

USACO TRAIN PROBLEM LIST

Collected by BirDOR
From <http://www.wzoi.org:88/usaco/>
Thanks for All Translators/Oiers/Readers

CONTEST

Chapter1

Section 1.1

- 1.1.1 Your Ride Is Here
- 1.1.2 Greedy Gift Givers
- 1.1.3 Friday the Thirteenth
- 1.1.4 Broken Necklace

Section 1.2

- 1.2.1 Milking Cows
- 1.2.2 Transformations
- 1.2.3 Name That Number
- 1.2.4 Palindromic Squares
- 1.2.5 Dual Palindromes

Section 1.3

- 1.3.1 Mixing Milk
- 1.3.2 Barn Repair
- 1.3.3 Calf Flac
- 1.3.4 Prime Cryptarithm

Section 1.4

- 1.4.1 Packing Rectangles
- 1.4.2 The Clocks
- 1.4.3 Arithmetic Progressions
- 1.4.4 Mother's Milk

Section 1.5

- 1.5.1 Number Triangles
- 1.5.2 Prime Palindromes
- 1.5.3 Superprime Rib
- 1.5.4 Checker Challenge

Chapter2

Section 2.1

- 2.1.1 The Castle
- 2.1.2 Ordered Fractions
- 2.1.3 Sorting a Three-Valued Sequence
- 2.1.4 Healthy Holsteins
- 2.1.5 Hamming Codes

Section 2.2

- 2.2.1 Preface Numbering
- 2.2.2 Subset Sums
- 2.2.3 Runaround Numbers

2.2.4 Party Lamps

Section 2.3

- 2.3.1 Longest Prefix
- 2.3.2 Cow Pedigrees
- 2.3.3 Zero Sum
- 2.3.4 Money Systems
- 2.3.5 Controlling Companies

Section 2.4

- 2.4.1 The Tamworth Two
- 2.4.2 Overfencing
- 2.4.3 Cow Tours
- 2.4.4 Bessie Come Home
- 2.4.5 Fractions to Decimals

Chapter3

Section 3.1

- 3.1.1 Agri-Net
- 3.1.2 Score Inflation
- 3.1.3 Humble Numbers
- 3.1.4 Shaping Regions
- 3.1.5 Contact
- 3.1.6 Stamps

Section 3.2

- 3.2.1 Factorials
- 3.2.2 Stringsobits
- 3.2.3 Spinning Wheels
- 3.2.4 Feed Ratios
- 3.2.5 Magic Squares
- 3.2.6 Sweet Butter

Section 3.3

- 3.3.1 Riding the Fences
- 3.3.2 Shopping Offers
- 3.3.3 Camelot
- 3.3.4 Home on the Range
- 3.3.5 A Game

Section 3.4

- 3.4.1 Closed Fences
- 3.4.2 American Heritage
- 3.4.3 Electric Fence
- 3.4.4 Raucous Rockers

Chapter4

Section 4.1

- 4.1.1 Beef McNuggets
- 4.1.2 Fence Rails
- 4.1.3 Fence Loops
- 4.1.4 Cryptcowgraphy

Section 4.2

- 4.2.1 Drainage Ditches
- 4.2.2 The Perfect Stall
- 4.2.3 Job Processing
- 4.2.4 Cowcycles

Section 4.3

- 4.3.1 Buy Low, Buy Lower
- 4.3.2 The Primes
- 4.3.3 Street Race
- 4.3.4 Letter Game

Section 4.4

- 4.4.1 Shuttle Puzzle
- 4.4.2 Pollutant Control
- 4.4.3 Frame Up

Chapter5

Section 5.1

- 5.1.1 Fencing the Cows
- 5.1.2 Starry Night
- 5.1.3 Musical Themes

Section 5.2

- 5.2.1 Snail Trails
- 5.2.2 Electric Fences
- 5.2.3 Wisconsin Squares

Section 5.3

- 5.3.1 Milk Measuring
- 5.3.2 Window Area
- 5.3.3 Network of Schools
- 5.3.4 Big Barn

Section 5.4

- 5.4.1 All Latin Squares
- 5.4.2 Canada Tour
- 5.4.3 Character Recognition
- 5.4.4 Betsy's Tour
- 5.4.5 Telecowmunication

Section 5.5

- 5.5.1 Picture
- 5.5.2 Hidden Password
- 5.5.3 Towfive

Chapter6

Section 6.1

- 6.1.1 Postal Vans
- 6.1.2 A Rectangular Barn
- 6.1.3 Cow XOR

CHAPTER 1

Section 1.1

★Your Ride Is Here 你要乘坐的飞碟在这里

一个众所周知的事实, 在每一彗星后面是一个不明飞行物 UFO. 这些不明飞行物时常来收集来自在地球上忠诚的支持者. 不幸地, 他们的空间在每次旅行只能带上一群支持者. 他们要做的是用一种聪明的方案让每一个团体人被彗星带走. 他们为每个彗星起了一个名字, 通过这些名字来决定一个团体是不是特定的彗星带走. 那个相配方案的细节在下面被给出;

你的工作要写一个程序来通过团体的名字和彗星的名字来决定一个组是否应该与在那一颗彗星后面的不明飞行物搭配.

团体的名字和彗星的名字都以下列各项方式转换成一个数字: 这个最后的数字代表名字中所有字母的信息, "A" 是 1 和 "Z" 是 26.

举例来说, 团体 "USACO" 会是 $21*19*1*3*15=17955$. 如果团体的数字 mod 47 等于彗星的数字 mod 47, 那么你要告诉这个团体准备好被带走!

写一个程序读入彗星的名字和团体的名字, 如果搭配打印 "GO" 否者打印 "STAY"

团体的名字和彗星的名字将会是没有空格或标点的一串大写字母 (不超过 6 个字母), 如:

Input	Output
COMETQ	GO
HVNGAT	
ABSTAR	STAY
USACO	

PROGRAM NAME: ride

INPUT FORMAT

第 1 行: 彗星的名字 (一个长度为 1 到 6 的字符串)

第 2 行: 团体的名字 (一个长度为 1 到 6 的字符串)

SAMPLE INPUT (file ride.in)

COMETQ
HVNGAT

OUTPUT FORMAT

单独一行包含 "STAR" 或 "GO".

SAMPLE OUTPUT (file ride.out)

GO

★Greedy Gift Givers 贪婪的礼物送礼者

对于一群要互送礼物的朋友, 你要确定每个人送出的礼物比收到的多多少 (and vice versa for those who view gift giving with cynicism).

在这一个问题中, 每个人都准备了一些钱来送礼物, 而这些钱将会被平均分给那些将收到他的礼物的人.

然而, 在任何一群朋友中, 有些人将送出较多的礼物 (可能是因为有较多的朋友), 有些人有准备了较多的钱.

给出一群朋友, 没有人的名字会长于 14 字符, 给出每个人将花在送礼上的钱, 和将收到他的礼物的人的列表,

请确定每个人收到的比送出的钱多的数目.

IMPORTANT NOTE

测试系统是 Linux 符合标准的 Unix 的协定.

用 '\n' 作为行的结束.

这和 Windows 系统用 '\n' 和 '\r' 作为行的结束是不同的.

你的程序不要被这困住了.

PROGRAM NAME: gift1

INPUT FORMAT

第 1 行: 人数 NP, $2 \leq NP \leq 10$

第 2 到 NP+1 行: 这 NP 个在组里人的名字 一个名字一行

第 NP+2 到最后: 这里的 NP 段内容是这样组织的:

第一行是将会送出礼物人的名字.

第二行包含二个数字: 第一个是原有的钱的数目 (在 0 到 2000 的范围里), 第二个 NGi 是将收到这个送礼者礼物的人的个数 如果 NGi 是非零的, 在下面 NGi 行列出礼物的接受者的名字, 一个名字一行.

SAMPLE INPUT (file gift1.in)

```
5
dave
laura
owen
vick
amr
dave
200 3
laura
owen
vick
owen
500 1
dave
amr
150 2
vick
owen
laura
```

```
0 2
amr
vick
vick
0 0
```

OUTPUT FORMAT

输出 NP 行

每行是一个的名字加上空格再加上收到的比送出的钱多的数目.

对于每一个人, 他名字的打印顺序应和他在输入的 2 到 NP+1 行中输入的顺序相同. 所有的送礼的钱都是整数.

每个人把相同数目的钱给每位要送礼的朋友, 而且尽可能多给, 不能给出的钱被送礼者自己保留.

SAMPLE OUTPUT (file gift1.out)

```
dave 302
laura 66
owen -359
vick 141
amr -150
```

★Friday the Thirteenth 黑色星期五

13 号又是星期五是一个不寻常的日子吗?

13 号在星期五比在其他日少吗?为了回答这个问题, 写一个程序来计算在 n 年里 13 日落在星期一, 星期二.....星期日的次数. 这个测试从 1900 年 1 月 1 日到 1900+n-1 年 12 月 31 日. n 是一个非负数且不大于 400.

这里有一些你要知道的:

1900 年 1 月 1 日是星期一.

4, 6, 11 和 9 月有 30 天. 其他月份除了 2 月有 31 天. 闰年 2 月有 29 天, 平年 2 月有 28 天.

年份可以被 4 整除的为闰年 (1992=4*498 所以 1992 年是闰年, 但是 1990 年不是闰年)

以上规则不适合于世纪年. 可以被 400 整除的世纪年为闰年, 否则为平年. 所以, 1700, 1800, 1900 和 2100 年是平年, 而 2000 年是闰年.

请不要预先算好数据!

PROGRAM NAME: friday

INPUT FORMAT

一个整数 n .

SAMPLE INPUT (file friday.in)

```
20
```

OUTPUT FORMAT

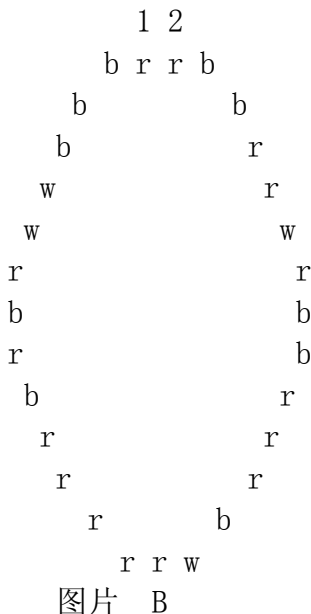
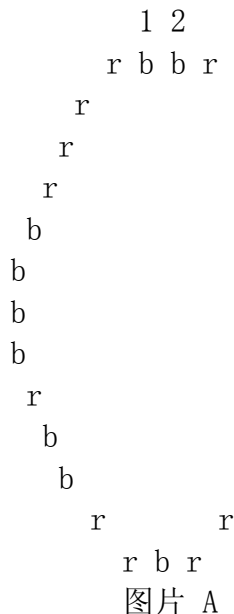
七个在一行且相分开的整数, 它们代表 13 日是星期六, 星期日, 星期一.....星期五的次数.

SAMPLE OUTPUT (file friday.out)

```
36 33 34 33 35 35 34
```

★Broken Necklace 破碎的项链

你有一条由 N 个红色的, 白色的, 或蓝色的珠子组成的项链 ($3 \leq N \leq 350$), 珠子是随意安排的. 这里是 $n=29$ 的二个例子:



r 代表 红色的珠子

b 代表 蓝色的珠子

w 代表 白色的珠子

第一和第二个珠子在图片中已经被作记号.

图片 A 中的项链可以用下面的字符串表示:

brbrrrbbrbrrrrrbrrbbrbrrrrrb .

假如你要在一点打破项链, 展开成一条直线, 然后从一端开始收集同颜色的珠子直到你遇到一个不同的颜色珠子, 在另一端做同样的事. (颜色可能与在这之前收集的不同) 确定应该在哪里打破项链来收集到最大数目的珠子. Example 举例来说, 在图片 A 中的项链, 可以收集到 8 个珠子, 在珠子 9 和珠子 10 或珠子 24 和珠子 25 之间打断项链. 在一些项链中, 包括白色的珠子如图片 B 所示. 当收集珠子的时候, 一个被遇到的白色珠子可以被当做红色也可以被当做蓝色. 表现项链的字符串将会包括三符号 r, b 和 w. 写一个程序来确定从一条被供应的项链最大可以被收集珠子数目.

PROGRAM NAME: beads

INPUT FORMAT

第 1 行: N , 珠子的数目

第 2 行: 一串度为 N 的字符串, 每个字符是 r, b 或 w.

SAMPLE INPUT (file beads.in)

29

wwwbbrwrbrbrrrbrrbrwrwwrbwrwrrb

OUTPUT FORMAT

单独的一行包含从被供应的项链可以被收集的珠子数目的最大值.

SAMPLE OUTPUT (file beads.out)

11

Section1.2

★Milking Cows 挤牛奶

三个农民每天清晨 5 点起床, 然后去牛棚给 3 头牛挤奶. 第一个农民在 300 时刻(从 5 点开始计时, 秒为单位)给他的牛挤奶, 一直到 1000 时刻. 第二个农民在 700 时刻开始, 在 1200 时刻结束. 第三个农民在 1500 时刻开始 2100 时刻结束. 期间最长的至少有一个农民在挤奶的连续时间为 900 秒(从 300 时刻到 1200 时刻), 而最长的无人挤奶的连续时间(从挤奶开始一直到挤奶结束)为 300 秒(从 1200 时刻到 1500 时刻).

你的任务是编一个程序, 读入一个有 N 个农民 ($1 \leq N \leq 5000$) 挤 N 头牛的工作时间列表, 计算以下两点(均以秒为单位):

- 最长至少有一人在挤奶的时间段.
- 最长的无人挤奶的时间段.

PROGRAM NAME: milk2

INPUT FORMAT

Line 1: 一个整数 N .

Lines 2.. N +1: 每行两个小于 1000000 的非负整数, 表示一个农民的开始时刻与结束时刻.

