迭代加深算法

埃及分数问题。在古埃及，人们使用单位分数的和（即1/a，a是自然数）表示一切有理数。例如，2/3=1/2+1/6，但不允许2/3=1/3+1/3，因为在加数中不允许有相同的。对于一个分数a/b，表示方法有很多种，其中加数少的比加数多的好，如果加数个数相同，则最小的分数越大越好。例如，19/45=1/5+1/6+1/18是最优方案。

输入整数a,b(0<a<b<500)a,b(0<a<b<500)，试编程计算最佳表达式。

样例输入：

495 499

样例输出：

Case 1: 495/499=1/2+1/5+1/6+1/8+1/3992+1/14970

#include <iostream>

#include <cstring>

#include <algorithm>

using namespace std;

typedef long long LL;

const int maxNum = 100 + 5;

LL a, b, maxd;

// 当前解和最终解

LL v[maxNum], ans[maxNum];

// 返回满足 1/c <= a/b 的最小c

inline int get\_first(LL a, LL b) {

return b / a + 1;

}

// 欧几里得算法

LL gcd(LL a, LL b) {

return b == 0 ? a : gcd(b, a % b);

}

// 如果当前解v比最优解ans更优，更新ans

bool better(int d) {

for(int i = d; i >= 0; i--) {

if(v[i] != ans[i]) {

return ans[i] == -1 || v[i] < ans[i];

}

}

return false;

}

// 当前深度为d,分母不能小于from,分数之和恰好为aa/bb

bool dfs(int d, int from, LL aa, LL bb) {

if(d == maxd) {

// aa/bb必须是埃及分数

if(bb % aa) {

return false;

}

// 深度为d时的分母

v[d] = bb / aa;

if(better(d)) {

memcpy(ans, v, sizeof(LL) \* (d + 1));

}

return true;

}

bool ok = false;

// 枚举的起点

from = max(from, get\_first(aa, bb));

for(int i = from; ; i++) {

// 剪枝：如果剩下的max+1-d个分数全部都是1/i,

// 加起来仍然不超过aa/bb，则无解

if(bb \* (maxd + 1 - d) <= i \* aa) {

break;

}

v[d] = i;

// 计算aa/bb - 1/i，设结果为a2/b2

LL b2 = bb \* i;

LL a2 = aa \* i - bb;

// 获取最大公约数

LL g = gcd(a2, b2);

if(dfs(d + 1, i + 1, a2 / g, b2 / g)) {

ok = true;

}

}

return ok;

}

int main() {

int kase = 0;

while(cin >> a >> b) {

int ok = 1;

// 迭代加深

for(maxd = 1; maxd <= 100; maxd++) {

memset(ans, -1, sizeof(ans));

if(dfs(0, get\_first(a, b), a, b)) {

ok = 1;

break;

}

}

cout << "Case " << ++kase << ": ";

if(ok) {

cout << a << "/" << b << "=";

for(int i = 0; i < maxd; i++) {

cout << "1/" << ans[i] << "+";

}

cout << "1/" << ans[maxd] << endl;

} else {

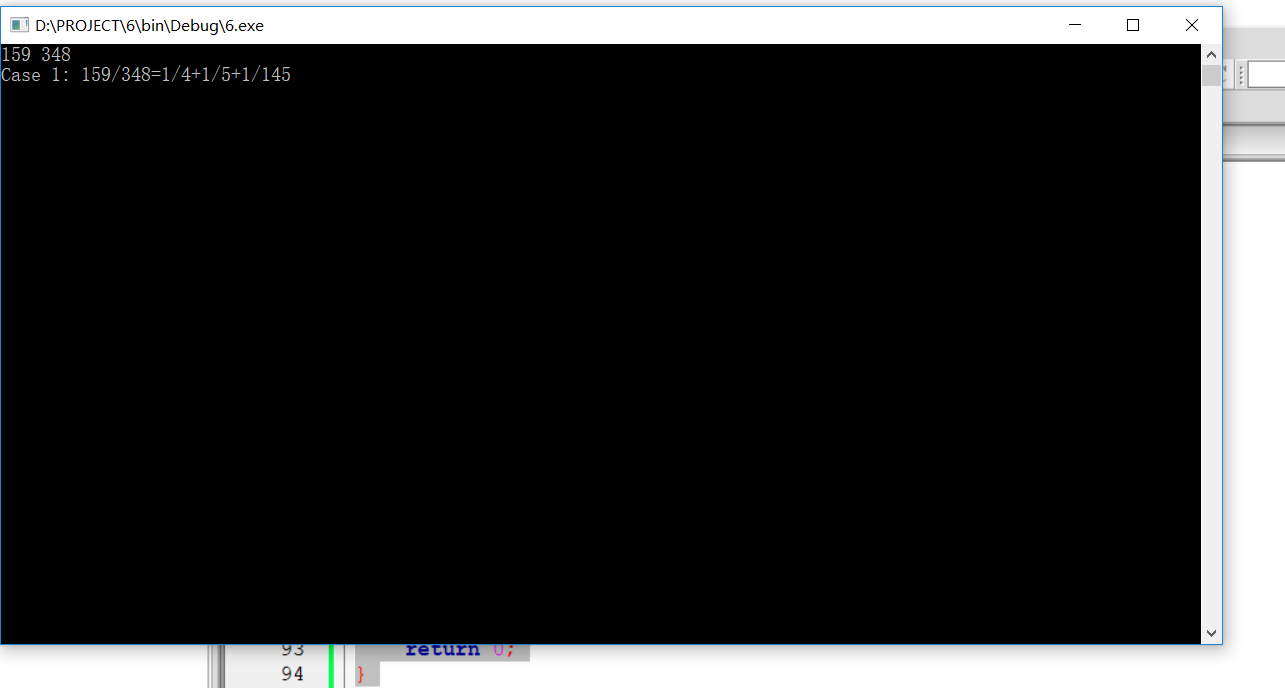
cout << "No solution." << endl;

