Lesson3

Work3-1

図3-3に従って人物ごとに何をしたかをまとめる

・村田さんの行ったこと

1.合図を聞いて１回戦めはパーを出した
2.合図を聞いて２回戦めはグーを出した
3.合図を聞いて３回戦めはグーを出した
4.何回勝ったかを答えた

・山田さんの行ったこと

1.合図を聞いて１回戦めはチョキを出した
2.合図を聞いて２回戦めはチョキを出した
3.合図を聞いて３回戦めはパーを出した 4.
何回勝ったかを答えた

・斎藤さんの行ったこと

1.じゃんけんを何回行うかを言った
2. １回戦めの合図を出した
3. １回戦めの勝敗を判定した
4.2~3の動作を２回戦目として行った
5.2~3の動作を３回戦目として行った
6.2人に何回勝ったかを聞いた
7.最終的な勝者を判定した

Work 3-2（P65）

SimpleJanken.javaを参考にしてPlayerクラスを完成させる

```
package lesson3_janken;

import java.util.Random;

/**
 * じゃんけんでプレイヤーのする動作
 * @author yuna
 *
 */
public class Player {
	//じゃんけんの手を表す定数
	public static final int STONE = 0; //グー
	public static final int SCISSORS = 1; //チョキ
	public static final int PAPER = 2; //パー

	//-----------------
	//プレイヤークラスの属性
	//-----------------

	//フィールド
	//プレイヤーの名前
	private String name_;
```

```
	//プレイヤーの勝った回数
	private int winCount_ = 0;

	//コンストラクタ
	//引数なしのコンストラクタも用意しておく
	Player(){}
	/**
	 * プレイヤーの名前を引数にとるコンストラクタ
	 * @author yuna
	 * @param name
	 */
	Player(String name){
		//名前を表すフィールドを初期化
		this.name_ = name;
	}
	/**
	 * プレイヤーの名前と勝利回数を引数にとるコンストラクタ
	 * @param name_
	 * @param winCount_
	 */
	Player(String name_, int winCount_){
		//名前を表すフィールドを初期化する
		this(name_);
		//勝利回数を表すフィールドを初期化
		this.winCount_ = winCount_;
	}

	//-----------------
	//プレイヤークラスの操作
	//-----------------

	//playerクラスの動作
	//じゃんけんの手を出す
	/**
	 * @author yuna
	 * @param なし
	 * @return じゃんけんの手を値にして返す<br>
	 * グー：0 、チョキー:1、パー:2
	 */
	public int showHand() {
		//ランダムクラスのインスタンスを生成する
		Random RandomNumber = new Random();
		//0から 3 の乱数を生成してプレイヤーの手を決める
		int playerHand = RandomNumber.nextInt(3);
		//プレイヤーの出す手を返却
		return playerHand;
	}

	/**
	 * 審判から勝敗を聞いて勝っていれば勝利回数を数える
	 * @author yuna
	 * @param result true:勝ち、false:負け
	 * @return 無し
	 */
	public void notifyResult(boolean result) {
		//勝った場合
		if(result = true) {
			//勝利回数カウントする
			winCount_ +=1;
```

```
                }
        }

        /**
         * 勝利回数を答える
         * @author yuna
         * @return 勝った回数を返却する
         */
        public int getWinCount() {
                //勝利回数を返却
                return    winCount_;
        }
}
```

