**// 典型的动态规划,用递归下的记忆化搜索来实现 POJ 1088**

**// 状态转移方程 合法的情况下:DP(i,j) = max( DP(i,j-1), DP(i,j+1), DP(i-1,j), DP(i+1,j) ) + 1;**

**#include <iostream>**

**using namespace std;**

**int matrix[100][100];// 保存原始数据**

**int cnt[100][100]; // 记录每一个点的最大滑雪长度**

**int row ,col;**

**int DP(int i, int j)**

**{**

**int max = 0;**

**// 如果已经处理过,直接返回(记忆化搜索效率之所以高的原因:不重复计算)**

**if (cnt[i][j] > 0)**

**{**

**return cnt[i][j];**

**}**

**// 以下四块语句,只对合法的i和j,进行递归(递归的重点就是:剪去所有不合法的,只处理所有合法的!!!)**

**if (j-1 >= 0)**

**{**

**if (matrix[i][j] > matrix[i][j-1])**

**{**

**if (max < DP(i, j-1))**

**{**

**max = DP(i, j-1);**

**}**

**}**

**}**

**if (j+1 <= col-1)**

**{**

**if (matrix[i][j] > matrix[i][j+1])**

**{**

**if (max < DP(i, j+1))**

**{**

**max = DP(i, j+1);**

**}**

**}**

**}**

**if (i-1 >= 0)**

**{**

**if (matrix[i][j] > matrix[i-1][j])**

**{**

**if (max < DP(i-1, j))**

**{**

**max = DP(i-1, j);**

**}**

**}**

**}**

**// 在这里我曾经很SB地将row错写成col,调试所有的行数等于列数的数据都没有问题,可是一提交就Wa**

**// 注意,行数可能不等于列数!!!!**

**if (i+1 <= row-1)**

**{**

**if (matrix[i][j] > matrix[i+1][j])**

**{**

**if (max < DP(i+1, j))**

**{**

**max = DP(i+1, j);**

**}**

**}**

**}**

**// 将结果记录在cnt数组中(记忆化搜索的重点)**

**// 如果左右上下都没有一个点的值比这个点的值大,则cnt[i][j] = max+1 = 1**

**// 否则将左右上下各点最大滑雪长度记录在max中, 则cnt[i][j] = max+1**

**// 这就是max为什么要初始化为0的原因.**

**return cnt[i][j] = max + 1;**

**}**

**int main()**

**{**

**int i, j;**

**cin>>row>>col;**

**// 初始化数据**

**for (i=0; i<=row-1; i++)**

**{**

**for (j=0; j<=col-1; j++)**

**{**

**cin>>matrix[i][j];**

**cnt[i][j] == 0;**

**}**

**}**

**// 处理每一个点,将其最大滑雪长度保存在cnt数组里面**

**for (i=0; i<=row-1; i++)**

**{**

**for (j=0; j<=col-1; j++)**

**{**

**DP(i, j);**

**}**

**}**

**// 遍历数组,求最大值,在这里因为将cnt错写成matrix而wa了一次,真不应该!!!**

**for (i=0; i<=row-1; i++)**

**{**

**for (j=0; j<=col-1; j++)**

**{**

**if (cnt[0][0] < cnt[i][j])**

**{**

**cnt[0][0] = cnt[i][j];**

**}**

**}**

**}**

**cout<<cnt[0][0]<<endl;**

**return 0;**

**}**