SAMPLE INPUT (file milk2.in)

```
3
300 1000
700 1200
1500 2100
```

OUTPUT FORMAT

一行, 两个整数, 即题目所要求的两个答案.

SAMPLE OUTPUT (file milk2.out)

```
900 300
```

★Transformations 方块转换

一块 $N \times N$ ($1 \leq N \leq 10$) 正方形的黑白瓦片的图案要被转换成新的正方形图案.

写一个程序来找出将原始图案按照以下列转换方法转换成新图案的最小方式:

- #1: 转 90 度: 图案按顺时针转 90 度.
- #2: 转 180 度: 图案按顺时针转 180 度.
- #3: 转 270 度: 图案按顺时针转 270 度.
- #4: 反射: 图案在水平方向翻转 (形成原图案的镜像).
- #5: 组合: 图案在水平方向翻转, 然后按照 #1-#3 之一转换.
- #6: 不改变: 原图案不改变.
- #7: 无效转换: 无法用以上方法得到新图案.

如果有多种可用的转换方法, 请选择序号最小的那个.

PROGRAM NAME: transform

INPUT FORMAT

第一行：单独的一个整数 N.

第二行到第 N+1 行：N 行每行 N 个字符（不是 “@” 就是 “-”）；这是转换前的正方形.

第 N+2 行到第 2*N+1 行：N 行每行 N 个字符（不是 “@” 就是 “-”）；这是转换后的正方形.

SAMPLE INPUT (file transform.in)

```
3
@-@
---
@@-
@-@
@--
--@
```

OUTPUT FORMAT

单独的一行包括 1 到 7 之间的一个数字（在上文已描述）表明需要将转换前的正方形变为转换后的正方形的

转换方法.

SAMPLE OUTPUT (file transform.out)

```
1
```

★Name That Number 命名那个数字

在威斯康辛州牛大农场经营者之中，都习惯于请会计部门用连续数字给母牛打上烙印.

但是，母牛用手机时并没感到这个系统的便利，它们更喜欢用它们喜欢的名字来呼叫它们的同伴，而不是用像这个的语句 “C’mon, #4734, get along.”.

请写一个程序来帮助可怜的牧牛工将一只母牛的烙印编号翻译成一个可能的名字.

因为母牛们现在都有手机了，使用那标准的按键的排布来把收到从数目翻译到文字：（除了为之外 “Q” 和 “Z”）

2: A, B, C	5: J, K, L	8: T, U, V
3: D, E, F	6: M, N, O	9: W, X, Y
4: G, H, I	7: P, R, S	

可接受的名字都被放在这样一个叫作 dict.txt 的文件中，它包含一连串的少于 5000 个可接受的牛名字。（所有的名字都是大写的），收到母牛的编号返回那些能从编号翻译出来并且在字典中的名字.

举例来说，编号 4734 能产生的下列各项名字：

```
GPDG GPDH GPDI GPEG GPEH GPEI GPGF GPFH GPGI GRDG GRDH GRDI
GREG GREH GREI GRFG GRFH GRFI GSDG GSDH GSDI GSEG GSEH GSEI
GSFG GSFH GSFI HPDG HPDH HPDI HPEG HPEH HPEI HPFG HPFH HPFI
HRDG HRDH HRDI HREG HREH HREI HRFH HRFI HSDG HSDH HSDI
HSEG HSEH HSEI HSFG HSFH HSFI IPDG IPDH IPDI IPEG IPEH IPEI
IPFG IPFH IPFI IRDG IRDH IRDI IREG IREH IREI IRFG IRFH IRFI
ISDG ISDH ISDI ISEG ISEH ISEI ISFG ISFH ISFI
```

碰巧，81 个中只有一个 “GREG” 是有效的（在字典中）.

Challenge One

写一个程序来对给出的编号打印出所有的有效名字，如果没有则输出 “NONE” .

编号可能有 12 位数字.

PROGRAM NAME: namenum

INPUT FORMAT

单独的一行包含一个编号(长度可能从 1 到 12).

SAMPLE INPUT (file namenum.in)

4734

OUTPUT FORMAT

以字典顺序输出一个有效名字的不负列表, 一行一个名字.

SAMPLE OUTPUT (file namenum.out)

GREG

★Palindromic Squares 回文平方数

回文数是指从左向右念和从右像做念都一样的数. 如 12321 就是一个典型的回文数.

给定一个进制 $B(2 \leq B \leq 20 \text{ 十进制})$, 输出所有的大于等于 1 小于等于 300 且它的平方用 B 进制表示时是回文数的数. 用 'A', 'B'表示 10, 11 等等.

PROGRAM NAME: palsquare

INPUT FORMAT

共一行, 一个单独的整数 B(B 用十进制表示).

SAMPLE INPUT (file palsquare.in)

10

OUTPUT FORMAT

每行两个数字, 第二个数是第一个数的平方, 且第二个数是回文数. (注意: 这两个数都应该在 B 那个进制下)

SAMPLE OUTPUT (file palsquare.out)

1 1

2 4

3 9

11 121

22 484

26 676

101 10201

111 12321

121 14641

202 40804

212 44944

264 69696

★Dual Palindromes 双重回文数

如果一个数从左往右读和从右往左读都是一样, 那么这个数就叫做“回文数”. 例如, 12321 就是一个回文数, 而 77778 就不是. 当然, 回文数的首和尾都应是非零的, 因此 0220 就不是回文数.

事实上, 有一些数(如 21), 在十进制时不是回文数, 但在其它进制(如二进制时为 10101)时就是回文数.

编一个程序, 从文件读入两个十进制数

N ($1 \leq N \leq 15$) S ($0 < S < 10000$)

然后找出前 N 个满足大于 S 且在两种以上进制 (二进制至十进制) 上是回文数的十进制数, 输出到文件上.

本问题的解决方案不需要使用大于 4 字节的整型变量.

PROGRAM NAME: dualpal

INPUT FORMAT

只有一行, 用空格隔开的两个数 N 和 S .

SAMPLE INPUT (file dualpal.in)

3 25

OUTPUT FORMAT

N 行, 每行一个满足上述要求的数, 并按从小到大的顺序输出.

SAMPLE OUTPUT (file dualpal.out)

26

27

28

Section 1.3

★Mixing Milk 混合牛奶

牛奶包装是一个如此低利润的生意, 所以尽可能低的控制初级产品 (牛奶) 的价格变的十分重要.

请帮助快乐的牛奶制造者 (Merry Milk Makers) 以可能的最廉价的方式取得他们所需的牛奶.

快乐的牛奶制造公司从一些农民那购买牛奶, 每个农民卖给牛奶制造公司的价格不一定相同.

而且, 如一只母牛一天只能生产一定量的牛奶, 农民每一天只有一定量的牛奶可以卖.

每天, 快乐的牛奶制造者从每个农民那购买一定量的牛奶, 少于或等于农民所能提供的最大值.

给出快乐牛奶制造者的每日的牛奶需求, 连同每个农民的可提供的牛奶量和每加仑的价格, 请计算快乐的牛奶制造者所要付出钱的最小值.

注意:

每天农民生产的牛奶的总数对快乐的牛奶制造者来说足够的.

PROGRAM NAME: milk

INPUT FORMAT

第 1 行: 二个整数, N 和 M .

第一个数值, N , ($0 \leq N \leq 2,000,000$) 是快乐的牛奶制造者的一天需要牛奶的数量.

第二个数值, M , ($0 \leq M \leq 5,000$) 是他们可能从农民那买到的数目.

第 2 到 $M+1$ 行: 每行二个整数, P_i 和 A_i .

P_i ($0 \leq P_i \leq 1,000$) 是农民 i 牛奶的价格.

A_i ($0 \leq A_i \leq 2,000,000$) 是农民 i 一天能卖给快乐的牛奶制造者的牛奶数量.

SAMPLE INPUT (file milk.in)

100 5

5 20

9 40

```
3 10
8 80
6 30
```

OUTPUT FORMAT

单独的一行包含单独的一个整数, 表示快乐的牛奶制造者拿到所需的牛奶所要的最小费用

SAMPLE OUTPUT (file milk.out)

```
630
```

★Barn Repair 修理牛棚

在一个暴风雨的夜晚, 农民约翰的牛棚的屋顶、门被吹飞了。好在许多牛正在度假, 所以牛棚没有住满。剩下的牛一个紧挨着另一个被排成一行来过夜。有些牛棚里有牛, 有些没有。所有的牛棚有相同的宽度。自门遗失以后, 农民约翰很快在牛棚之前竖立起新的木板。他的新木材供应者将会供应他任何他想要的长度, 但是供应者只能提供有限数目的木板。农民约翰想将他购买的木板总长度减到最少。给出 M ($1 \leq M \leq 50$), 可能买到的木板最大的数目; S ($1 \leq S \leq 200$), 牛棚的总数; C ($1 \leq C \leq S$) 牛棚里牛的数目, 和牛所在的牛棚的编号 $stall_number$ ($1 \leq stall_number \leq S$), 计算拦住所有有牛的牛棚所需木板的最小总长度。输出所需木板的最小总长度作为的答案。

PROGRAM NAME: barn1

INPUT FORMAT

第 1 行: M , S 和 C (用空格分开)

第 2 到 $C+1$ 行: 每行包含一个整数, 表示牛所占的牛棚的编号。

SAMPLE INPUT (file barn1.in)

```
4 50 18
3
4
6
8
14
15
16
17
21
25
26
27
30
31
40
41
42
43
```

OUTPUT FORMAT

单独的一行包含一个整数表示所需木板的最小总长度。

SAMPLE OUTPUT (file barn1.out)

25 {一种最优的安排是用板拦住牛棚 3-8, 14-21, 25-31, 40-43. }

★Calf Flac 最长的回文

据说如果你给无限只母牛和无限台巨型便携式电脑(有非常大的键盘), 那么母牛们会制造出世上最棒的回文.

你的工作就是去这些牛制造的奇观中寻找最长的回文.

寻找回文时不用理睬那些标点符号、空格(但应该保留下来以便做为答案输出), 只用考虑字母 'A' - 'Z' 和 'a' - 'z'.

要你寻找的最长的回文的文章是一个不超过 20,000 个字符的字符串.

我们将保证最长的回文不会超过 2,000 个字符(在除去标点符号、空格之前).

PROGRAM NAME: calfflac

INPUT FORMAT

一个不超过 20,000 个字符的文件.

SAMPLE INPUT (file calfflac.in)

Confucius say: Madam, I'm Adam.

OUTPUT FORMAT

输出的第一行应该包括找到的最长的回文的长度.

下一个行或几行应该包括这个回文的原文(没有除去标点符号、空格), 把这个回文输出到一行或多行(如果回文中包括换行符).

如果有多个回文长度都等于最大值, 输出那个前出现的.

SAMPLE OUTPUT (file calfflac.out)

11

Madam, I'm Adam

★Prime Cryptarithm 牛式

下面是一个乘法竖式, 如果用我们给定的那几个数字来取代*, 可以使式子成立的话, 我们就叫这个式子牛式.

```

  * * *
x   * *
-----
    * * *
  * * *
-----
 * * * *
```

数字只能取代*, 当然第一位不能为 0.

写一个程序找出所有的牛式.

PROGRAM NAME: crypt1

INPUT FORMAT

Line 1: 数字的个数.
Line 2: N个用空格分开的数字 (每个数字都 $\in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$) .

SAMPLE INPUT (file crypt1.in)

5
2 3 4 6 8

OUTPUT FORMAT

共一行, 一个数字. 表示牛式的总数. 下面是样例的那个牛式.

```

  2 2 2
x   2 2
-----
  4 4 4
  4 4 4
-----
  4 8 8 4
```

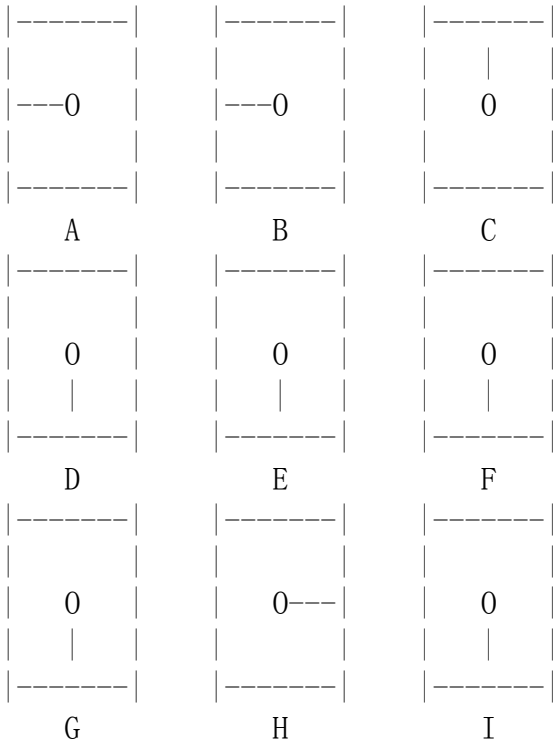
SAMPLE OUTPUT (file crypt1.out)

1

Section1.4

★The Clocks 时钟

IOI' 94 - Day 2
考虑将如此安排在一个 3X3 行列中的九个时钟:



目标要找一个最小的移动顺序次将所有的指针指向 12 点.

下面原表格列出了 9 种不同的旋转指针的方法, 每一种方法都叫一次移动.
 选择 1 到 9 号移动方法, 将会使在表格中对应的时钟的指针顺时针旋转 90 度.

移动方法	受影响的时钟
1	ABDE
2	ABC
3	BCEF
4	ADG
5	BDEFH
6	CFI
7	DEGH
8	GHI
9	EFHI

Example

```

9 9 12      9 12 12      9 12 12      12 12 12      12 12 12
6 6 6  5 -> 9 9 9  8->  9   9 9 4 ->  12 9 9 9->  12 12 12
6 3 6      6 6 6      9 9 9      12 9 9      12 12 12
  
```

[但这可能不是正确的方法, 请看下面]

PROGRAM NAME: clocks

INPUT FORMAT

第 1-3 行: 三个空格分开的数字, 每个数字表示一个时钟的初始时间, 3, 6, 9, 12.
 数字的含意和上面第一个例子一样.

