# オブジェクト指向プログラミング
# 期末試験

# 問題冊子

**「はじめ」の合図があるまで**
**問題冊子を開いてはいけません。**

この問題冊子は試験終了後、回収します。

・この問題冊子は、（表紙を除いて）12ページあります。

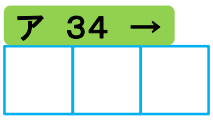
・この問題冊子は、7章に分かれており、各章に複数の問題があります。

・問題冊子に記載されているプログラムは、
全てプログラミング言語「Java」のプログラムです。

・**各章のプログラムは、章ごとに完結しており、**
**他の章のプログラムに依存することなく、章ごとに独立して動作します。**

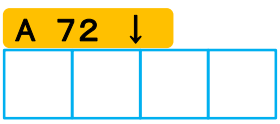
## 解答方法の例

例1）




カタカナの解答用紙に記入してください。
記入場所は、34番です。
記入方向は、横方向です。
記入する文字数は、3文字です。

例2）




アルファベットの解答用紙に記入してください。
記入場所は、72番です。
記入方向は、縦方向です。
記入する文字数は、4文字です。

# 1 基本事項(1／2)

●UMLの構成要素

(1)UML2.0には [A 46 →] [ ] [ ] 種類の図がある。

(2) [ア 22 →] [ ] [ ] [ ] [ ] [ ] [ ] [ ] 図や [ア 16 →] [ ] [ ] [ ] [ ] [ ] 図は [ア 23 ↓] [ ] [ ] [ ] [ ] [ ] [ ] ダイアグラムである。

(3) [ア 6 ↓] [ ] [ ] [ ] [ ] [ ] [ ] 図や [ア 7 ↓] [ ] [ ] [ ] 図は [ア 1 ↓] [ ] [ ] [ ] [ ] [ ] [ ] [ ] ダイアグラムである。

●継承と委譲

(1) [ア 24 ↓] [ ] [ ] [ ] [ ] [ ] は、[ア 8 →] [ ] [ ] [ ] [ ] は [ア 4 ↓] [ ] [ ] [ ] だが、[ア 2 →] [ ] [ ] [ ] [ ] [ ] [ ] [ ] は [ア 27 ↓] [ ] [ ] [ ] 。

(2) [ア 10 ↓] [ ] [ ] [ ] [ ] は、[ア 8 →] [ ] [ ] [ ] [ ] は [ア 13 ↓] [ ] [ ] [ ] [ ] だが、[ア 2 →] [ ] [ ] [ ] [ ] [ ] [ ] [ ] は [ア 18 →] [ ] [ ] [ ] 。

●インタフェースの特徴

(1) インタフェースを利用して、

① 異なる [ア 25 ↓] [ ] [ ] [ ] を同じものとして扱うことができる。

② ソフトウェアの [ア 25 ↓] [ ] [ ] [ ] を [ア 4 ↓] [ ] [ ] [ ] に [ア 12 →] [ ] [ ] [ ] [ ] できる。

③ 必要な [ア 25 ↓] [ ] [ ] [ ] だけを公開することができる。

(2) [ア 33 →] [ ] [ ] [ ] [ ] [ ] [ ] [ ] を [ア 12 →] [ ] [ ] [ ] [ ] するのは [ア 14 ↓] [ ] [ ] [ ] [ ] だが、[ア 7 ↓] [ ] [ ] [ ] を [ア 12 →] [ ] [ ] [ ] [ ] するのは [ア 4 ↓] [ ] [ ] [ ] である。

