試験報告書

プロジェクト１

グループ I

提出日：2018年5月25日

1. はじめに

本報告書は，ネットワーク対戦型オセロゲームシステムの試験方法および試験結果を記すものである．試験対象であるシステムのソースコードおよびクラス図は以下の通りである．

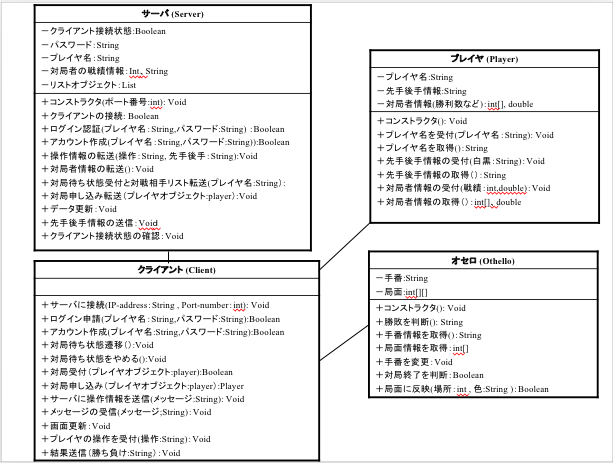
ソースコード一覧：

Player.java PlayerDriver.java (テスト用ドライバ)

Client.java ClientDriver.java (テスト用ドライバ)

Othello.java OthelloDriver.java(テスト用ドライバ)

Server.java PlayerArrayList.java

クラス図

1. 単体テスト
   1. Playerクラス

以下のドライバを用いてPlayerクラスの単体テストを行った．なお，スタブは使用していない．

package othello;

public class PlayerDriver {

public static void main(String[] args) {

Player playerdri = new Player("player", "password");

System.out.println("getName出力:" + playerdri.getName());

System.out.println("getPassword出力:" + playerdri.getPassword());

System.out.println("setRateで[2000]を入力");

playerdri.setRate(2000);

System.out.println("getRate出力:" + playerdri.getRate());

System.out.println("setWinで[50]を入力");

playerdri.setWin(50);

System.out.println("getWin出力:" + playerdri.getWin());

System.out.println("setDefeatで[30]を入力");

playerdri.setDefeat(30);

System.out.println("getDefeat出力:" + playerdri.getDefeat());

System.out.println("setDrawで[15]を入力");

playerdri.setDraw(15);

System.out.println("getDrawt出力:" + playerdri.getDraw());

System.out.println("setColorで[black]を入力");

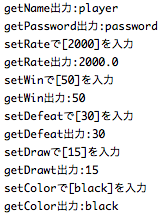
playerdri.setColor("black");

System.out.println("getColor出力:" + playerdri.getColor());

}

}

試験結果：



* 1. Othelloクラス

以下のドライバを用いてOthelloクラスの単体テストを行った．なお，スタブは使用していない．

package othello;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class OthelloDriver {

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

BufferedReader r = new BufferedReader(new InputStreamReader(System.in), 1);

Othello game = new Othello();

System.out.println("テスト：Othelloクラスのオブジェクトを初期化");

printStatus(game);

printGrids(game);

while(true){

System.out.println("石を置く場所(数字またはpass)をキーボードで入力してください");

String s = r.readLine();//文字列の入力

System.out.println(s + " が入力されました。手番は " + game.getTurn() + " です。");

game.putStone(Integer.parseInt(s), game.getTurn());

printStatus(game);

printGrids(game);

System.out.println("手番を変更します。\n");

game.chengeTurn();

}

}

//状態を表示する

public static void printStatus(Othello game){

int[] grids = game.getGrids();

System.out.println("checkWinner出力:" + game.checkWinner());

System.out.println("isGameover出力:" + game.isGameover(game.getTurn(),grids));

System.out.println("getTurn出力：" + game.getTurn());

}

//テスト用に盤面を表示する

public static void printGrids(Othello game){

int[] grids = game.getGrids();

int row = 8;

System.out.println("getRow出力：" + row);

System.out.println("Gridsテスト出力");

for(int i = 0 ; i < row \* row ; i++){

System.out.print(grids[i] + " ");

if(i % row == row - 1){

System.out.print("\n");

}

}

}

}

String s = r.readLine();//文字列の入力

System.out.println(s + " が入力されました。手番は " + game.getTurn() + " です。");

game.putStone(Integer.parseInt(s), game.getTurn());

printStatus(game);

printGrids(game);

System.out.println("手番を変更します。\n");

game.chengeTurn();

}

}

//状態を表示する

public static void printStatus(Othello game){

int[] grids = game.getGrids();

System.out.println("checkWinner出力:" + game.checkWinner());

