**NOIP2008 传球游戏（ball.pas/c/cpp）**

**【**问题描述】

上体育课的时候，小蛮的老师经常带着同学们一起做游戏。这次，老师带着同学们一起做传球游戏。

游戏规则是这样的：n个同学站成一个圆圈，其中的一个同学手里拿着一个球，当老师吹哨子时开始传球，每个同学可以把球传给自己左右的两个同学中的一个（左右任意），当老师再次吹哨子时，传球停止，此时，拿着球没传出去的那个同学就是败者，要给大家表演一个节目。

聪明的小蛮提出一个有趣的问题：有多少种不同的传球方法可以使得从小蛮手里开始传的球，传了m次以后，又回到小蛮手里。两种传球的方法被视作不同的方法，当且仅当这两种方法中，接到球的同学按接球顺序组成的序列是不同的。比如有3个同学1号、2号、3号，并假设小蛮为1号，球传了3次回到小蛮手里的方式有1->2->3->1和1->3->2->1，共2种。

【输入】

输入文件ball.in共一行，有两个用空格隔开的整数n，m（3<=n<=30，1<=m<=30）。

【输出】

输出文件ball.out共一行，有一个整数，表示符合题意的方法数。

【输入输出样例】

|  |  |
| --- | --- |
| ball.in | ball.out |
| 3 3 | 2 |

【数据范围】

40%的数据满足：3<=n<=30，1<=m<=20

100%的数据满足：3<=n<=30，1<=m<=30

**NOIP2009 细胞分裂 (cell.pas/c/cpp)**

**【**问题描述】

Hanks 博士是 BT (Bio-Tech，生物技术) 领域的知名专家。现在，他正在为一个细胞实验做准备工作：培养细胞样本。

Hanks 博士手里现在有 N 种细胞，编号从 1~N，一个第 i 种细胞经过 1 秒钟可以分裂为Si个同种细胞（Si为正整数）。现在他需要选取某种细胞的一个放进培养皿，让其自由分裂，进行培养。一段时间以后，再把培养皿中的所有细胞平均分入 M 个试管，形成 M 份样本，用于实验。Hanks 博士的试管数 M 很大，普通的计算机的基本数据类型无法存储这样大的M 值，但万幸的是，M 总可以表示为 m1的 m2次方，即M=m1^m2,其中 m1，m2均为基本数据类型可以存储的正整数。

注意，整个实验过程中不允许分割单个细胞，比如某个时刻若培养皿中有 4 个细胞，

Hanks 博士可以把它们分入 2 个试管，每试管内 2 个，然后开始实验。但如果培养皿中有 5个细胞，博士就无法将它们均分入 2 个试管。此时，博士就只能等待一段时间，让细胞们继续分裂，使得其个数可以均分，或是干脆改换另一种细胞培养。

为了能让实验尽早开始，Hanks 博士在选定一种细胞开始培养后，总是在得到的细胞“刚好可以平均分入 M 个试管”时停止细胞培养并开始实验。现在博士希望知道，选择哪种细胞培养，可以使得实验的开始时间最早。