# 1 基本事項(2／2)

## ●デザインパターン

(1) E.Gamma, Helm.R, R.Johnson, J.Vissides,の４人が著書

「Design Patterns」で A 43 ↓ 個のデザインパターンを提示した。

(2) デザインパターンは ア 32 → に依存しない。

(3) 状態によって動作を切り替えたい場合は

A 4 →

パターンの適用を検討すると良い。

(4) オブジェクトを１つしか作れないようにしたい場合は

A 47 →

パターンの適用を検討すると良い。

## ●関数型プログラミング

大原則: ア 31 → は、ア 13 → を持ってはならない。

原則1: ア 26 ↓ の値を ア 12 → してはいけない。

原則2: 関数を ア 26 ↓ に代入できる。

原則3: 処理の実行タイミングを遅らせることができる。

## ●オブジェクト指向プログラミングと関数型プログラミングの融合

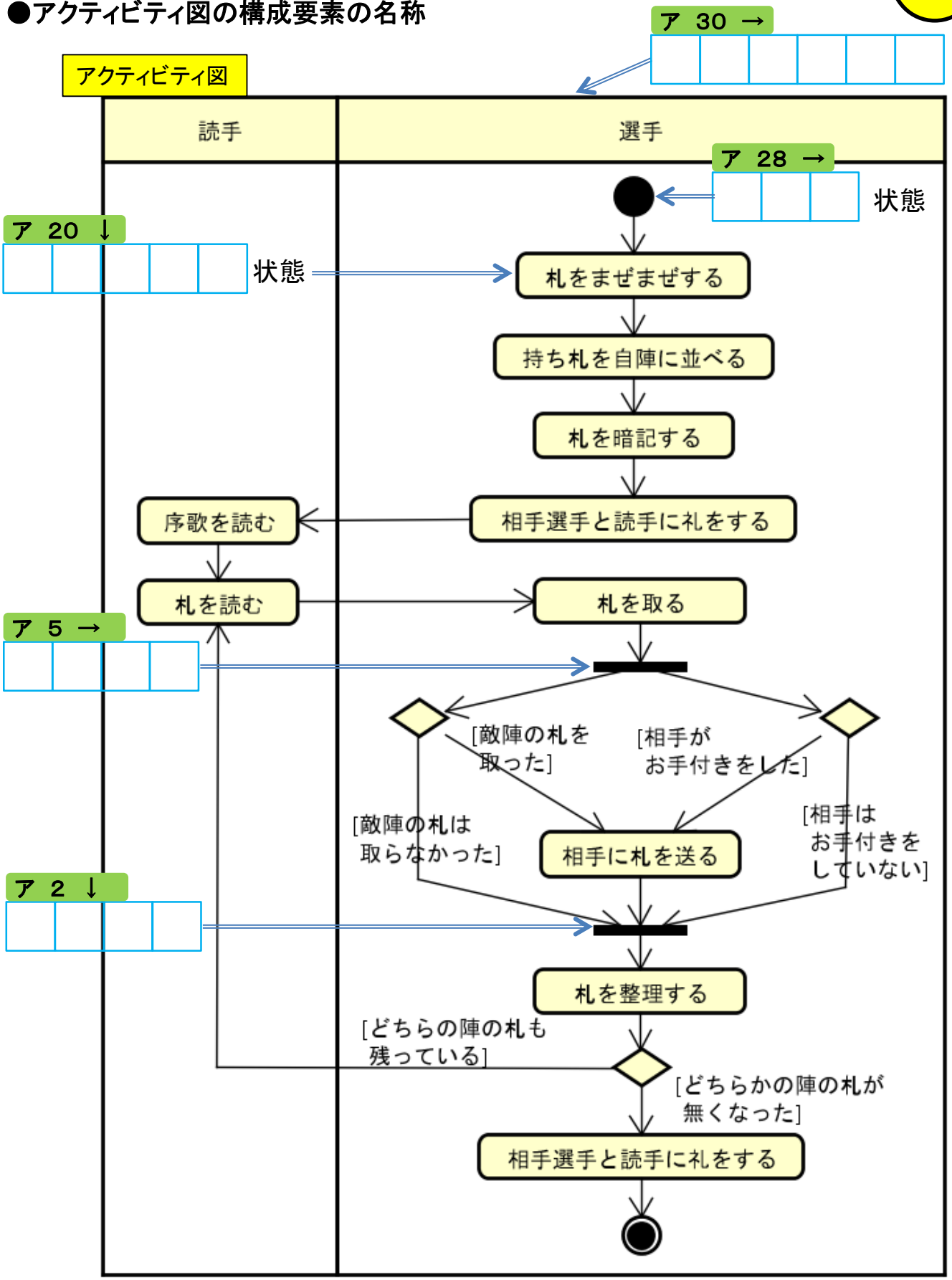
(1) ア 13 ↓ なシステムの構築には、ア 11 → の制御が必要。

(2) ア 17 ↓ したシステムの構築には、ア 13 → の無い処理が必要。

(3) したがって、オブジェクト指向プログラミングの考え方に基づいてシステムをモデル化し、関数型プログラミングの考え方に基づいて具体的な処理を記述すると良い。

