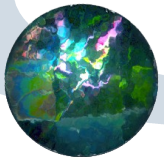
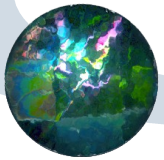
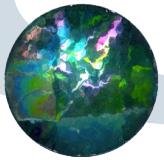


簡単なJavaプログラム その２



前回のプログラムで発生しそうな不都合

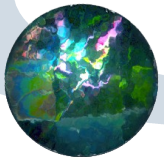
- 通常、ソートは、数字を並べ替えるのが目的ではない。
  - データを何かの順に並べ替える
  - データの種類に対応して、コードを作り替えることになる



クラスを使う例

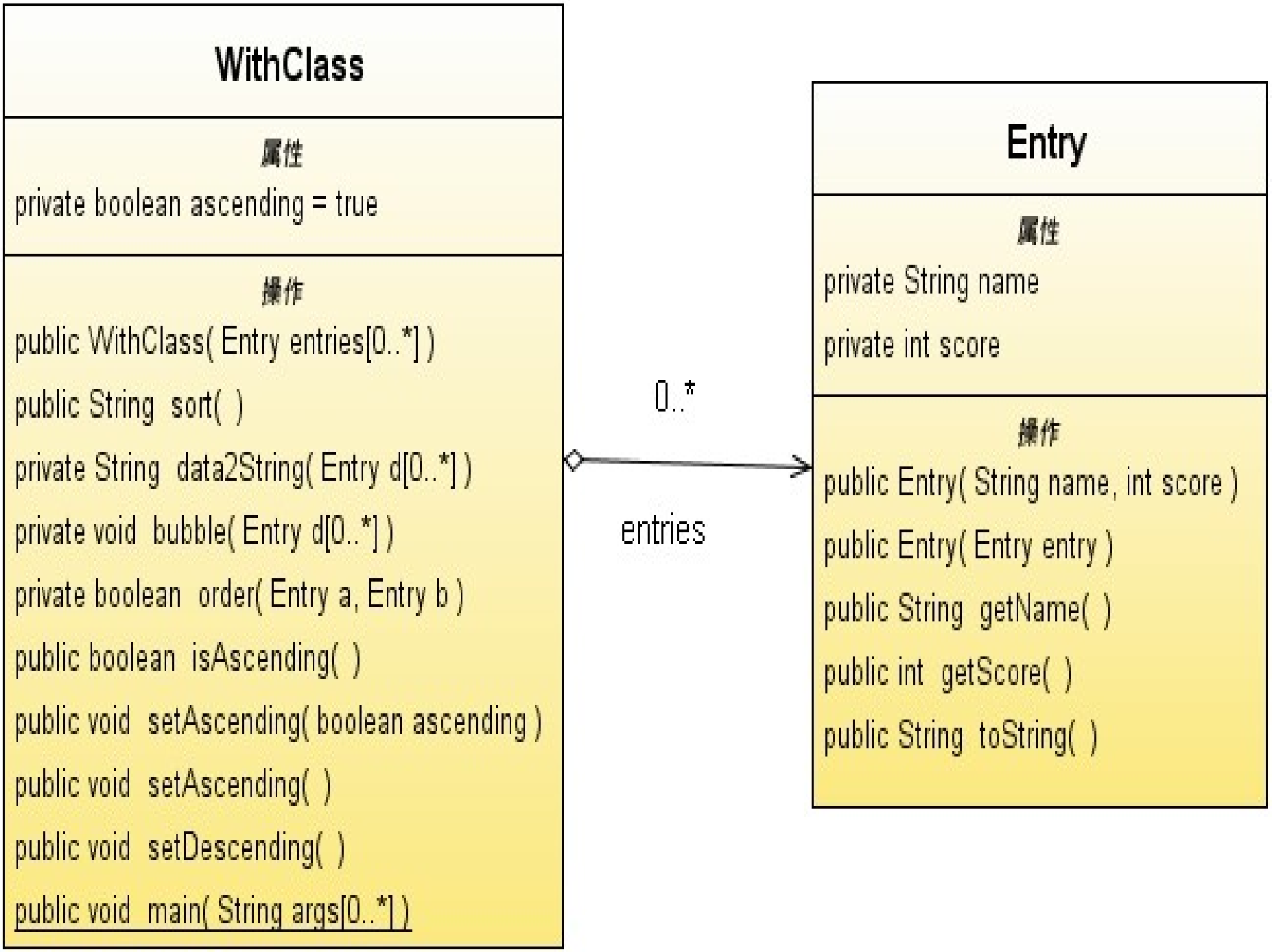
- 名前と点数を保持するクラスEntry
  - Entry.java
- 新しいインスタンスの生成new
  - 配列を一度に作ることもできる