【输入输出格式】

输入格式：

第一行有一个正整数 N，代表细胞种数。

第二行有两个正整数 m1，m2，以一个空格隔开，即表示试管的总数 M = m1^m2。

第三行有 N 个正整数，第 i 个数 Si表示第 i 种细胞经过 1 秒钟可以分裂成同种细胞的个数。

输出格式：

输出文件 cell.out 共一行，为一个整数，表示从开始培养细胞到实验能够开始所经过的最少时间（单位为秒）。

如果无论 Hanks 博士选择哪种细胞都不能满足要求，则输出整数-1。

【输入样例1】

1

2 1

3

【输出样例1】

-1

【输入样例2】

2

24 1

30 12

【输出样例2】

2

【输入输出说明】

经过 1 秒钟，细胞分裂成 3 个，经过 2 秒钟，细胞分裂成 9 个，……，可以看出无论怎么分裂，细胞的个数都是奇数，因此永远不能分入 2 个试管。

【输入输出样例2说明】

第 1 种细胞最早在3 秒后才能均分入24 个试管，而第2 种最早在2 秒后就可以均分（每试管144/(241)=6 个）。故实验最早可以在2 秒后开始。

【数据范围】

对于 50%的数据，有m1^m2 ≤ 30000。

对于所有的数据，有1 ≤N≤ 10000，1 ≤m1 ≤ 30000，1 ≤m2 ≤ 10000，1 ≤ Si ≤ 2,000,000,000。

**NOIP2010导弹拦截(missile.pas/c/cpp)**

【问题描述】

经过11 年的韬光养晦，某国研发出了一种新的导弹拦截系统，凡是与它的距离不超过其工作半径的导弹都能够被它成功拦截。当工作半径为0 时，则能够拦截与它位置恰好相同的导弹。但该导弹拦截系统也存在这样的缺陷：每套系统每天只能设定一次工作半径。而当天的使用代价，就是所有系统工作半径的平方和。

某天，雷达捕捉到敌国的导弹来袭。由于该系统尚处于试验阶段，所以只有两套系统投入工作。如果现在的要求是拦截所有的导弹，请计算这一天的最小使用代价。

【输入】

输入文件名missile.in。

第一行包含4 个整数x1、y1、x2、y2，每两个整数之间用一个空格隔开，表示这两套导弹拦截系统的坐标分别为(x1, y1)、(x2, y2)。

第二行包含1 个整数N，表示有N 颗导弹。接下来N 行，每行两个整数x、y，中间用一个空格隔开，表示一颗导弹的坐标(x, y)。不同导弹的坐标可能相同。

【输出】

输出文件名missile.out。

输出只有一行，包含一个整数，即当天的最小使用代价。

【提示】

两个点(x1, y1)、(x2, y2)之间距离的平方是(x1? x2)2+(y1?y2)2。

两套系统工作半径r1、r2 的平方和，是指r1、r2 分别取平方后再求和，即r12+r22。

【输入输出样例1】

|  |  |
| --- | --- |
| missile.in | missile.out |
| 0 0 10 0 | 18 |
| 2 |  |
| -3 3 |  |
| 10 0 |  |

【样例 1 说明】

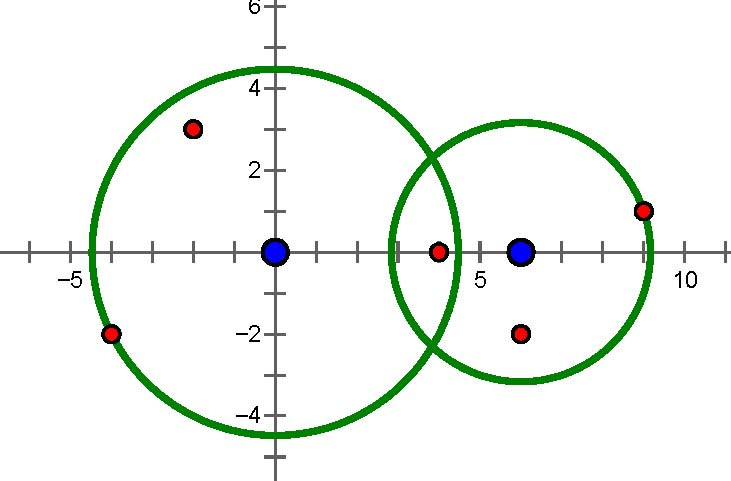
样例1 中要拦截所有导弹，在满足最小使用代价的前提下，两套系统工作半径的平方分别为18 和0。

【输入输出样例2】

|  |  |
| --- | --- |
| missile.in | missile.out |
| 0 0 6 0 | 30 |
| 5 |  |
| -4 -2 |  |
| -2 3 |  |
| 4 0 |  |
| 6 -2 |  |
| 9 1 |  |

