**迭代加深**

**迭代加深算法思想：**

就是在深度无上限的情况下，先预估一个深度（尽量小）进行搜索，如果没有找到解，再逐步放大深度搜索。这种方法虽然会导致重复的遍历 某些结点，但是由于搜索的复杂度是呈指数级别增加的，所以对于下一层搜索，前面的工作可以忽略不计；

总的来说，迭代加深搜索是在速度上接近广度优先搜索，空间上和深度优先搜索相当的搜索方式。由于在使用过程中引入了深度优先搜索，所以也可以当作深度优先搜索的优化方案。迭代加深搜索适用于当搜索深度没有明确上限的情况。

****前提：**题目一定要有解，否则会无限循环下去**。****

****剪枝操作:****

**使用迭代加深搜素时没有剪枝操作时，时间开销是非常大的（因为迭代加深搜索是通过限制每次dfs的最大深度进行的搜索。令maxd表示最大的搜索深度，那么dfs就只能在0~maxd之间来进行，如果在这个范围内找到了解，就退出大循环，否则maxd++，扩大搜索范围。所以可想而知，倘若没有高效及时的退出无解的情况，那么时间上的开销也是会比较大的。）**

****乐观估计函数：****从当前深度到找到最终的解“至少”还需要多少步，或者距离找到最终的解还需要扩展多少层。如果超出了当前限制的深度maxd，说明当前限制的最大深度下是不可能找到解的，直接退出。

**迭代加深算法实现（java）：**

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

**using namespace std;**

**bool flag=false;**

**char finop[20];**

**int reverseop[9]={5,4,7,6,1,0,3,2,-1};**

**int mp[24];**

**int pos[8][7]=**

**{**

**{ 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 center[8]={6,7,8,11,12,15,16,17};**

**void option(int op)**

**{**

**int temp=mp[pos[op][0]];**

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

**{**

**mp[pos[op][i]]=mp[pos[op][i + 1]];**

**}**

**mp[pos[op][6]]=temp;**

**}**

**bool cmp(int a,int b)**

**{**

**return a>b;**

**}**

**int get\_h()**

**{**

**int cnt[3]={0,0,0};**

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

**{**

**cnt[mp[center[i]] - 1]++;**

**}**

**sort(cnt,cnt+3,cmp);**

**return 8-cnt[0];**

**}**

**void dfs(int depth,int lastop,int maxdepth)**

**{**

**if(flag)**

**return;**

**if(depth>maxdepth||depth+get\_h()>maxdepth)**

**return;**

**if(get\_h()==0)**

**{**

**flag=true;**

**finop[depth]='\0';**

**cout<<finop<<endl;**

**cout<<mp[center[0]]<<endl;**

**return;**

**}**

**for(int nextop=0;nextop<8;nextop++)**

**{**

**if(nextop!=reverseop[lastop])**

**{**

**option(nextop);**

**finop[depth]=nextop+'A';**

**dfs(depth+1,nextop,maxdepth);**

**option(reverseop[nextop]);**

**}**

**}**

**}**

**int main()**

**{**

**while(1)**

**{**

**for(int i=0;i<24;i++)**

**{**

**cin>>mp[i];**

**if(i==0)**

**{**

**if(mp[i]==0)**

**return 0;**

**}**

**}**

**if(!get\_h())**

**{**

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

**cout<<mp[center[0]]<<endl;**

**continue;**

**}**

**else**

**{**

**flag=false;**

**for(int maxdepth=1;!flag;maxdepth++)**

**dfs(0,8,maxdepth);**

**}**

**}**

**return 0;**

**}**