adjecentVersionWithMultipleWaysOfWin()

{

int movex[] = {-1, 1, 0, 0, -1, 1, -1, 1};

int movey[] = {0, 0, 1, -1, 1, -1, -1, 1};

int frontx[]= {-2, 2, 0, 0, -2, 2, -2, 2};

int fronty[]= {0, 0, 2, -2, 2, -2, -2, 2};

int backx[] = {1, -1, 0, 0, 1, -1, 1, -1};

int backy[] = {0, 0, -1, 1, -1, 1, 1, -1};

int playerOneWin = 0,playerTwoWin = 0;

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int x = i;

int y = j;

for(int k = 0; k <= 7; k++) {

int adjOnex = x + movex[k];

int adjOney = y + movey[k];

playerOneWin = 0;

playerTwoWin = 0;

for(int l = k + 1; l <= 7; l++) {

int adjTwox = x + movex[l];

int adjTwoy = y + movey[l];

int frontadjOnex = x + frontx[k];

int frontadjOney = y + fronty[k];

int backadjOnex = x + backx[k];

int backadjOney = y + backy[k];

int frontadjTwox = x + frontx[l];

int frontadjTwoy = y + fronty[l];

int backadjTwox = x + backx[l];

int backadjTwoy = y + backy[l];

int mainGhor = simulationarr[i][j];

int a,b;

playerOneWin = 0;

playerTwoWin = 0;

if(adjOnex >= 0 && adjOnex <= 2 && adjOney >= 0 && adjOney <= 2 ) {

a = simulationarr[adjOnex][adjOney];

if(adjTwox >= 0 && adjTwox <= 2 && adjTwoy >= 0 && adjTwoy <= 2) {

b = simulationarr[adjTwox][adjTwoy];

if(a == b && a == mainGhor && a != 0) {

if(frontadjOnex >= 0 && frontadjOnex <= 2 && frontadjOney >= 0 && frontadjOney <= 2) {

int c = simulationarr[frontadjOnex][frontadjOney];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(backadjOnex >= 0 && backadjOnex <= 2 && backadjOney >= 0 && backadjOney <= 2) {

int c = simulationarr[backadjOnex][backadjOney];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(frontadjTwox >= 0 && frontadjTwox <= 2 && frontadjTwoy >= 0 && frontadjTwoy <= 2) {

int c = simulationarr[frontadjTwox][frontadjTwoy];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

if(backadjTwox >= 0 && backadjTwox <= 2 && backadjTwoy >= 0 && backadjTwoy <= 2) {

int c = simulationarr[backadjTwox][backadjTwoy];

if(c == 0) {

if(a == 1) {

playerOneWin++;

}

if(a == 2) {

playerTwoWin++;

}

}

}

}

}

}

if(playerOneWin > playerTwoWin && playerOneWin > 1) {

nameChange.setTitle("Server can win");

}

if(playerOneWin < playerTwoWin && playerTwoWin > 1) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

void adjacentAndDistant()

{

int mx[] = {-1, 1, 0, 0, -1, 1, -1, 1};

int my[] = {0, 0, 1, -1, 1, -1, -1, 1};

int fx[] = {-2, 2, 0, 0, -2, 2, -2, 2};

int fy[] = {0, 0, 2, -2, 2, -2, -2, 2};

int bx[] = {1, -1, 0, 0, 1, -1, 1, -1};

int by[] = {0, 0, -1, 1, -1, 1, 1, -1};

int durerx[] = {0,0,-2,2,-2,2,-2,2};

int durery[] = {2,-2,0,0,2,-2,-2,2};

int majkhanex[]= {0,0,-1,1,-1,1,-1,1};

int majkhaney[] = {1,-1,0,0,1,-1,-1,1};

for(int i = 0; i < 3; i++) {

for(int j = 0; j < 3; j++) {

int mainx = i;

int mainy = j;

int a = simulationarr[mainx][mainy];

for(int k = 0; k <= 7; k++) {

int adjx = i + mx[k];

int adjy = j + my[k];

if(adjx >= 0 && adjx <= 2 && adjy >= 0 && adjy <= 2) {

int b = simulationarr[adjx][adjy];

if(a == b && a != 0) {

int samnerx = i + fx[k];