}

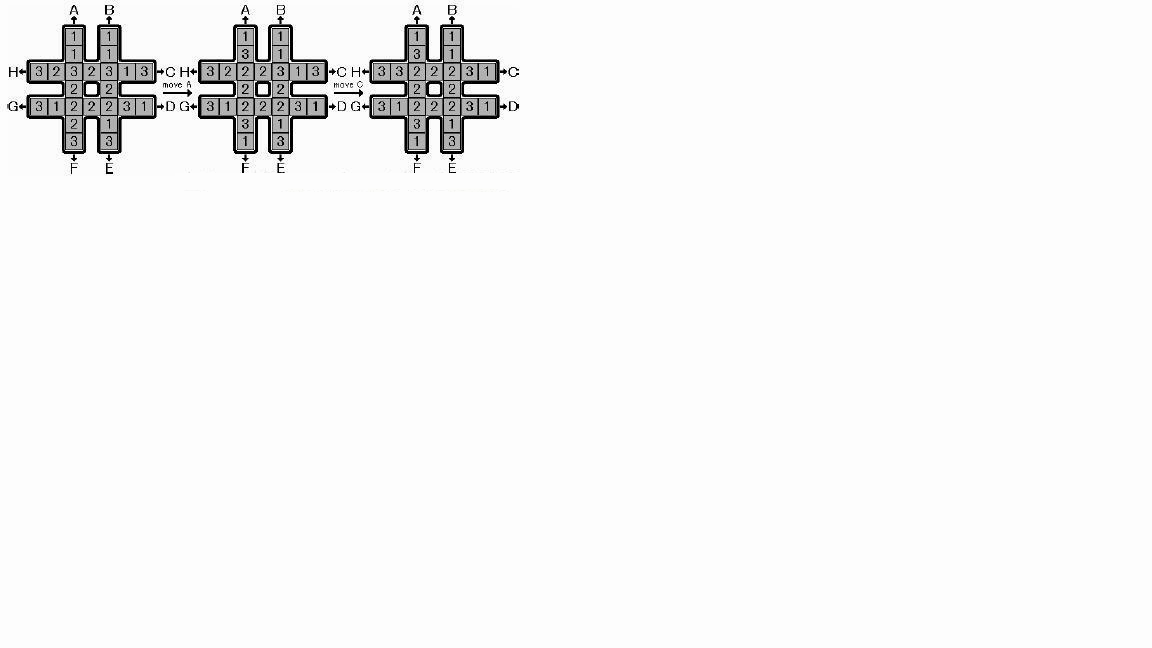
}

return 0;

}



给你一个如图所示地棋盘，上面有24个格子，每个格子上的数字为1，2或3。你可以向8个方向旋转棋盘，一次移动一格，8个方向为A-H编号。求最小步骤是的中间8个数字相同，若有多个答案输出字典序最小地答案。



#include<iostream>

using namespace std;

int step[10][10]={//代表A-H八种操作在读入的数字种的位置

{ 0, 2, 6,11,15,20,22},// A

{ 1, 3, 8,12,17,21,23},// B

{10, 9, 8, 7, 6, 5, 4},// C

{19,18,17,16,15,14,13},// D

{23,21,17,12, 8, 3, 1},//E

{22,20,15,11, 6, 2, 0},//F

{13,14,15,16,17,18,19},//G

{ 4, 5, 6, 7, 8, 9,10},//H

};

int index[9]={6,7,8,11,12,15,16,17};//中心位置的方块用于检查

int map[30];

char ans[100];

int check(){//检查中心的八个方块，是否数字相同

for(int i=0;i<8;i++){

if(map[index[0]]!=map[index[i]])

return 0;

}

return 1;

}

int move(int loc){//头位数字放到尾部，其他依次挪一格

int head=map[step[loc][0]];

for(int i=0;i<6;i++){

map[step[loc][i]]=map[step[loc][i+1]];

}

map[step[loc][6]]=head;

return 0;

}

int pre(){//估价函数

int most=100;

for(int i=1;i<=3;i++){

int k=0;

for(int j=0;j<8;j++)

if(map[index[j]]!=i)

k++;

most=min(most,k);

}

return most;

}

int dfs(int d,int dm ){

if(d==dm)

return check();

if(d+pre()>dm)

return 0;

for(int i=0;i<8;i++){

ans[d]=i+'A';

move(i);

if(dfs(d+1,dm)){

return 1;

}

if(i%2==0){//撤销刚才的操作

move((i+5)%8);

}

else

move((i+3)%8);

}

return 0;

}

int main(){

while(cin>>map[0]&&map[0]!=0){

for(int i=1;i<24;i++){

cin>>map[i];

}

// for(int i=0;i<24;i++){

// cout<<map[i];

// }

// cout<<endl;

if(check()){

cout<<"No moves needed"<<endl;

}

else{

int i=1;

while(!dfs(0,i)){

i++;

}

ans[i]='\0';

cout<<ans<<endl;

}

cout<<map[index[0]]<<endl;

}

return 0;

}

