SE-HHU Project01-四则运算

1、题目

编写一个四则运算程序，满足如下要求：

1.题目要求: 100 以内加减法，满足小学二年级数学口算需求

2.使用参数控制生成题目的个数

3.每道题目中出现的运算符个数不超过两个

4.程序- -次运行生成的题目不能重复，请思考关于重复的定义。生成的题目存入执行程序的当前目录 下的 Exercises.txt 文件， 格式如下:

1.四则运算题目 1

2.四则运算题目 2

5.再生成题目的同时，计算出所有题目的答案，并存入执行程序的当前目录下的 Answers.txt 文件, 格式如下:

1.答案 1

2.答案 2

6.估计需求分析、设计、编码、测试各阶段时间，记录实际工作中各项工作时间花费，并列表进行对 比

2、需求分析

1、100 以内的加减法（参与运算的数、过程答案及最终答案均小于等于 100 且大于等于 0）

2、不能出现重复算式（参与运算的数与符号至少有一个不同）

3、最多含有两个运算符（即一个或两个）

4、由用户决定生成算式的个数（设参数控制生成的算式数）

5、将算式和答案分别存到 Exercises.txt 与 Answers.txt 文件中

3、设计

1、分别设计 Equation01 与 Equation02 两个类存储两个数字运算算式与三个数字运算算式

2、设置 x、y、z 的随机数种子，范围在 10-100 之间（小部分微调为 10-50），运用 for 循环及 if 判断每次运算的数是否在 0-100 之间，若不成立则重新生成随机数

3、在类中构建 compare（）方法作为比较式子是否相同，包括数字及运算符是否重复

4、在类中构建 toString（）方法将算式导出为字符串

5、设置文件输出流将结果储存到对应文件

4、编码

class Main

```
package work01;

import java.util.*;
import java.io.*;
public class Main {
    public static void main(String [] args) throws IOException {
        Random r = new Random();
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int m = sc.nextInt();
        Equation01[] ep1 = new Equation01[n];
        Equation02[] ep2 = new Equation02[m];
        BufferedWriter bfw1 = new BufferedWriter(
                              new OutputStreamWriter(
                              new
FileOutputStream(".\\Exercises.txt")));
        BufferedWriter bfw2 = new BufferedWriter(
                              new OutputStreamWriter(
                              new
```

```
FileOutputStream(".\\Answers.txt")));
        for (int i = 0; i < n; i++) {
            ep1[i] = new Equation01();
            ep1[i].creat();
            for (int j = 0; j < i; j++) {
                if (ep1[i].compare(ep1[j])) {
                    ep1[i].creat();
                    j = 0;
                }
            }
        }
        for (int i = 0; i < m; i++) {
            ep2[i] = new Equation02();
            ep2[i].creat();
            for (int j = 0; j < i; j++) {
                if (ep2[i].compare(ep2[j])) {
                    ep2[i].creat();
                    j = 0;
                }
            }
        }
        for(int i = 0;i < n;i++){
            bfw1.write((i+1)+". "+ep1[i].toString()+"\n");
            bfw2.write((i+1)+". "+ep1[i].ans+"\n");
        }
        for(int i = 0;i < m;i++){
            bfw1.write((i+n+1)+". "+ep2[i].toString()+"\n");
            bfw2.write((i+n+1)+". "+ep2[i].ans+"\n");
        }
        bfw1.flush();
        bfw1.close();
        bfw2.flush();
        bfw2.close();
    }
}
```

class Equation01

```
package work01;

import java.util.*;
public class Equation01 {
    Random r = new Random();
```

```
    int x = r.nextInt(10, 100);
    int y = r.nextInt(10, 100);
    int ans = 0;
    int flag = 0;
    public void add() {
        while((x+y)>100) {
            x = r.nextInt(10, 100);
            y = r.nextInt(10, 100);
        }
        ans = x + y;
        flag = 0;
    }
    public void sub(){
        if (x < y){
            int temp = 0;
            temp = x;
            x = y;
            y = temp;
        }
        ans = x -y;
        flag = 1;
    }
    public void creat(){
        if(r.nextBoolean())
            this.add();
        else
            this.sub();
    }
    public boolean compare(Equation01 ep){
        if((ep.flag == this.flag)     //前面运行的运行中有与本次运行
结果相同的算式
            &&(ep.x == this.x)
            &&(ep.y == this.y))
            return true;                  //出现重复返回 true
        return false;                      //不重复返回 false
    }
    public String toString(){
        StringBuilder stb = new StringBuilder("");
        if(this.flag == 0)
            stb.append(x+"+"+y+"=");
        else if(this.flag == 1)
            stb.append(x+"-"+y+"=");
        return stb.toString();
```

```
    }
}
```