System.out.println("isGameover出力:" + game.isGameover(game.getTurn(),grids));

System.out.println("getTurn出力：" + game.getTurn());

}

//テスト用に盤面を表示する

public static void printGrids(Othello game){

int[] grids = game.getGrids();

int row = 8;

System.out.println("getRow出力：" + row);

System.out.println("Gridsテスト出力");

for(int i = 0 ; i < row \* row ; i++){

System.out.print(grids[i] + " ");

if(i % row == row - 1){

System.out.print("\n");

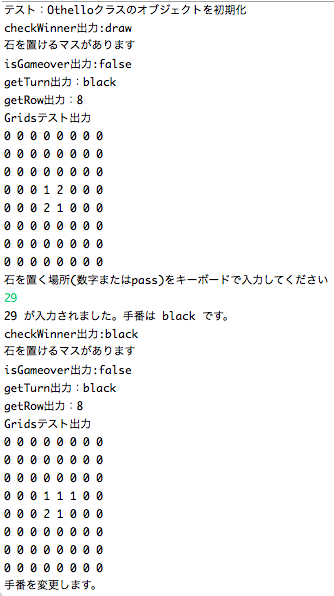
}

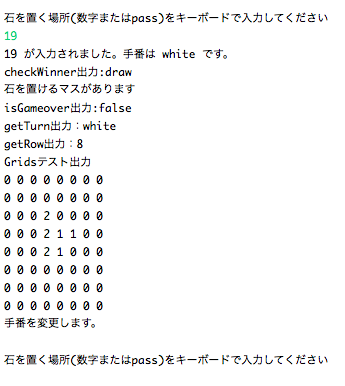
}

}

}

試験結果：





* 1. Clientクラス

以下のドライバを用いてClientクラスの単体テストを行った．なお，スタブはOthelloクラスとPlayerクラスを用意した．また、Clientクラスではサーバとのネットワーク接続が発生するが、エコーサーバで代用した。実際のサーバークラスとの接続は結合テストにおいて試験する．

package othello;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ClientDriver {

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

BufferedReader r = new BufferedReader(new InputStreamReader(System.in), 1);

Player player = new Player("PlayerA","1"); //プレイヤオブジェクトの用意

Othello game = new Othello(); //オセロオブジェクトを用意

Client clientdri = new Client(game); //引数としてオセロオブジェクトを渡す

System.out.println("テスト用サーバに接続します");

clientdri.connectServer("localhost", 10030);

System.out.println("接続しました");

System.out.println("オセロテストへ移行");

clientdri.mycolor = "black";

clientdri.playothello();

while(true){

System.out.println("石を置く場所を数字でキーボードで入力してください");

String s = r.readLine();//文字列の入力

System.out.println(s + " が入力されました。手番は " + game.getTurn() + " です。");

clientdri.receiveMessage(s);

System.out.println("テストメッセージ「" + s + "」を受信しました");

boolean putresult = game.putStone(Integer.parseInt(s), game.getTurn());

if(putresult == true) {

game.chengeTurn();

}

clientdri.updateDisp();

clientdri.Board.repaint();

}

}

}

InputStreamReader(System.in), 1);

Player player = new Player("PlayerA","1"); //プレイヤオブジェクトの用意

Othello game = new Othello(); //オセロオブジェクトを用意

Client clientdri = new Client(game); //引数としてオセロオブジェクトを渡す

System.out.println("テスト用サーバに接続します");

clientdri.connectServer("localhost", 10030);

System.out.println("接続しました");

System.out.println("オセロテストへ移行");

clientdri.mycolor = "black";

clientdri.playothello();

while(true){

System.out.println("石を置く場所を数字でキーボードで入力してください");

String s = r.readLine();//文字列の入力

System.out.println(s + " が入力されました。手番は " + game.getTurn() + " です。");

clientdri.receiveMessage(s);

System.out.println("テストメッセージ「" + s + "」を受信しました");

boolean putresult = game.putStone(Integer.parseInt(s), game.getTurn());

if(putresult == true) {

game.chengeTurn();

}

clientdri.updateDisp();

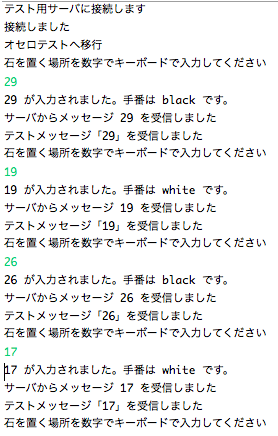
clientdri.Board.repaint();

