Java Homework 2

2020 年 10 月 3 日

1. 火柴棒游戏简介

一个基于操作火柴棒来使等式成立的火柴棒游戏

(i) 用户从命令行依次输入等号左边数字的最大位数, 等号左边数的个数, 题目的类型(移动, 添加, 移除), 以及需要操作的火柴棒数目

(ii) 系统随机自动生成一个符合输入条件的题目, 并展示在控制台(数字形式和火柴棒图形形式), 并提示用户输入答案

(iii) 用户可以把等式输入作为答案, 若正确, 则游戏胜利, 否则系统会提示答案错误, 再次输入

(iv) 用户可以通过输入“quit”来放弃作答, 这样的话游戏直接失败

(v) 若用户直接回车, 则显示正确答案

2. 运行演示

示例1:

等号左边数字最多2位, 等号左边2个数字, 游戏模式为“移动(火柴棒)”.

先输入错误答案(提示答案错误并要求再次输入), 再输入正确答案(提示答案正确, 游戏胜利).

示例2:

等号左边数字最多3位, 等号左边3个数字, 游戏模式为“添加(火柴棒)”.

先输入错误答案(提示答案错误并要求再次输入), 然后直接回车(显示正确答案).

示例3:

等号左边数字最多2位, 等号左边3个数字, 游戏模式为“移除(火柴棒)”.

输入“quit”退出游戏(提示失败).

```
number length in the left of '=':2
num of numbers in the left of '=':2
game mod(0:add, 1:rmv, 2:mov):2
num of matches to operate:2
OK to mov
original equation(for test):85+72=157
 _  _     _  _        _  _
 _||_  +  _| _| =   ||_   |
 _| _|    _||_      | _|  |
35+32=157
answer:
85+72=151
Wrong answer(85+72=151). Try again:
85+72=157
Right answer. Congratulations!

Process finished with exit code 0
```

图 1: 实例1

```
number length in the left of '=':3
num of numbers in the left of '=':3
game mod(0:add, 1:rmv, 2:mov):1
num of matches to operate:3
OK to rmv
original equation(for test):47+448+280=775
    _           _     _  _  _     _  _  _
|_||_| + |_||_||_| +  _||_|| | =   |  ||_
  | _|     |  ||_|   |_ |_||_|     |  | _|
49+448+280=775
answer:
47+448+280=776
Wrong answer(47+448+280=776). Try again:

Answer: 47+448+280=775

Process finished with exit code 0
```

图 2: 示例2

```
number length in the left of '=':2
num of numbers in the left of '=':3
game mod(0:add, 1:rmv, 2:mov):1
num of matches to operate:1
OK to rmv
original equation(for test):31+41+68=140
 _  _              _  _           _
 _|  | + |_|  | + |_ |_| =   ||_|| |
 _|  |     |  |   |_||_|     |  ||_|
37+41+68=140
answer:
quit
You failed, but this is not the end. Good luck.

Process finished with exit code 0
```

图 3: 示例3

3. 实现方法

3.1 主要逻辑

1. 用户输入对题目的要求后, 先生成一个符合要求的, 而且成立的等式(先生成等式的左半边, 然后通过运算得到右半边), 作为正确答案(这样可以保证题目一定是有解的).
2. 根据用户所选择的游戏模式构造题目, 具体操作为:
   - 移除: 在正确答案的基础上增加火柴棒
   - 添加: 在正确答案的基础上移除火柴棒
   - 移动: 在正确答案的基础上移动火柴棒
3. 根据用户输入进行判断, 将用户的答案(等式字符串)和正确答案(等式字符串)进行比较,
   - 若相等则用户作答正确, 游戏结束;
   - 若不正确而且用户是直接回车的, 则输出正确答案, 游戏结束;
   - 若不正确而且用户输入是"quit", 则提示失败, 游戏结束;
   - 若用户的输入是单纯的不正确, 则提示答案错误并要求再次输入.

3.2 程序主要结构

程序只有一个游戏主类, 但在主类下还有一些子结构:

- 辅助函数: rmSpaces, 用于移除字符串中的空格
- 辅助类: Pair, 类似于C++中的std::pair
- 火柴棒数字0-9: MatchDigit
- 火柴棒整数: MatchNumber, 由火柴棒数字组成
- 火柴棒表达式: MatchExp, 由火柴棒整数和运算符('-'或'+')组成
- 火柴棒等式: MatchEqu, 由左右两个表达式和等号组成
- main函数: 游戏运行的入口

```
1 public class MatchesGame {
2     public static String rmSpaces(String s){...}
3     public static class Pair<T1,T2>{...}
4     public static class MatchDigit{...}
5     public static class MatchNumber{...}
6     public static class MatchExp{...}
```

```
    public static class MatchEqu{...}
    public static void main(String[] args){...}
}
```

3.3 主要数据结构和算法

a. 火柴棒数字的表示

把一个火柴棒数字分成7段, 用一个7位的数组来表示, 数组某一位为1表示对应的数字段是填充的, 为0则是不填充的.