# 2 競技かるたの流れ

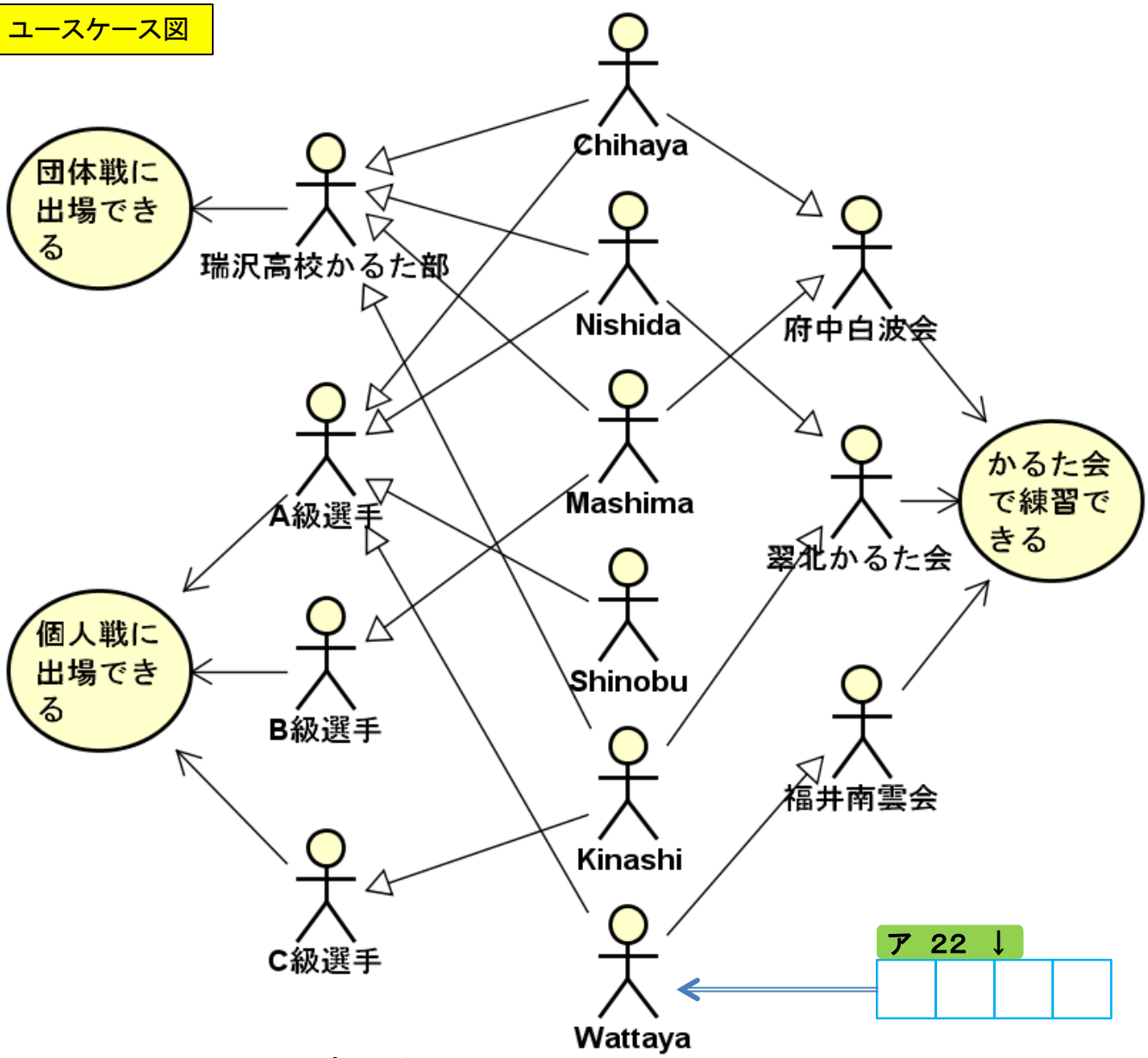
## ●アクティビティ図の構成要素の名称



# 3 全国高等学校かるた選手権大会

## ●ユースケース図の構成要素の名称

ユースケース図




## ●上のユースケース図からわかること

(1) A 1 ↓ [　　] は、かるた会では練習できない。

(2) A 26 → [　　] は、翠北かるた会に所属するA級選手である。

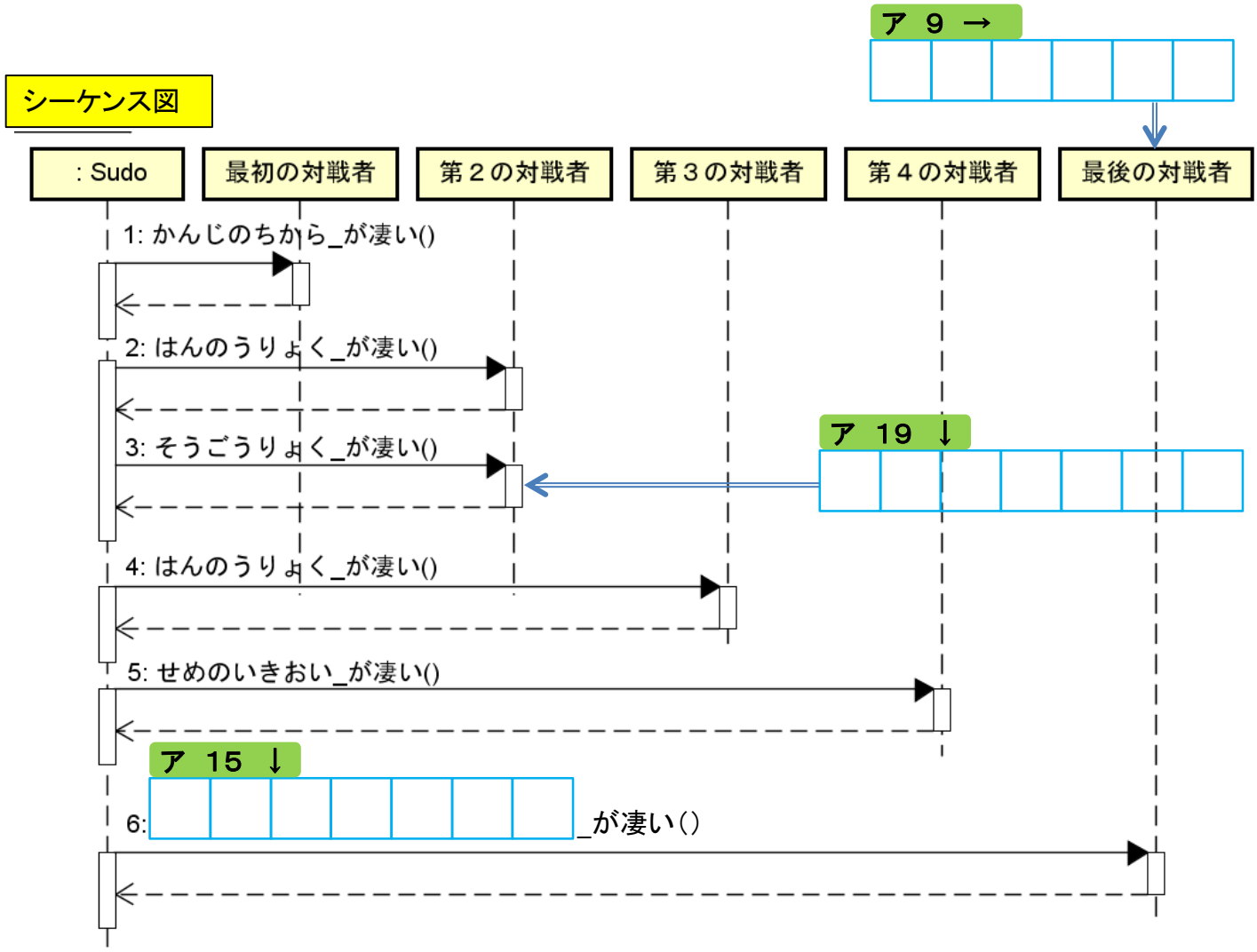
(3) A 21 ↓ [　　] と A 7 → [　　] は、
団体戦に出場できない。