}

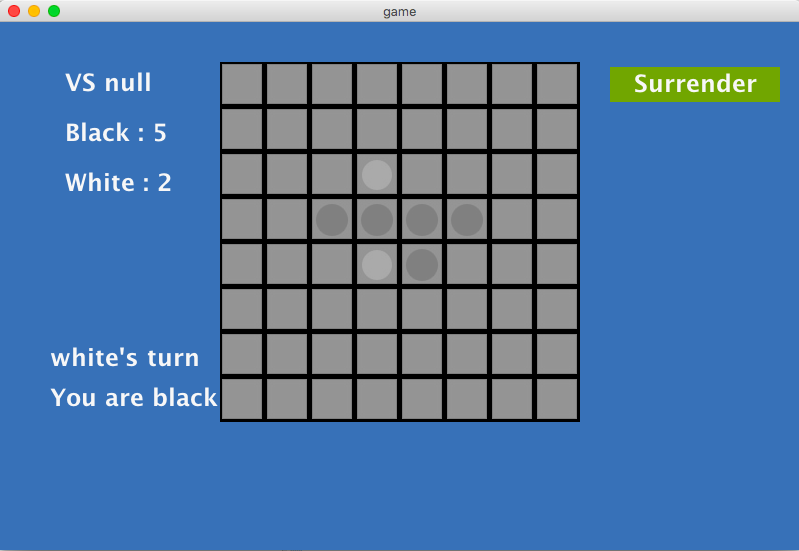
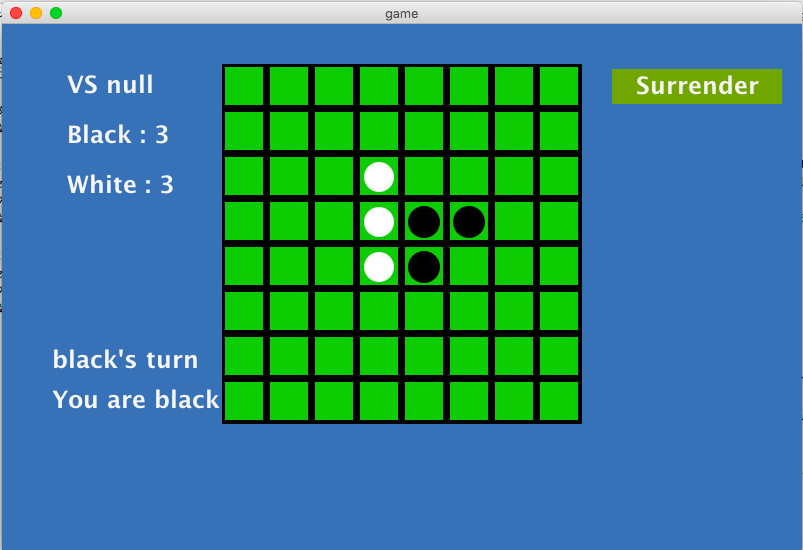
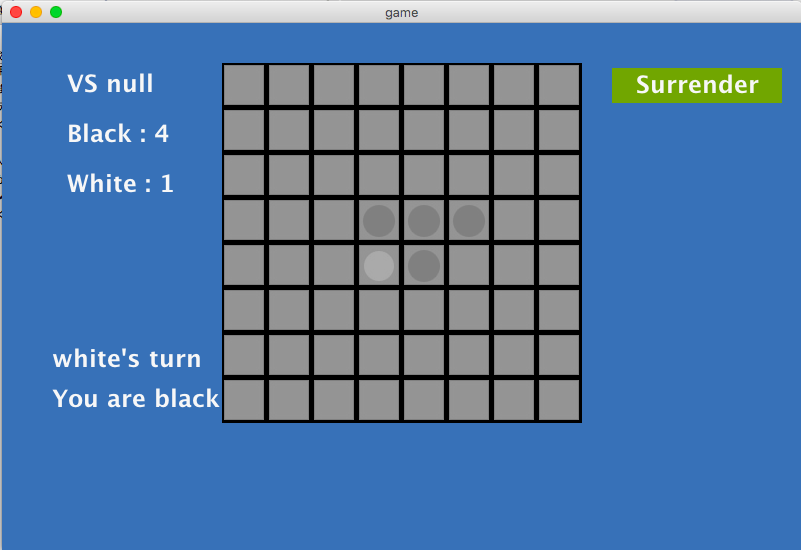
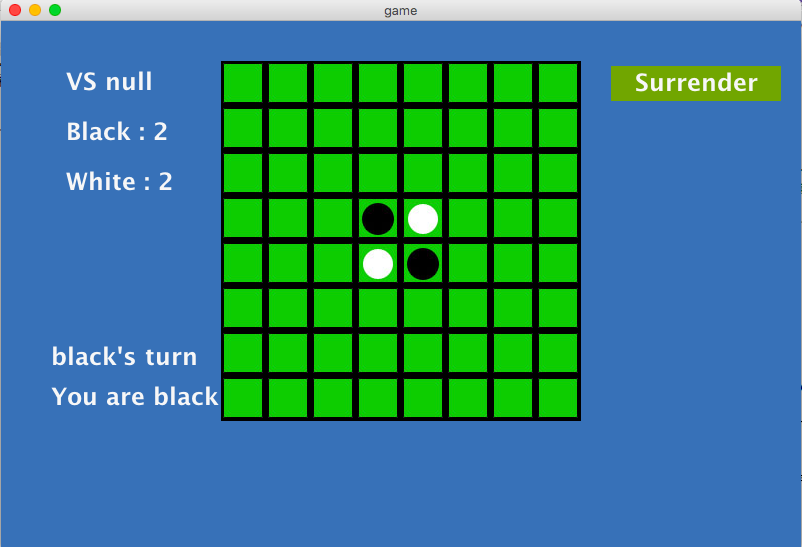
}