SAMPLE INPUT (file clocks.in)

```

9 9 12
6 6 6
6 3 6
  
```

OUTPUT FORMAT

单独的一行包括一个用空格分开的将所有指针指向 12:00 的最短移动顺序的列表.
 如果有多种方案, 输出那种使的连接起来数字最小的方案. (举例来说 5 2 4 6 < 9 3 1 1).

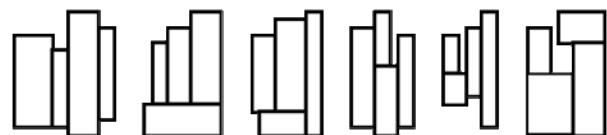
SAMPLE OUTPUT (file clocks.out)

```

4 5 8 9
  
```

★Packing Rectangles 铺放矩形块

I0I 95



给定 4 个矩形块, 找出一个最小的封闭矩形将这 4 个矩形块放入, 但不得相互重叠. 所谓最小矩形指该矩形面积最小.

所有 4 个矩形块的边都与封闭矩形的边相平行, 图 1 示出了铺放 4 个矩形块的 6 种方案. 这 6 种方案仅只是可能的基本铺放方案. 因为其它方案能由基本方案通过旋转和镜像反射得到.

可能存在满足条件且有着同样面积的各种不同的封闭矩形, 你应该输出所有这些封闭矩形的边长.

PROGRAM NAME: packrec

INPUT FORMAT

共有 4 行. 每一行用两个正整数来表示一个给定的矩形块的两个边长. 矩形块的每条边的边长范围最小是 1, 最大是 50.

SAMPLE INPUT (file packrec.in)

```
1 2
2 3
3 4
4 5
```

OUTPUT FORMAT

总行数为解的总数加 1. 第一行是一个整数, 代表封闭矩形的最小面积 (子任务 A). 接下来的每一行都表示一个解, 由数 P 和数 Q 来表示, 并且 $P \leq Q$ (子任务 B). 这些行必须根据 P 的大小按升序排列, P 小的行在前, 大的在后. 且所有行都应是不同的.

SAMPLE OUTPUT (file packrec.out)

```
40
4 10
5 8
```

★Arithmetic Progressions 等差数列

一个等差数列是一个能表示成 $a, a+b, a+2b, \dots, a+nb$ ($n=0, 1, 2, 3, \dots$)

在这个问题中 a 是一个非负的整数, b 是正整数.

写一个程序来找出在双平方数集合 S 中长度为 n 的等差数列.

双平方数集合是所有能表示成 p^2+q^2 的数的集合.

PROGRAM NAME: ariprog

INPUT FORMAT

第一行: $N(3 \leq N \leq 25)$, 要找的等差数列的长度.

第二行: $M(1 \leq M \leq 250)$, 搜索双平方数的上界 $0 \leq p, q \leq M$.

SAMPLE INPUT (file ariprog.in)

```
5
7
```

OUTPUT FORMAT

如果没有找到数列, 输出 'NONE'.

如果找到了, 输出一行或多行, 每行由于二个整数组成: a, b

这些行应该先按 b 排序再按 a 排序.

将不会有只多于 10,000 个等差数列.

SAMPLE OUTPUT (file ariprog.out)

```
1 4
37 4
2 8
29 8
1 12
5 12
```


13 12
17 12
5 20
2 24

★Mother's Milk 母亲的牛奶

农民约翰有三个容量分别是 A, B, C 升的桶, A, B, C 分别是三个从 1 到 20 的整数, 最初, A 和 B 桶都是空的, 而 C 桶是装满牛奶的. 有时, 约翰把牛奶从一个桶倒到另一个桶中, 直到被灌桶装满或原桶空了. 当然每一次灌注都是完全的. 由于节约, 牛奶不会有丢失.

写一个程序去帮助约翰找出当 A 桶是空的时候, C 桶中牛奶所剩量的所有可能性.

PROGRAM NAME: milk3

INPUT FORMAT

单独的一行包括三个整数 A, B 和 C.

SAMPLE INPUT (file milk3.in)

8 9 10

OUTPUT FORMAT

只有一行, 列出当 A 桶是空的时候, C 桶牛奶所剩量的所有可能性.

SAMPLE OUTPUT (file milk3.out)

1 2 8 9 10

SAMPLE INPUT (file milk3.in)

2 5 10

SAMPLE OUTPUT (file milk3.out)

5 6 7 8 9 10

Section 1.5

★Number Triangles 数字金字塔

考虑在下面被显示的数字金字塔.

写一个程序来计算从最高点开始在底部任意处结束的路径经过数字的和的最大.

每一步可以走到左下方的点也可以到达右下方的点.

7
3 8
8 1 0
2 7 4 4
4 5 2 6 5

在上面的样例中, 从 7 到 3 到 8 到 7 到 5 的路径产生了最大和:30

PROGRAM NAME: numtri

INPUT FORMAT

第一个行包含 R ($1 \leq R \leq 1000$) , 表示行的数目.
后面每行为这个数字金字塔特定行包含的整数.
所有的被供应的整数是非负的且不大于 100.

SAMPLE INPUT (file numtri.in)

```
5
7
3 8
8 1 0
2 7 4 4
4 5 2 6 5
```

OUTPUT FORMAT

单独的一行包含那个可能得到的最大的和.

SAMPLE OUTPUT (file numtri.out)

```
30
```

★Prime Palindromes 回文质数

因为 151 既是一个质数又是一个回文数(从左到右和从右到左是看一样的), 所以 151 是回文质数.
写一个程序来找出范围 $[a, b]$ ($5 \leq a < b \leq 100,000,000$) 间的所有回文质数;

PROGRAM NAME: pprime

INPUT FORMAT

第 1 行: 二个整数 a 和 b .

SAMPLE INPUT (file pprime.in)

```
5 500
```

OUTPUT FORMAT

输出一个回文质数的列表, 一行一个.

SAMPLE OUTPUT (file pprime.out)

```
5
7
11
101
131
151
181
191
313
353
373
383
```

★Superprime Rib 特殊的质数肋骨

农民约翰母牛总是产生最好的肋骨.

你能通过农民约翰和美国农业部标记在每根肋骨上的数字认出它们.

农民约翰确定他卖给买方的是真正的质数肋骨, 是因为从右边开始切下肋骨, 每次还剩下的肋骨上的数字都组成一个质数, 举例来说:

7 3 3 1

全部肋骨上的数字 7331 是质数; 三根肋骨 733 是质数; 二根肋骨 73 是质数; 当然, 最后一根肋骨 7 也是质数.

7331 被叫做长度 4 的特殊质数.

写一个程序对给定的肋骨的数目 N ($1 \leq N \leq 8$), 求出所有的特殊质数.

数字 1 不被看作一个质数.

PROGRAM NAME: sprime

INPUT FORMAT

单独的一行包含 N .

SAMPLE INPUT (file sprime.in)

4

OUTPUT FORMAT

按顺序输出长度为 N 的特殊质数, 每行一个.

SAMPLE OUTPUT (file sprime.out)

2333

2339

2393

2399

2939

3119

3137

3733

3739

3793

3797

5939

7193

7331

7333

7393

★Checker Challenge 跳棋的挑战>

检查一个如下的 6 x 6 的跳棋棋盘, 有六个棋子被放置在棋盘上, 使得每行, 每列, 每条对角线 (包括两条主对角线的所有对角线) 上都至多有一个棋子.

	1	2	3	4	5	6
1			0			
2				0		
3						0
4	0					
5			0			
6					0	

上面的布局可以用序列 2 4 6 1 3 5 来描述, 第 i 个数字表示在第 i 行的相应位置有一个棋子, 如下:

行号 1 2 3 4 5 6

列号 2 4 6 1 3 5

这只是跳棋放置的一个解. 请编一个程序找出所有跳棋放置的解. 并把它们以上面的序列方法输出. 解按字典顺序排列. 请输出前 3 个解. 最后一行是解的总个数.

特别注意: 对于更大的 N (棋盘大小 $N \times N$) 你的程序应当改进得更有效. 不要事先计算出所有解然后只输出, 这是作弊. 如果你坚持作弊, 那么你登陆 USACO Training 的帐号将被无警告删除

PROGRAM NAME: checker

INPUT FORMAT

一个数字 N ($6 \leq N \leq 13$) 表示棋盘是 $N \times N$ 大小的.

SAMPLE INPUT (checker.in)

6

OUTPUT FORMAT

前三行为前三个解, 每个解的两个数字之间用一个空格隔开. 第四行只有一个数字, 表示解的总数.

SAMPLE OUTPUT (checker.out)

2 4 6 1 3 5

3 6 2 5 1 4

4 1 5 2 6 3

4

CHAPTER 2

Section 2.1

★The Castle 城堡

以一个几乎超乎想像的运气, 农民约翰在他的生日收到了一张爱尔兰博彩的奖券. 这一张奖券成为了唯一中奖的奖券. 农民约翰赢得爱尔兰的乡下地方的一个传说中的城堡.

吹牛在他们威斯康辛州不算什么, 农民约翰想告诉他的牛所有有关城堡的事. 他想知道城堡有多少房间, 而且最大的房间有多大. 事实上, 他想去掉一面墙来制造一个更大的房间.

你的任务是帮助农民约翰去了解正确房间数目和大小.

城堡的平面图被分为 $M(\text{wide}) * N (1 \leq M, N \leq 50)$ 个小正方形.

每个这样的小正方形有 0 到 4 面墙.

城堡在它的外部边缘总是有墙壁的, 好遮挡风雨.

考虑这注解了一个城堡的平面图:

```

  1   2   3   4   5   6   7
#####
1 #   |   #   |   #   |   |   #
  #####---#####---#---#####---#
2 #   #   |   #   #   #   #   #
  #---#####---#####---#####---#
3 #   |   |   #   #   #   #   #
  #---#####---#####---#---#
4 # ->#   |   |   |   |   #   #
  #####
```

= 墙壁 -, | = 没有墙壁

-> = 移除这面墙能使得到的新房间最大

例子的城堡的大小是 7 x 4.

一个“房间”是平面图上有连接的“小正方形”的集合.

这个平面图包含五个房间. (它们的大小是 9, 7, 3, 1, 和 8 排列没有特别的顺序).

移除被箭作记号的墙壁来合并两个房间来制造最大的可能房间 (移除一面墙所能产生的).

城堡总是至少有二个房间并且总是有一面墙壁以可能被移除.

PROGRAM NAME: castle

INPUT FORMAT

地图以一个表格来储存, 每个数字描述一个小正方形, N 行每行 M 个数来描述这个平面图.

输入顺序符合那个在上面例子的编号方式.

每个描述小正方形的数字说明小正方形的四面的墙的分布情况, 它是下面 4 个数的和:

1: 在西面有墙

2: 在北面有墙

4: 在东面有墙

8: 在南面有墙

内部的墙壁是会被定义两次；小正方形(1, 1)南面的墙也被指出是小正方形(2, 1)北面的墙.

第 1 行: 二个被空格分开的整数: M 和 N

第 2 到: N+1 行: M x N 个整数, 每行 M 个.

SAMPLE INPUT (file castle.in)

```
7 4
11 6 11 6 3 10 6
7 9 6 13 5 15 5
1 10 12 7 13 7 5
13 11 10 8 10 12 13
```

OUTPUT FORMAT

输出包含一些行:

第 1 行: 城堡的房间数目.

第 2 行: 最大的房间的大小

第 3 行: 移除一面墙能得到的最大的房间的大小

第 4 行: 移除哪面墙

选择最佳的墙来移除, (选择最靠西的, 如果仍然不能确定, 再选择最靠南的. 编者注: 墙的位置应该由它的中点来定义)

(【原文】Choose the optimal wall to remove from the set of optimal walls by choosing the wall farther to the west (and then, if still tied, farthest to the south).)

墙壁由它在相邻的小正方形的西部或南方来命名

SAMPLE OUTPUT (file castle.out)

```
5
9
16
4 1 E
```

★Ordered Fractions 顺序的分数

输入一个自然数 N

请写一个程序来增序输出分母小于等于 N 的既约真分数

PROGRAM NAME: frac1

INPUT FORMAT

单独的一行 一个自然数 N(1..160)

SAMPLE INPUT (file frac1.in)

```
5
```

OUTPUT FORMAT

每个分数单独占一行

SAMPLE OUTPUT (file frac1.out)

```
0/1
1/5
1/4
1/3
```

2/5
1/2
3/5
2/3
3/4
4/5
1/1

★Sorting a Three-Valued Sequence 三值的排序

排序是一种很频繁的计算任务. 现在考虑最多只有三值的排序问题. 一个实际的例子是, 当我们给某项竞赛的优胜者按金银铜牌序的时候.

在这个任务中可能的值只有三种 1, 2 和 3. 我们用交换的方法把他排成升序的.

写一个程序计算出, 给定的一个 1, 2, 3 组成的数字序列, 排成升序所需的最少交换次数.

PROGRAM NAME: sort3

INPUT FORMAT

Line 1: N (1 ≤ N ≤ 1000)

Lines 2-N+1: 每行一个数字, 共 N 行. (1..3)

SAMPLE INPUT (file sort3.in)

9
2
2
1
3
3
3
2
3
1

OUTPUT FORMAT

共一行, 一个数字. 表示排成升序所需的最少交换次数.

SAMPLE OUTPUT (file sort3.out)

4

★Healthy Holsteins 健康的好斯坦奶牛

农民 JOHN 以拥有世界上最健康的奶牛为骄傲. 他知道每种饲料中所包含的的牛所需的最低的维他命量是多少. 请你帮助农夫喂养他的牛, 以保持他们的健康, 使喂给牛的饲料的种数最少.

