序列(seq)

时间限制：1 Sec 内存限制：128 MB

DESCRIPTION

有一个长度为 n 的非负整数序列 $a_1,a_2,a_3,\ldots a_n$，每次操作可以选择相邻的两个数 a_i,a_{i+1}，删去它们，然后在这个位置插入一个数 $\max(a_i,a_{i+1})$，此次操作的代价定义为 $\max(a_i,a_{i+1})$，求将这个序列长度变为 1 的最少代价。

INPUT

第一行为一个正整数 n，表示序列的长度。
第二行有 n 个非负整数 $a_1,a_2,a_3,\ldots a_n$，表示这个序列。

OUTPUT

一行一个数，表示最少代价。

SAMPLE INPUT:

3
1 2 3

SAMPLE OUTPUT

5

HINT

对于 30%的数据, n<=10
对于 50%的数据, n<=100
对于 70%的数据, n<=1000
对于 90%的数据, n<=100,000
对于 100%的数据, n<=1,000,000

样例解释：先删去 1 和 2，加回 2，再删去 2 和 3，加回 3，总代价为 5

软件安装(install)

时间限制：1 Sec　内存限制：128 MB

Description

现在我们的手头有 N 个软件，对于一个软件 i，它要占用 W_i 的磁盘空间，它的价值为 V_i。我们希望从中选择一些软件安装到一台磁盘容量为 M 计算机上，使得这些软件的价值尽可能大（即 V_i 的和最大）。

但是现在有个问题：软件之间存在依赖关系，即软件 i 只有在安装了软件 j（包括软件 j 的直接或间接依赖）的情况下才能正确工作（软件 i 依赖软件 j)。幸运的是，一个软件最多依赖另外一个软件。如果一个软件不能正常工作，那么它能够发挥的作用为 0。

我们现在知道了软件之间的依赖关系：软件 i 依赖软件 D_i。现在请你设计出一种方案，安装价值尽量大的软件。一个软件只能被安装一次，如果一个软件没有依赖则 D_i=0，这时只要这个软件安装了，它就能正常工作。

Input

第 1 行：N, M （0<=N<=100, 0<=M<=500）
第 2 行：$W_1, W_2, ... W_i, ..., W_n$ （0<=W_i<=M ）
第 3 行：$V_1, V_2, ..., V_i, ..., V_n$ （0<=Vi<=1000 ）
第 4 行：$D_1, D_2, ..., D_i, ..., D_n$ （0<=D_i<=N, D_i≠i ）

Output

一个整数，代表最大价值。

Sample Input

```
3 10

5 5 6

2 3 4

0 1 1
```

Sample Output

```
5
```

Hint

数据范围及约定

测试点	n	m	v	约定
1,2	<=20	<=500	<=1000	无
3,4	<=100	<=500	<=1000	Di 均为 0
5,6,7	<=100	<=500	<=1000	依赖关系不构成环
8,9,10	<=100	<=500	<=1000	无

图书管理员（book）

时间限制: 1 Sec 内存限制: 128 MB

题目描述

图书馆里有 N 个书架，每个书架可以放 M 本书。小 C 是一位优秀的图书管理员，所以他决定整理图书馆中的所有图书，如果可以的话，把所有图书放回它们原来的位置。他打算用下面的方法来整理图书：

1.如果书架上的某本书的左边或右边是空的，就可以把这本书书向左或向右移动。

2.把一本书从书架上拿起来，再放回到另一个空的位置（可以是任意一个书架）。

可是没过多久小 C 就累了。如果他手上已经有了一本书，他就不能再移动其他的书。而且因为把书拿出来又放回去实在太麻烦了，所以小 C 希望尽可能少地将书从书架上拿起来（即步骤 2）。请帮小 C 算出，他最少需要将书拿出来多少次，即步骤 2 的操作次数。

输入

输入第一行，两个整数 N 和 M (1 <= N,M <= 1000)。

接下来 N 行，第 i+1 行包含 M 个整数，表示第 i 个书架中原本书的摆放位置。

再接下来 N 行，第 i+N+1 行包含 M 个整数，表示第 i 个书架整理完后所有书的正确位置。

所有的书都用 1 到 K 进行编号，K 为书的数量。数字 0 表示这个位置是空的。如果数字不为 0，则放有某一本书，数字表示书的编号。每一本书都会在开始和最终情况中出现。

输出

输出仅一行，一个整数表示最少需要将书拿出来的步数，如果不能将所有书摆会原来的位置，就输出-1。

样例

Input	Input
2 4 1 0 2 0 3 5 4 0 2 1 0 0 3 0 4 5	3 3 1 2 3 4 5 6 7 8 0 4 2 3 6 5 1 0 7 8
Output	**Output**
2	4

提示

对于第一组样例，小 C 会先将书 1 向右移一格，然后拿起书 2 将它放到第一个书架的第一个位置。再拿起书 5，并放到第二个书架的第四个位置。一共需要将书拿起来两次。

对于 30%的数据，每一本书在开始和结尾都位于同一个书架。

对于另外 30%的数据，N,M 不超过 300。

请使用效率较高的读入方法。