(4) 府中白波会で個人戦に出場できるB級選手は、A 8 ↓ [　　] である。

# 4 練習試合（1／2）

北央学園かるた部の須藤部長が、瑞沢高校かるた部の7人の部員のうちの5人の部員と練習試合をしました。

## ●シーケンス図の構成要素の名称




## ●シーケンス図とクラス図からわかること

(1) 最初の対戦者は A 34 ↓ である。

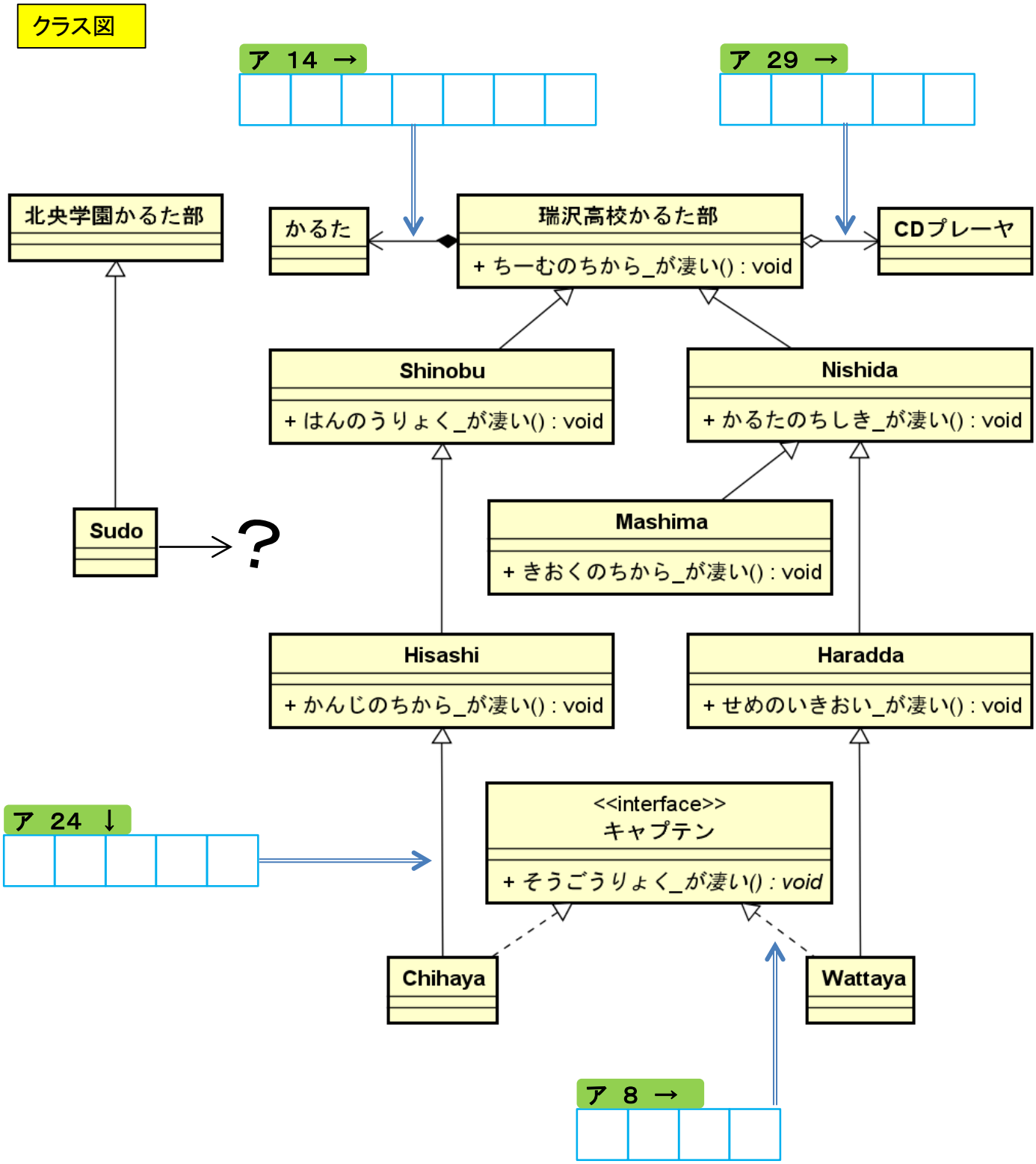
(2) 第2の対戦者は A 35 ↓ である。

(3) 第3の対戦者は A 33 ↓ である。

(4) A 22 ↓ と A 23 ↓ は対戦しなかった。

# 4 練習試合（2／2）

## ●クラス図の構成要素の名称



# 5 指導者たち

**プログラムを完成させてください。**

プログラム

```
public  class  Master {
  public String hello( ){
    return “今やるべきことから目を逸らすのは逃げではないのか。”;
  }
}
```

```
public  class  Wataya  [A 44 →]  [A 37 ↓]  {
  public String  [A 15 →]  ( ){
    return “ルール破って、罰も受けん。ほんなんで相手にしてくれる神様はえんぞ。”;
  }
}
```

```
public  class  Harada  [A 44 →]  [A 36 ↓]  {
  public String  [A 31 →]  ( ){
    return “青春ぜんぶ懸けたって強くなれない？　懸けてから言いなさい。”;
  }
}
```

```
public  class  Karutakai {
    public  static  void  main(String[ ]  args ) {

        Master kitano = new Master();
        System.out.println( “北野「” +  kitano. [A 38 →] () + “」” );

        Wataya  wataya = new Wataya( );
        System.out.println( “綿谷「” + wataya. [A 28 ↓] () + “」” );

        Master  harada = new Harada( );
        System.out.println( “原田「” + harada. [A 41 →] () + “」” );
    }
}
```

実行結果

```
北野「今やるべきことから目を逸らすのは逃げではないのか。」
綿谷「青春ぜんぶ懸けたって強くなれない？　懸けてから言いなさい。」
原田「青春ぜんぶ懸けたって強くなれない？　懸けてから言いなさい。」
```

# 6 一生懸命

**プログラムを完成させるとともに、正しい実行結果を表示させてください。**

プログラム

```
public   interface   Student {
                    A 15 →
   public  String  [     ] ( );
}
                                                    A 30 ↓
public  class  Masashi  [                                 ]  Student{
                    A 15 ↓
   public String  [     ] ( ) {
      return “一生懸命はたのしいぞ。";
   }
}
                                                    A 30 ↓
public  class  Mashima  [                                 ]  Student{
                    A 15 ↓
   public String  [     ] ( ) {
      return “一生懸命ってきついな。";
   }
}

public  class  Isshokenmei {
   public  static  void  main( String[ ]  args ) {
      A 16 ↓
      [       ]  student;
      student = new Mashima( );
                    A 2 →
      student = new  [       ] ( );
      System.out.println(  student.reply( ) );
   }
}
```

