冲刺 NOIP2015 模拟赛 2

（请选手务必仔细阅读本页内容）

一、题目概况

中文名称	解压密码	开销预算	华容道	分配任务
英文名称	passworda	expense	PuzzleNOIP2013	batch
输入文件	passworda.in	expense.in	PuzzleNOIP2013.in	batch.in
输出文件	passworda.out	expense.out	PuzzleNOIP2013.out	batch.out
测试点时限	1000 毫秒	1000 毫秒	1000 毫秒	1000 毫秒
测试点数目	10	10	20	10
测试点分值	10	10	5	10
比较方式	全文比较			
题目类型	传统	传统	传统	传统
内存上限	128 兆字节	128 兆字节	128 兆字节	128 兆字节

二、提交源程序文件名

C	passworda.c	expense.c	PuzzleNOIP2013.c	batch.c
C++	passworda.cpp	expense.cpp	PuzzleNOIP2013.cpp	batch.cpp
Pascal	passworda.pas	expense.pas	PuzzleNOIP2013.pas	batch.pas

三、编译命令

C	gcc -Wall -std=c99 -DCONTEST -o foo src.c -lm
C++	g++ -Wall -std=c++11 -DCONTEST -o foo src.cpp -lm
Pascal	fpc -Mtp -v0 -dCONTEST -Sgic -Tlinux -ofoo src.pas -lm

注意事项：

1、 需要为每个题目建立英文小写的子目录。

2、 文件名（程序名和输入输出文件名）必须使用英文小写。

3、 C/C++中函数 main 的返回值类型必须是 int，程序正常结束时返回值必须是 0。

4、 评测时采用的机器配置为：Intel Pentium G2020 2.90 GHz × 2 处理器，4GB 内存。
上述时限以此配置为准。

5、 特别提醒：评测在 CentOS 6.7 x86_64 操作系统上进行，各语言的编译器版本如下：
GCC 4.4.7，FPC 2.6.4。

1. 破译密码

【问题描述】

有了防护伞，并不能完全避免 2012 的灾难。地球防卫小队决定去求助外星种族的帮助。经过很长时间的努力，小队终于收到了外星生命的回信。但是外星人发过来的却是一串密码。只有解开密码，才能知道外星人给的准确回复。

解开密码的第一道工序就是解压缩密码，外星人对于连续的若干个相同的子串“x”会压缩为“[DX]”的形式（D 是一个整数且 0＜D≤99），比如说字符串“CBCBCBCB”就压缩为“[4CB]”或者“[2[2CB]]”，类似于后面这种压缩之后再压缩的我们称之为二重压缩。如果是“[2[2[2CB]]]"，则是三重。

现在我们给你外星人发送过来的密码，请你对其进行解压缩。

【输入】

1 行：一个字符串，外星人发送过来的密码

【输出】

1 行：一个字符串，解压缩后得到的密码

【输入输出样例】

Input	Output
输入样例一： AC[3FUN] 输入样例二： [2BILI]	输出样例一： ACFUNFUNFUN 输出样例二： BILIBILI

【数据说明】

数据范围：

对于 50%的数据： 解压后的字符串长度在 1,000 以内，最多只有三重压缩。

对于 100%的数据：解压后的字符串长度在 20,000 以内，最多只有十重压缩。

保证只包含数字、大写字母、‘[’和‘]’。

2. 开销预算

【问题描述】

Farmer John 是一个令人惊讶的会计学天才，他已经明白了他可能会花光他的钱，这些钱本来是要维持农场每个月的正常运转的。他已经计算了他以后 N(1＜=N＜=100,000)个工作日中每一天的花费 moneyi(1＜=moneyi＜=10,000)，他想要为他连续的 M(1＜=M＜=N)个被叫做“清算月”的结帐时期做一个预算，每一个“清算月”包含一个工作日或更多连续的工作日，每一个工作日都仅被包 含在一个“清算月”当中。

FJ 的目标是安排这些“清算月”，使得每个清算月的花费中最大的那个花费达到最小，从而来决定他的月度支出限制。

【输入】

第一行包含两个整数 N,M，用单个空格隔开。

接下来 N 行，每行包含一个 1 到 10000 之间的整数，按顺序给出接下来 N 天里每天的开销。

【输出】

一个整数，能够维持每个月农场正常运转的钱数，即最大清算月度开销的最小值。

【输入输出样例 1】

Input	Output
7 5 100 400 300 100 500 101 400	500

【数据说明】

1＜=N＜=100,000，1＜=M＜=N；

每一天的花费 moneyi(1＜=moneyi＜=10,000)

【样例说明】

输入细节

这里有 7 个工作日来被 5 个“清算月”划分。他花费 100，400，300，100，500，101 和 400 元在他的每个工作日。

输出细节

如果 FJ 安排他的月度预算，他将把前两天划分在一个月中，把第三天、第四天划分在一个月当中，最后的三个工作日各自在一个月当中，所以他一个月最多花费 500 元，其他的方法总是得出一个较大的结果。

100 400	300 100	500	101	400	每天花费
---1---	---2---	-3-	-4-	-5-	月度标号
500	400	500	101	400	月度花费

3. 华容道

【问题描述】

小 B 最近迷上了华容道， 可是他总是要花很长的时间才能完成一次。于是，他想到用编程来完成华容道：给定一种局面， 华容道是否根本就无法完成，如果能完成， 最少需要多少时间。