【样例2 说明】

样例中的导弹拦截系统和导弹所在的位置如下图所示。要拦截所有导弹，在满足最小使用代价的前提下，两套系统工作半径的平方分别为20 和10。



【数据范围】

对于10%的数据，N = 1

对于20%的数据，1 ≤ N ≤ 2

对于40%的数据，1 ≤ N ≤ 100

对于70%的数据，1 ≤ N ≤ 1000

对于100%的数据，1 ≤ N ≤ 100000，且所有坐标分量的绝对值都不超过1000。

**NOIP2011瑞士轮(swiss.cpp/c/pas)**

**【**背景**】**

在双人对决的竞技性比赛，如乒乓球、羽毛球、国际象棋中，最常见的赛制是淘汰赛和循环赛。前者的特点是比赛场数少，每场都紧张刺激，但偶然性较高。后者的特点是较为公平，偶然性较低，但比赛过程往往十分冗长。

本题中介绍的瑞士轮赛制，因最早使用于 1895 年在瑞士举办的国际象棋比赛而得名。它可以看作是淘汰赛与循环赛的折衷，既保证了比赛的稳定性，又能使赛程不至于过长。

**【**问题描述**】**

2\*N 名编号为1~2N 的选手共进行R 轮比赛。每轮比赛开始前，以及所有比赛结束后，都会按照总分从高到低对选手进行一次排名。选手的总分为第一轮开始前的初始分数加上已参加过的所有比赛的得分和。总分相同的，约定编号较小的选手排名靠前。

每轮比赛的对阵安排与该轮比赛开始前的排名有关：第 1 名和第2 名、第3 名和第4名、……、第2K–1 名和第2K 名、…… 、第 2N–1 名和第2N 名，各进行一场比赛。每场比赛胜者得1 分，负者得0 分。也就是说除了首轮以外，其它轮比赛的安排均不能事先确定，而是要取决于选手在之前比赛中的表现。

现给定每个选手的初始分数及其实力值，试计算在 R 轮比赛过后，排名第Q 的选手编号是多少。我们假设选手的实力值两两不同，且每场比赛中实力值较高的总能获胜。

**【**输入**】**

输入文件名为 swiss.in。

输入的第一行是三个正整数 N、R、Q，每两个数之间用一个空格隔开，表示有2\*N 名选手、R 轮比赛，以及我们关心的名次Q。

第二行是 2\*N 个非负整数s1, s2, …, s2N，每两个数之间用一个空格隔开，其中si 表示编号为i 的选手的初始分数。

第三行是 2\*N 个正整数w1, w2, …, w2N，每两个数之间用一个空格隔开，其中wi 表示编号为i 的选手的实力值。

**【**输出**】**

输出文件名为 swiss.out。

输出只有一行，包含一个整数，即 R 轮比赛结束后，排名第Q 的选手的编号。

**【**输入输出样例**】**

|  |  |
| --- | --- |
| swiss.in | swiss.out |
| 2 4 2  7 6 6 7  10 5 20 15 | 1 |

**【**输入输出样例说明**】**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 本轮对阵 | 本轮结束后的得分 | | | |
| 选手编号 | / | ① | ② | ③ | ④ |
| 初始 | / | 7 | 6 | 6 | 7 |
| 第1轮 | ①—④ ②—③ | 7 | 6 | 7 | 8 |
| 第2轮 | ④—① ③—② | 7 | 6 | 8 | 9 |
| 第3 轮 | ④—③ ①—② | 8 | 6 | 9 | 9 |
| 第4 轮 | ③—④ ①—② | 9 | 6 | 10 | 9 |

**【**数据范围**】**

对于 30%的数据，1≤N≤100；

对于 50%的数据，1≤N≤10,000；

对于 100%的数据，1≤N≤100,000，1≤R≤50，1≤Q≤2N，0≤s1, s2, …, s2N ≤，1≤w1,w2, …, w2N ≤。

**2012摆花 (flower.cpp/c/pas)**

【问题描述】

小明的花店新开张，为了吸引顾客，他想在花店的门口摆上一排花，共 m 盆。通过调查顾客的喜好，小明列出了顾客最喜欢的 n 种花，从 1 到 n 标号。为了在门口展出更多种花，规定第 i 种花不能超过 ai盆，摆花时同一种花放在一起，且不同种类的花需按标号的从小到 大的顺序依次摆列。 试编程计算，一共有多少种不同的摆花方案。