実行結果

```
                ア 21 →
一生懸命  [     ] 。
```

# 7 大里小学校かるた大会（1／4）

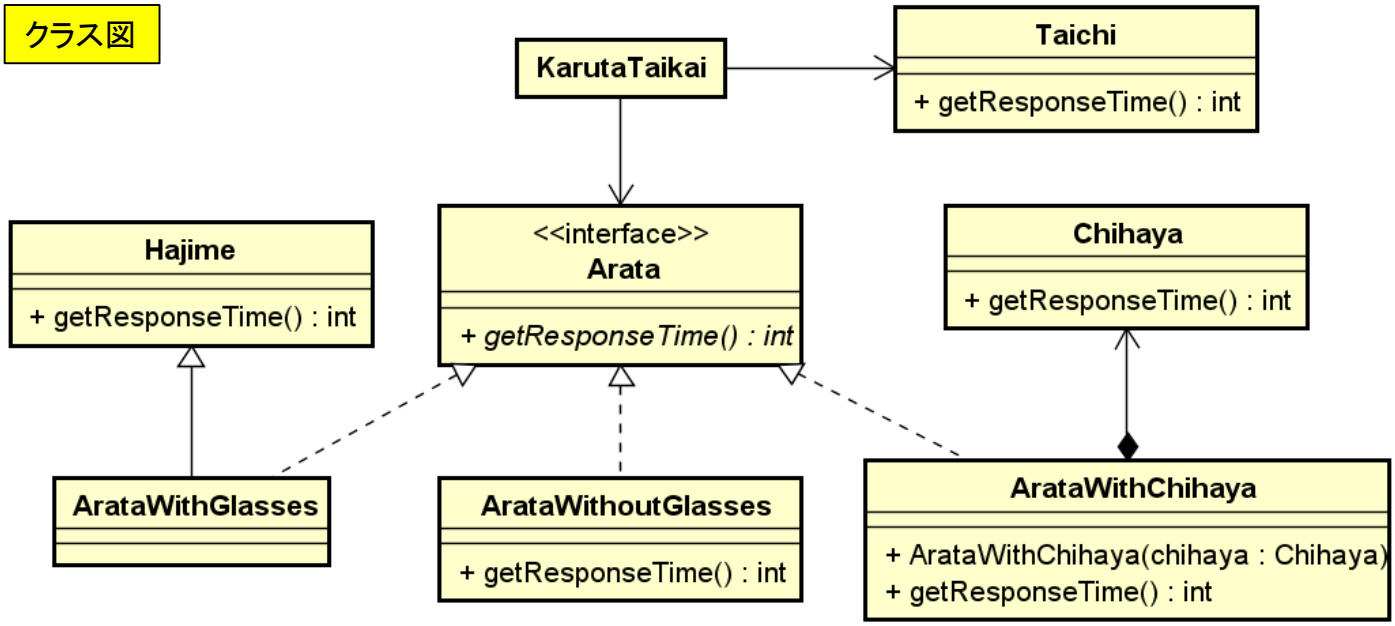
**クラス図をもとにプログラムを完成させてください。**

【参考①】
綿谷新は、競技かるた名人戦を７連覇した綿谷始の孫で、小さい頃から祖父と一緒にかるたをしてきました。
そんな新が、東京の大里小学校に転校することになりました。
大里小学校では毎年、校内かるた大会が開催されており、新もかるた大会に参加することになりました。

【参考②】
新は、学年成績１位の真島太一と対戦することになります。
新は、圧倒的な強さで札を取り続けますが、新の実力に気づいた太一は、新の眼鏡を隠してしまいます。
眼鏡を無くした新は、札に書かれている文字が読めなくなってしまいますが、それでも、暗記していた札の位置を頼りに全ての札を取り続けます。

【参考③】
このままでは勝てないと悟った太一は、全ての札の位置を動かしてしまいます。
札が見えないだけでなく、札の位置まで変わってしまっては、さすがの新もどうしようもありません。お手付きが続き負けそうになってしまいます。
その時、この対戦を見ていた千早が、新のかわりに自分が太一と戦うと言い出します。

クラス図




**●クラス図とプログラムからわかること**

(1) Arataは [A 9 →] [ ][ ][ ][ ][ ] パターンを使って実装されている。

(2) AarataWithGlassesは [ア 24 ↓] [ ][ ][ ][ ][ ] を使って実装されている。

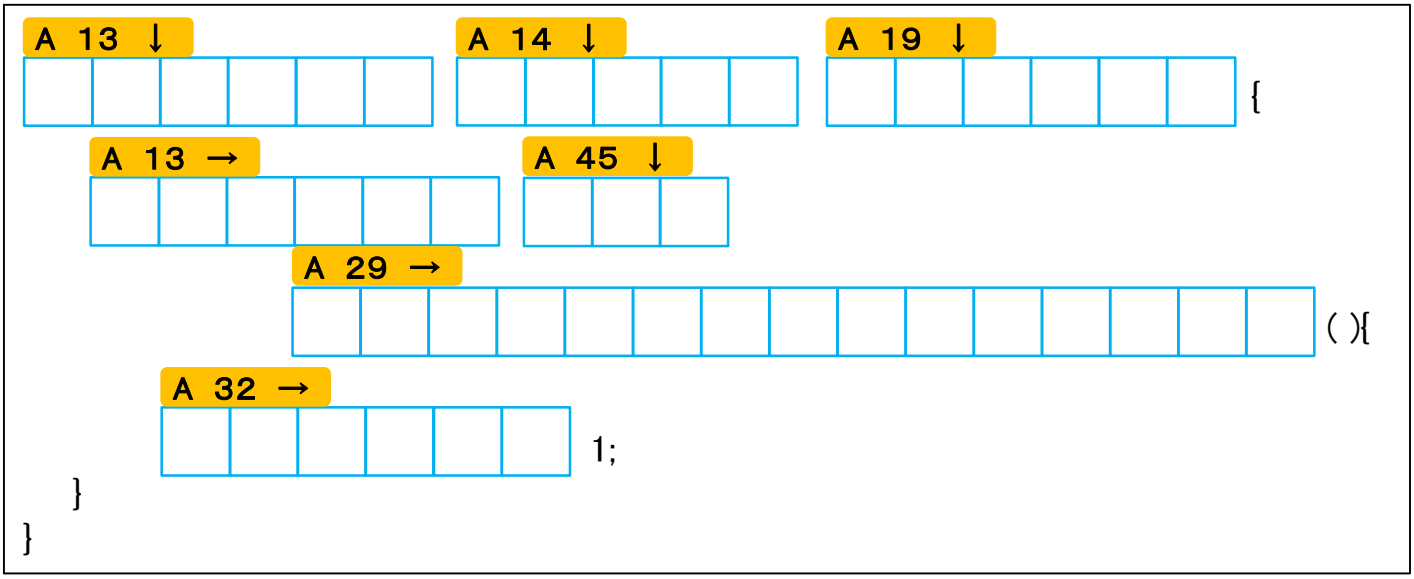
(3) AarataWithChihayaは [ア 3 ↓] [ ][ ][ ][ ] を使って実装されている。

