1218：取石子游戏

题目描述

有两堆石子,两个人轮流去取。每次取的时候,只能从较多的那堆石子里取,并且取的数目必须是较少的那堆石子数目的整数倍，最后谁能够把一堆石子取空谁就算赢。

比如初始的时候两堆石子的数目是25和7。
25 7 --> 11 7 --> 4 7 --> 4 3 --> 1 3 --> 1 0
选手1取　选手2取　选手1取　选手2取　选手1取

最后选手1（先取的）获胜，在取的过程中选手2都只有唯一的一种取法。
给定初始时石子的数目，如果两个人都采取最优策略，请问先手能否获胜。

输入

输入包含多数数据。每组数据一行，包含两个正整数a和b，表示初始时石子的数目。

输入以两个0表示结束。

输出

如果先手胜，输出"win"，否则输出"lose"。

输入样例

```
34 12
15 24
0 0
```

输出样例

```
win
lose
```

提示

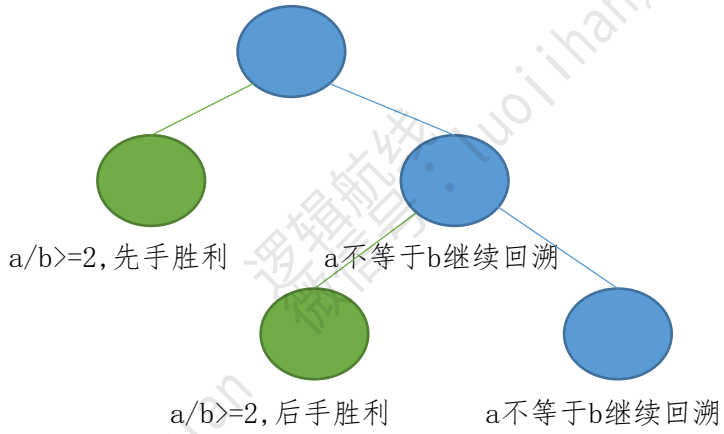
假设石子数目为(a,b)且a >= b,如果[a/b] >= 2则先手必胜,如果[a/b]<2,那么先手只有唯一的一种取法。[a/b]表示a除以b取整后的值。

解析

根据题目描述以及提示的内容，我们可以很容易的判断出来先遇到a/b>=2的那个人必然胜利。因此，整个决策树就两种情况：

1、大于等于2——胜利
2、继续模石子

下面，我们来构造决策树：



那么，为什么先达到a/b>=2的情况的人会胜利呢？我们来看分析：

第一种情况：

初始　7　4

先手取4，剩余 3　4

后手取3，剩余 3　1

先手取3，剩余 0　1

先手胜利。

第二种情况：

初始　11　4

先手取4，剩余 7　4

后手取4，剩余 3　4

先手取3，剩余 3　1

后续去3，剩余 0　1

后手胜利。

但是，当面对这种情况的时候，先手会取4吗？因为双方都足够聪明，所以，必然不会啊，先手一定会先取8，来保证自己必胜。

我们继续扩大初始值，将第一堆增加到15、19甚至更多的时候，你会发现先手总是能够一步到达第一种情况。15时可以一次拿12,19时可以一次拿16，这样只要第一堆数量是第二堆的两倍以上，先手就是必胜的。

当然，如果在最初的时刻，如果两堆补的比例小于2的时候，我们就只能看谁的运气好，先拿到2倍的堆数了。

编码

根据决策树，我们可以很清晰的构造出代码逻辑

```
#include<bits/stdc++.h>
using namespace std;
bool Dfs(int a, int b) {
    //优先达到条件者获胜
    if (a / b >= 2 || a == b) {
        return true;
    } else {
        //继续回溯，如果后手先遇到，则后手胜利
        return !Dfs(b, a - b);
    }
}
int main() {
    //定义输入的a，b两堆石子
    int a, b;
    //持续获得输入
    while ((cin >> a >> b) && !(a == 0 && b == 0)) {
        //保证多的石子在前
        if (b > a) {
            swap(a, b);
        }
        //调用回溯，直到遇到第一次a/b>=2
        if (Dfs(a, b))
            cout << "win" << endl;
        else
            cout << "lose" << endl;
    }
    return 0;
}
```