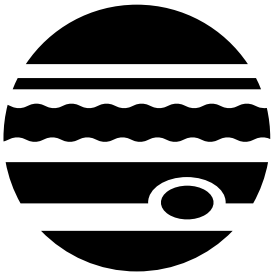


Java Homework2
MatchGame Report

Date: 18-10-9

火柴棒游戏报告

一、作业要求

1.用户从命令行输入最大数字的位数(如1位数、2位数、3位数);

2.用户从命令行输入提示数(2或3)，表示等号左式数字的个数;

3.用户从命令行输入题目类型编号(移动、移除、添加)，以及 火柴棒根数;

4.系统随机自动生成火柴棒游戏，并展示(直接用数字展示);

5.用户输入答案，系统验证是否正确;若正确，则提示正确;若错误，则让用户继续输入;

6.若用户直接回车，则显示正确答案。

二、思路及步骤

第一次编写比较大型的JAVA程序，首先需要对我们的需求进行分析，通过作业的五点要求，我首先明确了需要在几个类里实现功能：

1、需要一个处理输入输出的模块(IOStream)

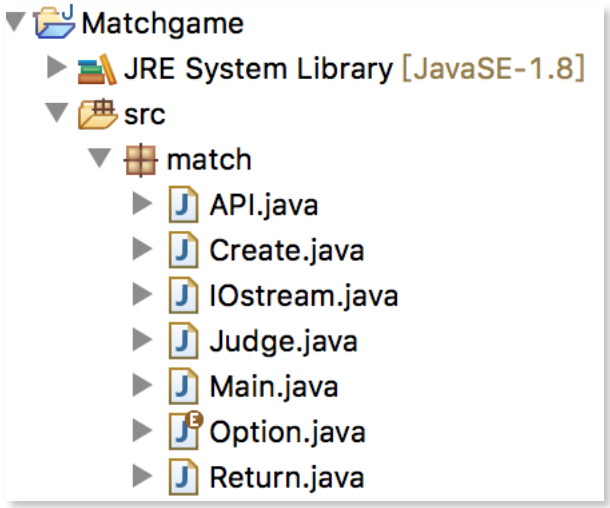
2、需要有一个生成题目的模块(Create)

3、需要有一个判断答案的模块(Judge)

4、需要有一个连接1、2、3模块的模块(API)

5、需要有一个主程序(可忽略)

因而，程序的基本框架如下：



通过对需求分析，可以发现这个程序有两个难点功能需要实现：

1、如何通过用户的输出创造出一道题目？

2、如何解析并判断用户的答案是否正确？

除此之外，还有一些比较简单的功能可以首先实现：

1、输入输出主界面的文字，比如“请输入……”；

2、主程序的循环；

3、为操作建立枚举类；

4、将输入输出与判断、生成模块连接。

因而我首先先建立了基本框架，并没有先急着写核心函数。期间我发现，我们所要创造的不等式其实可以先由等式变形形成，虽然计算机对生成可变化的不等式的能力很差，但是对生成等式的能力很好。因而Create部分也可以先写好通过用户键入及随机数功能建立等式的基本函数Equality()。建立好了基本框架之后，我开始构思对数字的处理方式：

我对数字的表达最开始的想法是：由于火柴棍的表达方式和硬件课程中的七段数码管显示类似，所以我最开始想为每一个数字都创建一个7位的数组来表示，但是这个表达很快就遇到了难点：这种表达方法对加减符号来说并不适用，而且这种方法只能做到表示，如果要对某一个火柴棒进行移动，就会变得非常麻烦，需要检测很多相关的数组。

很快我推翻了这个设想，后来我又想到或许可以建立一个链表桶，将每个数字加减一根火柴棒的关系连接在一起。但是这样又遇到了另一个问题：这样形成的链表会变成双向的？还是应该做成单向的？这样的结构大量使用指针会不会使得程序变得非常混乱？这个构想很快也被舍弃了。

由于上文的构想，我发现其实这些数字的变化都非常地有规律，甚至可以枚举出来到底有多少种变换方式！因为这个题目最终是与火柴棒的根数有关的，不妨这样记录每一个数字：

0	1	2	3	4	5	6	7	8	9	+	-
6根	2根	5根	5根	4根	5根	6根	3根	7根	6根	2根	1根

我们对火柴棒的操作一共有三个：添加、移除、移动。其实我们只需要考虑一种情况就好了，因为移动可以看作先添加再减少，移除也可以看成添加的逆过程。这样列出来之后还有一个问题，就是火柴棍的移动又并不是简单的数字加减的关系，比如2和5都是5根火柴棒，但是2无法移动变成5。所以我还应该再建立一个关于变化的关系图表，这样的图表表明了一个数字要增加几根火柴棒可以变成另一个数字：

增加一根	1->7	3->9	5->6	6->8	9->8	- -> +	0->8
增加两根	4->9	5->8	2->8	1->4	7->3		
增加三根	1->3	7->0					

这样列出之后，就极小地缩短了我需要写的情况范围。接下来的工作就变得简单明了了。

三、关键函数

1、Create类详解

Equality函数主要通过传入进Ceate类的参数来创建一个等式，并且把这个等式的所

```
for( int i = 0 ; i < hint_Number ; i ++ ) {
    numbers[i] = my_Random.nextInt(max_Random-1)+1;//生成1～（10^n-1）的随机数
    equality = equality + numbers[i];
    if( i < hint_Number - 1 ) {//
        options[i] = my_Random.nextInt(2)+4;//生成4～5的随机数
        equality = equality + Option.getName(options[i]);
    }
    if( i > 0 ) {//从第二个开始判断加或减
        if( options[i-1] == Option.PLUS.getIndex() ) {//加
            sum += numbers[i];
        }
        else if( options[i-1] == Option.MINUS.getIndex() ) {//减
            sum -= numbers[i];
        }
    }
    else sum = numbers[i];//第一个直接等于
}
```

