P1080 国王游戏

题目描述

恰逢 H 国国庆，国王邀请n位大臣来玩一个有奖游戏。首先，他让每个大臣在左、右手上面分别写下一个整数，国王自己也在左、右手上各写一个整数。然后，让这 n 位大臣排成一排，国王站在队伍的最前面。排好队后，所有的大臣都会获得国王奖赏的若干金币，每位大臣获得的金币数分别是：排在该大臣前面的所有人的左手上的数的乘积除以他自己右手上的数，然后向下取整得到的结果。

国王不希望某一个大臣获得特别多的奖赏，所以他想请你帮他重新安排一下队伍的顺序，使得获得奖赏最多的大臣，所获奖赏尽可能的少。注意，国王的位置始终在队伍的最前面。

输入格式

第一行包含一个整数n，表示大臣的人数。

第二行包含两个整数 a和 b，之间用一个空格隔开，分别表示国王左手和右手上的整数。

接下来 n行，每行包含两个整数a aa 和 bbb，之间用一个空格隔开，分别表示每个大臣左手和右手上的整数。

输出格式

一个整数，表示重新排列后的队伍中获奖赏最多的大臣所获得的金币数。

输入样例

```
3
1 1
2 3
7 4
4 6
```

输出样例

```
2
```

解析

需要注意弄清楚本题的题意：**“排在该大臣前面的所有人的左手上的数的乘积除以他自己右手上的数”**，这里是不包括他自己的。那么测试数据则可以表示如下：

按1,2,3排列

1	1		
2	3	1/3	0
7	4	2/4	0
4	6	14/6	2

按1,3,2排列

1	1		
2	3	1/3	0
4	6	2/6	0
7	4	8/4	2

……

若国王左手为a0，右手b0

大臣1：左手a1，右手b1

大臣2：左手a2，右手b2

则答案

$$ans_1 = max(\frac{a_0}{b_1}, \frac{a_0 * a_1}{b_2})$$

如果交换大臣1和大臣2，那么答案

$$ans_2 = max(\frac{a_0}{b_2}, \frac{a_0 * a_2}{b_1})$$

而最终的答案：$ans = min(ans_1, ans_2)$

显然有 $\frac{a_0 * a_2}{b_1} > \frac{a_0}{b_1}$，$\frac{a_0 * a_1}{b_2} > \frac{a_0}{b_2}$

所以 $ans = min(\frac{a_0 * a_2}{b_1}, \frac{a_0 * a_1}{b_2})$

如果交换两个大臣的位置决策更优，那么 $\frac{a_0 * a_2}{b_1} < \frac{a_0 * a_1}{b_2}$

两边同时约掉a0，最终则有：$a_1 * b_1 > a_2 * b_2$

将示例数据按照如上规则进行排序后，再进行计算，则有：

1	1	积	金币	商
2	3	6	1/3	0
4	6	24	2/6	0
7	4	28	8/4	2

因此，最优的排序方式即为：以a*b为关键字将大臣从小到大排序即可，然后选取全局最大值即为答案。注意本题数据结果较大，全程都应该使用高精度进行计算。

编码

```
#include <bits/stdc++.h>

using namespace std;

//输入n个数据;
int n;
//乘法最终结果的长度
int lens = 1;
//除法最终结果的长度
int lena = 1;
//最终结果的长度
int lenm = 1;
//乘法的结果数组
int sum[10010] = {0, 1};
//除法的数据结果
int ans[10010];
//最终的数据结果
int maxn[10010] = {0, 1};


//定义结构体数据
struct tmp {
    //定义大臣左边的数据和右边的数据
    int l, r;

    //重载小于计算
    bool operator<(const tmp x) const {
        return l * r < x.l * x.r;
    }
} coin[1001];

//高精度乘法
void muti(int x) {
    int tmp = 0;
```

```
    for (int i = 1; i <= lens; i++) {
        sum[i] *= x;
    }
    for (int i = 1; i <= lens; i++) {
        tmp += sum[i];
        sum[i] = tmp % 10;
        tmp /= 10;
    }
    while (tmp != 0) {
        lens++;
        sum[lens] = tmp % 10;
        tmp /= 10;
    }
}

//高精度除法
void cut(int x) {
    memset(ans, 0, sizeof(ans));
    lena = lens;
    int tmp = 0;
    for (int i = lena; i >= 1; i--) {
        tmp *= 10;
        tmp += sum[i];
        if (tmp >= x) {
            ans[i] = tmp / x;
            tmp %= x;
        }
    }
    while (ans[lena] == 0) {
        if (lena == 1) {
            break;
        }
        lena--;
    }
}

//计算全局最大值
void max() {
    //新的结果的长度大于全局最大值，则进行更新
    if (lena > lenm) {
        for (int i = 1; i <= lena; i++) {
            maxn[i] = ans[i];
        }
        lenm = lena;
    }
    //当计算出的结果位数等于当前全局最大值的时候
    else if (lena == lenm) {
        for (int i = lena; i >= 1; i--) {
            //从最高位开始，如果找到当前的全局最大值小于新值
```

```
            //则用新值进行替换
            if (maxn[i] < ans[i]) {
                for (int j = 1; j <= lena; j++) {
                    maxn[j] = ans[j];
                }
                lenm = lena;
                break;
            }
        }
    }
}

int main() {
    cin >> n;
    //优先读入国王的数据
    cin >> coin[0].l >> coin[0].r;
    //再读入大臣的数据
    for (int i = 1; i <= n; i++) {
        scanf("%d %d", &coin[i].l, &coin[i].r);
    }
    //按照l*r的结果从小到大排序所有大臣的数据
    sort(coin + 1, coin + n + 1);
    //开始进行计算
    for (int i = 1; i <= n; i++) {
        //计算乘法，当前大臣之前的所有人左手的积
        muti(coin[i - 1].l);
        //计算除法，当前大臣能得到的钱
        cut(coin[i].r);
        //在所有最大值当中选择一个最小的
        max();
    }
    //高精度逆序输出
    for (int i = lenm; i >= 1; i--) {
        cout << maxn[i];
    }
}
```