int samnery = j + fy[k];

int pichonerx = i + bx[k];

int pichonery = j + by[k];

if(samnerx >= 0 && samnerx <= 2 && samnery >= 0 && samnery <= 2) {

int c = simulationarr[samnerx][samnery];

if(c == 0) {

for(int l = 0; l <= 7; l++) {

int distx = i + durerx[l];

int disty = j + durery[l];

int moddhox = i + majkhanex[l];

int moddhoy = j + majkhaney[l];

if(distx >= 0 && distx <= 2 && disty >= 0 && disty <= 2 && moddhox >= 0 && moddhoy <= 2) {

int d = simulationarr[distx][disty];

int e = simulationarr[moddhox][moddhoy];

if(a == d && a != 0 && e == 0) {

if(a == 1) {

nameChange.setTitle("Server can win");

}

if(a == 2) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

if(pichonerx >= 0 && pichonerx <= 2 && pichonery >= 0 && pichonery <= 2) {

int c = simulationarr[pichonerx][pichonery];

if(c == 0) {

for(int l = 0; l <= 7; l++) {

int distx = i + durerx[l];

int disty = j + durery[l];

int moddhox = i + majkhanex[l];

int moddhoy = j + majkhaney[l];

if(distx >= 0 && distx <= 2 && disty >= 0 && disty <= 2 && moddhox >= 0 && moddhoy <= 2) {

int d = simulationarr[distx][disty];

int e = simulationarr[moddhox][moddhoy];

if(a == d && a != 0 && e == 0) {

if(a == 1) {

nameChange.setTitle("Server can win");

}

if(a == 2) {

nameChange.setTitle("Client can win");

}

}

}

}

}

}

}

}

}

}

}

}

}

package clienttictactoefirst;

public class WinnerDeterminator {

boolean playerOne;

boolean playerTwo;

int simulationarr[][];

public WinnerDeterminator(int arr[][])

{

playerOne = false;

playerTwo = false;

simulationarr = arr;

movesOfPlayerOne();

movesOfPlayerTwo();

}

public void movesOfPlayerOne()

{

if(simulationarr[0][0] == 1 && simulationarr[1][1] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][1] == 1 && simulationarr[2][0] == 1) {

playerOne = true;

}

else if(simulationarr[0][0] == 1 && simulationarr[1][0] == 1 && simulationarr[2][0] == 1) {

playerOne = true;

}

else if(simulationarr[0][1] == 1 && simulationarr[1][1] == 1 && simulationarr[2][1] == 1) {

playerOne = true;

}

else if(simulationarr[0][2] == 1 && simulationarr[1][2] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else if(simulationarr[0][0] == 1 && simulationarr[0][1] == 1 && simulationarr[0][2] == 1) {

playerOne = true;

}

else if(simulationarr[1][0] == 1 && simulationarr[1][1] == 1 && simulationarr[1][2] == 1) {

playerOne = true;

}

else if(simulationarr[2][0] == 1 && simulationarr[2][1] == 1 && simulationarr[2][2] == 1) {

playerOne = true;

}

else {

playerOne = false;

}

}

public void movesOfPlayerTwo()

{

if(simulationarr[0][0] == 2 && simulationarr[1][1] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][1] == 2 && simulationarr[2][0] == 2) {

playerTwo = true;

}

else if(simulationarr[0][0] == 2 && simulationarr[1][0] == 2 && simulationarr[2][0] == 2) {

playerTwo = true;

}

else if(simulationarr[0][1] == 2 && simulationarr[1][1] == 2 && simulationarr[2][1] == 2) {

playerTwo = true;

}

else if(simulationarr[0][2] == 2 && simulationarr[1][2] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else if(simulationarr[0][0] == 2 && simulationarr[0][1] == 2 && simulationarr[0][2] == 2) {

playerTwo = true;

}

else if(simulationarr[1][0] == 2 && simulationarr[1][1] == 2 && simulationarr[1][2] == 2) {

playerTwo = true;

}

else if(simulationarr[2][0] == 2 && simulationarr[2][1] == 2 && simulationarr[2][2] == 2) {

playerTwo = true;

}

else {

playerTwo = false;

}

System.out.println("class e one = " + playerOne + " two = " + playerTwo);

}

}