给出牛所需的最低的维他命, 输出喂给牛需要哪些种类的饲料, ? 宜 璧闹掷喇 钉佟?/FONT>

PROGRAM NAME: holstein

INPUT FORMAT

第 1 行:一个整数 $V(1 \leq V \leq 25)$, 表示需要的维他命的种类数.

第 2 行: V 个整数 ($1 \leq \text{每个数} \leq 1000$), 表示牛每天需要的维他命的最小量.

第 3 行:一个整数 $G(1 \leq G \leq 15)$, 表示可用来喂牛的饲料的数量. 下面 G 行, 第 i 行表示编号为 i 饲料包含的各种维他命的量的多少.

SAMPLE INPUT (file holstein.in)

```
4
100 200 300 400
3
50 50 50 50
200 300 200 300
900 150 389 399
```

OUTPUT FORMAT

输出文件只有一行, 包括:

牛必需的最小的饲料种数 P

后面有 P 个数, 表示所选择的饲料编号 (按从小到大排列).

SAMPLE OUTPUT (file holstein.out)

```
2 1 3
```

★Hamming Codes 海明码

给出 N, B 和 D : 找出 N 个编码 ($1 \leq N \leq 64$), 每个编码有 B 位 ($1 \leq B \leq 8$), 使得两两编码之间至少有 D 个单位的“海明距离” ($1 \leq D \leq 7$). “海明距离”是指对于两个编码, 他们的二进制表示法中的不同二进制位的数目. 看下面的两个编码 $0x554$ 和 $0x234$ 之间的区别 ($0x554$ 表示一个十六进制数, 每个位上分别是 5, 5, 4):

$0x554 = 0101\ 0101\ 0100$

$0x234 = 0010\ 0011\ 0100$

不同的二进制位: xxx xx

因为有五个位不同, 所以“海明距离”是 5.

PROGRAM NAME: hamming

INPUT FORMAT

一行, 包括 N, B, D .

SAMPLE INPUT (file hamming.in)

```
16 7 3
```

OUTPUT FORMAT

N 个编码 (用十进制表示), 要排序, 十个一行. 如果有多解, 你的程序要输出这样的解: 假如把它化为 2^B 进制的数, 它的值要最小.

SAMPLE OUTPUT (file hamming.out)

```
0 7 25 30 42 45 51 52 75 76
82 85 97 102 120 127
```


Section2.2

★Preface Numbering 序言页码

一类书的序言是以罗马数字标页码的. 传统罗马数字用单个字母表示特定的数值, 一下是标准数字表:

I 1 L 50 M 1000

V 5 C 100

X 10 D 500

最多 3 个可以表示为 10^n 的数字 (I, X, C, M) 可以连续放在一起, 表示它们的和:

III=3

CCC=300

可表示为 5×10^n 的字符 (V, L, D) 从不连续出现.

除了下一个规则, 一般来说, 字符以递减的顺序接连出现:

CCLXVIII = $100+100+50+10+5+1+1+1 = 268$

有时, 一个可表示为 10^n 的数出现在一个比它大的数前 (I 在 V 或 X 前面, X 在 L 或 C 前面, 等等). 在这种情况下, 数值等于后面的那个数减去前面的那个数:

IV = 4

IX = 9

XL = 40

像 XD, IC, 和 XM 这样的表达是非法的, 因为前面的数比后面的数小太多. 对于 XD (490 的错误表达), 可以写成 CDXC; 对于 IC (99 的错误表达), 可以写成 XCIX; 对于 XM (990 的错误表达), 可以写成 CMXC.

给定 $N (1 \leq N < 3,500)$, 序言的页码数, 请统计在第 1 页到第 N 页中, 有几个 I 出现, 几个 V 出现, 等等 (从小到大的顺序). 不要输出并没有出现过的字符.

比如 $N = 5$, 那么页码数为: I, II, III, IV, V. 总共有 7 个 I 出现, 2 个 V 出现.

PROGRAM NAME: preface

INPUT FORMAT

一个整数 N.

SAMPLE INPUT (preface.in)

5

OUTPUT FORMAT

每行一个字符和一个数字 k, 表示这个字符出现了 k 次. 字符必须按数字表中的递增顺序输出.

SAMPLE OUTPUT (preface.out)

I 7

V 2

★Subset Sums 集合

对于从 1 到 N 的连续整集合, 能划分成两个子集合, 且保证每个集合的数字和是相等的. 举个例子, 如果 $N=3$, 对于 $\{1, 2, 3\}$ 能划分成两个子集合, 他们每个的所有数字和是相等的:

$\{3\}$ and $\{1, 2\}$

这是唯一一种分发（交换集合位置被认为是同一种划分方案, 因此不会增加划分方案总数）
如果 $N=7$, 有四种方法能划分集合 $\{1, 2, 3, 4, 5, 6, 7\}$, 每一种分发的子集合各数字和是相等的:

$\{1, 6, 7\}$ and $\{2, 3, 4, 5\}$ {注 $1+6+7=2+3+4+5$ }

$\{2, 5, 7\}$ and $\{1, 3, 4, 6\}$

$\{3, 4, 7\}$ and $\{1, 2, 5, 6\}$

$\{1, 2, 4, 7\}$ and $\{3, 5, 6\}$

给出 N , 你的程序应该输出划分方案总数, 如果不存在这样的划分方案, 则输出 0. 程序不能预存结果直接输出.

PROGRAM NAME: subset

INPUT FORMAT

输入文件只有一行, 且只有一个整数 N

SAMPLE INPUT (file subset.in)

7

OUTPUT FORMAT

输出划分方案总数, 如果不存在则输出 0.

SAMPLE OUTPUT (file subset.out)

4

★Runaround Numbers 循环数

循环数是那些不包括 0 这个数字的没有重复数字的整数（比如说, 81362）并且同时具有一个有趣的性质, 就像这个例子:

如果你从最左边的数字开始（在这个例子中是 8）数最左边这个数字个数字到右边（回到最左边如果数到了最右边）, 你会停止在另一个新的数字（如果没有停在一个不同的数字上, 这个数就不是循环数）. 就像: 8 1 3 6 2 从最左边接下去数 8 个数字: 1 3 6 2 8 1 3 6 所以下一个数字是 6. 重复这样做（这次从“6”开始数 6 个数字）并且你会停止在一个新的数字上: 2 8 1 3 6 2, 也就是 2.

再这样做（这次数两个）: 8 1

再一次（这次一个）: 3

又一次: 6 2 8 这是你回到了起点, 在从每一个数字开始数 1 次之后. 如果你在从每一个数字开始数一次以后没有回到起点, 你的数字不是一个循环数.

给你一个数字 M (在 1 到 9 位之间), 找出第一个比 M 大的循环数, 并且一定能用一个无符号长整形数装下.

PROGRAM NAME: runround

INPUT FORMAT

仅仅一行, 包括 M

SAMPLE INPUT (file runround.in)

81361

OUTPUT FORMAT

仅仅一行, 包括第一个比 M 大的循环数.

SAMPLE OUTPUT (file runround.out)

81362

★Party Lamps 派对灯

在 IOI98 的节日宴会上, 我们有 N ($10 \leq N \leq 100$) 盏彩色灯, 他们分别从 1 到 N 被标上号码.

这些灯都连接到四个按钮:

按钮 1: 当按下此按钮, 将改变所有的灯: 本来亮着的灯就熄灭, 本来是关着的灯被点亮.

按钮 2: 当按下此按钮, 将改变所有奇数号的灯.

按钮 3: 当按下此按钮, 将改变所有偶数号的灯.

按钮 4: 当按下此按钮, 将改变所有序号是 $3 \cdot K + 1$ ($K \geq 0$) 的灯. 例如: 1, 4, 7...

一个计数器 C 记录按钮被按下的次数.

当宴会开始, 所有的灯都亮着, 此时计数器 C 为 0.

你将得到计数器 C ($0 \leq C \leq 10000$) 上的数值和经过若干操作后所有灯的状态. 写一个程序去找出所有灯最后可能的与所给出信息相符的状态, 并且没有重复.

PROGRAM NAME: lamps

INPUT FORMAT

不会有灯会在输入中出现两次.

第一行: N .

第二行: C 最后显示的数值.

第三行: 最后亮着的灯, 用一个空格分开, 以 -1 为结束.

第四行: 最后关着的灯, 用一个空格分开, 以 -1 为结束.

SAMPLE INPUT (file lamps.in)

```
10
1
-1
7 -1
```

在这个样例中, 有 10 盏灯, 只有 1 个按钮被按下. 最后 7 号灯是关着的.

OUTPUT FORMAT

每一行是所有灯可能的最后状态 (没有重复). 每一行有 N 个字符, 第 1 个字符表示 1 号灯, 最后一个字符表示 N 号灯. 0 表示关闭, 1 表示亮着. 这些行必须从小到大排列 (看作是二进制数).

如果没有可能的状态, 则输出一行 'IMPOSSIBLE'.

SAMPLE OUTPUT (file lamps.out)

```
0000000000
0101010101
0110110110
```

在这个样例中, 有三种可能的状态:

所有灯都关着

1, 4, 7, 10 号灯关着, 2, 3, 5, 6, 8, 9 亮着.

1, 3, 5, 7, 9 号灯关着, 2, 4, 6, 8, 10 亮着.

Section2.3

★Longest Prefix 最长前缀

在生物学中, 一些生物的结构是用包含其要素的大写字母序列来表示的. 生物学家对于把长的序列

分解成较短的（称之为元素的）序列很感兴趣.

如果一个集合 P 中的元素可以通过串联（允许重复；串联, 相当于 Pascal 中的 “+” 运算符）组成一个序列 S , 那么我们认为序列 S 可以分解为 P 中的元素. 并不是所有的元素都必须出现. 举个例子, 序列 ABABACABAAB 可以分解为下面集合中的元素:

{A, AB, BA, CA, BBC}

序列 S 的前面 K 个字符称作 S 中长度为 K 的前缀. 设计一个程序, 输入一个元素集合以及一个大写字母序列, 计算这个序列最长的前缀的长度.

PROGRAM NAME: prefix

INPUT FORMAT

输入数据的开头包括 1..200 个元素（长度为 1..10 ）组成的集合, 用连续的以空格分开的字符串表示. 字母全部是大写, 数据可能不止一行. 元素集合结束的标志是一个只包含一个 “.” 的行. 集合中的元素没有重复. 接着是大写字母序列 S , 长度为 1..200,000, 用一行或者多行的字符串来表示, 每行不超过 76 个字符. 换行符并不是序列 S 的一部分.

SAMPLE INPUT (file prefix.in)

A AB BA CA BBC

ABABACABAABC

OUTPUT FORMAT

只有一行, 输出一个整数, 表示 S 能够分解成 P 中元素的最长前缀的长度.

SAMPLE OUTPUT (file prefix.out)

11

★Cow Pedigrees 奶牛家谱

农民约翰准备购买一群新奶牛. 在这个新的奶牛群中, 每一个母亲奶牛都生两小奶牛. 这些奶牛间的关系可以用二叉树来表示. 这些二叉树总共有 N 个节点 ($3 \leq N < 200$). 这些二叉树有如下性质: 每一个节点的度是 0 或 2. 度是这个节点的孩子的数目.

树的高度等于 K ($1 < K < 100$). 高度是从根到任何叶子的最长的路径上的节点的数目; 叶子是指没有孩子的节点.

有多少不同的家谱结构? 如果一个家谱的树结构不同于另一个的, 那么这两个家谱就是不同的. 输出可能的家谱树的个数除以 9901 的余数.

PROGRAM NAME: nocows

INPUT FORMAT

第 1 行: 两个空格分开的整数, N 和 K .

SAMPLE INPUT (file nocows.in)

5 3

OUTPUT FORMAT

第 1 行: 一个整数, 表示可能的家谱树的个数除以 9901 的余数.

SAMPLE OUTPUT (file nocows.out)

2

OUTPUT DETAILS:

有 5 个节点, 高为 3 的两个不同的家谱:



★Zero Sum 和为零

请考虑一个由 1 到 N ($N=3, 4, 5 \dots 9$) 的数字组成的递增数列: 1 2 3 ... N.

现在请在数列中插入 “+” 表示加, 或者 “-” 表示减, 抑或是 “ ” 表示空白, 来将每一对数字组合在一起 (请不在第一个数字前插入符号).

计算该表达式的结果并注意你是否得到了和为零.

请你写一个程序找出所有产生和为零的长度为 N 的数列.

PROGRAM NAME: zerosum

INPUT FORMAT

单独的一行表示整数 N ($3 \leq N \leq 9$).

SAMPLE INPUT (file zerosum.in)

7

OUTPUT FORMAT

按照 ASCII 码的顺序, 输出所有在每对数字间插入 “+”, “-”, 或 “ ” 后能得到和为零的数列. (注意: 就算两个数字之间没有插入符号也应该保留空格)

SAMPLE OUTPUT (file zerosum.out)

```
1+2-3+4-5-6+7
1+2-3-4+5+6-7
1-2 3+4+5+6+7
1-2 3-4 5+6 7
1-2+3+4-5+6-7
1-2-3-4-5+6+7
```

★Money Systems 货币系统

母牛们不但创建了他们自己的政府而且选择了建立了自己的货币系统.

[In their own rebellious way],, 他们对货币的数值感到好奇.

传统地, 一个货币系统是由 1, 5, 10, 20 或 25, 50, 和 100 的单位面值组成的.

母牛想知道有多少种不同的方法来用货币系统中的货币来构造一个确定的数值.

举例来说, 使用一个货币系统 {1, 2, 5, 10, ...} 产生 18 单位面值的一些可能的方法是: 18x1, 9x2, 8x2+2x1, 3x5+2+1, 等等其它.

写一个程序来计算有多少种方法用给定的货币系统来构造一定数量的面值.