【输入】

输入文件 flower.in，共 2 行。 第一行包含两个正整数 n 和 m，中间用一个空格隔开。 第二行有 n 个整数，每两个整数之间用一个空格隔开，依次表示 a1、a2、……an。

【输出】

输出文件名为 flower.out。 输出只有一行，一个整数，表示有多少种方案。注意：因为方案数可能很多，请输出 方案数对 1000007 取模的结果。

【输入输出样例 1】

|  |  |
| --- | --- |
| flower.in | flower.out |
| 2 4  3 2 | 2 |

【输入输出样例说明】

有 2 种摆花的方案，分别是(1，1，1，2)， (1，1，2，2)。括号里的 1 和 2 表示两种花， 比如第一个方案是前三个位置摆第一种花，第四个位置摆第二种花。

【数据范围】

对于 20%数据，有 0<n≤8，0<m≤8，0≤ai≤8； 对于 50%数据，有 0<n≤20，0<m≤20，0≤ai≤20； 对于 100%数据，有 0<n≤100，0<m≤100，0≤ ai≤100。

**NOIP2013小朋友的数字 (number.cpp/c/pas)**

【问题描述】

有 n 个小朋友排成一列。每个小朋友手上都有一个数字,这个数字可正可负。规定每个小朋友的特征值等于排在他前面(包括他本人)的小朋友中连续若干个(最少有一个)小朋友手上的数字之和的最大值。

作为这些小朋友的老师,你需要给每个小朋友一个分数,分数是这样规定的:第一个小朋友的分数是他的特征值,其它小朋友的分数为排在他前面的所有小朋友中(不包括他本人),小朋友分数加上其特征值的最大值。

请计算所有小朋友分数的最大值,输出时保持最大值的符号,将其绝对值对 p 取模后输出。

【输入】

输入文件为 number.in。

第一行包含两个正整数 n、p，之间用一个空格隔开。

第二行包含 n 个数，每两个整数之间用一个空格隔开，表示每个小朋友手上的数字。

【输出】

输出文件名为 number.out。

输出只有一行，包含一个整数，表示最大分数对 p 取模的结果。

【输入输出样例 1】

|  |  |
| --- | --- |
| number.in | number.out |
| 5 997  1 2 3 4 5 | 21 |

【输入输出样例说明】

小朋友的特征值分别为 1、3、6、10、15，分数分别为 1、2、5、11、21，最大值 21

对 997 的模是 21。

【输入输出样例 2】

|  |  |
| --- | --- |
| number.in | number.out |
| 5 7  -1 -1 -1 -1 -1 | -1 |

【输入输出样例说明】

小朋友的特征值分别为-1、-1、-1、-1、-1，分数分别为-1、-2、-2、-2、-2，最大值

-1 对 7 的模为-1，输出-1。

【数据范围】

对于 50%的数据，1 ≤ n ≤ 1,000，1 ≤ p ≤ 1,000所有数字的绝对值不超过 1000；

对于 100%的数据，1 ≤ n ≤ 1,000,000， 1≤ p ≤ 109， 其他数字的绝对值均不超过 109。

**NOIP2014螺旋矩阵**

【问题描述】

一个 n 行 n 列的螺旋矩阵可由如下方法生成:从矩阵的左上角(第 1 行第 1 列)出发,初始时向右移动;如果前方是未曾经过的格子,则继续前进,否则右转;重复上述操作直至经过矩阵中所有格子。根据经过顺序,在格子中依次填入 1, 2, 3, ... , n 2 ,便构成了一个螺旋矩阵。

