题目大意：

输入n，得到编号为0~n-1的木块，分别摆放在顺序排列编号为0~n-1的位置。现对这些木块进行操作，操作分为四种。

1、move a onto b：把木块a、b上的木块放回各自的原位，再把a放到b上；

2、move a over b：把a上的木块放回各自的原位，再把a放到含b的堆上；

3、pile a onto b：把b上的木块放回各自的原位，再把a连同a上的木块移到b上；

4、pile a over b：把a连同a上木块移到含b的堆上。

当输入quit时，结束操作并输出0~n-1的位置上的木块情况

#include<iostream>

#include<algorithm>

#include<vector>

#define maxn 30

using namespace std;

int n;

vector<int> pile[maxn];

void find\_block(int a,int& p,int& h)//找木块a所在的pile和height，以引用的形式返回调用者

{

for(p=0;p<n;p++)

for(h=0;h<pile[p].size();h++)

if(pile[p][h]==a) return;

}

void clear\_above(int p,int h)//把第p堆高度为h的木块上方的所有木块移回原位

{

for(int i=h+1;i<pile[p].size();i++)

{

int b=pile[p][i];

pile[b].push\_back(b);

}

pile[p].resize(h+1);

}

void pile\_onto(int p,int h,int p2)

{

for(int i=h;i<pile[p].size();i++)

pile[p2].push\_back(pile[p][i]);

pile[p].resize(h);

}

int main()

{

// freopen("input.txt","r",stdin);

int a,b;

cin>>n;

string s1,s2;

for(int i=0;i<n;i++)

pile[i].push\_back(i);

while(cin>>s1>>a>>s2>>b)

{

int pa,pb,ha,hb;

find\_block(a,pa,ha);

find\_block(b,pb,hb);

if(pa==pb)continue;

if(s2=="onto") clear\_above(pb,hb);

if(s1=="move") clear\_above(pa,ha);

pile\_onto(pa,ha,pb);

}

for(int i=0;i<n;i++)

{

cout<<i<<":";

for(int j=0;j<pile[i].size();j++)

cout<<' '<<pile[i][j];

cout<<"\n";

}

return 0;

}

Vector是一个不定长数组

若a是一个vector，可以用a.size()读取它的大小，a.resize()改变大小，

1. push\_back()像尾部添加元素，a.pop\_back()删除最后一个元素

push\_back完之后a的长度就会自动加1

vector看上去像是“一等公民”，因为他们可以直接赋值，还可以作为函数的参数或者返回值，而无需像传递数组那样另外用一个变量指定元素个数

vector也是第一个元素为0

同样可以a[x][y]这样二维，只不过区别于普通数组的是它的长度随着变量大小而变