class Equation02

```
package work01;

import java.util.*;
public class Equation02 {
    Random r = new Random();
    int x = r.nextInt(10,100);
    int y = r.nextInt(10,100);
    int z = r.nextInt(10,100);
    int ans = 0;
    int flag1 = 0;
    int flag2 = 0;      //0 值表示加 1 值表示减
    public void firstAdd(){
        flag1 = 0;
        if(r.nextBoolean()){
            while((x+y+z)>100) {
                x = r.nextInt(10, 50);
                y = r.nextInt(10, 50);
                z = r.nextInt(10,50);
            }
            flag2 = 0;
        }else {
            while((x + y) > 100) {
                x = r.nextInt(10, 100);
                y = r.nextInt(10, 100);
            }
            while((x + y - z) < 0){
                z = r.nextInt(10,80);
            }
            flag2 = 1;
        }
    }
    public void firstSub(){
        flag1 = 1;
        if (x < y){
            int temp = 0;
            temp = x;
            x = y;
            y = temp;
```

```
        }
        if(r.nextBoolean()){
            while((x - y + z) > 100)
                z = r.nextInt(10,100);
            flag2 = 0;
        }else{
            while(x - y < 10){
                x = r.nextInt(30, 100);
                y = r.nextInt(10, 100);
            }
            while((x - y - z) < 0){
            z = r.nextInt(10,50);
        }
            flag2 = 1;
        }
    }
    public void creat(){
        if(r.nextBoolean())
            this.firstAdd();
        else
            this.firstSub();
    }
    public boolean compare(Equation02 ep){
        if((ep.flag1 == this.flag1)
                &&(ep.flag2 == this.flag2)
                &&(ep.x == this.x)
                &&(ep.y == this.y)
                &&(ep.z == this.z))
            return true;
        return false;
    }
    public String toString() {
        StringBuilder stb = new StringBuilder("");
        switch (this.flag1){
            case 0 : stb.append(x + "+" + y );
                ans = x + y ;
                break;
            case 1 : stb.append(x + "-" + y );
                ans = x - y ;
                break;
            default: return "";
        }
        switch (this.flag2){
            case 0 : stb.append("+" + z + "=");
```

```
                ans += z;
                break;
            case 1 : stb.append("-" + z + "=");
                ans -= z;
                break;
            default: return "";
        }
        return stb.toString();
    }
}
```

5、测试

全程测试无明显 bug，如果算式数过多会出现运行时间明显加长，具体原因为所用随机出的算式大概率与之前出现的重复，导致运行时间增长。可通过增设二维或三维数组来解决，但会导致空间急剧增加，故而不推荐使用

Exercises.txt 文件

1. 40-37=
2. 43-29=
3. 75-57=
4. 53+19=
5. 38+52=
6. 67+25=
7. 79-41=
8. 65-27=
9. 34+24=
10. 55-44=
11. 10+20=
12. 80-58=
13. 84-56=
14. 83+13=
15. 48-22=
16. 45-39+58=
17. 61-26+36=
18. 27-11+63=
19. 29+49-77=
20. 31+43-42=
21. 16+49+30=
22. 82-12-63=
23. 22+40+31=
24. 22+12+16=

25. 91-36+43=
26. 77-22-43=
27. 70-11-42=
28. 63-47+69=
29. 50-13-37=
30. 10+32+41=

Answers.txt 文件

1. 3
2. 14
3. 18
4. 72
5. 90
6. 92
7. 38
8. 38
9. 58
10. 11
11. 30
12. 22
13. 28
14. 96
15. 26
16. 64
17. 71
18. 79
19. 1
20. 32
21. 95
22. 7
23. 93
24. 50
25. 98
26. 12
27. 17
28. 85
29. 0
30. 83

6、时间分析表格

阶段	估计时间	实际时间
需求分析	5min	10min
设计	10min	20min
编码	3h	6h
测试	30min	5min