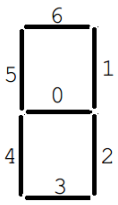

图 4: 火柴棒数字

火柴棒数字附加的操作:

- 求值: 返回火柴棒数字表示的数字
- 添加若干火柴
- 移除若干火柴

```
public static class MatchDigit{
    byte[] seg = new byte[7];

    public MatchDigit(){seg = new byte[7];}
    public MatchDigit(int n){...}
    public int value(){...}
    public void print(){...}
    public int matchesCnt(){...}
    public int reMatchesCnt(){...}

    public int addMatches(int cnt){...}
    public int rmvMatches(int cnt){...}
}
```

b. 题目构造算法

根据题目要求构造题目

```
    input: num_len, num_cnt
    String equ_str = genEquation(num_len, num_cnt);
    MatchEqu euqation = new MatchEqu(equ_str);

    if(mod==0){//need to add
        while(equation cannot be changed to satisfy the requirement){
            // generate a new equation
            equ_str = genEquation(num_len, num_cnt);
            euqation = new MatchEqu(equ_str);
        }
    }else if(mod==1){//need to rmv
        ... //similar to the case mod==0
    }else if(mod==2){//need to mov
        ... //similar to the case mod==0
    }
    output: equation, equ_str;
```

在等式中添加一定数量的火柴棒:
-调用火柴棒表达式的添加若干火柴棒的方法
-调用火柴棒整数的添加若干火柴棒的方法
-调用火柴棒数的添加若干火柴棒的方法

```
public static class MatchEqu{
    MatchExp left;
    MatchExp right;
    ...
    public int addMatches(int cnt){
        cnt = left.addMatches(cnt);
        //the return of method "addMatches" is the number of left matches to change
        if(cnt==0) return 0;
        cnt = right.addMatches(cnt);
        return cnt;//cnt is the num of unfinished changes
    }
}
```

从等式中移除一定数量的火柴棒:
-调用火柴棒表达式的移除若干火柴棒的方法
-调用火柴棒整数的移除若干火柴棒的方法
-调用火柴棒数的移除若干火柴棒的方法

```
public static class MatchEqu{
    MatchExp left;
    MatchExp right;
    ...
    public int rmvMatches(int cnt){
        cnt = left.rmvMatches(cnt);
        //the return of method "addMatches" is the number of left
        ↪  matches to change
        if(cnt==0) return 0;
        cnt = right.rmvMatches(cnt);
        return cnt;//cnt is the num of unfinished changes
    }
}
```

从等式中移动一定数量的火柴棒:
-调用火柴棒表达式的移动若干火柴棒的方法
-调用火柴棒整数的移动若干火柴棒的方法

```
public static class MatchEqu{
    MatchExp left;
    MatchExp right;
    ...
    public int movMatches(int cnt){
        Pair<Integer,Integer> mov = new Pair<Integer,
        ↪  Integer>(cnt,cnt);
        mov=left.movMatches(mov.first, mov.second);
        if(mov.first==0 && mov.second==0) return 0;
        mov=right.movMatches(mov.first, mov.second);
        if(mov.first>0) return -1*mov.first;
        return mov.second;
        //return the num of unfinished changes
    }
}
```

---

4. 总结与感悟

本次的Java作业看起来比较简单, 但实际做起来还是需要考虑很多东西的, 比如如何做到题目的随机性, 如何保证题目一定有解, 如何保证构造出的题目仍然是由数组成的等式(没有0-9,+,-,=之外的其他符号), 如何构造数据结构以使对火柴棒的操作更加便捷直观, 等等.

在完成作业的过程中, 用到了许多Java的基础知识, 对于之前没有学过Java的自己来说, 这次作业给了我一次充分锻炼自己Java编程能力的机会, 通过完成这次作业, 我对Java语言的许多特性都有了了解.

作业的不足之处大概在于代码的略显臃肿, 以及算法的粗糙, 没有将Java语言的特性充分利用, 希望在接下来的学习过程中可以不断磨砺自己对Java的理解, 写出更好的程序.