Work3-3（P79）

Judgeクラスを完成させる

```
package lesson3_janken;

/**
 * じゃんけんの判定を行うクラス
 * @author yuna
 * @createDate 2023/6/16
 */
public class Judge {

        //じゃんけんの手を表す定数
        public static final int HAND_STONE = 0; //グー
        public static final int HAND_SCISSORS = 1; //チョキ
        public static final int HAND_PAPER = 2; //パー

        /**
         * じゃんけんを開始する
         * @param player1 判定対象プレイヤー 1
         * @param player2 判定対象プレイヤー 2
         * @return なし
         * @author yuna
         */
        public void startJanken(player player1, player player2) {
                //じゃんけんの開始を合図する
                System.out.println("【じゃんけん開始】\n");
                //じゃんけんをする回数を表す変数
                int numberOfGames = 3;
                //じゃんけんする回数ループする
                for(int count=0; count<numberOfGames; count++) {
                        //何回戦目かを表示する
                        System.out.println("\n【"+(count+1)+"回戦目】");
                        //2人のプレイヤーの内どちらが勝者かを判定する
                        player winner = judgeJanken(player1,player2);
                        //引き分けでない場合
                        if(winner != null) {
                                //誰が勝ったかを表示する
                                System.out.println("\n"+winner.getName() + "が勝ちました！\n");
                                //勝ったプレイヤーに結果を伝える
                                winner.notifyResult(true);
                        }
                        //引き分けの場合
```

```
                    else {
                            //引き分けだったことを表示する
                            System.out.println("\n引き分けです！\n");
                    }
            }
            //じゃんけんの終了を合図する
            System.out.println("【じゃんけん終了】");
            //最終的な勝者の判定をする
            player finalWinner = judgeFinalWinner(player1,player2);
            //何対何で勝敗がどうなったかを表示する
            System.out.print(player1.getWinCount() +"対"+ player2.getWinCount() + "で
            "); //引き分けでない場合
            if(finalWinner != null) {
                    //誰が勝ったかを表示する
                System.out.println(finalWinner.getName()+"の勝ちです！\n");
            }
            //引き分けの場合
            else {
                    //引き分けたことを表示する
                    System.out.println("引き分けです\n");
            }
    }

    private player judgeJanken(player player1, player player2) {
            //勝者を表す変数、NULLは引き分けを表す
            player winner = null;
            //プレイヤー 1 の出す手を取り出す
            int player1Hand = player1.showHand();
            //プレイヤー 2 の出す手を取り出す
            int player2Hand = player2.showHand();
            //プレイヤー 1 の出す手を表示する
            printHand(player1Hand);
            //vsを表示して勝負を表す
            System.out.print("vs.");
            //プレイヤー 1 の出す手を表示する
            printHand(player2Hand);
            //改行
            System.out.println();
            //プレイヤー 1 が勝つ場合
            if(
                    //プレイヤー 1 がグーでプレイヤー 2 がチョキの場合
                      (player1Hand==HAND_STONE &&
                      player2Hand==HAND_SCISSORS) //または、プレイヤー 1 がチョキ
                      でプレイヤー2がパーの場合
                      || (player1Hand==HAND_SCISSORS &&
                      player2Hand==HAND_PAPER) //または、プレイヤー 1 がパーでプレ
                      イヤー 2 がグーの場合
                      || (player1Hand==HAND_PAPER &&
            player2Hand==HAND_STONE) ) {
                      //プレイヤー 1 を勝者にする
                      winner = player1;
            }
            //プレイヤー2が勝つ場合
            else if(
                    //プレイヤー 2 がグーでプレイヤー 1 がチョキの場合
                      (player2Hand==HAND_STONE &&
                      player1Hand==HAND_SCISSORS) //または、プレイヤー 2 がチョキ
                      でプレイヤー 1 がパーの場合
                      || (player2Hand==HAND_SCISSORS &&
```

```
                player1Hand==HAND_PAPER) //または、プレイヤー２がパーでプレ
                イヤー１がグーの場合
                || (player2Hand==HAND_PAPER &&
        player1Hand==HAND_STONE) ) {
                //プレイヤー２を勝者にする
                winner = player2;
        }
        //勝った人を返す、nullは引き分け
        return winner;
}

/**
 * 最終的な勝者を判定する
 * @param player1
 * @param player2
 * @return 最終的な勝者が誰か返す
 */
private player judgeFinalWinner(player player1, player player2){
        //最終的な勝者を表す変数、nullは引き分けを表す
        player finalWinner = null;
        //プレイヤー１の勝利数を取り出す
        int player1Count = player1.getWinCount();
        //プレイヤー２の勝利数を取り出す
        int player2Count = player2.getWinCount();

        //プレイヤー１の方が多く勝った場合
        if(player1Count > player2Count) {
                //最終的な勝者をプレイヤー１とする
                finalWinner = player1;
        }
        //プレイヤー２の方が多く勝った場合
        else if(player1Count < player2Count) {
                //最終的な勝者をプレイヤー２とする
                finalWinner = player2;
        }
        //最終的な勝者が誰か返却する
        return finalWinner;
}


/**
 * プレイヤーが誰かを引数に取ってプレイヤーの出す手を表示す
 る * @param player
 */
static void printHand(player player) {
        //引数にとったプレイヤーの出す手を取り出す
        int PlayerHand = player.showHand();
        //出す手がグー(0)の時
        if(PlayerHand==HAND_STONE) {
                //グーを表示する
                System.out.print(player.getName() + ":グー");
        }
        //出す手がチョキ(1)の時
        if(PlayerHand==HAND_SCISSORS) {
                //チョキを表示する
                System.out.print(player.getName() + ":チョキ");
        }
        //出す手がチョキ(2)の時
        if(PlayerHand==HAND_PAPER) {
```

```
                        //チョキを表示する
                        System.out.print(player.getName() + ":パー");
                }
        }

        /**
         * 引数に出す手を表す値をとってプレイヤーの出す手を表示す
         る * @param hand
         */
        static void printHand(int hand) {
                //出す手がグー(0)の時
                if(hand==HAND_STONE) {
                        //グーを表示する
                        System.out.print("グー");
                }
                //出す手がチョキ(1)の時
                if(hand==HAND_SCISSORS) {
                        //チョキを表示する
                        System.out.print( "チョキ");
                }
                //出す手がチョキ(2)の時
                if(hand==HAND_PAPER) {
                        //チョキを表示する
                        System.out.print("パー");
                }
        }
}
```