T1 飞翔(fly)

题意

有一只菜鹰，从地图上 $(1,1)$ 飞到 (n,m) ,但是你只能在格子的边上飞 (就是曼哈顿距离) ，有些特殊格子能飞对角线，求最短距离

Solve

考场思路&&正解思路

本题咱写法和正解一样呢（

因为这个题的特殊飞行方式，这只傻鹰要么飞曼哈顿距离要么飞对角线

很容易就可以发现最短路的走法只会有向上向右和右上对角线三种走法

根据简单的数学原理，走一格对角线比用曼哈顿距离走一格短了 $(2-(2^{\frac{1}{1}}))\times 100m$

所以走对角线走的越多越好

想一想就可以发现这玩意不就是一发 LIS

如果我们把每一个特殊格子的 $point.y$ 赋值到 $date[point.x]$上

走最多的对角线 $\Longleftrightarrow$ 走在最长的上升子序列（严格）上

那么代码就很简单了

Code

```cpp
#include<cstdio>
#include<cstring>
#include<algorithm>
#include<iostream>
#include<vector>
using namespace std;
int n,m,k;
vector<int> date[100005];
int dp[100005];
int len;
```

```
struct POINT{int x,y;}way[1100];

bool CMP(POINT a,POINT b)

{

if (a.x!=b.x) return a.x<b.x;

return a.y>b.y;

}

int main()

{

memset(dp,0x3f,sizeof(dp));

scanf("%d %d %d",&n,&m,&k);

for (int i=1,x,y;i<=k;i++)

scanf("%d %d",&way[i].x,&way[i].y);

sort(way+1,way+1+k,CMP);

int pre=-1,cnt=0;

for (int i=1;i<=k;i++)

{

if (way[i].x==pre) way[i].x=cnt;

else

{

way[i].x=++cnt;

pre=way[i].x;

}

}

for(int i=1;i<=k;i++)

date[way[i].x].push_back(way[i].y);

for (int i=1,pos;i<=cnt;i++)

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

{
```

```
pos=lower_bound(dp+1,dp+1+k,date[i][j])-dp;

dp[pos]=date[i][j];

len=max(len,pos);

}

long double ans=(n+m)*100;

ans=ans-len*100*2+

(len*1.41421356237309504880168872420970*100);//注意精度

long long Ans1=floor(ans),Ans2=ceil(ans);

if (ans-Ans1<Ans2-ans) printf("%lld",Ans1);

else printf("%lld",Ans2);

return 0;

}
```

Ex

你以为这题就完事了吗，这破题如果你是直接把每一个特殊格子的 $point.y$ 赋值到 $date[point.x]$上，那么恭喜你无了，同一个 x 可以上可能有多个 y 和它对应

办法很简单，你且看上面程序的那个 $sort$

它是按照 x 的升序 y 的降序排的，这样就可以避免dp数组被同一个 x 上的数据重复更新，原理不明白建议重新温习LIS优化的本质

此题当然还可以拿最短路对着特殊点暴力建图过，复杂度与 LIS 是一样的

End

T2 拼图(puzzling)