有信息存储在Create的成员内。

```
public String Inequality ( ){
    String equality = Equality();
    CountAmount( equality );
    String inequality;
    //System.out.println(equality);

    //有些等式可能无法满足题目要求凑成不等式，所以需要restart一个新的等式
    //这里采用一次递归来得到这些不能凑成不等式的等式的新的等式

    if( Option.getName(option_Number) == "添加") {
        inequality = Minus(equality);
        if( restart_Flag == true ) {
            restart_Flag = false;
            return Inequality();
        }
        else {
            equality_Str = equality;
            return inequality;
            }
    }
```

Inequality函数通过Equality()得到一串等式的字符串，通过之前提到的检测方法，加入随机数判断，对数字进行随机的增加或减少。由于有一些等式存在着不能改变以符合题意的情况，此时restart_Flag会使得Inequality函数自身递归，启用一个新的Equality使得等式符合题意。

2、Judge类详解

Judge类的工作相对于Create来说简单很多。我采用了比较简单的检测方法（可能存在一些边界情况）。Judge首先将Flag置为false，第一次检测是用户输入的是否是一个等式，如果这个检测通过了，再进入下一个检测，检测用户输入的等式的火柴棒数目是否满足create创造的火柴棒数目与用户输入的需要操作的火柴棒数目相同。因为Judge的很多内容都是Create的逆过程，因而就不赘述了。

```
public void JudgeGame( Create create ) {
    match_Amount = 0;
    numbers = new int[create.numbers.length];
    options = new int[create.options.length];
    flag = false;
    JudgeEquality(create);
    flag = false;
    JudgeAmount(create);
}
```

四、结果及说明

1、测试1

```
~火柴棒游戏开始~
请输入最大数字的位数:1
读入成功.
请输入提示数:2
读入成功.
请输入题目类型(1:移除;2:移动;3:添加):1
读入成功.
请输入操作火柴棒数目:1
数据读入完毕!正在生成游戏...
游戏如下:
8-6=3
您需要移除1根火柴棒.
请输入您的答案:
输入的是回车!正在为您输出正确答案...
正确答案如下:
9-6=3
~火柴棒游戏开始~
请输入最大数字的位数:
```

测试1完成最基本的测试，程序生成火柴棒游戏正常，输入输出正常，输入回车后会显示正确答案并继续游戏。

2、测试2

```
～火柴棒游戏开始～
请输入最大数字的位数:2
读入成功.
请输入提示数:1
读入失败,提示数过大或过小.
请输入提示数:2
读入成功.
请输入题目类型(1:移除;2:移动;3:添加):2
读入成功.
请输入操作火柴棒数目:1
数据读入完毕!正在生成游戏...
游戏如下:
88-42=30
您需要移动1根火柴棒.
请输入您的答案:
输入的是回车!正在为您输出正确答案...
正确答案如下:
80-42=38
```

测试2测试了题目2的生成，以及对两位数的题目的生成。

3、测试3

```
～火柴棒游戏开始～
请输入最大数字的位数:3
读入成功.
请输入提示数:3
读入成功.
请输入题目类型(1:移除;2:移动;3:添加):1
读入成功.
请输入操作火柴棒数目:2
数据读入完毕!正在生成游戏...
游戏如下:
545+863-307=501
您需要移除2根火柴棒.
请输入您的答案:1+2+3=2
1+2+3=2
好可惜,差一点点就正确了,再想想?
请输入您的答案:
输入的是回车!正在为您输出正确答案...
正确答案如下:
545+263-307=501
```

测试3测试了三位数，以及对输入题目的判断。

4、测试4

```
～火柴棒游戏开始～
请输入最大数字的位数:2
读入成功.
请输入提示数:3
读入成功.
请输入题目类型(1:移除;2:移动;3:添加):3
读入成功.
请输入操作火柴棒数目:1
数据读入完毕!正在生成游戏...
游戏如下:
22-54+50=16
您需要添加1根火柴棒.
请输入您的答案:
输入的是回车!正在为您输出正确答案...
正确答案如下:
22-54+50=18
```

测试4测试了题目3的生成。

五、作业心得

这次的Java作业对我来说还是很有难度的，在没有什么可参考代码的情况下，自己独立完成了近700行的代码，感觉还是很有成就感的。而且这次写代码也比较讲究方法，因为题目比较难，一开始我对自己的程序要求就比较简单，希望能生成最简单的一位算数问题就可以了。后来发现自己想的方法确实可以推广，目前还暂时没有发现对高位数有什么Bug。

同时这次作业也算是对之前C++的面向对象的一次复习，因为一个假期都没有怎么写代码，所以所最开始写的时候非常混乱，最开始甚至给所有类的所有参数都用的是 public static 的类型。eclipse的编译环境非常适合我这样自己写了一堆之后代码格式糟糕，乱用全局变量的人，它的提示很多情况下都减少了我的后期工作。

这次还学到了很多Java中对字符串处理的一些操作，对随机数的操作，以及Java里的package与class之间的关系。Java的良好的封装性面向对象的其他特性随着作协的不断深入我也有了更深的认识。