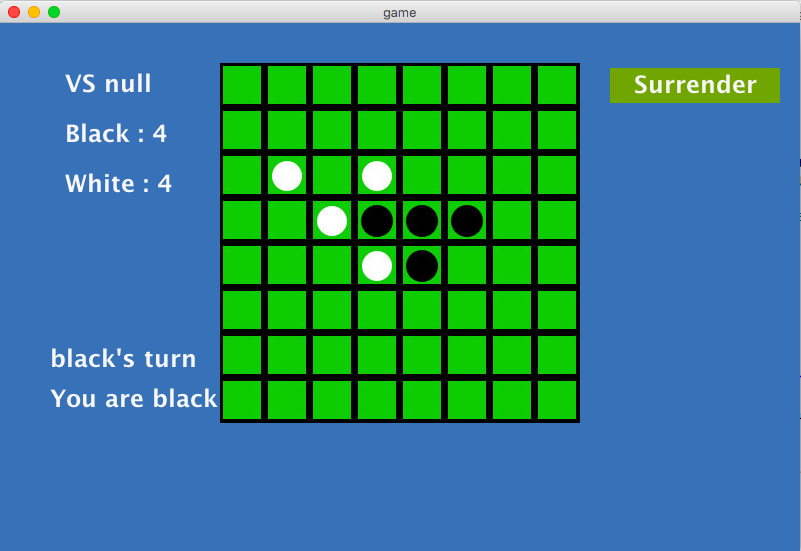
}

試験結果（コンソール出力）：



試験結果（GUI）：





* 1. Serverクラス

ServerクラスはClientクラスとの接続が必要であり、ログインや対局関係のテストはClientDriverクラスとの接続ではなく、Clientクラスとの接続が必要となるため、単体テストではなく、結合テストで試験を行う。

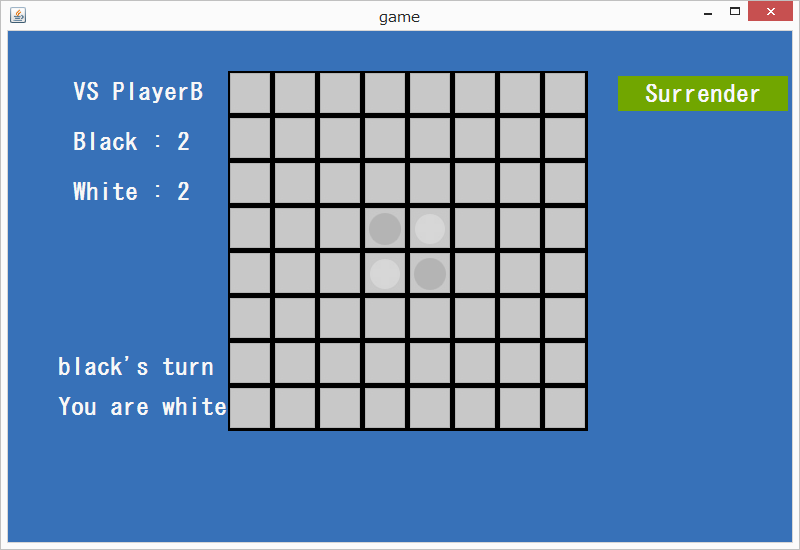
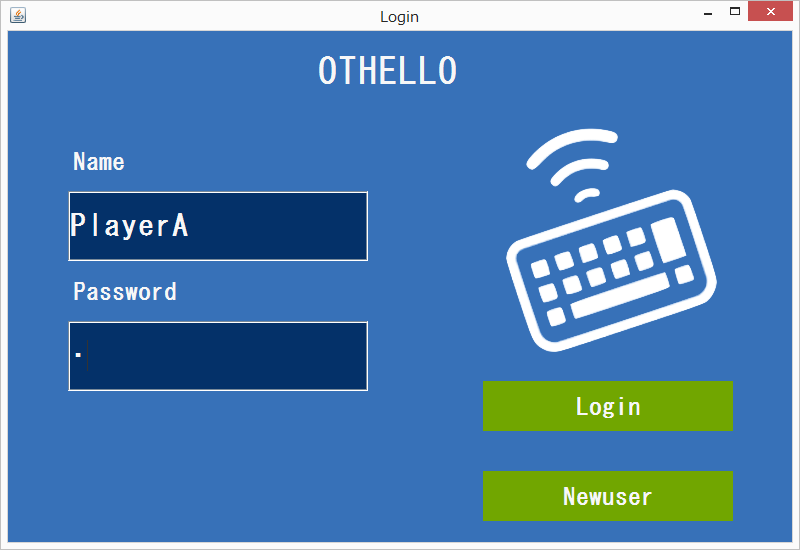
1. 結合テスト