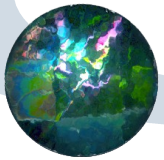
```
new Entry[]{
        new Entry("Bob", 90),
        new Entry("Mary", 70),
        new Entry("Tom", 95),
        new Entry("Mark", 85),
        new Entry("Betty", 80)
    }
```



- メインクラスの変更：WithClass.java
- 順序を確かめるメソッド

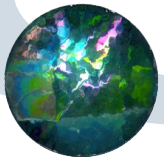
```
private boolean order(Entry a, Entry b) {
    boolean ans = false;
    if (this.isAscending() && (a.getScore() < b.getScore())) {
        ans = true;
    }
    return ans;
}
```


WithClass
屬性
private boolean ascending = true
操作
public WithClass(Entry entries[0..*])
public String sort( )
private String data2String(Entry d[0..*])
private void bubble(Entry d[0..*])
private boolean order(Entry a, Entry b)
public boolean isAscending( )
public void setAscending(boolean ascending)
public void setAscending( )
public void setDescending( )
public void main(String args[0..*])
0..*
entries
Entry
屬性
private String name
private int score
操作
public Entry(String name, int score)
public Entry(Entry entry)
public String getName( )
public int getScore( )
public String toString( )



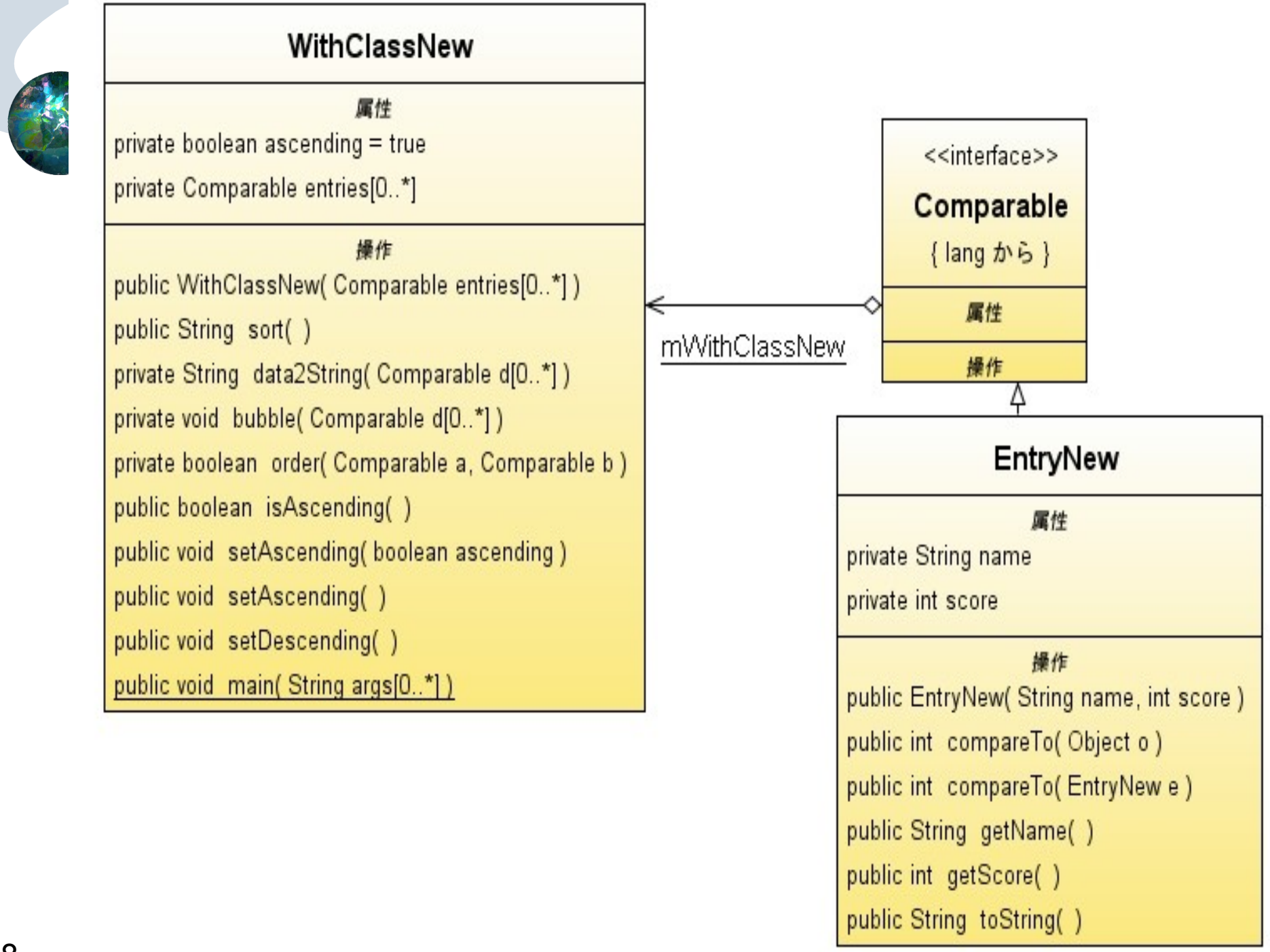
何が問題か・何を学ぶか

- メインのクラスがEntryクラスの中身を知らねばならない
  - 別のデータには別のプログラムが必要になる
- データの実装とデータを処理する過程を分離
  - クラスの抽象化
  - 抽象的データ構造
  - 抽象的インターフェイス
  - デザインパターン



インターフェイスの利用

- java.lang.Comparable
  - 要素の比較を定義する抽象インターフェイス
  - 「比較できる」と実際の比較方法を分離
- 新しいEntryクラス：EntryNew.java
- 新しいWithClassNewクラス：WithClassNew.java


WithClassNew
属性
private boolean ascending = true
private Comparable entries[0..*]
操作
public WithClassNew(Comparable entries[0..*])
public String sort( )
private String data2String(Comparable d[0..*])