小 B 玩的华容道与经典的华容道游戏略有不同，游戏规则是这样的：

1. 在一个 n*m 棋盘上有 n*m 个格子，其中有且只有一个格子是空白的，其余 n*m-1 个格子上每个格子上有一个棋子， 每个棋子的大小都是 1*1 的；

2. 有些棋子是固定的，有些棋子则是可以移动的；

3. 任何与空白的格子相邻（有公共的边） 的格子上的棋子都可以移动到空白格子上。

游戏的目的是把某个指定位置可以活动的棋子移动到目标位置。

给定一个棋盘，游戏可以玩 q 次，当然，每次棋盘上固定的格子是不会变的， 但是棋盘上空白的格子的初始位置、 指定的可移动的棋子的初始位置和目标位置却可能不同。第 i 次玩的时候，空白的格子在第 EXi 行第 EYi 列，指定的可移动棋子的初始位置为第 SXi 行第 SYi 列， 目标位置为第 TXi 行第 TYi 列。

假设小 B 每秒钟能进行一次移动棋子的操作， 而其他操作的时间都可以忽略不计。 请你告诉小 B 每一次游戏所需要的最少时间，或者告诉他不可能完成游戏。

【输入】

第一行有 3 个整数，每两个整数之间用一个空格隔开，依次表示 n、 m 和 q；

接下来的 n 行描述一个 n*m 的棋盘，每行有 m 个整数，每两个整数之间用一个空格隔开，每个整数描述棋盘上一个格子的状态，0 表示该格子上的棋子是固定的，1 表示该格子上的棋子可以移动或者该格子是空白的。

接下来的 q 行，每行包含 6 个整数依次是 EXi、 EYi、 SXi、 SYi、 TXi、 TYi， 每两个整数之间用一个空格隔开，表示每次游戏空白格子的位置， 指定棋子的初始位置和目标位置。

【输出】

输出有 q 行，每行包含 1 个整数，表示每次游戏所需要的最少时间，如果某次游戏无法完成目标则输出-1。

【输入输出样例】

Input	Output
3 4 2 0 1 1 1 0 1 1 0 0 1 0 0 3 2 1 2 2 2 1 2 2 2 3 2	2 -1

【数据说明】

对于 30%的数据， $1 \le n, m \le 10$， $q = 1$；

对于 60%的数据， $1 \le n, m \le 30$， $q \le 10$；

对于 100%的数据， $1 \le n, m \le 30$， $q \le 500$。

【样例说明】

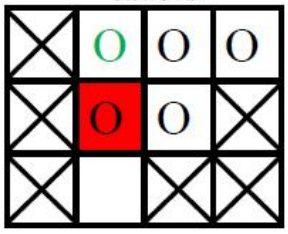
【输入输出样例说明】

棋盘上划叉的格子是固定的，红色格子是目标位置，圆圈表示棋子，其中绿色圆圈表示目标棋子。

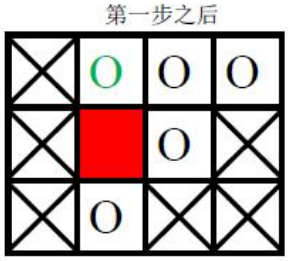
1. 第一次游戏，空白格子的初始位置是 (3, 2)（图中空白所示），游戏的目标是将初始位置在(1, 2)上的棋子（图中绿色圆圈所代表的棋子）移动到目标位置(2, 2)（图中红色的格子）上。

移动过程如下：

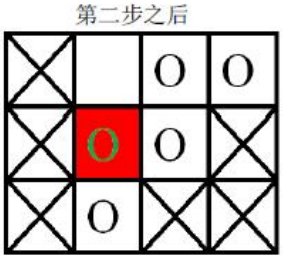
初始状态



第一步之后

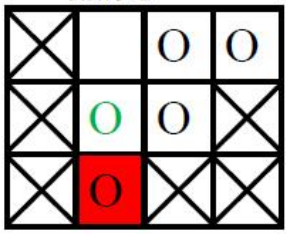


第二步之后



2. 第二次游戏，空白格子的初始位置是（1, 2）（图中空白所示），游戏的目标是将初始位置在（2, 2）上的棋子（图中绿色圆圈所示）移动到目标位置 (3, 2)上。

初始状态



要将指定块移入目标位置，必须先将空白块移入目标位置，空白块要移动到目标位置，必然是从位置（2，2）上与当前图中目标位置上的棋子交换位置，之后能与空白块交换位置的只有当前图中目标位置上的那个棋子，因此目标棋子永远无法走到它的目标位置，游戏无法完成。

4. 分配任务

【问题描述】

N 个任务排成一个序列在一台机器上等待完成（顺序不得改变），这 N 个任务被分成若干批，每批包含相邻的若干任务。从时刻 0 开始，这些任务被分批加工，第 i 个任务单独完成所需的时间是 Ti。在每批任务开始前，机器需要启动时间 S，而完成这批任务所需的时间是各个任务需要时间的总和（同一批任务将在同一时刻完成）。每个任务的费用是它的完成时刻乘以一个费用系数 Fi。请确定一个分组方案，使得总费用最小。

例如：S=1；T={1,3,4,2,1}；F={3,2,3,3,4}。如果分组方案是{1,2}、{3}、{4,5}，则完成时间分别为{5,5,10,14,14}，费用 C={15,10,30,42,56}，总费用就是 153。

【输入】

第一行是 N(1＜=N＜=5000)。

第二行是 S(0＜=S＜=50)。

下面 N 行每行有一对数，分别为 Ti 和 Fi，均为不大于 100 的正整数，表示第 i 个任务单独完成所需的时间是 Ti 及其费用系数 Fi。

【输出】

一个数，最小的总费用。

【输入输出样例】

Input	Output
5 1 1 3 3 2 4 3 2 3 1 4	153

【数据说明】

1＜=N＜=5000

0＜=S＜=50