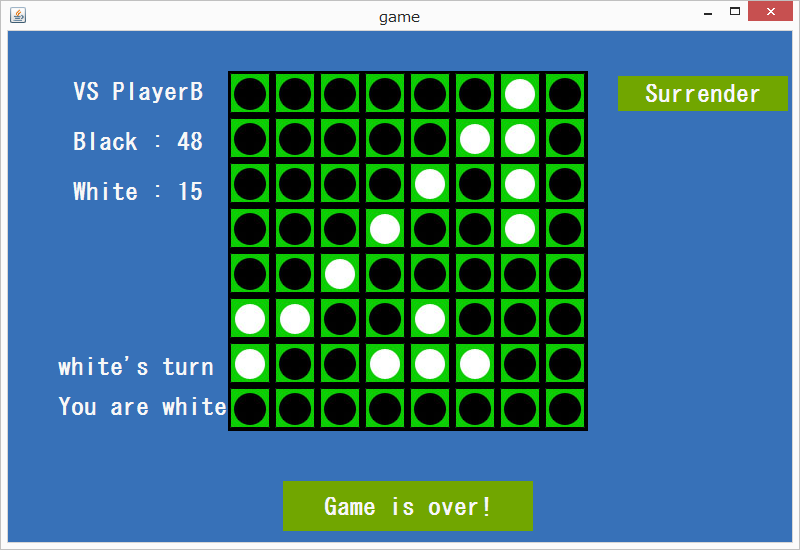
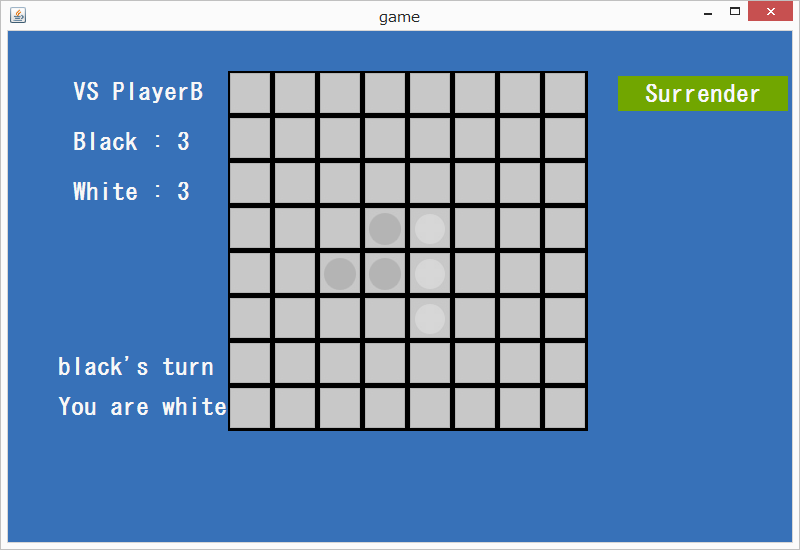
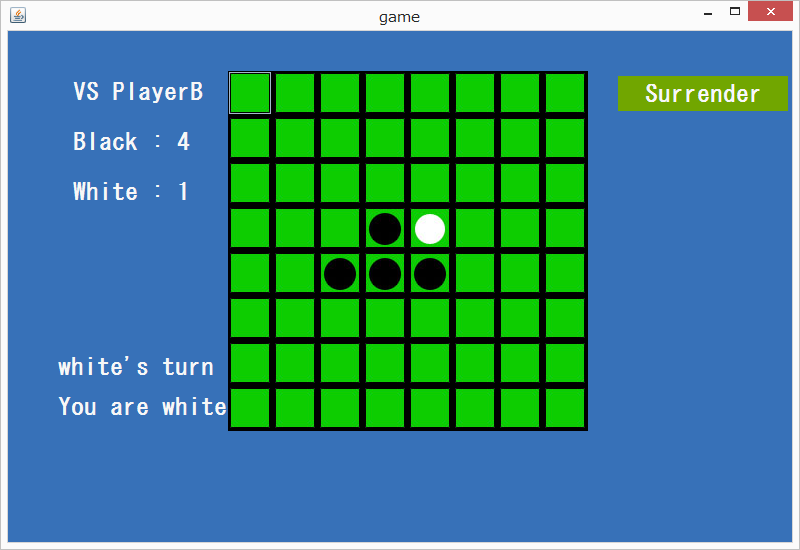
以下の手順に従い，結合テストを行った．