private void bubble(Comparable d[0..*])
private boolean order(Comparable a, Comparable b)
public boolean isAscending( )
public void setAscending(boolean ascending)
public void setAscending( )
public void setDescending( )
public void main(String args[0..*])
mWithClassNew
<<interface>>
Comparable
{ lang から }
属性
操作
EntryNew
属性
private String name
private int score
操作
public EntryNew(String name, int score)
public int compareTo(Object o)
public int compareTo(EntryNew e)
public String getName( )
public int getScore( )
public String toString( )

WithClass.java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package firstSample;

/**
 *
 * @author tadaki
 */
public class WithClass {

    private boolean ascending = true;
    private Entry entries[];

    public WithClass(Entry entries[]) {
        this.entries =entries;
    }

    public String sort() {
        bubble(entries);
        return data2String(entries);
    }

    private String data2String(Entry d[]) {//配列内の数値を文字列化する
        String nl = System.getProperty("line.separator");

        StringBuffer buffer = new StringBuffer();
        for (int i = 0; i < d.length - 1; i++) {
            buffer.append(d[i].toString());
            buffer.append(nl);
        }
        int n = d.length - 1;
        buffer.append(d[n].toString());
        buffer.append(nl);
        return buffer.toString();
    }

    private void bubble(Entry d[]) {//泡立ち法
        for (int j = d.length - 1; j >= 1; j--) {//後ろからループを回す
            for (int i = 0; i < j; i++) {
                //順序が逆の場合
                if (!order(d[i], d[i + 1])) {
                    Entry c = d[i];
```