题意

就是题目说的，拼好拼图

Solve

这题数据范围 $n \leq 5$,如此之小，爆搜几乎不加任何剪枝就能过

但是坑点很多就是了，咱也不好说什么

Code

```
/*
像这种题，多写函数方便调试，debug的输出写好看点有利于调试
*/
#include<cstdio>
#include<cstring>
#include<algorithm>
#include<iostream>
using namespace std;
int n,m,k;
struct BLOCK
{
int type;
int dir;
int n,m;
char Map[10][10];
BLOCK()
{type=dir=0,memset(Map,0,sizeof(Map));}
void print()
{
for (int i=1;i<=n;i++)
{
for(int j=1;j<=m;j++)
cout<<Map[i][j];
cout<<'\n';
}
cout<<'\n';
}
```

```
}block[10];

int len,size;

int Map[15][15];

inline bool check(int x,int y,int pos)

{

for (int i=x;i<x+block[pos].n;i++)

for(int j=y;j<y+block[pos].m;j++)

if (Map[i][j]&&block[pos].Map[i-x+1][j-y+1]=='1') return false;

return true;

}

inline void col(int x,int y,int pos)

{

for (int i=x;i<x+block[pos].n;i++)

for(int j=y;j<y+block[pos].m;j++)

if (block[pos].Map[i-x+1][j-y+1]=='1') Map[i][j]=pos;

}

inline void recol(int x,int y,int pos)

{

for (int i=x;i<x+block[pos].n;i++)

for(int j=y;j<y+block[pos].m;j++)

if (block[pos].Map[i-x+1][j-y+1]=='1') Map[i][j]=0;

}

bool use[10];

inline void debug()

{

for(int i=1;i<=len;i++)

{

for(int j=1;j<=len;j++)
```

```
cout<<Map[i][j];
cout<<'\n';
}
cout<<'\n';
}
void dfs(int x,int y)
{
if (y>len) {dfs(x+1,1);return ;}
if (x==len+1)
{
for(int i=1;i<=len;i++)
for (int j=1;j<=len;j++)
if (!Map[i][j]) return ;
for(int i=1;i<=len;i++)
{
for (int j=1;j<=len;j++)
cout<<Map[i][j];
cout<<endl;
}
exit(0);
}
for (int i=1;i<=n;i++)
{
if (use[i]) continue;
else if (check(x,y,i))
{
col(x,y,i);use[i]=true;
for (int j=i+1;j<=n;j++)
```

```
{
if (use[j]) continue;
if (check(x,y,j))
{
col(x,y,j);use[j]=true;
dfs(x,y);
recol(x,y,j);use[j]=false;
}
}
dfs(x,y+1);
recol(x,y,i);use[i]=false;
}
}
dfs(x,y+1);
}
int main()
{
cin>>n;
for(int t=1,x,y;t<=n;t++)
{
cin>>block[t].n>>block[t].m;
for (int i=1;i<=block[t].n;i++)
for (int j=1;j<=block[t].m;j++)
{
cin>>block[t].Map[i][j];
while (block[t].Map[i][j]!='0'&&block[t].Map[i][j]!='1')
cin>>block[t].Map[i][j];
if (block[t].Map[i][j]=='1') size++;
```

```
}
for (int i=block[t].n;i>=1;i--)
{
bool flag=true;
for (int j=1;j<=block[t].m;j++)
if (block[t].Map[i][j]=='1') {flag=false;break;}
if (!flag) break;
block[t].n--;
}
}
len=sqrt(size);
if (len*len!=size)
{
cout<<-1;
return 0;
}
dfs(1,1);
cout<<-1;
return 0;
}
```

T3 比赛(match)

题意

组建N支篮球队进行单循环比赛（任意两队之间比赛一场）

这N支球队需要安排一个比赛日程表

比赛分成N轮进行，每轮比赛都有N/2场比赛同时进行、并有一支球队轮空

即每支球队共参加N-1场比赛。

Solve

考场思路

暴力填表哇！

反正 $n \leq 499$,$\Theta(n^3)$ 完全可过

Code

```
#include<cstdio>
#include<cstring>
#include<algorithm>
#include<iostream>
using namespace std;
int n,m,k,T,Map[510][510];
bool book[510];
int temp[1000],head=1,tail;
int main()
{
cin>>n>>m>>T;
for(int i=1;i<=n;i++)
{
cin>>Map[m][i];
temp[++tail]=Map[Map[m][i]][i]=Map[m][i];
}
book[m]=true;
for(int tt=1,pos;tt<=n-1;tt++)
{
temp[++tail]=temp[head++];
bool flag=false;
for(int i=1;i<=n;i++)
```

```
{
if (book[i]) continue;
if (flag) break;
flag=true;pos=i;
for (int j=1;j<=n;j++)
if (Map[i][j])
if (Map[i][j]!=temp[head+j-1])
{
flag=false;
break;
}
}
for (int i=1;i<=n;i++)
Map[pos][i]=temp[head+i-1];
}
for (int i=1;i<=n;i++)
if (Map[i][T]==-1) cout<<i<<' ';
else cout<<Map[i][T]<<' ';
return 0;
}
```

正解

这是个找规律题，有式子可推，因为这所有球队的对场都是轮换的，就是说

一个球队的所有对场是 1 2 3

那下一个就是 2 3 1

下下个就是 3 1 2

完事，推式子就行了

Code

```
#include<cmath>
#include<iostream>
#include<vector>
#include<cstring>
using namespace std;
int n,m,T,l,date[500500],ans[50050];
int main()
{
cin>>n>>m>>T;
for(int i=1;i<=n;i++)
cin>>date[i],date[i+n]=date[i];
for(int i=1;i<=n;i++)
if(date[i]==m){l=i;break;}
for(int i=1;i<=n;i++)
ans[date[i]]=date[l+(T+n-i)%n];
for(int i=1;i<=n;i++) cout<<ans[i]<<' ';
return 0;
}
```

T4 羽毛(feather)

题意

给定n个集合的大小，问在相邻集合没有交集的情况下，并集的最小大小(1与n相邻)

Solve

本题是个结论题，可以有严格的数学证明然而我不会又能咋办捏

Code

```
#include<cstdio>
```

```cpp
#include<cstring>
#include<algorithm>
#include<iostream>
using namespace std;
int n,a[20005],ans,tot;
int main()
{
scanf("%d",&n);
for(int i=1;i<=n;i++)
{
scanf("%d",&a[i]);
ans=max(ans,a[i]+a[i-1]);
tot+=a[i];
}
ans=max(ans,a[1]+a[n]);
ans=max(ans,(tot+(n/2)-1)/(n/2));
printf("%d",ans);
return 0;
}
```