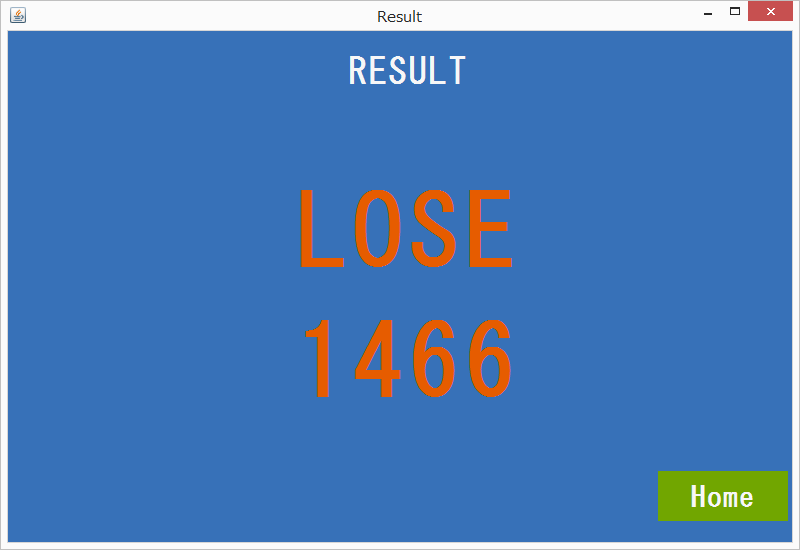
1. Serverプログラムを起動
2. Clientプログラム１を起動し、プレイヤ「PlayerA」でログイン
3. Clientプログラム２を起動し、プレイヤ「PlayerB」でログイン
4. PlayerAをマッチング画面に移行させ、その後PlayerBをマッチング画面に移行させて、PlayerBがPlayerAに対局を申し込み、対局を開始する
5. 対局を進める
6. ゲーム終了後、成績を確認し、Clientプログラム１を終了し、サーバから切断
7. ゲーム終了後、成績を確認し、Clientプログラム２を終了し、サーバから切断