下图是一个 n = 4 时的螺旋矩阵。

现给出矩阵大小 n 以及 i 和 j,请你求出该矩阵中第 i 行第 j 列的数是多少。

【输入】

输入共一行,包含三个整数 n,i,j,每两个整数之间用一个空格隔开,分别表示矩阵大小、待求的数所在的行号和列号。

【输出】

输出共一行,包含一个整数,表示相应矩阵中第 i 行第 j 列的数。

【输入样例】

4 2 3

【输出样例】

14

【数据说明】

对于 50%的数据,1 ≤ n ≤ 100;

对于 100%的数据,1 ≤ n ≤ 30,000,1 ≤ i ≤ n,1 ≤ j ≤ n。

-------------------------------------------------------------------------------

首先处理特殊情况：

1、N=1直接输出1

2、如果要找的数字在边界上，直接计算即可

中间情况

1、首先算出外围有多少数字

2、分四个方向计算数字排位

以上为最简单的模拟方法，方法直观，还有更简洁的方法自行思考

**NOIP2015求和 (sum.cpp/c/pas)**

【问题描述】

一条狭长的纸带被均匀划分出了 n 个格子，格子编号从 1 到 n。每个格子上都染了一种颜色𝑐𝑜𝑙𝑜𝑟𝑖（用[1，m]当中的一个整数表示），并且写了一个数字𝑛𝑢𝑚𝑏𝑒𝑟𝑖。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **5** | **5** | **3** | **2** | **2** | **2** |
| 1 | 2 | 3 | 4 | 5 | 6 |

编号

定义一种特殊的三元组：(x, y, z)，其中 x，y，z 都代表纸带上格子的编号，这里的三元组要求满足以下两个条件：

1. 𝑥, 𝑦, 𝑧都是整数, 𝑥 < 𝑦 < 𝑧, 𝑦 − 𝑥 = 𝑧 − 𝑦
2. 𝑐𝑜𝑙𝑜𝑟𝑥 = 𝑐𝑜𝑙𝑜𝑟𝑧

满足上述条件的三元组的分数规定为(x + z) ∗ (𝑛𝑢𝑚𝑏𝑒𝑟𝑥 + 𝑛𝑢𝑚𝑏𝑒𝑟𝑧)。整个纸带的分数规定为所有满足条件的三元组的分数的和。这个分数可能会很大，你只要输出整个纸带的分数除以 10,007 所得的余数即可。

【输入格式】

输入文件名为 sum.in。

第一行是用一个空格隔开的两个正整数𝑛和𝑚，𝑛代表纸带上格子的个数，𝑚代表纸带上颜色的种类数。

第二行有𝑛个用空格隔开的正整数，第𝑖个数字𝑛𝑢𝑚𝑏𝑒𝑟𝑖代表纸带上编号为𝑖的格子上面写的数字。

第三行有𝑛个用空格隔开的正整数，第𝑖个数字𝑐𝑜𝑙𝑜𝑟𝑖代表纸带上编号为𝑖的格子染的颜色。

【输出格式】

输出文件名为 sum.out。 共一行，一个整数，表示所求的纸带分数除以 10,007 所得的余数。

【输入输出样例 1】

|  |  |  |
| --- | --- | --- |
| **sum.in** | **sum.out** | |
| 6 2  5 5 3 2 2 2  2 2 1 1 2 1 | 82 |

【输入输出样例 1 说明】

纸带如题目描述中的图所示。

所有满足条件的三元组为：(1,3, 5), (4, 5, 6)。

所以纸带的分数为(1 + 5) ∗ (5 + 2) + (4 + 6) ∗ (2 + 2) = 42 + 40 = 82。

【输入输出样例 2】

|  |  |
| --- | --- |
| **sum.in** | **sum.out** |
| 15 4  5 10 8 2 2 2 9 9 7 7 5 6 4 2 4  2 2 3 3 4 3 3 2 4 4 4 4 1 1 1 | 1388 |