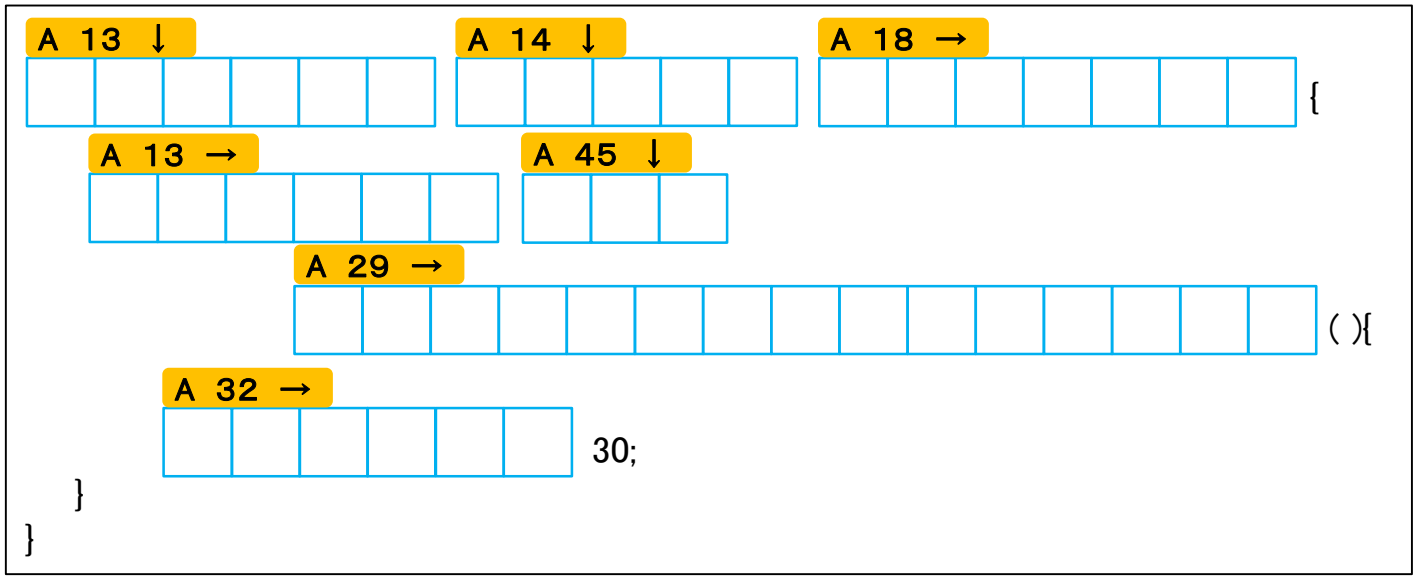
(4) このプログラムを最後まで実行すると、

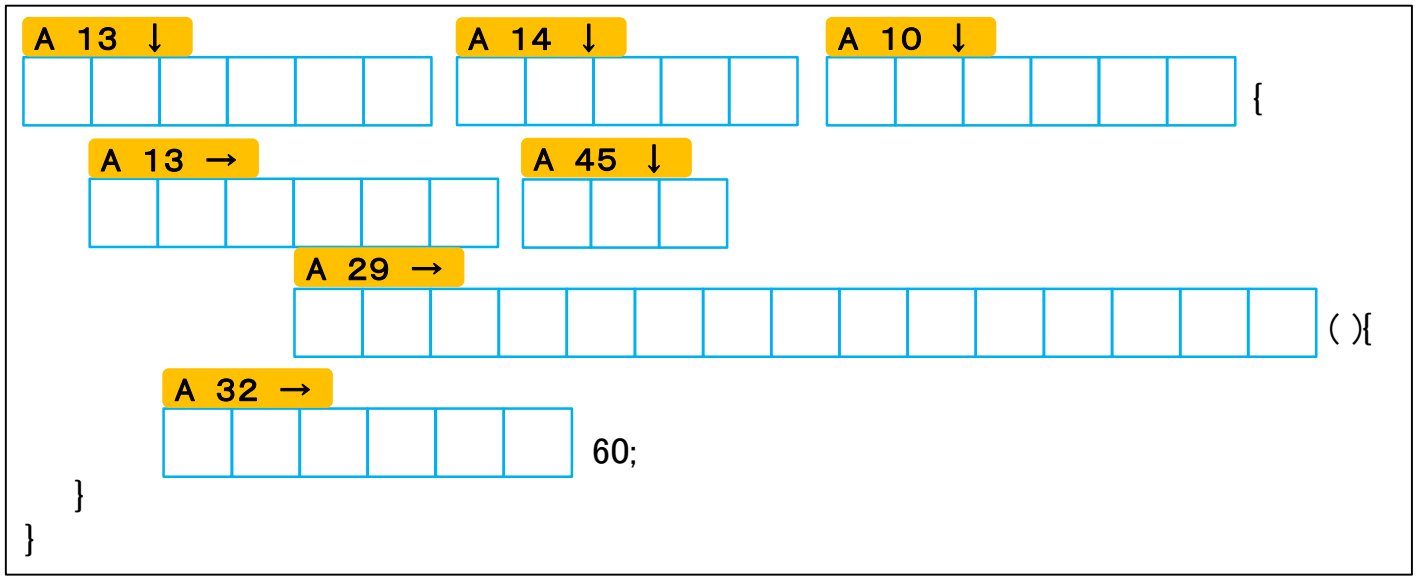
新が取った枚数は [A 20 →] [ ][ ] 枚、太一が取った枚数は [A 17 ↓] [ ][ ] 枚になる。

