面向对象JML系列第二次代码作业指导书

题目描述

写在前面：请勿提交官方包代码，仅提交自己实现的类。更不要将官方包的JML或代码粘贴到自己的类中，否则以作弊、抄袭论处。

基本目标

本次作业最终需要实现一个社交关系模拟系统。可以通过各类输入指令来进行数据的增删查改等交互。

基本任务

本次作业的程序主干逻辑我们均已经实现，只需要同学们完成剩下的部分，即：

- **通过继承官方提供的接口`Person`、`Network`和`Group`，来实现自己的`Person`、`Network`和`Group`类。**

每个模块的**接口定义源代码和对应的JML规格**都已在接口源代码文件中给出，各位同学需要**准确理解JML规格**，然后使用Java来实现相应的接口，并保证**代码实现严格符合对应的JML规格。**

具体来说，各位同学需要新建三个类`MyPerson`、`MyNetwork`和`MyGroup`（仅举例，具体类名可自行定义并配置），并实现相应的接口方法，每个方法的代码实现需要严格满足给出的JML规格定义。

当然，还需要同学们在主类中通过调用官方包的`Runner`类，并载入自己实现的`Person`、`Network`和`Group`类，来使得程序完整可运行，具体形式下文中有提示。

测试模式

针对本次作业提交的代码实现，课程将使用公测+互测+bug修复的黑箱测试模式，具体测试规则参见下文。

类规格

Person类

`Person`的具体接口规格见官方包的开源代码，此处不加赘述。

除此之外，`Person`类必须实现一个构造方法

```
public class MyPerson implements Person {
    public MyPerson(int id, String name, BigInteger character, int age);
}
```

构造函数的逻辑为完成以下操作并生成一个`Person`对象。

接收人的属性：

`id`：独一无二的ID

`name`：姓名

`character`：性格

`age`：年龄

请确保构造函数正确实现，且类和构造函数均定义为`public`。`Runner`内将自动获取此构造函数进行`Person`实例的生成。

Network类

`Network`的具体接口规格见官方包的开源代码，此处不加赘述。

除此之外，`Network`类必须实现一个构造方法

```
public class MyNetwork implements Network {
    public MyNetwork();
}
```

构造函数的逻辑为生成一个空的`Network`对象。

请确保构造函数正确实现，且类和构造函数均定义为`public`。`Runner`内将自动获取此构造函数进行`Network`实例的生成。

Group类

`Group`的具体接口规格见官方包的开源代码，此处不加赘述。

除此之外，`Group`类必须实现一个构造方法

```
public class MyGroup implements Group {
    public MyGroup(int id);
}
```

构造函数的逻辑为完成以下操作并生成一个`Group`对象。

接收组的属性：

`id`：独一无二的id

请确保构造函数正确实现，且类和构造函数均定义为`public`。`Runner`内将自动获取此构造函数进行`Group`实例的生成。

输入输出

本次作业将会下发输入输出接口和全局测试调用程序，前者用于输入输出的解析和处理，后者会实例化同学们实现的类，并根据输入接口解析内容进行测试，并把测试结果通过输出接口进行输出。

输出接口的具体字符格式已在接口内部定义好，各位同学可以阅读相关代码，这里我们只给出程序黑箱的字符串输入输出。

关于main函数内对于`Runner`的调用，参见以下写法。

```
package xxx;

import com.oocourse.spec1.main.Runner;

public class xxx {
    public static void main(String[] args) throws Exception {
        Runner runner = new Runner(MyPerson.class, MyNetwork.class,
MyGroup.class);
        runner.run();
    }

}
```

规则

- 输入一律在标准输入中进行，输出一律在标准输出。
- 输入内容以指令的形式输入，一条指令占一行，输出以提示语句的形式输出，一句输出占一行。
- 输入使用官方提供的输入接口，输出使用官方提供的输出接口。

指令格式一览(括号内为变量类型)

- 基本格式：`指令字符串` `参数1` `参数2`...