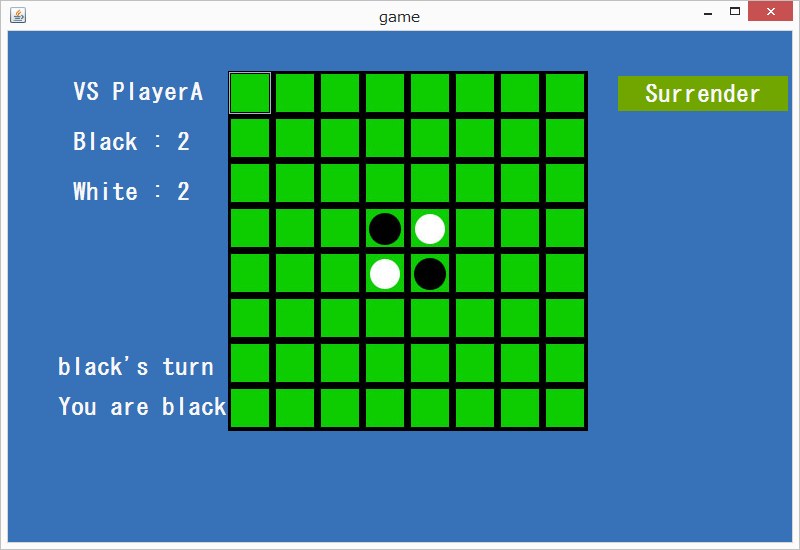
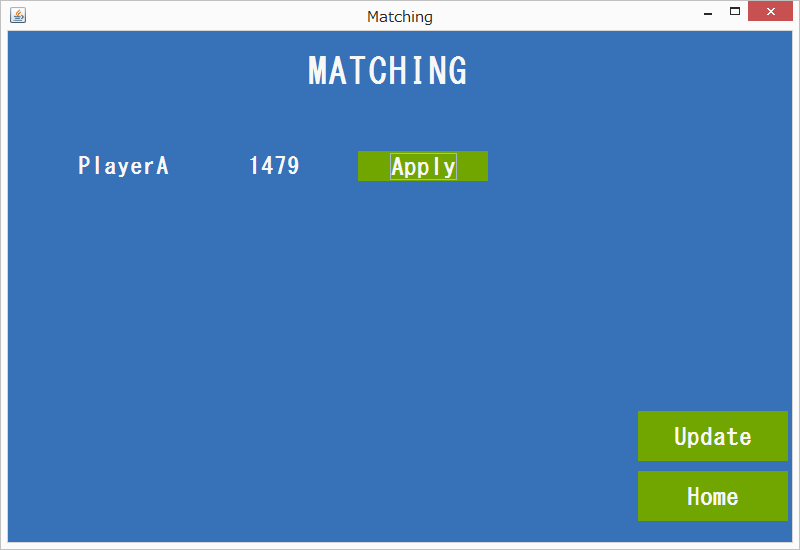
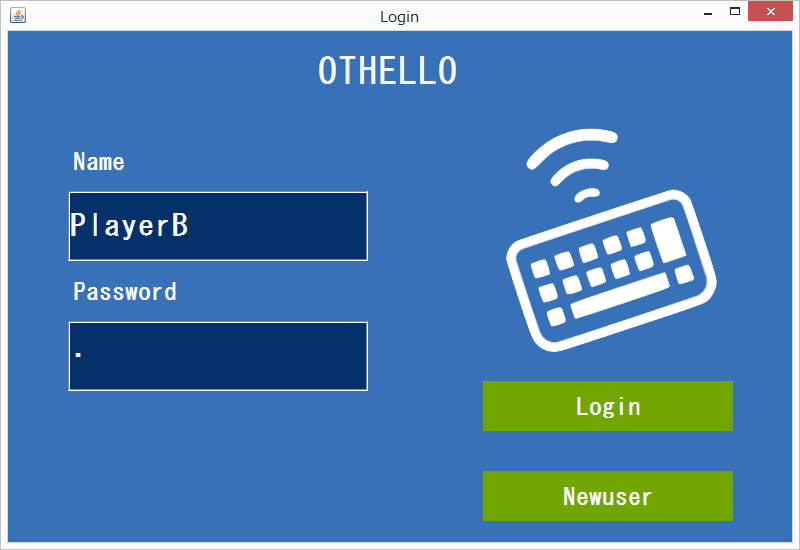
試験結果を以下に示す．