# 7 大里小学校かるた大会（2／4）

プログラム



# 7 大里小学校かるた大会（3／4）

プログラム

```
public A 25 ↓ [          ] Arata {
    public int getResponseTime();
}
```

```
public class AratawithGlasses
    A 44 → [          ]
    A 12 → [          ]
    A 40 → [          ]
    A 3 ↓ [          ] {

}
```

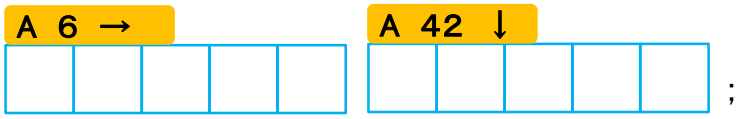
```
// 眼鏡をなくし、札を動かされてしまった新は、
// 札が読まれてから札を取るまで180秒かかる。

public class AratawithoutGlasses
    A 40 → [          ]
    A 3 ↓ [          ] {
    A 27 ↓ [     ]
    public [     ] getResponseTime() {
        return 180;
    }
}
```

```
public class AratawithChihaya
    A 40 → [          ]
    A 3 ↓ [          ] {
    A 39 → [       ]
    A 11 → [          ] ;
    public AratawithChihaya(Chihaya chihaya){
        myChihaya=chihaya;
    }
    A 27 ↓ [     ]
    public [     ] getResponseTime() {
        A 11 → [          ]
        return [          ] .getResponseTime();
    }
}
```

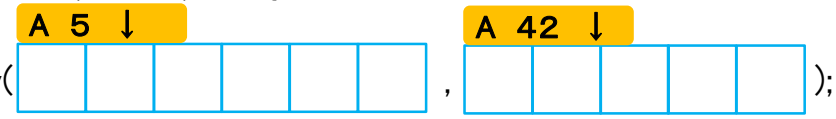
# 7 大里小学校かるた大会（4／4）

プログラム

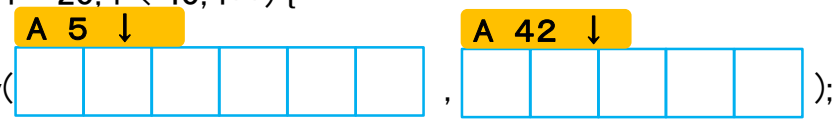
```
public class KarutaTaikai {
    static int arataScore = 0;
    static int taichiScore = 0;
    public static void main(String[] args) {
        Chihaya chihaya = new Chihaya();
        Taichi taichi = new Taichi();
```




```
        arata = new AarataWithGlasses();
        for (int i = 0; i < 20; i++) {
```

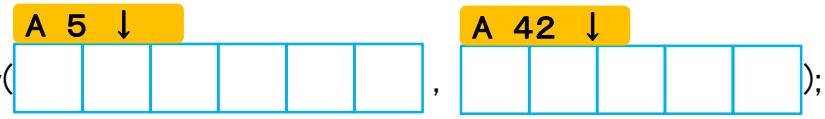



```
        }

        arata = new AarataWithoutGlasses();
        for (int i = 20; i < 40; i++) {
```




```
        }

        arata = new AarataWithChihaya(
```


```
);
        for (int i = 40; i < 50; i++) {
```




```
        }

        System.out.println("太一が取った枚数＝" + taichiScore + "枚");
        System.out.println("新が取った枚数＝" + arataScore + "枚");
    }
    public static void play(Taichi taichi, Arata arata) {
        if (taichi.getResponseTime() <= arata.getResponseTime()) {
            taichiScore++;
        } else {
            arataScore++;
        }
    }
}
```