保证总数将会适合 long long (C/C++) 和 Int64 (Free Pascal).

PROGRAM NAME: money

INPUT FORMAT

货币系统中货币的种类数目是 V . ($1 \leq V \leq 25$)

要构造的数量钱是 N . ($1 \leq N \leq 10,000$)

第 1 行: 二整数, V 和 N

第 2 .. $V+1$ 行: 可用的货币 V 个整数 (每行一个 每行没有其它的数).

SAMPLE INPUT (file money.in)

```
3 10
```

```
1 2 5
```

OUTPUT FORMAT

单独的一行包含那个可能的构造的方案数.

SAMPLE OUTPUT (file money.out)

```
10
```

★Controlling Companies 控制公司

有些公司其他公司的部分所有者, 因为他们获得了其他公司发行的股票的一部分. 例如, 福特公司拥有马自达公司 12% 的股票. 据说, 如果至少满足了以下条件之一, 公司 A 就可以控制公司 B 了:

公司 A = 公司 B.

公司 A 拥有大于 50% 的公司 B 的股票.

公司 A 控制 K ($K \geq 1$) 个公司, 记为 C_1, \dots, C_K , 每个公司 C_i 拥有 $x_i\%$ 的公司 B 的股票, 并且 $x_1 + \dots + x_K > 50\%$.

你将被给予一系列的三对数 (i, j, p) , 表明公司 i 享有公司 j 的 $p\%$ 的股票. 计算所有的数对 (h, s) , 表明公司 h 控制公司 s . 至多有 100 个公司.

写一个程序读入三对数 (i, j, p) , i, j 和 p 是都在范围 $(1..100)$ 的正整数, 并且找出所有的数对 (h, s) , 使得公司 h 控制公司 s .

PROGRAM NAME: concom

INPUT FORMAT

第一行: N , 表明接下来三对数的数量.

第二行到第 $N+1$ 行: 每行三个整数作为一个三对数 (i, j, p) , 如上文所述.

SAMPLE INPUT (file concom.in)

```
3
```

```
1 2 80
```

```
2 3 80
```

```
3 1 20
```

OUTPUT FORMAT

输出零个或更多的控制其他公司的公司. 每行包括两个整数表明序号为第一个整数的公司控制了序号为第二个整数的公司. 将输出的每行以第一个数字升序排列 (并且第二个数字也升序排列来避免并列). 请不要输出控制自己的公司.

SAMPLE OUTPUT (file concom.out)

```
1 2
```

```
1 3
```

```
2 3
```

Section2.4

★The Tamworth Two 两只塔姆沃斯牛

两只牛在森林里故意走丢了. 农民 John 开始用他的专家技术追捕这两头牛. 你的任务是模拟他们的行为(牛和 John). 追击在 10x10 的平面网格内进行. 一个格子可以是: 一个障碍物, 两头牛(它们总在一起), 或者农民 John.

两头牛和农民 John 可以在同一个格子内(当他们相遇时), 但是他们都不能进入有障碍的格子.

一个格子可以是:

. 空地 * 障碍物 C 两头牛 F 农民 John

这里有一个地图的例子::

```
*...*.
.....*
...*...*
.....
...*.F...
*...*.
...*.....
..C.....*
...*.*...
.*.*.....
```

牛在地图里以固定的方式游荡. 每分钟, 它们可以向前移动或是转弯. 如果前方无障碍且不会离开地图, 它们会按照原来的方向前进一步. 否则它们会用这一分钟顺时针转 90 度.

农民 John, 深知牛的移动方法, 他也这么移动.

每次(每分钟)农民 John 和两头牛的移动是同时的. 如果他们在移动的时候穿过对方, 但是没有在同一格相遇, 我们不认为他们相遇了. 当他们在某分钟末在某格子相遇, 那么追捕结束. 开始时, John 和牛都面向北方.

PROGRAM NAME: ttwo

INPUT FORMAT

Lines 1-10:

每行 10 个字符, 表示如上文描述的地图.

SAMPLE INPUT (file ttwo.in)

```
*...*.
.....*
...*...*
.....
...*.F...
*...*.
...*.....
..C.....*
...*.*...
.*.*.....
```

OUTPUT FORMAT

输出一个数字, 表示 John 需要多少时间才能抓住牛们. 输出 0, 如果 John 无法抓住牛.

SAMPLE OUTPUT (file ttwo.out)

49

★Overfencing 穿越栅栏

农夫 John 在外面的田野上搭建了一个巨大的用栅栏围成的迷宫. 幸运的是, 他在迷宫的边界上留出了两段栅栏作为迷宫的出口. 更幸运的是, 他所建造的迷宫是一个“完美的”迷宫: 即你能从迷宫中的任意一点找到一条走出迷宫的路.

给定迷宫的宽 W ($1 \leq W \leq 38$) 及长 H ($1 \leq H \leq 100$).

$2*H+1$ 行, 每行 $2*W+1$ 的字符以下面给出的格式表示一个迷宫. 然后计算从迷宫中最“糟糕”的那一个点走出迷宫所需的步数. (即使从这一点以最优的方式走向最靠近的出口, 它仍然需要最多的步数) 当然了, 牛们只会水平或垂直地在 X 或 Y 轴上移动, 他们从来不走对角线. 每移动到一个新的方格算作一步 (包括移出迷宫的那一步)

这是一个 $W=5, H=3$ 的迷宫:

```
+---+---+---+
|           |
+--+ +--+ +--+
|           | | |
+ +---+ +--+
| |         |
+--+ +---+---+
```

如上图的例子, 栅栏的柱子只出现在奇数行或奇数列. 每个迷宫只有两个出口.

PROGRAM NAME: maze1

INPUT FORMAT

第一行: W 和 H (用空格隔开)

第二行至第 $2*H+2$ 行: 每行 $2*W+1$ 个字符表示迷宫

SAMPLE INPUT (file maze1.in)

```
5 3
+---+---+---+
|           |
+--+ +--+ +--+
|           | | |
+ +---+ +--+
| |         |
+--+ +---+---+
```

OUTPUT FORMAT

输出一个单独的整数, 表示能保证牛从迷宫中任意一点走出迷宫的最小步数.

SAMPLE OUTPUT (file maze1.out)

9

★Bessie Come Home 回家

现在是晚餐时间, 而母牛们在外面分散的牧场中.

农民约翰按响了电铃, 所以她们开始向谷仓走去.

你的工作是要指出哪只母牛会最先到达谷仓 (在给出的测试数据中, 总会有且只有一只速度最快的

在挤奶的时候(晚餐前), 每只母牛都在她自己的牧场上, 一些牧场上可能没有母牛.

有时,两个牧场(可能是自我相同的)之间会有超过一条道路相连.

因此,所有的母牛最后都能到达谷仓,并且母牛总是走最短的路径.

牧场被标记为'a'..'z'和'A'..'Y',在用大写字母表示的牧场中有一只母牛,小写字母中则没有.

PROGRAM NAME: comehome

第 1 行: 整数 $P(1 \leq P \leq 10000)$, 表示连接牧场(谷仓)的道路的数目.

被道路连接牧场的标记和道路的长度($1 \leq \text{长度} \leq 1000$).

5

B d 3

C e 9

d Z 8

e Z 3

单独的一行包含二个项目:

最先到达谷仓的母牛所在的牧场的标记, 和这只母牛走过的路径的长度.

B 11

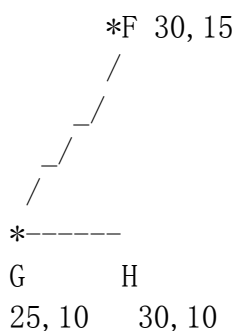
农民 John 的农场里有很多牧区. 有的路径连接一些特定的牧区. 一片所有连通的牧区称为一个牧场. 但是就目前而言, 你能看到至少有两个牧区不连通. 这样, 农民 John 就有多个牧区了. John 想在农场里添加一条路径(注意, 恰好一条). 对这条路径有以下限制:

一个牧场的直径就是牧场中最远的两个牧区的距离(本题中所提到的所有距离指的都是最短的距离). 考虑如下的有 5 个牧区的牧场, 牧区用 “*” 表示, 路径用直线表示. 每一个牧区都有自己的坐标:

1	15, 15
2	20, 15
3	20, 20
4	15, 20
5	10, 17.5



这个牧场的直径大约是 12.07106, 最远的两个牧区是 A 和 E, 它们之间的最短路径是 A-B-E. 这里是另一个牧场:



这两个牧场都在 John 的农场上. John 将会两个牧场中各选一个牧区, 然后用一条路径连起来, 使得连通后这个新的更大的牧场有最小的直径.

注意, 如果两条路径中途相交, 我们不认为它们是连通的. 只有两条路径在同一个牧区相交, 我们才认为它们是连通的.

输入文件包括牧区、它们各自的坐标,还有一个如下的对称邻接矩阵:

A	B	C	D	E	F	G	H
A	0	1	0	0	0	0	0
B	1	0	1	1	1	0	0
C	0	1	0	0	1	0	0
D	0	1	0	0	1	0	0
E	0	1	1	1	0	0	0
F	0	0	0	0	0	0	1
G	0	0	0	0	0	1	0
H	0	0	0	0	0	0	1

输入文件至少包括两个不连通的牧区.

请编程找出一条连接两个不同牧场的路径,使得连上这条路径后,这个更大的新牧场有最小的直径.

PROGRAM NAME: cowntour

INPUT FORMAT

第 1 行: 一个整数 N ($1 \leq N \leq 150$), 表示牧区数

第 2 到 N+1 行: 每行两个整数 X,Y ($0 \leq X, Y \leq 100000$), 表示 N 个牧区的坐标. 注意每个牧区的坐标都是不一样的.

第 $N+2$ 行到第 $2*N+1$ 行: 每行包括 N 个数字 (0 或 1) 表示如上文描述的对称邻接矩阵.

SAMPLE INPUT (file cowtour.in)

```

8
10 10
15 10
20 10
15 15
20 15
30 15
25 10
30 10
01000000
10111000
01001000

```

01001000
01110000
00000010
00000101
00000010

OUTPUT FORMAT

只有一行, 包括一个实数, 表示所求答案. 数字保留六位小数.

SAMPLE OUTPUT (file cowtour.out)

22.071068

★Fractions to Decimals 分数化小数

写一个程序, 输入一个形如 N/D 的分数(N 是分子, D 是分母), 输出它的小数形式。

如果小数有循环节的话, 把循环节放在一对圆括号中。例如,

$1/3 = .33333333$ 写成 $0.(3)$

$41/333 = 0.123123123...$ 写成 $0.(123)$

用 $xxx.0$ 成表示整数

典型的转化例子:

$1/3 = 0.(3)$

$22/5 = 4.4$

$1/7 = 0.(142857)$

$2/2 = 1.0$

$3/8 = 0.375$

$45/56 = 0.803(571428)$

PROGRAM NAME: fracdec

INPUT FORMAT

单独的一行包括被空格分开的 N 和 D , $1 \leq N, D \leq 100000$ 。

SAMPLE INPUT (file fracdec.in)

45 56

OUTPUT FORMAT

小数的表示方法上面说的很明白了, 如果输出的长度超过 76 个字符, 每行输出 76 个。

SAMPLE OUTPUT (file fracdec.out)

0.803(571428)

CHAPTER 3

Section 3.1

★Agri-Net 最短网络

农民约翰被选为他们镇的镇长！他其中一个竞选承诺就是在镇上建立起互联网，并连接到所有的农场。当然，他需要你的帮助。

约翰已经给他的农场安排了一条高速的网络线路，他想把这条线路共享给其他农场。为了用最小的消费，他想铺设最短的光纤去连接所有的农场。你将得到一份各农场之间连接费用的列表，你必须找出能连接所有农场并所用光纤最短的方案。每两个农场间的距离不会超过 100000

PROGRAM NAME: agrinet

INPUT FORMAT

第一行：农场的个数， N ($3 \leq N \leq 100$)。

第二行.. 结尾：后来的行包含了一个 $N \times N$ 的矩阵，表示每个农场之间的距离。理论上，他们是 N 行，每行由 N 个用空格分隔的数组成，实际上，他们限制在 80 个字符，因此，某些行会紧接着另一些行。当然，对角线将会是 0，因为不会有线路从第 i 个农场到它本身。

SAMPLE INPUT (file agrinet.in)

```
4
0 4 9 21
4 0 8 17
9 8 0 16
21 17 16 0
```

OUTPUT FORMAT

只有一个输出，其中包含连接到每个农场的光纤的最小长度。

SAMPLE OUTPUT (file agrinet.out)

```
28
```

★Score Inflation 总分

学生在我们 USACO 的竞赛中的得分越多我们越高兴。我们试着设计我们的竞赛以便人们能尽可能的多得分，这需要你的帮助。我们可以从几个种类中选取竞赛的题目，这里的一个“种类”是指一个竞赛题目的集合，解决集合中的题目需要相同多的时间并且能得到相同的分数。

你的任务是写一个程序来告诉 USACO 的职员，应该从每一个种类中选取多少题目，使得解决题目的总耗时在竞赛规定的时间内并且总分最大。

输入包括竞赛的时间， M ($1 \leq M \leq 10,000$) (不要担心，你要到了训练营中才会有长时间的比赛) 和 N ，“种类”的数目 $1 \leq N \leq 10,000$ 。

后面的每一行将包括两个整数来描述一个“种类”：

第一个整数说明解决这种题目能得的分数 ($1 \leq \text{points} \leq 10000$)，第二整数说明解决这种题目所

需的时间($1 \leq \text{minutes} \leq 10000$).

你的程序应该确定我们应该从每个“种类”中选多少道题目使得能在竞赛的时间中得到最大的分数. 来自任意的“种类”的题目数目可能任何非负数(0 或更多). 计算可能得到的最大分数.

