Dp模板题。用f[i][j]表示到达(i,j)的最优值，状态转移方程：

f[i][j]=min(f[i-1][j-1],min(f[i-1][j],f[i-1][j+1]))+a[i][j];

不过本题是环状的，考虑变环为链，将数组开两倍。不过本题的状态只会由相邻的状态转移而来，可以有更好的处理方法，遍历时对j==1和j==m的情况进行特判就行了。如下：

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

for(int j=1;j<=m;j++)

{

if(j==1) f[i][j]=min(f[i-1][m],min(f[i-1][1],f[i-1][2]));

else if(j==m) f[i][j]=min(f[i-1][m],min(f[i-1][m-1],f[i-1][1]));

else f[i][j]=min(f[i-1][j-1],min(f[i-1][j],f[i-1][j+1]));

f[i][j]+=a[i][j];

}

运行一下大数据，过了，然而……

输入：

3 1

1

1

1

输出：

1

正确输出：3

貌似程序出了点问题。

当m==1时，特判j==m，会使用f[i][0]的值，由于没更新，f[i][0]的值为0，导致最终结果出错(其实我当时已经想到了，准备改的，但是急着做题，后来给忘了)。

于是

if(m==1)

{

int ans=0;

for(int i=1;i<=n;i++) ans+=a[i][1];

return printf("%d",ans),0;

}

这下就好了。

完整代码：

#include<cstdio>

#include<algorithm>

#include<iostream>

#include<cstring>

using namespace std;

const int N=3e3+10,Inf=1e8;

int a[N+1][N+1];

int n,m;

long long f[N+1][N+1],ans=Inf;

int

main()

{

freopen("compress.in","r",stdin);

freopen("compress.out","w",stdout);

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

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

for(int j=1;j<=m;j++)

{

scanf("%d",&a[i][j]);

a[i][j+m]=a[i][j];

}

if(m==1)

{

int ans=0;

for(int i=1;i<=n;i++) ans+=a[i][1];

return printf("%d",ans),0;

}

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

for(int j=1;j<=m;j++)

{

if(j==1) f[i][j]=min(f[i-1][m],min(f[i-1][1],f[i-1][2]));

else if(j==m) f[i][j]=min(f[i-1][m],min(f[i-1][m-1],f[i-1][1]));

else f[i][j]=min(f[i-1][j-1],min(f[i-1][j],f[i-1][j+1]));

f[i][j]+=a[i][j];

}

for(int i=1;i<=m;i++) ans=min(ans,f[n][i]);

printf("%lld",ans);

return 0;

}

看来以后测试程序还是要多测试几组特殊数值啊。

貌似本题可以用循环数组避开讨论，但求模运算的速度实在不敢恭维。

关于讨论，可以使用三目运算符进行优化，但恐怕效果不会很明显。