对于n行输入，各行之间的独立的，只需要找到各行的最大值即可。它们的和即为所求。

DP[i][j]表示当前行i位置到j位置获得的最大分数

　　转移方程:

　　　　DP[i][j]=max(DP[i+1][j]+a[i]\*2^x,DP[i][j-1]+a[i]\*2^x);

由于数据较大需要用到高精。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | //矩阵取数游戏 (NOIP2007)  //动态规划 高精度  #include<stdio.h>  #include<algorithm>  #include<string.h>  using namespace std;  const int maxn=85;  const int maxm=85;  const int maxl=30;  int N,M;  int a[maxn];  int DP[maxn][maxm][maxl];  int ys[maxn][maxl];  int ans[maxl];  int tmp[maxl];  void Multi(int \*x,int \*y,int z)  {  memset(tmp,0,sizeof(tmp));  tmp[0]=y[0];  for(int i=1;i<=tmp[0];i++)  {  tmp[i]=y[i]\*z;  }  for(int i=1;i<=tmp[0];i++)  {  tmp[i+1]+=tmp[i]/100000;  tmp[i]%=100000;  }  while(tmp[tmp[0]+1])  {  tmp[0]++;  tmp[tmp[0]+1]+=tmp[tmp[0]]/100000;  tmp[tmp[0]]%=100000;  }  for(int i=0;i<30;i++)x[i]=tmp[i];  return ;  }  void Add(int \*x,int \*y,int \*z)  {  memset(tmp,0,sizeof(tmp));  if(y[0]>z[0])tmp[0]=y[0];  else tmp[0]=z[0];  for(int i=1;i<=tmp[0];i++)  {  tmp[i]=y[i]+z[i];  }  for(int i=1;i<=tmp[0];i++)  {  tmp[i+1]+=tmp[i]/100000;  tmp[i]%=100000;  }  if(tmp[tmp[0]+1])tmp[0]++;  for(int i=0;i<30;i++)x[i]=tmp[i];  return ;  }  bool Max(int \*x,int \*y)  {  if(x[0]>y[0])return 1;  else if(y[0]>x[0])return 0;  else  {  for(int i=x[0];i>=1;i--)  {  if(x[i]>y[i])return 1;  else if(x[i]<y[i])return 0;  }  }  return 1;  }  int main()  {  scanf("%d %d",&N,&M);  ys[0][0]=1;  ys[0][1]=1;  for(int i=1;i<=M;i++)  {  Multi(ys[i],ys[i-1],2);  }  ans[0]=1;ans[1]=0;  for(int i=1;i<=N;i++)  {  for(int j=1;j<=M;j++)  {  scanf("%d",&a[j]);  Multi(DP[j][j],ys[M],a[j]);  }  for(int k=2;k<=M;k++)  {  for(int x=1;x<=M-k+1;x++)  {  int y=x+k-1;  int xm[31],ym[31],cm[31];  Multi(cm,ys[M-k+1],a[x]);  Add(xm,cm,DP[x+1][y]);  Multi(cm,ys[M-k+1],a[y]);  Add(ym,cm,DP[x][y-1]);  if(Max(xm,ym))  {  memcpy(DP[x][y],xm,sizeof(int)\*30);  }  else  {  memcpy(DP[x][y],ym,sizeof(int)\*30);  }  }  }  Add(ans,ans,DP[1][M]);  }  printf("%d",ans[ans[0]]);  for(int i=ans[0]-1;i>=1;i--)  {  printf("%05d",ans[i]);  }  printf("\n");  return 0;  } |