

# 試験報告書

プロジェクト 1

グループ G

提出日：2019 年 5 月 31 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          EchoServer.java (テスト用ドライバ)

クラス図

プレイヤー(Player)	オセロ(Othello)	クライアント(Client)	サーバ(Server)
<div>名前:String 手番の色:String パスワード:String 勝ち数:int 負け数:int 引き分け数:int 投了数:int 切断数:int 自分のレート:int</div> <div>+プレイヤー名、パスワードの保存(得点:名前:String,パスワード:String):Void +勝ち数の保存(勝ち数:int):Void +負け数の保存(負け数:int):Void +投了数の保存(投了数:int):Void +切断数の保存(切断数:int):Void +レート値の保存(レート値:int):Void +自分の色の保存(手番の色:String):Void +手番の色の入れ替え(Void):Void +プレイヤーの情報(得点:String):String +名前の取得(Void):String +色の取得(Void):String +パスワードの取得(Void):Void +レートの取得(Void):int +勝ち数の取得(Void):int +負け数の取得(Void):int +引き分け数の取得(Void):int +投了数の取得(Void):int +引き分け数の加算(Void):Void +投了数の加算(Void):Void +切断数の加算(Void):Void +レートの加算(勝敗:Boolean,相手のレート:int):Void</div>	<div>操作の情報:int 探索用配列:int[] 盤面の座標:int 盤面:int[] 表示用盤面:int[] ターン:int 勝敗:Boolean ゲーム終了フラグ:Boolean 引き分けフラグ:Boolean</div> <div>+コンストラクタ(Void):Void +ターン入れ替え(Void):Void +盤面表示(Void):Void +盤面リセット(Void):Void +盤面操作(石を置く場所:String):Void +勝敗判定(Void):Boolean +盤面更新(Void):Boolean +おける場所判定(Void):Boolean +石を表返す操作(Void):Void +勝敗結果(Void):Void +コマンド操作(置いた場所:String):int +盤面取得(Void):int[] +表示用盤面取得(Void):int[] +ターン取得(Void):String +勝ち数取得(Void):String +引き分けフラグ取得(Void):Boolean +ゲーム終了フラグ取得(Void):Boolean</div>	<div>ボタン:JButton リスト:JList ラベル:JLabel アイコン:ImageIcon 送信用オブジェクト:PrintWriter 受信データ用文字ストリーム:InputStreamReader オセロ:Othello プレイヤー:Player 盤面:int[]</div> <div>+コンストラクタ(ゲーム:Othello,プレイヤー:Player):Void +ラベル表示(Void):Void +サーバ接続(IPアドレス:String,ポート番号:int):Void +メッセージ送信(メッセージ:String):Void +ゲーム画面(Void):Void +ランキング画面(Void):Void +main(String args[]):Void Receiver +コンストラクタ(Socket:Socket):Void +受信スレッド(Void):Void +ゲームスレッド(Void):Void +ディスプレイ初期化(Void):Void +メッセージ受信(タイプ:String,メッセージ:String):Void +ランキング表示(ランキングリスト:String[]):String[] +ランキングソート(String[]):String[] +画面表示(Void):Void +画面更新(Void):Void +ActionPerformed(ActionEvent):Void +ログイン(Void):Void +新規登録(Void):Void +ホーム画面(Void):Void +ログイン画面(Void):Void +新規登録画面(Void):Void</div>	<div>サーバ:ServerSocket ポート番号:int 書き込み:FileWriter 書き込み:PrintWriter ユーザリスト:ArrayList&lt;String&gt; ゲームルーム:ArrayList&lt;Receiver&gt; ゲームルームリスト:ArrayList&lt;ArrayList&lt;Receiver&gt;&gt; ユーザ情報リスト:ArrayList&lt;ArrayList&lt;String&gt;&gt;</div> <div>+コンストラクタ(ポート番号:int):Void +ユーザデータ取得(Void):Void +ユーザ数加算(ユーザReceiver):Void +ユーザ数減算(ユーザReceiver):Void +ルーム入室(ユーザReceiver):Void +ルーム退出(ユーザReceiver):Void +切断処理(ルーム:ArrayList&lt;Receiver&gt;&gt;ユーザReceiver):Void +ユーザデータ保存(名前:String,値:Value):Void +ユーザデータ更新(Void):Void +受信スレッド(Void):Void +main(String args[]):Void Receiver +コンストラクタ(Socket:Socket):Void +受信スレッド(ユーザReceiver):Void +ゲームスレッド(Void):Void +メッセージ送信(メッセージ:String):Void +ランキング取得(Void):String +ログイン(Void):Void +ランキング取得(Void):String +ユーザ認証(名前:String,パスワード:String):Void +新規登録(名前:String,パスワード:String):Void +切断時のレート処理(Void):String</div>

