Excel单元格数值统计

知识点递归 循环数组

时间限制：2s 空间限制：256MB 限定语言：不限

题目描述：

Excel工作表中对选定区域的数值进行统计的功能非常实用。
仿照Excel的这个功能，请对给定表格中选中区域中的单元格进行求和统计，并输出统计结果。

为简化计算，假设当前输入中每个单元格内容仅为数字或公式两种。

如果为数字，则是一个非负整数，形如3、77

如果为公式，则固定以=开头，且仅包含下面三种情况：

等于某单元格的值，例如=B12

两个单元格的双目运算（仅为+或-），形如=C1-C2、C3+B2

单元格和数字的双目运算（仅为+或-），形如=B1+1、100-B2

注意：

公式内容都是合法的，例如不存在,=C+1、=C1-C2+B3，=5、=3+5

不存在循环引用，例如A1=B1+C1、C1=A1+B2

内容中不存在空格、括号

输入描述：

第一行两个整数rows cols,表示给定表格区域的行数和列数，1<=rows<=20，1<=cols<=26。
接下来rows行，每行cols个以空格分隔的字符串，表示给定表格values的单元格内容。
最后一行输入的字符串，表示给定的选中区域，形如A1:C2。

输出描述：

一个整数，表示给定选中区域各单元格中数字的累加总和，范围-2,147,483,648 ~ 2,147,483,647

补充说明：

表格的行号1~20，列号A~Z,例如单元格B3对应values[2][1]。
输入的单元格内容（含公式）中的数字均为十进制，值范围[0,100]。
选中区域：冒号左侧单元格表示选中区域的左上角，右侧表示右下角，如可以为B2:C10、B2:B5、B2:Y2、B2:B2,无类似C2:B2、C2:A1的输入。

示例1

输入：

```
5 3
10 12 =C5
15 5 6
7 8 =3+C2
6 =B2-A1 =C2
7 5 3
B2:C4
```

输出：

```
29
```

示例2

输入：

```
1 3
1 =A1+C1 3
A1:C1
```

输出：

```
8
```

解题思路：

```
public class Main {

    public static String[][] strings;

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int row = sc.nextInt();
        int col = sc.nextInt();
        sc.nextLine();

        strings = new String[row][col];
        for(int i=0; i<row; i++){
            strings[i] = sc.nextLine().split(" ");
        }

        String[] nums = sc.nextLine().split(":");
        int[] start = zuobiao(nums[0]);
        int[] end = zuobiao(nums[1]);

        int sum = 0;
        for( int i=start[0]; i<=end[0]; i++){
            for(int j=start[1]; j<=end[1]; j++){
                String temp = strings[i][j];
                if(temp.contains("=")){
                    sum += jisuan(temp);
                }else {
                    sum += Integer.valueOf(temp);
                }
            }
        }

        System.out.println(sum);
    }

    public static int jisuan(String s){

        s = s.replace("=","");
        boolean jiafa = true;     //是否为加法运算
        boolean isDigit = true;         //是否为纯数字
        int num1 = 0;
        int num2 = 0;
        String temp = "";
```

```
        for(int i=0; i<s.length(); i++){
            char c = s.charAt(i);
            if(c == '-' || c == '+'){
                if(c == '-'){
                    jiafa = false;
                }
                if(isDigit){       //纯数字
                    num1 = Integer.valueOf(temp);
                } else {
                    int[] ints = zuobiao(temp); //先求出其坐标位置
                    String str = strings[ints[0]][ints[1]];
                    if(str.contains("=")){    //如果此坐标位置还是一个算式需要继续求值
                        num1 = jisuan(str);
                    }else {
                        num1 = Integer.valueOf(str);
                    }
                }
                temp = "";
                isDigit = true;
            }else {
                if(Character.isLetter(c)){
                    isDigit = false;       //包含字母则非纯数字
                }
                temp += c;
            }
            if(i == s.length()-1){
                if(isDigit){       //纯数字
                    num2 = Integer.valueOf(temp);
                } else {
                    int[] ints = zuobiao(temp); //先求出其坐标位置
                    String str = strings[ints[0]][ints[1]];
                    if(str.contains("=")){    //如果此坐标位置还是一个算式需要继续求值
                        num2 = jisuan(str);
                    }else {
                        num2 = Integer.valueOf(str);
                    }
                }
            }
        }

        return jiafa ? num1 + num2 : num1 - num2;
    }

    public static int[] zuobiao(String s){
```

```
            int y = s.charAt(0) - 'A';
            String num = "";
            for(int i=1; i<s.length(); i++){
                num += s.charAt(i);
            }

            return new int[]{ Integer.valueOf(num) - 1, y};
        }
}
```