IDA*

简介&思想

迭代加深：有时候广搜太多深搜也太多，主要在于枝数太多、深度太深，因此枚举深搜的深度并不断深搜。
时间复杂度由于搜索树的性质只与最深一层有关。
IDA*迭代加深+估价函数（估计深度）

例题

骑士精神

来源：洛谷p2324

概述：5×5的棋盘上有12个黑棋子、12个白棋子、一个空位，棋子只能按规则（和中国象棋的马一样）走，并且只能走到空位。存在一种规定的目标形态，计算15步以内能到达目标形态的最少步数，否则输出-1。

答案dfs+估价函数（统计和目标形态不同的位置个数）；注意棋子移动的细节。

```
#include<cstdio>
#include<string>
#include<cstring>
#include<map>
using namespace std;
int T,ans;
int dp[8][25]={{-1,-1,-1,-1,-1, -1,-1,0,1,2, -1,-1,5,6,7 ,-1,-1,10,11,12
,-1,-1,15,16,17},{-1,-1,-1,-1,-1, -1,-1,-1,-1,-1, -1,0,1,2,3, -1,5,6,7,8,
-1,10,11,12,13},{-1,-1,-1,-1,-1, -1,-1,-1,-1,-1, 1,2,3,4,-1, 6,7,8,9,-1,
11,12,13,14,-1},{-1,-1,-1,-1,-1, 2,3,4,-1,-1, 7,8,9,-1,-1, 12,13,14,-1,-1,
17,18,19,-1,-1},{7,8,9,-1,-1, 12,13,14,-1,-1, 17,18,19,-1,-1,
22,23,24,-1,-1, -1,-1,-1,-1,-1},{11,12,13,14,-1, 16,17,18,19,-1,
21,22,23,24,-1, -1,-1,-1,-1,-1, -1,-1,-1,-1,-1},{-1,10,11,12,13,
-1,15,16,17,18, -1,20,21,22,23, -1,-1,-1,-1,-1,
-1,-1,-1,-1,-1},{-1,-1,5,6,7, -1,-1,10,11,12, -1,-1,15,16,17,
-1,-1,20,21,22, -1,-1,-1,-1,-1}};
string End="111110111100*110000100000";
map<string,int>vist,D,H;
void init()
{
    vist.clear();
    D.clear();
    H.clear();
    ans=16;
}
void dfs(string s,int d,int h,int f,int p)
{
    if(s==End)
```

```
    {
        ans=min(ans,d);
        return;
    }
    if(f>=ans)
        return;
    d++;
    for(int i=0;i<8;i++)
    {
        int p_=dp[i][p];
        if(p_!=-1)
        {
            string s_=s;
            s_[p]=s_[p_],s_[p_]='*';
            if(!vist[s_])
            {
                vist[s_]=1;
                int h_=h-(s[p]!=End[p])-
(s[p_]!=End[p_])+(s_[p]!=End[p])+(s_[p_]!=End[p_]);
                D[s_]=d,H[s_]=h_;
                dfs(s_,d,h_,d+(h_*4/5),p_);
            }
            else if(d<D[s_])
            {
                D[s_]=d;
                dfs(s_,d,H[s_],d+(H[s_]*4/5),p_);
            }
        }
    }
}
int main()
{
    scanf("%d",&T);
    while(T--)
    {
        init();
        string s;
        s.resize(25);
        for(int i=0;i<25;i+=5)
            scanf("%s",&s[i]);
        int h=0,f,p;
        for(int i=0;i<25;i++)
        {
            if(s[i]!=End[i])
                h++;
            if(s[i]=='*')
                p=i;
        }
        f=h*4/5;
        vist[s]=1;
```

```
        D[s]=0;
        H[s]=h;
        if(f<=15)
            dfs(s,0,h,f,p);
        printf(ans<16?"%d\n":"-1\n",ans);
    }
    return 0;
}
```

铁盘整理

来源：洛谷p2534

概述：有若干铁盘摞在一起，每次操作可以令底部若干铁盘不动，反转上面所有铁盘。铁盘具有不同的半径，求使所有铁盘半径从上到下递增的最少操作次数。

答案IDA*:迭代加深+估价函数（将铁盘大小离散，故目标状态相邻的铁盘大小相差为1；每次翻转只改变翻转与不动的边界处两个铁盘的大小关系，因此估价函数即统计相邻铁盘大小相差不为1的铁盘对数）

```
#include<cstdio>
#include<algorithm>
#include<map>
#include<cmath>
#define inf 0x3f3f3f3f
using namespace std;
int n,ans=inf,a[20];
struct unit{int r,p;}plate[20];
bool cmp(unit x,unit y)
{
    return x.r<y.r;
}
int H(int *s)
{
    int sum=0;
    for(int i=1;i<=n;i++)
        sum+=(abs(s[i+1]-s[i])!=1);
    return sum;
}
void turn(int l,int r)
{
    for(int i=l;i<=(l+r)>>1;i++)
        swap(a[i],a[l+r-i]);
}
void dfs(int dm,int d,int h,int last)
{
    if(!h)
    {
        ans=d;
        return;
    }
```

```
    if(d+h>dm)
        return;
    d++;
    for(int i=2,h_;i<=n;i++)
        if(i!=last)
        {
            turn(1,i);
            h_=H(a);
            dfs(dm,d,h_,i);
            turn(1,i);
            if(ans!=inf)
                return;
        }
}
int main()
{
    scanf("%d",&n);
    for(int i=1;i<=n;i++)
    {
        scanf("%d",&plate[i].r);
        plate[i].p=i;
    }
    sort(plate+1,plate+n+1,cmp);
    for(int i=1;i<=n;i++)
        a[plate[i].p]=i;
    a[n+1]=n+1;
    int h_=H(a);
    for(int i=1;ans==inf;i++)
        dfs(i,0,h_,1);
    printf("%d",ans);
    return 0;
}
```

总结