  本次作业涉及指令如下：

  实际上为了减小输入量，真实输入为简写

  ```
  add_person id(int) name(String) character(BigInterger) age(int)
  add_relation id(int) id(int) value(int)
  query_value id(int) id(int)
  query_conflict id(int), id(int)
  query_acquaintance_sum id(int)
  compare_age id(int) id(int)
  compare_name id(int) id(int)
  query_name_rank id(int)
  query_people_sum
  query_circle id(int) id(int)

  add_group id(int)
  add_to_group id(int) id(int)

  query_group_sum
  query_group_people_sum id(int)
  query_group_relation_sum id(int)
  query_group_value_sum id(int)
  query_group_conflict_sum id(int)
  query_group_age_mean id(int)
  query_group_age_var id(int)
  ```

样例

#	标准输入	标准输出
1	ap 1 jack 1 100 ap 2 mark 1 100 ar 1 2 100 qv 1 2 qc 1 2	Ok Ok Ok 100 0
2	ap 1 jack 1 100 ap 2 mark 1 100 ar 1 2 100 qas 1 ca 1 2 qnr 1	Ok Ok Ok 1 = 1
3	ap 1 jack 1 100 ap 2 mark 1 100 ar 1 2 100 qps qc 1 2	Ok Ok Ok 2 0
4	ap 1 jack 1 100 ap 2 mark 1 100 ar 1 2 100 ag 1 atg 1 1	Ok Ok Ok Ok Ok
5	ap 1 jack 1 100 ap 2 mark 1 100 ar 1 2 100 ag 1 atg 1 1 atg 1 2 qgs qgps 1 qgrs 1	Ok Ok Ok Ok Ok ginf 1 1 1
6	ap 1 jack 1 100 ap 2 mark 1 100 ar 1 2 100 ag 1 atg 1 1 atg 1 2 qgvs 1 qgcs 1 qgam 1 qgav 1	Ok Ok Ok Ok Ok ginf 0 1 100 0

关于判定

数据基本限制

指令条数不多于`100000`

`add_person` 指令条数不超过 `5000`

`query_name_rank` 指令条数不超过 `333`

`query_circle` 指令条数不超过 `333`

`add_group` 指令条数不超过 `10`

`age(int)` 值在$[0, 200]$中

强测中保证仅有20%的数据含有 `10000` 以上的指令数

标程复杂度瓶颈为O(n*group_sum)，n为指令条数

互测数据限制

指令条数不多于 `5000`

`add_person` 指令条数不超过 `5000`

`query_name_rank` 指令条数不超过 `333`

`query_circle` 指令条数不超过 `333`

`add_group` 指令条数不超过 `10`

`age(int)` 值在$[0, 200]$中

`name(String)` 长度不超过10

`character(BigInteger)` 值在$[0, 6666666666666666666]$中

测试模式

公测和互测都将使用指令的形式模拟容器的各种状态，从而测试各个接口的实现正确性，即是否满足JML规格的定义。**可以认为，只要代码实现严格满足JML，就能保证正确性。**

任何满足规则的输入，程序都应该保证不会异常退出，如果出现问题即视为未通过该测试点。

程序的最大运行cpu时间为6.66s，虽然保证强测数据有梯度，但是还是请注意时间复杂度的控制。

提示&说明

- 如果还有人不知道标准输入、标准输出是啥的话，那在这里解释一下
  - 标准输入，直观来说就是屏幕输入
  - 标准输出，直观来说就是屏幕输出
  - 标准异常，直观来说就是报错的时候那堆红字
  - 想更加详细的了解的话，请去百度
- 本次作业中可以自行组织工程结构。任意新增 `java` 代码文件。只需要保证两个类的继承与实现即可。
- **关于本次作业容器类的设计具体细节，本指导书中均不会进行过多描述，请自行去官方包开源仓库中查看接口的规格，并依据规格进行功能的具体实现，必要时也可以查看Runner的代码实现。**
- 开源库地址：传送门
- 推荐各位同学在课下测试时使用Junit单元测试来对自己的程序进行测试
  - Junit是一个单元测试包，**可以通过编写单元测试类和方法，来实现对类和方法实现正确性的快速检查和测试。**还可以查看测试覆盖率以及具体覆盖范围（精确到语句级别），以帮助编程者全面无死角的进行程序功能测试。

    - Junit已在评测机中部署（版本为Junit4.12，一般情况下确保为Junit4即可），所以项目中可以直接包含单元测试类，在评测机上不会有编译问题。
    - 此外，Junit对主流Java IDE（Idea、eclipse等）均有较为完善的支持，可以自行安装相关插件。推荐两篇博客：
        - Idea下配置Junit
        - Idea下Junit的简单使用
    - 感兴趣的同学可以自行进行更深入的探索，百度关键字：`Java Junit`。
- **不要试图通过反射机制来对官方接口进行操作**，我们有办法进行筛查。此外，在互测环节中，如果发现有人试图通过反射等手段hack输出接口的话，请邮件HugeGun@buaa.edu.cn或私聊助教进行举报，**经核实后，将直接作为无效作业处理。**