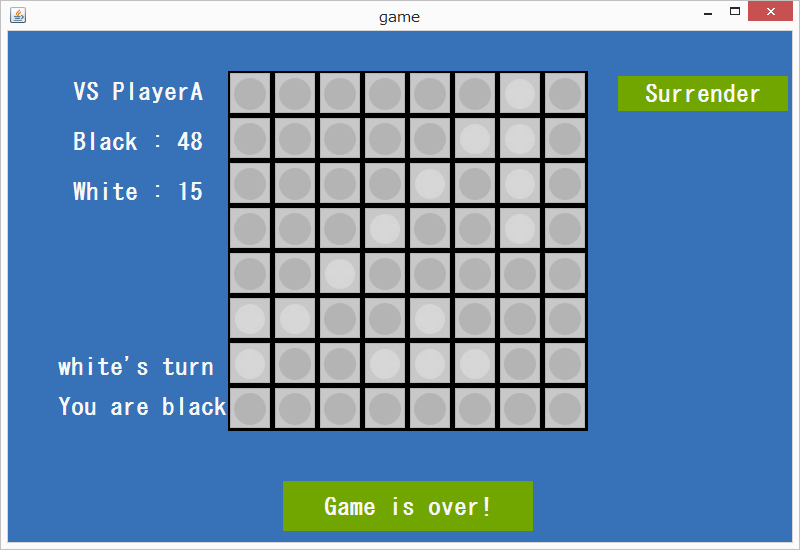
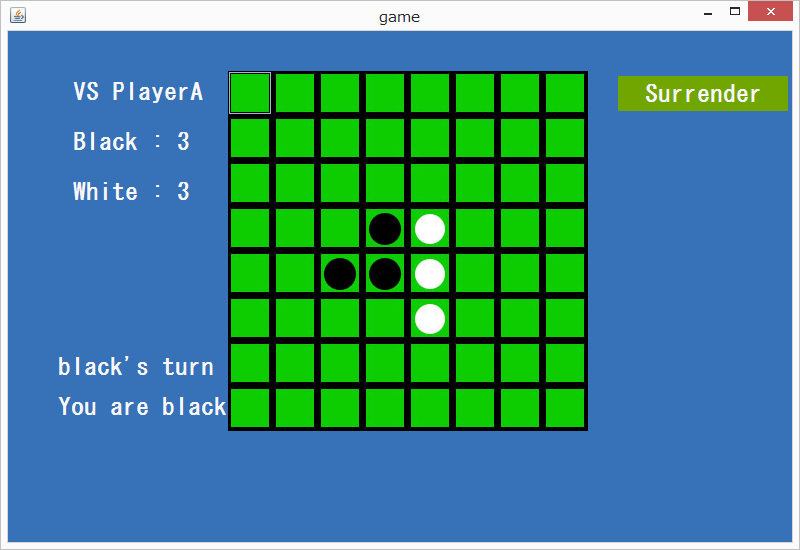
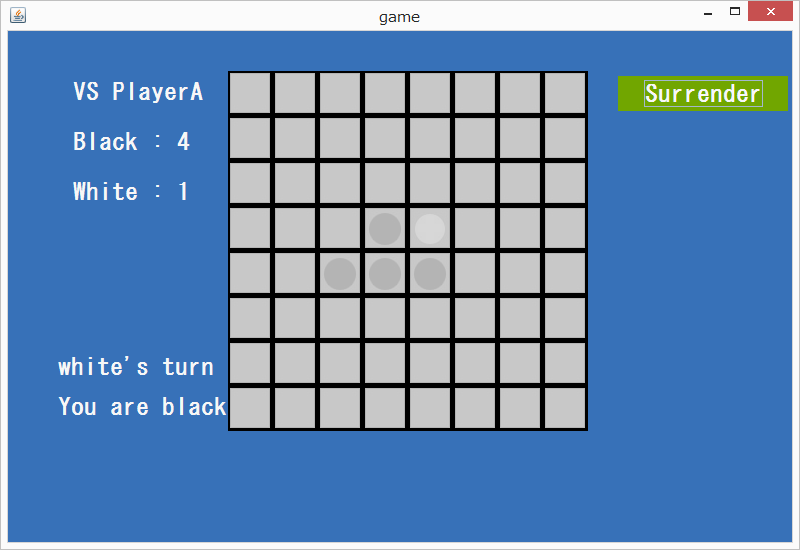
クライアント側：

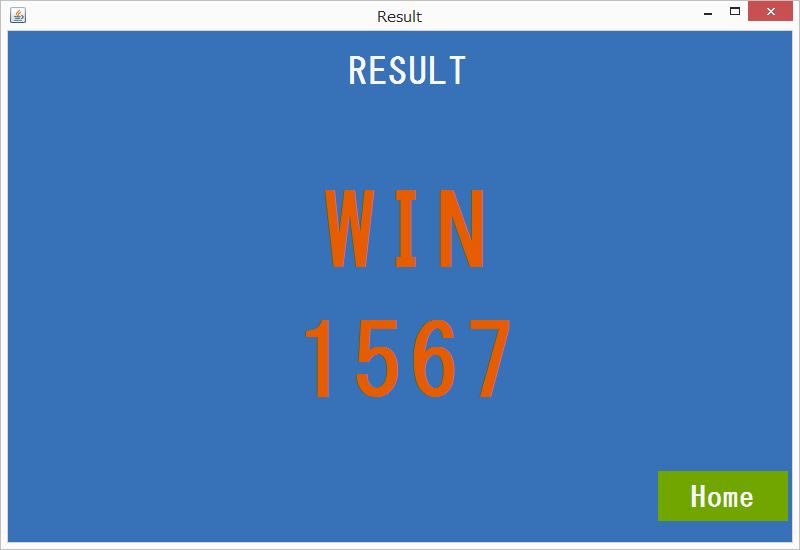












サーバ側：

サーバが起動しました．

start Thread1

1がせつぞくしました

login permit

start Thread2

2がせつぞくしました

login permit

name : PlayerB

rate : 1554.5932907572255

name : PlayerB

rate : 1554.5932907572255

相手側に申し込み送信完了

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

スレッドナンバー2に送信しました

スレッドナンバー1に送信しました

プレイヤ 1との接続が切れました．

0

1

ユーザ名PlayerBはオンライン状態です

2

プレイヤ 2との接続が切れました．0

1

2

オンライン状態のPlayerはいません

decrement