PROGRAM NAME: inflate

INPUT FORMAT

第 1 行: M, N--竞赛的时间和题目“种类”的数目.

第 2-N+1 行: 两个整数:每个“种类”题目的分数和耗时.

SAMPLE INPUT (file inflate.in)

```
300 4
100 60
250 120
120 100
35 20
```

OUTPUT FORMAT

单独的一行包括那个在给定的限制里可能得到的最大的分数.

SAMPLE OUTPUT (file inflate.out)

```
605
{从第 2 个“种类”中选两题第 4 个“种类”中选三题}
```

★Humble Numbers 丑数

对于一给定的素数集合 $S = \{p_1, p_2, \dots, p_K\}$,

来考虑那些质因数全部属于 S 的数的集合. 这个集合包括, p_1 , p_1p_2 , p_1p_1 , 和 $p_1p_2p_3$ (还有其它). 这是个对于一个输入的 S 的丑数集合.

注意:我们不认为 1 是一个丑数.

你的工作是对于输入的集合 S 去寻找集合中的第 N 个丑数. `longint` (signed 32-bit) 对于程序是足够的.

PROGRAM NAME: humble

INPUT FORMAT

第 1 行: 二个被空间分开的整数: K 和 N , $1 \leq K \leq 100$, $1 \leq N \leq 100,000$.

第 2 行: K 个被空间分开的整数:集合 S 的元素

SAMPLE INPUT (file humble.in)

```
4 19
2 3 5 7
```

OUTPUT FORMAT

单独的一行, 写上对于输入的 S 的第 N 个丑数.

SAMPLE OUTPUT (file humble.out)

```
27
```

★Shaping Regions 形成的区域

N 个不同的颜色的不透明的长方形 ($1 \leq N \leq 1000$) 被放置在一张宽为 A 长为 B 的白纸上. 这些长方形被放置时, 保证了它们的边于白纸的边缘平行. 所有的长方形都放置在白纸内, 所以我们会

看到不同形状的各种颜色. 坐标系统的原点 (0, 0) 设在这张白纸的左下角, 而坐标轴则平行于边缘.

PROGRAM NAME: rect1

INPUT FORMAT

每行输入的是放置长方形的办法.

第一行输入的是那个放在底的长方形(即白纸).

第 1 行: A , B 和 N, 由空格分开 ($1 \leq A, B \leq 10,000$)

第 2 到 N+1 行: 为五个整数 llx, lly, urx, ury, color 这是一个长方形的左下角坐标, 右上角坐标和颜色.

颜色 1 和底部白纸的颜色相同.

SAMPLE INPUT (file rect1.in)

```
20 20 3
2 2 18 18 2
0 8 19 19 3
8 0 10 19 4
```

OUTPUT FORMAT

输出文件应该包含一个所有能被看到颜色连同该颜色的总面积的清单(即使颜色的区域不是连续的), 按 color 的增序顺序.

不要显示没有区域的颜色.

SAMPLE OUTPUT (file rect1.out)

```
1 91
2 84
3 187
4 38
```

★Contact 联系

奶牛们开始对用电波望远镜扫描牧场外的宇宙感兴趣. 最近, 他们注意到了一种非常奇怪的脉冲调制微波被从星系的中央发射出来. 他们希望知道电波是否是被某些地外生命发射出来的, 还是仅仅是普通的星星的心跳.

帮助奶牛们用一个能够分析他们在文件中记下的记录的工具来找到真相. 他们在寻找长度在 A 到 B 之间(含)在每天的数据文件中重复得最多的比特序列 ($1 \leq A \leq B \leq 12$). 他们在找那些重复得最多的比特序列. 一个输入限制告诉你应输出多少频率最多的序列.

符合的序列可能会重叠, 并且至少重复一次的序列会被计数.

PROGRAM NAME: contact

INPUT FORMAT

第一行: 三个用空格分隔的整数: A, B, N; ($1 \leq N \leq 50$)

第二行及以后: 一个最多 200,000 字符的序列, 全是 0 或 1; 每行有 80 个字符, 除了可能的最后一行.

SAMPLE INPUT (file contact.in)

```
2 4 10
01010010010001000111101100001010011001111000010010011110010000000
```

在样例里, 序列 100 出现了 12 次, 而序列 1000 出现了 5 次. 次数最多的序列是 00, 出现了 23 次.

OUTPUT FORMAT

输出 N 个频率最高的序列(按照频率由高到低的次序). 由短到长排列频率相同的这些序列, 如果长短相同, 按二进制大小排列. 如果出现的序列个数小于 N, 输出存在的序列.

对于每个存在的频率, 先输出单独包含该频率的一行, 再输出以空格分隔的这些频率. 每行六个 (除非少于六个剩下) .

SAMPLE OUTPUT (file contact.out)

```
23
00
15
01 10
12
100
11
11 000 001
10
010
8
0100
7
0010 1001
6
111 0000
5
011 110 1000
4
0001 0011 1100
```

★Stamps 邮票

已知一个 N 枚邮票的面值集合 (如, $\{1 \text{ 分}, 3 \text{ 分}\}$) 和一个上限 K —— 表示信封上能够贴 K 张邮票. 计算从 1 到 M 的最大连续可贴出的邮资.

例如, 假设有 1 分和 3 分的邮票; 你最多可以贴 5 张邮票. 很容易贴出 1 到 5 分的邮资 (用 1 分邮票贴就行了), 接下来的邮资也不难:

$$6 = 3 + 3$$

$$7 = 3 + 3 + 1$$

$$8 = 3 + 3 + 1 + 1$$

$$9 = 3 + 3 + 3$$

$$10 = 3 + 3 + 3 + 1$$

$$11 = 3 + 3 + 3 + 1 + 1$$

$$12 = 3 + 3 + 3 + 3$$

$$13 = 3 + 3 + 3 + 3 + 1.$$

然而, 使用 5 枚 1 分或者 3 分的邮票根本不可能贴出 14 分的邮资. 因此, 对于这两种邮票的集合和上限 $K=5$, 答案是 $M=13$.

PROGRAM NAME: stamps

INPUT FORMAT

第 1 行: 两个整数, K 和 N . K ($1 \leq K \leq 200$) 是可用的邮票总数. N ($1 \leq N \leq 50$) 是邮票面值的数量.

第 2 行 .. 文件末: N 个整数, 每行 15 个, 列出所有的 N 个邮票的面值, 面值不超过 10000.

SAMPLE INPUT (file stamps.in)

5 2

1 3

OUTPUT FORMAT

第 1 行: 一个整数, 从 1 分开始连续的可用集合中不多于 K 张邮票贴出的邮资数.

SAMPLE OUTPUT (file stamps.out)

13

Section3.2

★Factorials 阶乘

N 的阶乘写作 $N!$ 表示小于等于 N 的所有正整数的乘积. 阶乘会很快的变大, 如 $13!$ 就必须用 32 位整数类型来存储, $70!$ 即使用浮点数也存不下了. 你的任务是找到阶乘最后面的非零位. 举个例子, $5! = 1*2*3*4*5 = 120$ 所以 $5!$ 的最后面的非零位是 2, $7! = 1*2*3*4*5*6*7 = 5040$, 所以最后面的非零位是 4.

PROGRAM NAME: fact4

INPUT FORMAT

共一行, 一个整数不大于 4, 220 的整数 N.

SAMPLE INPUT (file fact4.in)

7

OUTPUT FORMAT

共一行, 输出 $N!$ 最后面的非零位.

SAMPLE OUTPUT (file fact4.out)

4

★Stringsobits01 串

考虑排好序的 N ($N \leq 31$) 位二进制数.

你会发现, 这很有趣. 因为他们是排列好的, 而且包含所有可能的长度为 N 且含有 1 的个数小于等于 L ($L \leq N$) 的数.

你的任务是输出第 I ($1 \leq I \leq$ 长度为 N 的二进制数的个数) 大的, 长度为 N , 且含有 1 的个数小于等于 L 的那个二进制数.

PROGRAM NAME: kimbits

INPUT FORMAT

共一行, 用空格分开的三个整数 N, L, I .

SAMPLE INPUT (file kimbits.in)

5 3 19

OUTPUT FORMAT

共一行, 输出满足条件的第 I 大的二进制数.

SAMPLE OUTPUT (file kimbits.out)

10011

★Spinning Wheels 纺车的轮子

一架纺车有五个纺轮, 这五个不透明的轮子边缘上都有一些缺口. 这些缺口必须被迅速而准确地排列好. 每个轮子都有一个起始标记 (在 0 度), 这样所有的轮子都可以在统一的已知位置开始转动. 轮子按照 'plus degrees' 方向旋转, 所以从起始位置开始, 在一定的时间内, 它们依次转过 1 度, 2 度等等 (虽然这些轮子很可能不会同时转过这些角度).

这是一个整数问题. 轮子不会转过 1.5 度或 23.51234123 度这样的角度. 例如, 轮子可能在一秒钟内转过 20 到 25 度甚至 30 到 40 度 (如果转得快的话).

这个问题中的所有角度都限制在 $0 \leq \text{角度} \leq 359$ 这个范围内. 轮子转过 359 度后接下来就是 0 度. 每个轮子都有一个确定的旋转速度, 以秒作为单位. $1 \leq \text{速度} \leq 180$.

轮子上的缺口的起始角度和缺口大小 (或长度) 各由一个整数表示, 都以度为单位. 在一个轮子上, 两个缺口之间至少有一度的间隔.

在起始位置, 设时间为 0, 所有的轮子的起始标记排列成一条直线. 你的程序必须计算, 最早出现每个的轮子上的缺口同其他轮子上的缺口对准 (也就是一束光可以通过五个轮子上的五个缺口) 情况的时间. 这些缺口在任意一个角度对准.

PROGRAM NAME: spin

INPUT FORMAT

输入中的五行对应五个轮子.

第一个数字表示轮子的转动速度. 下一个数字是缺口的数目 W . $1 \leq W \leq 5$. 接下来的 W 对数字表示每个缺口的起始角度和长度.

SAMPLE INPUT (file spin.in)

```
30 1 0 120
50 1 150 90
60 1 60 90
70 1 180 180
90 1 180 60
```

OUTPUT FORMAT

只有一行, 包括一个整数, 表示光能够通过这五个轮子的最早时间. 如果无解, 输出 'none' 小写, 不含引号).

SAMPLE OUTPUT (file spin.out)

9

★Feed Ratios 饲料调配

农夫约翰从来只用调配得最好的饲料来为他的奶牛. 饲料用三种原料调配成: 大麦, 燕麦和小麦. 他知道自己饲料精确的配比, 在市场上是买不到这样的饲料的. 他只好购买其他三种混合饲料 (同样都由三种麦子组成), 然后将它们混合, 来调配他的完美饲料.

给出三组整数, 表示 大麦:燕麦:小麦 的比例, 找出用这三种饲料调配 $x:y:x$ 的饲料的方法. 例如, 给出目标饲料 3:4:5 和三种饲料的比例:

1:2:3
3:7:1
2:1:2

你必须编程找出使这三种饲料用量最少的方案,要是不能用这三种饲料调配目标饲料,输出“NONE”.“用量最少”意味着三种饲料的用量(整数)的和必须最小.

对于上面的例子,你可以用 8 份饲料 1, 2 份饲料 2, 和 5 份饲料 3, 来得到 7 份目标饲料:

$$8*(1:2:3) + 1*(3:7:1) + 5*(2:1:2) = (21:28:35) = 7*(3:4:5)$$

表示饲料比例的整数,都是小于 100(数量级)的非负整数.表示各种饲料的份数的整数,都小于 100.一种混合物的比例不会由其他混合物的比例直接向加得到.

PROGRAM NAME: ratios

INPUT FORMAT

Line 1: 三个用空格分开的整数,表示目标饲料

Line 2..4: 每行包括三个用空格分开的整数,表示农夫约翰买进的饲料的比例

SAMPLE INPUT (file ratios.in)

3 4 5 1 2 3 3 7 1 2 1 2

OUTPUT FORMAT

输出文件要包括一行,这一行要么有四个整数,要么是“NONE”.前三个整数表示三种饲料的份数,用这样的配比可以得到目标饲料.第四个整数表示混合前三中饲料后得到的目标饲料的份数.

SAMPLE OUTPUT (file ratios.out)

8 1 5 7

★Magic Squares 魔板

在成功地发明了魔方之后,拉比克先生发明了它的二维版本,称作魔板.这是一张有 8 个大小相同的格子的魔板:

1 2 3 4
8 7 6 5

我们知道魔板的每一个方格都有一种颜色.这 8 种颜色用前 8 个正整数来表示.可以用颜色的序列来表示一种魔板状态,规定从魔板的左上角开始,沿顺时针方向依次取出整数,构成一个颜色序列.对于上图的魔板状态,我们用序列(1, 2, 3, 4, 5, 6, 7, 8)来表示.这是基本状态.

这里提供三种基本操作,分别用大写字母“A”,“B”,“C”来表示(可以通过这些操作改变魔板的状态):

“A”:交换上下两行;

“B”:将最右边的一行插入最左边;

“C”:魔板中央作顺时针旋转.

下面是对基本状态进行操作的示范:

A:	8	7	6	5	B:	4	1	2	3	C:	1	7	2	4
	1	2	3	4		5	8	7	6		8	6	3	5

对于每种可能的状态,这三种基本操作都可以使用.

你要编程计算用最少的基本操作完成基本状态到特殊状态的转换,输出基本操作序列.

PROGRAM NAME: msquare

INPUT FORMAT

只有一行,包括 8 个整数,用空格分开(这些整数在范围 1——8 之间),表示目标状态.

SAMPLE INPUT (file msquare.in)

2 6 8 4 5 7 3 1

OUTPUT FORMAT

Line 1: 包括一个整数, 表示最短操作序列的长度.

Line 2: 在字典序中最早出现的操作序列, 用字符串表示, 除最后一行外, 每行输出 60 个字符.

