**第一题：**

终端公司的零售店需要定期去仓库提取货物，假设零售店和仓库在一个矩阵上：相邻点的距离为 1 ；只能通过上下左右走动。  
矩阵元素的值仅为三种：0，表示仓库； -1，表示障碍； 1，表示零售店。 注：障碍无法通过，其它可以通过。

为了将取货效率最大化，需要计算每个零售店走到最近仓库的最小距离，并输出这些最小距离的和：

* 无法到达仓库的零售店，不参与距离和的计算；
* 没有零售店或者没有仓库的话，返回0;

**解答要求**时间限制：3000ms, 内存限制：256MB

**输入**

第一行为两个数字 m 和 n，表示数组的行数和列数，m和n的范围均为 [1,300) 。  
接下来的 m 行表示一个 m\*n 的数组，每行的元素间以空格分割。

**输出**

一个整数，表示所计算的最小距离和。

**样例**

输入样例 1 复制

3 3

1 -1 0

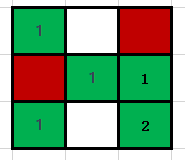
0 1 1

1 -1 1

输出样例 1

6

提示样例 1

如下图所示，共有5个零售店（绿色），2个仓库（红色），2个障碍（白色），零售店上标注的数字表示到最近仓库的最小距离：  
  
位置[2][2]的零售店，距离[0][2]的仓库为 2，距离[1][0]的仓库为 3，因此到最近仓库的最小距离为 2 ； 其余零售店到最近仓库的最小距离都为 1。所以，所有零售店到仓库的最小距离和为 1 + 1 + 1 + 1 + 2 = 6 。

输入样例 2 复制

2 3

0 -1 1

1 -1 1

输出样例 2

1

提示样例 2

位置[0][2]和[1][2]的零售店无法到达唯一的仓库[0][0]，只有[1][0]的零售店可以到达，且最近距离为 1 。

**第二题：**

软件攻城狮小E准备下班的时候，发现办公室只剩下他自己了，但是灯全部还开着，一方面他想关掉灯节省资源，可另一方面他又得赶班车。办公室的布局可用一个正方形矩阵表示，其中：  
0：表示空位，可以通过；  
1：表示桌子，不可通过；  
2：表示一盏灯（假设灯和开关一体的），可以通过；  
3：表示小E工位（只有一个），可以通过；  
4：表示办公室门口（只有一个），可以通过；

现给定办公室布局信息及赶班车所需的**最少**时间，请帮小E计算在不耽误赶班车的情况，**最多**来得及熄掉几盏灯；如果小E无论如何都赶不上班车，则输出-1。 假设：

* 小E每一秒可以向上/下/左/右移动一个单位。
* 最后到达办公室门口，出门口即认为可赶上班车。

**解答要求**时间限制：1000ms, 内存限制：256MB

**输入**

* 首行一个整数 timeLimit，表示赶班车所需的**最少**时间，即小E要在 timeLimit 秒内（小于或等于）赶到办公室门口（可经过而暂不出去），其值范围：[1, 20000]
* 第二行一个整数 size，表示办公室正方矩形的边长，其值范围：[2, 50]
* 随后 size 行，每行 size 个整数，表示办公室的布局， 值为0,1,2,3,4其中一个。（其中 2 所表示的灯的数量 小于等于 11）

**输出**

一个整数，代表小E最多能熄掉几盏灯；或者 -1。

**样例**

输入样例 1 复制

8

5

0 0 0 0 0

0 3 0 2 0

0 0 2 0 0

0 0 0 0 0

0 0 0 0 4

输出样例 1

2

提示样例 1

小E可以把两盏灯都关掉，用时刚好8秒（其中一条可行路径为：右->右->下->左->下->下->右->右）。

输入样例 2 复制

3

4

0 0 0 0

0 0 0 2

0 0 4 0

0 0 0 3

输出样例 2

0

提示样例 2

小E来不及关灯，但可以走到门口，故返回0。

输入样例 3 复制

10

3

3 1 2

0 1 0

0 1 4

输出样例 3

-1

提示样例 3

小E给3张桌子挡着过不去，无论如何都赶不上班车，故输出-1。

**第三题：**

某系统的进程可能占有和等待一些资源，现给出在某一时刻dump的这些进程占有和等待的资源信息，请按照如下简化规则分析哪些进程发生了死锁，请**升序**返回所有死锁的进程ID列表，或空列表 []。  
简化规则如下：

  如果某个进程 *P* 的**任一**等待资源被占有，则该进程必须等待，直到这些资源被释放； 等待期间，该进程不会释放所占有的资源。

  如果进程 *P* 所等待的资源全部都未被其他进程占有，则该进程**必将**释放所占有的资源。

基于上，如果某个进程因为所等待的资源一直被占有而无限等待下去，则认为该进程发生了死锁。

**输入**

第一行为一个整数num，表示进程个数；  
第二行开始的num行，依次表示每个进程占有和等待资源的情况，格式为：进程ID (占有资源列表) (等待资源列表)。

0 <= 进程ID <= 100, 0 <= 资源编号 <= 300  
每一行的输入中不会包含重复的资源编号

输入保证：每个资源最多**只会**被一个进程占有。

**输出**

升序返回所有死锁的进程ID列表，或空列表 []

**样例1**

输入：

5

100 () (20)

1 (40 20) (10)

2 (10) (30 100)

3 (100 300) (40 0)

0 () (30)

输出：

[1 2 3 100]

解释：

         进程1 等待被进程2 占有的资源10；进程2 等待被进程3 占有的资源100；进程3 等待被进程1 占有的资源40。这三个进程，都因为所等待的资源一直被占有而无限等待下去，所以这三个进程都发生了死锁。

         因为进程1 死锁，资源20 一直被占有，导致进程100 也死锁。

         进程0 等待的资源30 未被占有，因此进程0 不死锁。

**样例2**

输入：

4

2 () (40 30)

1 (20) (30 40)

3 () ()

9 (40) (30)

输出：

[]

解释：

对于每个进程，其所等待的资源全部都未被其他进程占有，不死锁。

样例3

输入：

6

1 (10) (20 50)

2 (20) (30 60)

3 (30) (40)

5 (50) ()

6 (101) (202)

7 (202) (101)

输出：

[6 7]

解释：

无