【输入输出样例 3】见选手目录下的 sum/sum3.in 和 sum/sum3.ans。

【数据说明】

对于第 1 组至第 2 组数据，1 ≤ 𝑛 ≤ 100, 1 ≤ 𝑚 ≤ 5；

对于第 3 组至第 4 组数据，1 ≤ 𝑛 ≤ 3000, 1 ≤ 𝑚 ≤ 100；

对于第 5 组至第 6 组数据，1 ≤ 𝑛 ≤ 100000, 1 ≤ 𝑚 ≤ 100000，且不存在出现次数超过 20 的颜色；

对于全部 10 组数据， 1 ≤ 𝑛 ≤ 100000, 1 ≤ 𝑚 ≤ 100000, 1 ≤ 𝑐𝑜𝑙𝑜𝑟𝑖 ≤ 𝑚, 1 ≤𝑛𝑢𝑚𝑏𝑒𝑟𝑖 ≤ 100000。

**NOIP2016海港(port.cpp/c/pas)**

【问题描述】

小K是一个海港的海关工作人员，每天都有许多船只到达海港，船上通常有很多来自不同国家的乘客。小K对这些到达海港的船只非常感兴趣，他按照时间记录下了到达海港的每一艘船只情况；对于第i艘到达的船，他记录了这艘船到达的时间ti (单位：秒)，船上的乘 客数星ki，以及每名乘客的国籍 x(i,1), x(i,2),… x(i,k)。小K统计了n艘船的信息，希望你帮忙计算出以每一艘船到达时间为止的24小时(24小时=86400秒）内所有乘船到达的乘客来自多少个不同的国家。

形式化地讲，你需要计算n条信息。对于输出的第i条信息，你需要统计满足 ti - 86400 < tp <= ti的船只p，在所有的x(p,j)中，总共有多少个不同的数。

## 【输入输出格式】

**输入格式：**

第一行输入一个正整数n，表示小K统计了 n艘船的信息。接下来n行，每行描述一艘船的信息：前两个整数ti和ki分别表示这艘船到达海港的时间和船上的乘客数量，接下来ki个整数x(i,j)表示船上乘客的国7。保证输入的ti是递增的，单位是秒；表示从小K第一次上班开始计时，这艘船在第 ti 秒到达海港。保证 IMG_256，IMG_257 ，IMG_258， IMG_259。其中IMG_260表示所有的ki的和。

**输出格式：**

输出n行，第i行输出一个整数表示第i艘船到达后的统计信息。

## 【输入输出样例】

**输入样例#1：**

3

1 4 4 1 2 2

2 2 2 3

10 1 3

**输出样例#1：**

3

4

4

**输入样例#2：**

4

1 4 1 2 2 3

3 2 2 3

86401 2 3 4

86402 1 5

**输出样例#2：**

3

3

3

4

【样例解释1】

第一艘船在第1秒到达海港，最近24小时到达的船是第一艘船，共有4个乘客， 分别是来自国家4,1,2,2，共来自3个不同的国家；

第二艘船在第2秒到达海港，最近24小时到达的船是第一艘船和第二艘船，共有 4 + 2 = 6个乘客，分别是来自国家4,1,2,2,2,3，共来自4个不同的国家；

第三艘船在第10秒到达海港，最近24小时到达的船是第一艘船、第二艘船和第 三艘船，共有4+ 2+1=7个乘客，分别是来自国家4,1,2,2,2,3,3，共来自4个不同 的国家。

【样例解释2】

第一艘船在第1秒到达海港，最近24小时到达的船是第一艘船，共有4个乘客，分别是来自国家1,2,2,3，共来自3个不同的国家。

第二艘船在第3秒到达海港，最近24小时到达的船是第一艘船和第二艘船，共有4+2=6个乘客，分别是来自国家1,2,2,3,2,3，共来自3个不同的国家。

第三艘船在第86401秒到达海港，最近24小时到达的船是第二艘船和第三艘船，共有2+2=4个乘客，分别是来自国家2,3,3,4，共来自3个不同的国家。

第四艘船在第86402秒到达海港，最近24小时到达的船是第二艘船、第三艘船和第四艘船，共有2+2+1=5个乘客，分别是来自国家2,3,3,4,5，共来自4个不同的国家。