## 2. 単体テスト

### (ア) 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。

```
public class PlayerDriver {  
    public static void main(String [] args) throws Exception{  
        Player player = new PlayerSample10;  
  
        System.out.println("setNamePass で「ID:情報太郎、PASSWORD:zyouhou」を入力します");  
        player.setNamePass("情報太郎","zyouhou");  
        System.out.println("getName 出力: " + player.getName());  
        System.out.println("getPass 出力: " + player.getPass());  
  
        System.out.println("setWinCount で 1 を入力");  
        player.setWinCount(1);  
        System.out.println("getWinCount 出力: " + player.getWinCount());  
  
        System.out.println("setLoseCount で 2 を入力");  
        player.setLoseCount(2);  
        System.out.println("getLoseCount 出力: " + player.getLoseCount());  
  
        System.out.println("setResignCount で 3 を入力");  
        player.setResignCount(3);  
        System.out.println("getResignCount 出力: " + player.getResignCount());  
  
        System.out.println("setDrawCount で 4 を入力");  
        player.setDrawCount(4);  
        System.out.println("getDrawCount 出力: " + player.getDrawCount());  
  
        System.out.println("setRate で 3000 を入力");  
        player.setRate(3000);  
        System.out.println("getRate 出力: " + player.getRate());  
  
        System.out.println("setColor で Black を入力");  
        player.setColor("Black");  
        System.out.println("getColor 出力: " + player.getColor());  
    }  
}
```

```
System.out.println("colorChange を実行");
player.colorChange();
System.out.println("getColor 出力: " + player.getColor());

System.out.println("setDisConnectedCount で 5 を入力");
player.setDisConnectedCount(5);

System.out.println("getStatus 出力 : " + player.getStatus());

System.out.println("addDrawCount を実行");
System.out.println("getDrawCount 出力: " + player.getDrawCount());

System.out.println("addResignCount を実行");
System.out.println("getResignCount 出力: " + player.getResignCount());

System.out.println("addRate で true,2000 を入力");
player.addRate(true,2000);
System.out.println("getStatus 出力 : " + player.getStatus());

System.out.println("addRate で false,2000 を入力");
player.addRate(false,2000);
System.out.println("getStatus 出力 : " + player.getStatus());
}
}
```

試験結果：

```

C:\Users\SHOJI\Desktop\OthelloDriver\PlayerDriver>java PlayerDriver
setNamePassで「ID:情報太郎、PASSWORD:zyouhou」を入力します
getName出力: 情報太郎
getPass出力: zyouhou
setWinCountで1を入力
getWinCount出力: 1
setLoseCountで2を入力
getLoseCount出力: 2
setResignCountで3を入力
getResignCount出力: 3
setDrawCountで4を入力
getDrawCount出力: 4
setRateで3000を入力
getRate出力: 3000
setColorでBlackを入力
getColor出力: Black
colorChangeを実行
getColor出力: White
setDisconnectedCountで5を入力
getStatus出力: 情報太郎,zyouhou,1,2,4,3,5,3000
addDrawCountを実行
getDrawCount出力: 4
addResignCountを実行
getResignCount出力: 3
addRateでtrue,2000を入力
getStatus出力: 情報太郎,zyouhou,2,2,4,3,5,3150
addRateでfalse,2000を入力
getStatus出力: 情報太郎,zyouhou,2,3,4,3,5,2893

```

#### (イ) 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。

```

import java.io.*;

public class OthelloDriver {

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

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

        Othello game = new Othello(); //初期化
        System.out.println("テスト 1 : Othello クラスのオブジェクトを初期化した結果 :
");

        printStatus(game);
        printGrids(game);
        while(true){

            System.out.println("石を置く場所(数字または giveup)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。");

            if(game.gamestart(s)){

```

```

        System.out.println("手番を変更します。¥n");
        game.turnShift();
    }
    printStatus(game);
    printGrids(game);
    }
    //状態を表示する
    public static void printStatus(Othello game){
        System.out.println("checkWinner 出力:" + game.getWinColor());
        System.out.println("isGameover 出力:" + game.getJudge());
        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();

        System.out.println("getGrids テスト出力 : (BLACK=1,WHITE=2,EMPTY=0)");
        for(int i = 0 ; i < 8 ; i++){
            for(int j = 0 ; j < 8 ; j++){
                System.out.print(grids[i][j] + " ");
            }
            System.out.print("¥n");
        }
    }
}

