题目描述：

给定一组不等式，判断是否成立并输出不等式的最大差(输出浮点数的整数部分)，要求：1）不等式系数为 double 类型，是一个二维数组；2）不等式的变量为 int 类型，是一维数组；3）不等式的目标值为 double 类型，是一维数组；4）不等式约束为字符串数组，只能是：">",">=","<","<=","="，例如,不等式组：

a11*x1+a12*x2+a13*x3+a14*x4+a15*x5<=b1;

a21*x1+a22*x2+a23*x3+a24*x4+a25*x5<=b2;

a31*x1+a32*x2+a33*x3+a34*x4+a35*x5<=b3;

最大差=max{ (a11*x1+a12*x2+a13*x3+a14*x4+a15*x5-b1), (a21*x1+a22*x2+a23*x3+a24*x4+a25*x5-b2), (a31*x1+a32*x2+a33*x3+a34*x4+a35*x5-b3) }，类型为整数(输出浮点数的整数部分)

输入描述：

1）不等式组系数(double 类型)：

a11,a12,a13,a14,a15

a21,a22,a23,a24,a25

a31,a32,a33,a34,a35

2）不等式变量(int 类型)：

x1,x2,x3,x4,x5

3）不等式目标值(double 类型)：b1,b2,b3

4)不等式约束(字符串类型):<=,<=,<=

输入：

a11,a12,a13,a14,a15;a21,a22,a23,a24,a25;a31,a32,a33,a34,a35;x1,x2,x3,x4,x5;b1,b2,b3;<=,<=,<=

输出描述：

true 或者 false, 最大差

示例 1

输入：

```
2.3,3,5.6,7,6;11,3,8.6,25,1;0.3,9,5.3,66,7.8;1,3,2,7,5;340,670,80.6;<=,<=,<=
```

输出：

```
false 458
```

说明：

示例 2

输入：

```
2.36,3,6,7.1,6;1,30,8.6,2.5,21;0.3,69,5.3,6.6,7.8;1,13,2,17,5;340,67,300.6;<=,>=,<=
```

输出：

```
false 758
```

```
#include <bits/stdc++.h>
using namespace std;

int findSymbolIndex (const string& input) {
    for (int i = input.size()-1; i >=0; i--) {
        if (input[i] == ';') return i;
    }
    return -1;
}

void getSymbols(const string& input, int symbol_index, vector<string> &symbols) {
    string symbol = "";
    for (int i = symbol_index+1; i < input.size(); i++) {
        if (input[i] == ',') {
            symbols.push_back(symbol);
            symbol = "";
            continue;
        }
        symbol.push_back(input[i]);
    }
    symbols.push_back(symbol);
}

int getNumVar(const string& input) {
    int result = 1;
    for (auto c : input) {
        if (c == ',') result++;
        if (c == ';') break;
    }
    return result;
}

void getAll(const string& input, int num_func, int num_var,
        vector<vector<double>> &paras,
        vector<double> &vars,
        vector<double> &values) {
    paras.reserve(num_func);
    vars.reserve(num_var);
    values.reserve(num_func);
    for (int i=0; i< num_func; i++) {
        vector<double> para;
        para.reserve(num_var);
        for (int j=0; j < num_var; j++) {
            para.push_back(0.0);
        }
        paras.push_back(para);
    }
    for (int i=0; i< num_var; i++) {
        vars.push_back(0.0);
    }
    for (int i=0; i< num_func; i++) {
        values.push_back(0.0);
    }

    double t=0.0;
    bool point=false;
    int point_num=1;
    int t_num = 0;
    for (auto c : input) {
        if (c >= '0' && c <= '9') {
            double tt = c-'0';
            if (point) {
                t += tt / pow(10, point_num);
                point_num++;
            } else {
                t *= 10;
                t += tt;
            }
        } else if (c == '.') {
            point=true;
        } else {
            if (t_num < num_func * num_var) {
                int t_func = t_num/num_var;
                int t_var = t_num % num_var;
                paras[t_func][t_var] = t;
            } else if (t_num < num_func * num_var + num_var) {
                int t_var = t_num - num_func * num_var;
                vars[t_var] = t;
            } else {
                int t_func = t_num - num_func * num_var - num_var;
                values[t_func] = t;
            }
            t=0.0;
            point=false;
            point_num=1;
            t_num++;
            if (t_num == num_func * num_var + num_var + num_func) return;
        }
    }
}
```

```
int main() {
        vector<vector<double>> paras;
        vector<double> vars;
        vector<double> values;

        string input;
        cin >> input;
        int symbol_index = findSymbolIndex(input);

        vector<string> symbols;
        getSymbols(input, symbol_index, symbols);
        int num_func = symbols.size();
        int num_var = getNumVar(input);
        getAll(input, num_func, num_var,
                paras,
                vars,
                values);

        int max;
        bool tf=true;
        for (int i=0; i < num_func; i++) {
                double value = 0.0;
                for (int j=0; j < num_var; j++) {
                        value += paras[i][j] * vars[j];
                }
                if (i==0) {
                        max = value-values[i];
                } else {
                        if (value-values[i] > max) {
                                max = value-values[i];
                        }
                }

                if (symbols[i] == ">") {
                        if (!(value > values[i])) {
                                tf = false;
                        }
                } else if (symbols[i] == ">=") {
                        if (!(value >= values[i])) {
                                tf = false;
                        }
                } else if (symbols[i] == "<") {
                        if (!(value < values[i])) {
                                tf = false;
```

```
            }
        } else if (symbols[i] == "<=") {
            if (!(value <= values[i])) {
                tf = false;
            }
        } else if (symbols[i] == "=") {
            if (!(value == values[i])) {
                tf = false;
            }
        }
    }

    int max_int;
    if (max > 0) {
        max_int = floor(max);
    } else {
        max_int = ceil(max);
    }

    if (tf) cout << "true ";
    else cout << "false ";
    cout << max_int;
}
```