**NOIP2017棋盘 (chess.cpp/c/pas)**

【问题描述】

有一个m × m的棋盘，棋盘上每一个格子可能是红色、黄色或没有任何颜色的。你现在要从棋盘的最左上角走到棋盘的最右下角。

任何一个时刻，你所站在的位置必须是有颜色的（不能是无色的），你只能向上、下、左、右四个方向前进。当你从一个格子走向另一个格子时，如果两个格子的颜色相同，那你不需要花费金币；如果不同，则你需要花费 1 个金币。

另外，你可以花费 2 个金币施展魔法让下一个无色格子暂时变为你指定的颜色。但这个魔法不能连续使用，而且这个魔法的持续时间很短，也就是说，如果你使用了这个魔法，走到了这个暂时有颜色的格子上，你就不能继续使用魔法；只有当你离开这个位置，走到一个本来就有颜色的格子上的时候，你才能继续使用这个魔法，而当你离开了这个位置（施展魔法使得变为有颜色的格子）时，这个格子恢复为无色。 现在你要从棋盘的最左上角，走到棋盘的最右下角，求花费的最少金币是多少？

【输入格式】

输入文件名为chess.in。

数据的第一行包含两个正整数 m，n，以一个空格分开，分别代表棋盘的大小，棋盘上有颜色的格子的数量。 接下来的 n 行，每行三个正整数 x，y，c，分别表示坐标为（x，y）的格子有颜色 c。其中 c=1 代表黄色，c=0 代表红色。相邻两个数之间用一个空格隔开。棋盘左上角的坐标为（1, 1），右下角的坐标为（m, m）。 棋盘上其余的格子都是无色。保证棋盘的左上角，也就是（1，1）一定是有颜色的。

【输出格式】

输出文件名为chess.out。

输出一行，一个整数，表示花费的金币的最小值，如果无法到达，输出-1。

【输入输出样例 1】

|  |  |
| --- | --- |
| **chess.in** | **chess.out** |
| 5 7  1 1 0   1. 2 0 2. 2 1 3. 3 1 4. 4 0 5. 4 1 6. 5 0 | 8 |

【输入输出样例 1 说明】

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

从（1，1）开始，走到（1，2）不花费金币从（1，2）向下走到（2，2）花费 1 枚金币从（2，2）施展魔法，将（2，3）变为黄色，花费 2 枚金币从（2，2）走到（2，3）不花费金币从（2，3）走到（3，3）不花费金币从（3，3）走到（3，4）花费 1 枚金币从（3，4）走到（4，4）花费 1 枚金币从（4，4）施展魔法，将（4，5）变为黄色，花费 2 枚金币，从（4，4）走到（4，5）不花费金币从（4，5）走到（5，5）花费 1 枚金币共花费 8 枚金币。

【输入输出样例 2】

|  |  |
| --- | --- |
| **chess.in** | **chess.out** |
| 5 5  1 1 0   1. 2 0 2. 2 1 3. 3 1   5 5 0 | -1 |

【输入输出样例 2 说明】

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

从（1，1）走到（1，2），不花费金币从（1，2）走到（2，2），花费 1 金币施展魔法将（2，3）变为黄色，并从（2，2）走到（2，3）花费 2 金币从（2，3）走到（3，3）不花费金币 从（3，3）只能施展魔法到达（3，2），（2，3），（3，4），（4，3）,而从以上四点均无法到达（5，5），故无法到达终点，输出－1 。

【数据规模与约定】

对于 30%的数据，1 ≤ m ≤ 5， 1 ≤ n ≤ 10。

对于 60%的数据，1 ≤ m ≤ 20， 1 ≤ n ≤ 200。 对于 100%的数据，1 ≤ m≤100， 1 ≤ n ≤ 1,000。