```

試験結果：

```
C:\Users\SHOJI\Desktop\othelloDriver\OthelloDriver>java OthelloDriver
```

```
board board board board board board board board
board board board board board board board board
board board board board board board board board
board board board white black board board board
board board board black white board board board
board board board board board board board board
board board board board board board board board
board board board board board board board board
テスト 1 : Othelloクラスのオブジェクトを初期化した結果 :
checkWinner出力:not_finished
isGameOver出力:false
getTurn出力:Black
石を置く場所(数字またはgiveup)をキーボードで入力してください
26
26 が入力されました。手番は Black です。
黒のターン

board board board board board board board board
board board board board board board board board
board board board board board board board board
board board black black black board board board
board board board black white board board board
board board board board board board board board
board board board board board board board board
board board board board board board board board
手番を変更します。

checkWinner出力:not_finished
isGameOver出力:false
getTurn出力:White
石を置く場所(数字またはgiveup)をキーボードで入力してください
```

#### (ウ) 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、スタブは使用していない。

```
import java.io.*;

public class ClientDriver{
    public static void main(String [] args) throws Exception{
        BufferedReader r = new BufferedReader(new InputStreamReader(System.in), 1);
        Player player = new Player(); //プレイヤーオブジェクトの用意
        player.setNamePass("test user","aa"); //名前を受付
        Othello game = new Othello(); //オセロオブジェクトを用意
        Client oclient = new Client(game, player); //引数としてオセロオブジェクトとプレイヤーオブジェクトを渡す
        oclient.setVisible(true);
        System.out.println("テスト用サーバに接続します");
        oclient.connectServer("localhost", 10000);
        System.out.println("受信用テストメッセージを入力してください");
        while(true){
```

```

        String s = r.readLine();
        String t = r.readLine();
        oclient.receiveMessage(s,t);
        System.out.println("テストメッセージ「" + s + t + "」を受信しました");
        System.out.println("テスト操作を行った後、受信用テストメッセージを入力して
        ください");
    }
}
}

```

試験結果：

```
C:\Users\SHOJI\Desktop\othelloDriver\ClientDriver>java ClientDriver
```

```

□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □
□ □ □ ○ ● □ □ □ □
□ □ □ ● ○ □ □ □ □
□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □

```

```

テスト用サーバに接続します
受信用テストメッセージを入力してください
gamematch

```

```

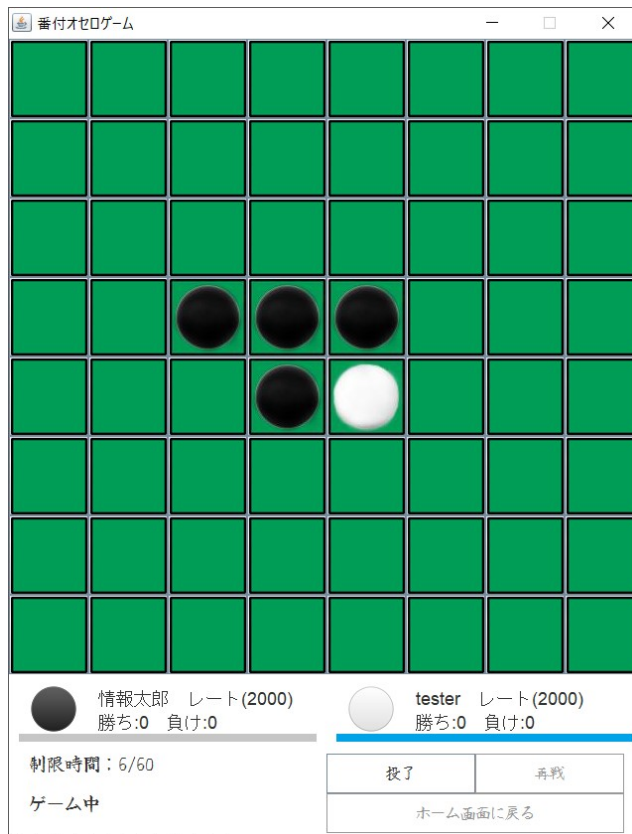
テストメッセージ「gamematch」を受信しました
テスト操作を行った後、受信用テストメッセージを入力してください
command
26
黒のターン

```

```

□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □
□ □ ● ● ● □ □ □ □
□ □ □ ● ○ □ □ □ □
□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □

```



(エ) Server クラス

```
import java.net.*;
import java.io.*;

public class EchoServer {
    static int port = 10000;

    public static void main(String[] args) {
        try {
            ServerSocket server = new ServerSocket(port);
            Socket sock = null;
            System.out.println("サーバが起動しました");
            while(true) {
                try {
                    sock = server.accept(); // クライアントからの接続を待つ

                    System.out.println("クライアントと接続しました");
                    BufferedReader in = new BufferedReader(
```



```
        new InputStreamReader(sock.getInputStream()));
        PrintWriter out = new PrintWriter(sock.getOutputStream());
        String s;
        while((s = in.readLine()) != null) { // 一行受信
            out.print(s + "¥r¥n"); // 一行送信
            out.flush();
            System.out.println(s);
        }
        sock.close(); // クライアントからの接続を切断
        System.out.println("切断しました");
    } catch (IOException e) {
        System.err.println(e);
    }
}

} catch (IOException e) {
    System.err.println(e);
}

}

}
```

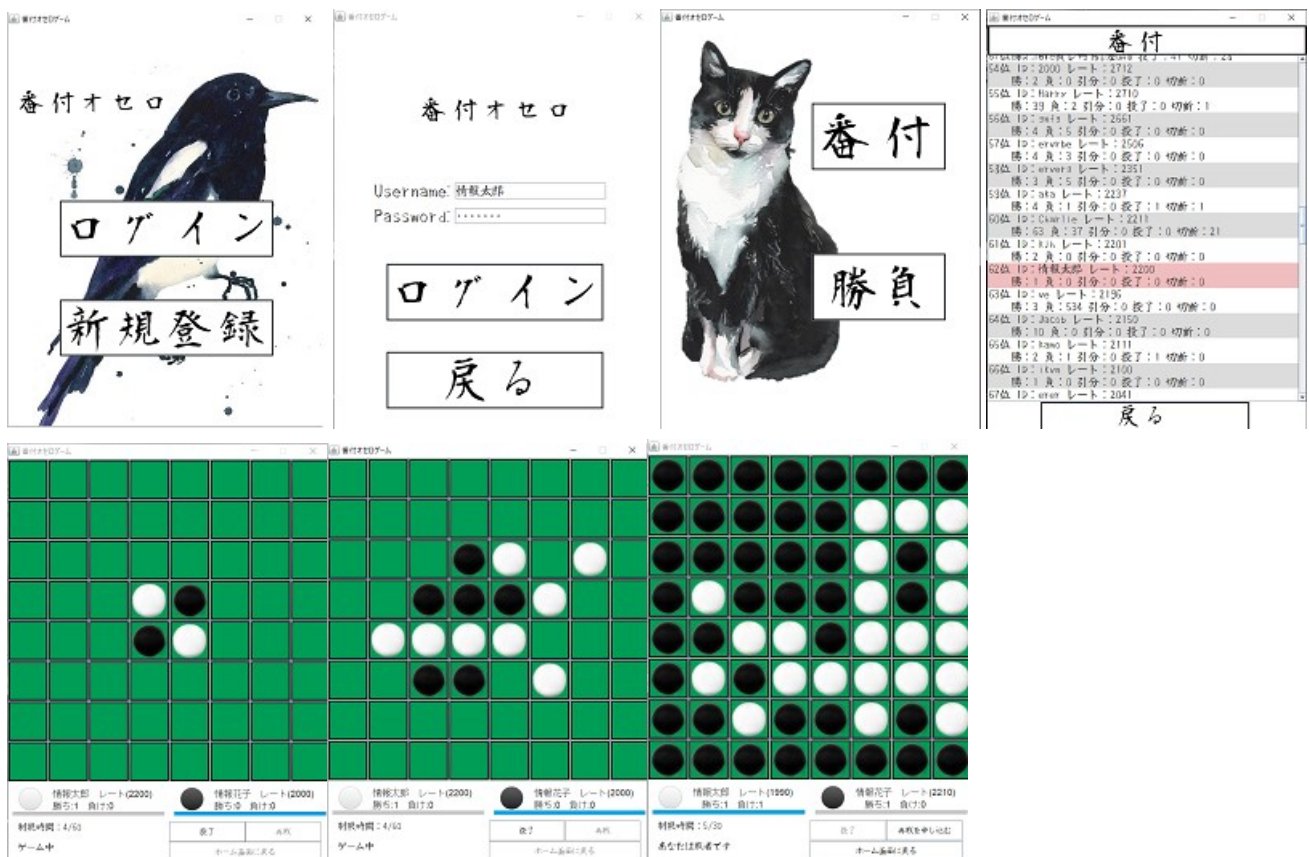
### 3. 結合テスト

以下の手順に従い、結合テストを行った。

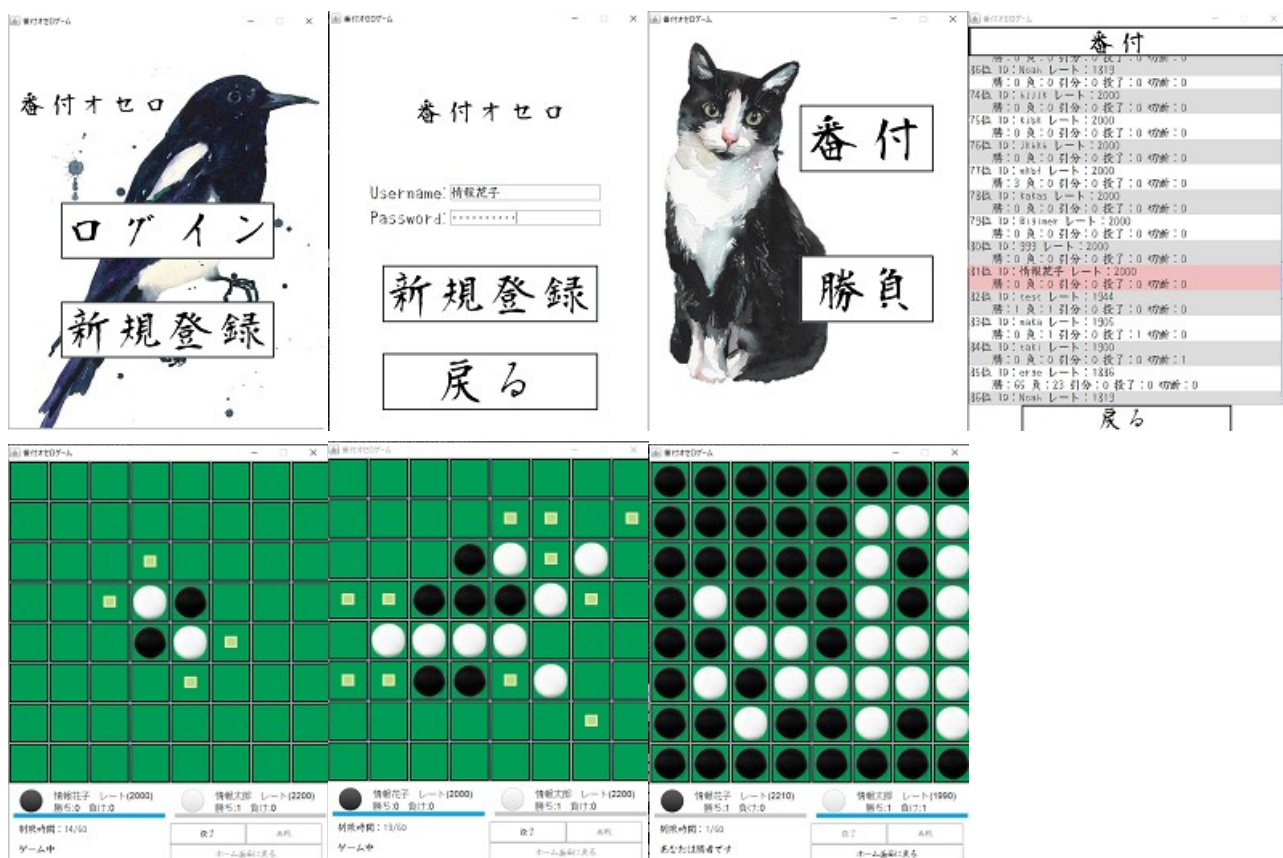
1. Server プログラムを起動
2. Client1 はログインを選択
3. Client2 は新規登録を選択
4. Client プログラム1を起動し、プレイヤ名「情報太郎」パスワード「zyouhou」を入力
5. Client プログラム2を起動し、プレイヤ名「情報花子」パスワード「zyouhou2」を入力
6. 番付を選択しランキングを表示し、戻る
7. ゲームを開始する
8. ゲームを進行する
9. ゲーム終了後、Client プログラム1を終了し、サーバから切断
10. ゲーム終了後、Client プログラム2を終了し、サーバから切断

試験結果を以下に示す。

クライアント 1 側：



クライアント 2 側：



サーバ側：

```
C:\Users\SHOUJI\Desktop\othello1>java Server
新規登録しました
接続人数 : 2
部屋[0]の人数 : 1
現在の部屋の数 : 1
情報花子 : Access Error
情報太郎 : Access Error
```