SAMPLE OUTPUT (file msquare.out)

7 BCABCCB

★Sweet Butter 香甜的黄油

农夫 John 发现做出全威斯康辛州最甜的黄油的方法: 糖. 把糖放在一片牧场上, 他知道 N ($1 \leq N \leq 500$) 只奶牛会过来舔它, 这样就能做出能卖好价钱的超甜黄油. 当然, 他将付出额外的费用在奶牛上.

农夫 John 很狡猾. 像以前的 Pavlov, 他知道他可以训练这些奶牛, 让它们在听到铃声时去一个特定的牧场. 他打算将糖放在那里然后下午发出铃声, 以至他可以在晚上挤奶.

农夫 John 知道每只奶牛都在各自喜欢的牧场 (一个牧场不一定只有一头牛). 给出各头牛在的牧场和牧场间的路线, 找出使所有牛到达的路程和最短的牧场 (他将把糖放在那)

PROGRAM NAME: butter

INPUT FORMAT

第一行: 三个数: 奶牛数 N , 牧场数 ($2 \leq P \leq 800$), 牧场间道路数 C ($1 \leq C \leq 1450$)

第二行到第 $N+1$ 行: 1 到 N 头奶牛所在的牧场号

第 $N+2$ 行到第 $N+C+1$ 行: 每行有三个数: 相连的牧场 A 、 B , 两牧场间距离 D ($1 \leq D \leq 255$), 当然, 连接是双向的

SAMPLE INPUT (file butter.in)

3 4 5

2

3

4

1 2 1

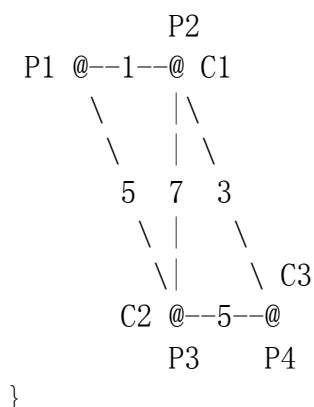
1 3 5

2 3 7

2 4 3

3 4 5

{样例图形



}

OUTPUT FORMAT

一行 输出奶牛必须行走的最小的距离和

SAMPLE OUTPUT (file butter.out)

8 {说明:放在 4 号牧场最优}

Section3.3

★Riding the Fences 骑马修栅栏

农民 John 每年有很多栅栏要修理. 他总是骑着马穿过每一个栅栏并修复它破损的地方.

John 是一个与其他农民一样懒的人. 他讨厌骑马, 因此从来不两次经过一个一个栅栏. 你必须编一个程序, 读入栅栏网络的描述, 并计算出一条修栅栏的路径, 使每个栅栏都恰好被经过一次. John 能从任何一个顶点(即两个栅栏的交点)开始骑马, 在任意一个顶点结束.

每一个栅栏连接两个顶点, 顶点用 1 到 500 标号(虽然有的农场并没有 500 个顶点). 一个顶点上可连接任意多(≥ 1)个栅栏. 所有栅栏都是连通的(也就是你可以从任意一个栅栏到达另外的所有栅栏).

你的程序必须输出骑马的路径(用路上依次经过的顶点号码表示). 我们如果把输出的路径看成一个 500 进制的数, 那么当存在多组解的情况下, 输出 500 进制表示法中最小的一个(也就是输出第一个数较小的, 如果还有多组解, 输出第二个数较小的, 等等).

输入数据保证至少有一个解.

PROGRAM NAME: fence

INPUT FORMAT

第 1 行: 一个整数 F ($1 \leq F \leq 1024$), 表示栅栏的数目

第 2 到 $F+1$ 行: 每行两个整数 i, j ($1 \leq i, j \leq 500$) 表示这条栅栏连接 i 与 j 号顶点.

SAMPLE INPUT (fence.in)

```
9
1 2
2 3
3 4
4 2
4 5
2 5
5 6
5 7
4 6
```

OUTPUT FORMAT

输出应当有 $F+1$ 行, 每行一个整数, 依次表示路径经过的顶点号. 注意数据可能有多组解, 但是只有上面题目要求的那一组解是认为正确的.

SAMPLE OUTPUT (fence.out)

```
1
2
3
```

4
2
5
4
6
5
7

★Shopping Offers 商店购物

在商店中,每一种商品都有一个价格(用整数表示)。例如,一朵花的价格是 2 zorkmids (z),而一个花瓶的价格是 5z。为了吸引更多的顾客,商店举行了促销活动。

促销活动把一个或多个商品组合起来降价销售,例如:

三朵花的价格是 5z 而不是 6z,

两个花瓶和一朵花的价格是 10z 而不是 12z。

编写一个程序,计算顾客购买一定商品的花费,尽量利用优惠使花费最少。尽管有时候添加其他商品可以获得更少花费,但是你不能这么做。

对于上面的商品信息,购买三朵花和两个花瓶的最少花费是:以优惠价购买两个花瓶和一朵花(10z),以原价购买两朵花(4z)。

PROGRAM NAME: shopping

INPUT FORMAT

输入文件包括一些商店提供的优惠信息,接着是购物清单。

第一行 优惠商品的种类数 ($0 \leq s \leq 99$)。

第二行..第 s+1 行 每一行都用几个整数来表示一种优惠方式。第一个整数 n ($1 \leq n \leq 5$),表示这种优惠方式由 n 种商品组成。后面 n 对整数 c 和 k 表示 k ($1 \leq k \leq 5$) 个编号为 c ($1 \leq c \leq 999$) 的商品共同构成这种优惠,最后的整数 p 表示这种优惠的优惠价 ($1 \leq p \leq 9999$)。优惠价总是比原价低。

第 s+2 行 这一行有一个整数 b ($0 \leq b \leq 5$),表示需要购买 b 种不同的商品。

第 s+3 行..第 s+b+2 行 这 b 行中的每一行包括三个整数:c, k, 和 p。c 表示唯一的商品编号 ($1 \leq c \leq 999$),k 表示需要购买的 c 商品的数量 ($1 \leq k \leq 5$),p 表示 c 商品的原价 ($1 \leq p \leq 999$)。最多购买 $5 \times 5 = 25$ 个商品。

SAMPLE INPUT (file shopping.in)

```
2
1 7 3 5
2 7 1 8 2 10
2
7 3 2
8 2 5
```

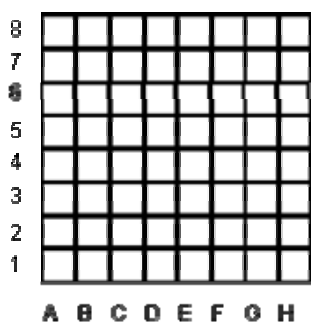
OUTPUT FORMAT

只有一行,输出一个整数:购买这些物品的最低价格。

SAMPLE OUTPUT (file shopping.out)

```
14
```

★Camelot 亚瑟王的宫殿

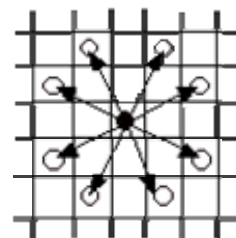


很久以前, 亚瑟王和他的骑士习惯每年元旦去庆祝他们的友谊. 在回忆中, 我们把这些是看作是一个有一人玩的棋盘游戏. 有一个国王和若干个骑士被放置在一个由许多方格组成的棋盘上, 没有两个骑士在同一个方格内.

这个例子是标准的 8*8 棋盘

国王可以移动到任何一个相邻的方格, 从●移动到○(右上图) 但前提是他不掉出棋盘之外.

一个骑士可以从●移动到○(右下图) 但前提是他不掉出棋盘之外



玩家的任務就是把所有的棋子移动到同一个方格里——用最小的步数. 为了完成这个任务, 他必须按照上面所说的规则去移动棋子. 玩家必须选择一个骑士跟国王一起行动, 其他的单独骑士则自己一直走到集中点. 骑士和国王一起走的时候, 只算一个人走的步数.

写一个程序去计算他们集中在一起的最小步数, 而且玩家必须自己找出这个集中点. 当然, 这些棋子可以在棋盘的 anywhere 集合.



PROGRAM NAME: camelot

INPUT FORMAT

第一行: 两个用空格隔开的整数: R, C 分别为棋盘行和列的长. 不超过 26 列, 40 行.

第二行.. 结尾: 输入文件包含了一些有空格隔开的字母/数字对, 一行有一个或以上. 第一对为国王的位置, 接下来是骑士的位置. 可能没有骑士, 也可能整个棋盘都是骑士. 行从 1 开始, 列从大写字母 A 开始.

SAMPLE INPUT (file camelot.in)

```
8 8
D 4
A 3 A 8
H 1 H 8
```

国王位置在 D4. 一共有四个骑士, 位置分别是 A3, A8, H1 和 H8.

OUTPUT FORMAT

单独一行表示棋子集中在一个方格的最小步数.

SAMPLE OUTPUT (file camelot.out)

```
10
```

HINT

他们集中在 B5.

骑士 1: A3 - B5 (1 步)

骑士 2: A8 - C7 - B5 (2 步)

骑士 3: H1 - G3 - F5 - D4 (picking up king) - B5 (4 步)

骑士 4: H8 - F7 - D6 - B5 (3 步)

$1 + 2 + 4 + 3 = 10$ 步.

★Home on the Range 家的范围

农民约翰在一片边长是 N ($2 \leq N \leq 250$) 英里的正方形牧场上放牧他的奶牛. (因为一些原因, 他的奶牛只在正方形的牧场上吃草.)

遗憾的是, 他的奶牛已经毁坏一些土地. (一些 1 平方英里的正方形)

农民约翰需要统计那些可以放牧奶牛的正方形牧场(至少是 2×2 的, 在这些较大的正方形中没有小于 1×1 的部分被分割毁坏).

你的工作要在被供应的数据组里面统计所有不同的正方形放牧区域($>2 \times 2$)的个数.

当然, 放牧区域可能是重叠.

PROGRAM NAME: range

INPUT FORMAT

第 1 行: N, 牧区的边长.

第 2 到 $n+1$ 行: N 个没有空格分开的字符.

0 表示 "那一个区段被毁坏了"; 1 表示 "准备好被吃".

SAMPLE INPUT (file range.in)

```
6
101111
001111
111111
001111
101101
111001
```

OUTPUT FORMAT

输出那些存在的正方形的大小和个数, 一种一行.

SAMPLE OUTPUT (file range.out)

```
2 10
3 4
4 1
```

★A Game 游戏

有如下一个双人游戏: N ($2 \leq N \leq 100$) 个正整数的序列放在一个游戏平台上, 两人轮流从序列的两端取数, 取数后该数字被去掉并累加到本玩家的得分中, 当数取尽时, 游戏结束. 以最终得分多者为胜.

编一个执行最优策略的程序, 最优策略就是使自己能得到在当前情况下最大的可能的总分的策略.

你的程序要始终为第二位玩家执行最优策略.

PROGRAM NAME: game1

INPUT FORMAT

第一行: 正整数 N , 表示序列中正整数的个数.

第二行至末尾: 用空格分隔的 N 个正整数 (大小为 $1-200$).

SAMPLE INPUT (file game1.in)

```
6
4 7 2 9
5 2
```

OUTPUT FORMAT

只有一行, 用空格分隔的两个整数: 依次为玩家一和玩家二最终的得分.

SAMPLE OUTPUT (file game1.out)

```
18 11
```

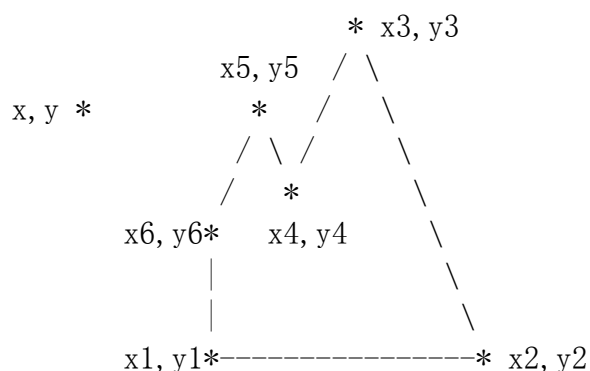

Section3.4

★Closed Fences 闭合的栅栏

一个闭合的栅栏是平面上的一些不相交的首尾相连的线段形成的多边形, 有 N 个角(顶点) ($3 < N < 200$). 顶点不重合, 它以逆时针方式以数组 $\{x_i, y_i\}$ 给出 ($i=1, 2, \dots, N$).

每一对相邻的顶点都是一条栅栏. 因此共有 N 条栅栏 (定义 $x_{N+1}=x_1, y_{N+1}=y_1$).

这里有一个栅栏的例子和一个点 x, y :



请编写一个程序实现下面的任务:

检查输入的顶点列表 $\{x_i, y_i\}$, $i=1, 2, \dots, N$, 判断它是否为一个合法的闭合栅栏.

找出所有可以被站在点 (x, y) 处的人所能看到的栅栏(忽略人的高度), 因为有的栅栏会被另外的栅栏所挡住. 只有当存在从 (x, y) 发射的一条射线第一个穿过栅栏 i 时, 栅栏 i 是可以被看见的. 如果栅栏是平行于目光的, 它并不认为是可以看见的. 在上面的例子里, 线段 $[x_3, y_3-x_4, y_4]$, $[x_5, y_5-x_6, y_6]$, $[x_6, y_6-x_1, y_1]$ 是可以被 (x, y) 看见的.

PROGRAM NAME: fence4

INPUT FORMAT

第一行: N , 表示闭合栅栏的顶点数.

第二行: 两个整数 x 和 y , 表示观测者的位置. 两个整数都是 16 位的.

第 3 到 $N+2$ 行: 每行一对整数 (x, y) 表示对应闭合栅栏的第 k 个顶点的坐标. 坐标以逆时针顺序给出. 整数绝对值不超过 1000.