```
                    d[i] = d[i + 1];
                    d[i + 1] = c;
                }
            }
        }
    }

    private boolean order(Entry a, Entry b) {
        boolean ans = false;
        if (this.isAscending() && (a.getScore() < b.getScore())) {
            ans = true;
        }
        return ans;
    }

    public boolean isAscending() {
        return ascending;
    }

    public void setAscending(boolean ascending) {
        this.ascending = ascending;
    }

    public void setAscending() {
        setAscending(true);
    }

    public void setDescending() {
        setAscending(false);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        WithClass withClass = new WithClass(new Entry[]{
            new Entry("Bob", 90),
            new Entry("Mary", 70),
            new Entry("Tom", 95),
            new Entry("Mark", 85),
            new Entry("Betty", 80)
        });
        System.out.println(withClass.sort());
```

```
    }
}
```

Entry.java

```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package firstSample;

/**
 *
 * @author tadaki
 */
public class Entry {
    private String name;
    private int score;
    public Entry(String name, int score){
        this.name=name; this.score=score;
    }

    public Entry(final Entry entry){
        this.name=entry.getName();
        this.score=entry.getScore();
    }

    public String getName() {
        return name;
    }

    public int getScore() {
        return score;
    }

    public String toString(){
        return getName()+":"+getScore();
    }
}
```

WithClassNew.java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package firstSample;

/**
 * java.lang.Comparableを実装したクラステンプレートT を使うことを指示
 * @author tadaki
 */
public class WithClassNew<T extends Comparable<T>> {

    private boolean ascending = true;
    private T entries[];

    public WithClassNew(T entries[]) {
        this.entries = entries;
    }

    public String sort() {
        bubble(entries);
        return data2String(entries);
    }

    private String data2String(T d[]) {//配列内の数値を文字列化する
        String nl = System.getProperty("line.separator");

        StringBuffer buffer = new StringBuffer();
        for (int i = 0; i < d.length - 1; i++) {
            buffer.append(d[i].toString());
            buffer.append(nl);
        }
        int n = d.length - 1;
        buffer.append(d[n].toString());
        buffer.append(nl);
        return buffer.toString();
    }

    private void bubble(T d[]) {//泡立ち法
        for (int j = d.length - 1; j >= 1; j--) {//後ろからループを回す
            for (int i = 0; i < j; i++) {
                if (!order(d[i], d[i + 1])) {
                    T c = d[i];
                    d[i] = d[i + 1];
```

```
                    d[i + 1] = c;
                }
            }
        }
    }

    private boolean order(T a, T b) {
        boolean ans = false;
        if (this.isAscending() && (a.compareTo(b) < 0)) {
            ans = true;
        }
        return ans;
    }

    public boolean isAscending() {
        return ascending;
    }

    public void setAscending(boolean ascending) {
        this.ascending = ascending;
    }

    public void setAscending() {
        setAscending(true);
    }

    public void setDescending() {
        setAscending(false);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        EntryNew e[] = new EntryNew[]{
            new EntryNew("Bob", 90),
            new EntryNew("Mary", 70),
            new EntryNew("Tom", 95),
            new EntryNew("Mark", 85),
            new EntryNew("Betty", 80)
        };

        WithClassNew<EntryNew> withClass =
                new WithClassNew<EntryNew>(e);
```

```
            System.out.println(withClass.sort());

        }
    }
```

EntryNew.java

```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package firstSample;

/**
 *
 * @author tadaki
 */
public class EntryNew
        implements java.lang.Comparable<EntryNew> {

    private String name;
    private int score;

    public EntryNew(String name, int score) {
        this.name = name;
        this.score = score;
    }

    public int compareTo(EntryNew e) {
        if (e.getScore() > score) {
            return -1;
        }
        if (e.getScore() < score) {
            return 1;
        }
        return 0;
    }

    public String getName() {
        return name;
    }

    public int getScore() {
        return score;
    }

    public String toString() {
        return getName() + ":" + getScore();
    }
}
```