SAMPLE INPUT (file fence4.in)

```
13
5 5
0 0
7 0
5 2
7 5
5 7
3 5
4 9
1 8
2 5
0 9
-2 7
```



```
0 3
-3 1
```

OUTPUT FORMAT

如果给出的序列不是一个合法的闭合栅栏, 那么输出文件只有一行, 为“NOFENCE”(不包括引号). 否则, 输出是一个可见栅栏的列表. 栅栏用两端的顶点表示, 顶点的输出顺序以输入文件中的顺序为准. 把栅栏按照最后一个点在输入文件中的顺序排序. 如果两条栅栏的最后一个点是一样的, 就以它们第一个点的顺序排序.

SAMPLE OUTPUT (file fence4.out)

```
7
0 0 7 0
5 2 7 5
7 5 5 7
5 7 3 5
-2 7 0 3
0 0 -3 1
0 3 -3 1
```

★American Heritage 美国血统

农夫约翰非常认真地对待他的奶牛们的血统. 然而他不是一个真正优秀的记帐员. 他把他的奶牛们的家谱作成二叉树, 并且把二叉树以更线性的”树的中序遍历“和”树的前序遍历“的符号加以记录而不是用图形的方法.

你的任务是在被给予奶牛家谱的”树中序遍历“和”树前序遍历“的符号后, 创建奶牛家谱的”树的后序遍历“的符号. 每一头奶牛的姓名被译为唯一的字母. (你可能已经知道你可以在知道树的两种遍历以后可以经常地重建这棵树.) 显然, 这里的树不会有多余 26 个的顶点.

右图是在样例输入和样例输出中的树的图形表达方式:

树的中序遍历是打印左子树, 根和右子树. 树的前序遍历是打印根, 左子树和右子树.

树的后序遍历是打印左子树, 右子树和根.

PROGRAM NAME: heritage

INPUT FORMAT

第一行: 树的中序遍历

第二行: 同样的树的前序遍历

SAMPLE INPUT (file heritage.in)

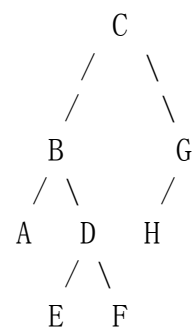
```
ABEDFCHG
CBADEFGH
```

OUTPUT FORMAT

单独的一行表示该树的后序遍历.

SAMPLE OUTPUT (file heritage.out)

```
AEFDBHGC
```



★Electric Fences 电网

农夫约翰已经决定建造电网. 他已经把他的农田围成一些奇怪的形状, 现在必须找出安放电源的最

佳位置.

对于段电网都必须从电源拉出一条电线. 电线可以穿过其他电网或者跨过其他电线. 电线能够以任意角度铺设, 从电源连接到一段电网的任意一点上(也就是, 这段电网的端点上或者在其之间的任意一点上). 这里所说的“一段电网”指的是呈一条线段状的电网, 并不是连在一起的几段电网. 若几段电网连在一起, 那么也要分别给这些电网提供电力.

已知所有的 F ($1 \leq F \leq 150$) 段电网的位置 (电网总是和坐标轴平行, 并且端点的坐标总是整数, $0 \leq X, Y \leq 100$). 你的程序要计算连接电源和每段电网所需的电线的最小总长度, 还有电源的最佳坐标.

电源的最佳坐标可能在农夫约翰的农田中的任何一个位置, 并不一定是整数.

PROGRAM NAME: fence3

INPUT FORMAT

第一行包括 F ——电网的数量.

下面的 F 行每行包括两个 X, Y 对, 表示这段电网的两个端点.

SAMPLE INPUT (file fence3.in)

```
3
0 0 0 1
2 0 2 1
0 3 2 3
```

OUTPUT FORMAT

只有一行, 输出三个浮点数, 相邻两个之间留一个空格. 假定你的电脑的输出库会正确地对小数进行四舍五入. 这三个数是: 电源最佳坐标的 X 值, 电源最佳坐标的 Y 值, 和需要的电线的总长度 (要最小).

SAMPLE OUTPUT (file fence3.out)

```
1.6 3.7
```

★Raucous Rockers “破锣摇滚” 乐队

你刚刚继承了流行的“破锣摇滚”乐队录制的尚未发表的 N ($1 \leq N \leq 20$) 首歌的版权. 你打算从中精选一些歌曲, 发行 M ($1 \leq M \leq 20$) 张 CD. 每一张 CD 最多可以容纳 T ($1 \leq T \leq 20$) 分钟的音乐, 一首歌不能分装在两张 CD 中.

不巧你是一位古典音乐迷, 不懂如何判定这些歌的艺术价值. 于是你决定根据以下标准进行选择: 歌曲必须按照创作的时间顺序在 CD 盘上出现. 选中的歌曲数目尽可能地多.

PROGRAM NAME: rockers

INPUT FORMAT

第一行: 三个整数: N, T, M .

第二行: N 个整数, 分别表示每首歌的长度, 按创作时间顺序排列.

SAMPLE INPUT (file rockers.in)

```
4 5 2
4 3 4 2
```

OUTPUT FORMAT

一个整数, 表示可以装进 M 张 CD 盘的乐曲的最大数目.

SAMPLE OUTPUT (file rockers.out)

```
3
```

CHAPTER 4

Section 4.1

★Beef McNuggets 麦香牛块

农夫布朗的奶牛们正在进行斗争, 因为它们听说麦当劳正在考虑引进一种新产品: 麦香牛块. 奶牛们正在想尽一切办法让这种可怕的设想泡汤. 奶牛们进行斗争的策略之一是“劣质的包装”. “看,” 奶牛们说, “如果你用只有一次能装 3 块、6 块或 10 块的三种包装盒装麦香牛块, 你就不可能满足想要一次只想买 1、2、4、5、7、8、11、14 或 17 块麦香牛块的顾客了. 劣质的包装意味着劣质的产品.”

你的任务是帮助这些奶牛. 给出包装盒的种类数 N ($1 \leq N \leq 10$) 和 N 个代表不同种类包装盒容纳麦香牛块个数的正整数 ($1 \leq i \leq 256$), 输出顾客不能用上述包装盒 (每种盒子数量无限) 买到麦香牛块的最大块数. 如果在限定范围内所有购买方案都能得到满足, 则输出 0.

范围限制是所有不超过 2,000,000,000 的正整数.

PROGRAM NAME: nuggets

INPUT FORMAT

第 1 行: 包装盒的种类数 N

第 2 行到 $N+1$ 行: 每个种类包装盒容纳麦香牛块的个数

SAMPLE INPUT (file nuggets.in)

```
3
3
6
10
```

OUTPUT FORMAT

输出文件只有一行数字: 顾客不能用包装盒买到麦香牛块的最大块数或 0 (如果在限定范围内所有购买方案都能得到满足).

SAMPLE OUTPUT (file nuggets.out)

```
17
```

★Fence Rails 栅栏的木料

农民 John 准备建一个栅栏来围住他的牧场. 他已经确定了栅栏的形状, 但是他在木料方面有些问题. 当地的杂货储存商扔给 John 一些木板, 而 John 必须从这些木板中找出尽可能多所需的木料.

当然, John 可以切木板. 因此, 一个 9 英尺的木板可以切成一个 5 英尺和一个 4 英尺的木料 (当

然也能切成 3 个 3 英尺的, 等等). John 有一把梦幻之锯, 因此他在切木料时, 不会有木料的损失. 所需要的木料规格都已经给定. 你不必切出更多木料, 那没有用.

PROGRAM NAME: fence8

INPUT FORMAT

第 1 行: N ($1 \leq N \leq 50$), 表示提供的木板的数目

第 2 行到第 $N+1$ 行: N 行, 每行包括一个整数, 表示各个木板的长度.

第 $N+2$ 行: R ($1 \leq R \leq 1023$), 所需木料的数目

第 $N+3$ 行到第 $N+R+1$ 行: R 行, 每行一个整数 ($1 \leq R_i \leq 128$), 表示所需木料的长度.

SAMPLE INPUT (file fence8.in)

```
4
30
40
50
25
10
15
16
17
18
19
20
21
25
24
30
```

OUTPUT FORMAT

只有一行, 一个数字, 表示能切除的最多的所需木料的树木. 当然, 并不是任何时候都能切出所有所需木料.

SAMPLE OUTPUT (file fence8.out)

```
7
```

★Fence Loops 篱笆回路

农夫布朗的牧场上的篱笆已经失去控制了. 它们分成了 $1 \sim 200$ 英尺长的线段. 只有在线段的端点处才能连接两个线段, 有时给定的一个端点上会有两个以上的篱笆. 结果篱笆形成了一张网分割了布朗的牧场. 布朗想将牧场恢复原样, 出于这个考虑, 他首先得知道牧场上哪一块区域的周长最小.

布朗将他的每段篱笆从 1 到 N 进行了标号 (N =线段的总数). 他知道每段篱笆的有如下属性:

该段篱笆的长度

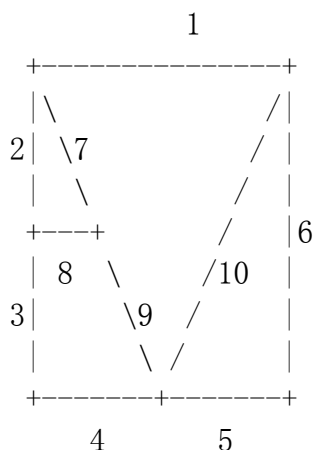
该段篱笆的一端所连接的另一段篱笆的标号

该段篱笆的另一端所连接的另一段篱笆的标号

幸运的是, 没有篱笆连接它自身.

对于一组有关篱笆如何分割牧场的数据, 写一个程序来计算出所有分割出的区域中最小的周长.

例如, 标号 $1 \sim 10$ 的篱笆由下图的形式组成 (下面的数字是篱笆的标号):



上图中周长最小的区域是由 2, 7, 8 号篱笆形成的.

PROGRAM NAME: fence6

INPUT FORMAT

第 1 行:	N (1 ≤ N ≤ 100)
第 2 行到第 3*N+1 行:	<p>每三行为一组, 共 N 组信息:</p> <p>每组信息的第 1 行有 4 个整数: s, 这段篱笆的标号 (1 ≤ s ≤ N); Ls, the 这段篱笆的长度 (1 ≤ Ls ≤ 255); N1s (1 ≤ N1s ≤ 8) 与本段篱笆的一端所相邻的篱笆的数量; and N2s 与本段篱笆的另一端所相邻的篱笆的数量. (1 ≤ N2s ≤ 8).</p> <p>每组信息的第 2 行有 N1s 个整数, 分别描述与本段篱笆的一端所相邻的篱笆的标号.</p> <p>每组信息的第 3 行有 N2s 个整数, 分别描述与本段篱笆的另一端所相邻的篱笆的标号.</p>

SAMPLE INPUT (file fence6.in)

```

10
1 16 2 2
2 7
10 6
2 3 2 2
1 7
8 3
3 3 2 1
8 2
4
4 8 1 3
3
9 10 5
5 8 3 1
9 10 4
6
6 6 1 2
5
1 10
7 5 2 2

```

```
1 2
8 9
8 4 2 2
2 3
7 9
9 5 2 3
7 8
4 5 10
10 10 2 3
1 6
4 9 5
```

OUTPUT FORMAT

输出的内容为单独的一行, 用一个整数来表示最小的周长.

SAMPLE OUTPUT (file fence6.out)

```
12
```

★Cryptcowgraphy 解密牛语

农民 Brown 和 John 的牛们计划协同逃出它们各自的农场. 它们设计了一种加密方法用来保护它们的通讯不被他人知道.

如果一头牛有信息要加密, 比如 "International Olympiad in Informatics", 它会随机地把 C, O, W 三个字母插到到信息中 (其中 C 在 O 前面, O 在 W 前面), 然后它把 C 与 O 之间的文字和 O 与 W 之间的文字的位置换过来. 这里是两个例子:

International Olympiad in Informatics

->

CnOIWternational Olympiad in Informatics

International Olympiad in Informatics

->

International Cin InformaticsOolympiad W

为了使解密更复杂, 牛们会在一条消息里多次采用这个加密方法 (把上次加密的结果再进行加密).

一天夜里, John 的牛们收到了一条经过多次加密的信息. 请你写一个程序判断它是不是这条信息经过加密 (或没有加密) 而得到的:

Begin the Escape execution at the Break of Dawn

PROGRAM NAME: cryptcow

INPUT FORMAT

一行, 不超过 75 个字符的加密过的信息.

SAMPLE INPUT (file cryptcow.in)

Begin the EscCutition at the BreOape execWak of Dawn

OUTPUT FORMAT

一行, 两个整数. 如果能解密成上面那条逃跑的信息, 第一个整数应当为 1, 否则为 0; 如果第一个数为 1, 则第二个数表示此信息被加密的次数, 否则第二个数为 0.

SAMPLE OUTPUT (file cryptcow.out)

```
1 1
```

Section4.2

★Drainage Ditches 草地排水

在农夫约翰的农场上, 每逢下雨, 贝茜最喜欢的三叶草地就积聚了一潭水. 这意味着草地被水淹没了, 并且小草要继续生长还要花相当长一段时间. 因此, 农夫约翰修建了一套排水系统来使贝茜的草地免除被大水淹没的烦恼 (不用担心, 雨水会流向附近的一条小溪). 作为一名一流的技师, 农夫约翰已经在每条排水沟的一端安上了控制器, 这样他可以控制流入排水沟的水流量.

农夫约翰知道每一条排水沟每分钟可以流过的水量, 和排水系统的准确布局 (起点为水潭而终点为小溪的一张网). 需要注意的是, 有些时候从一处到另一处不只有一条排水沟.

根据这些信息, 计算从水潭排水到小溪的最大流量. 对于给出的每条排水沟, 雨水只能沿着一个方向流动, 注意可能会出现雨水环形流动的情形.

PROGRAM NAME: ditch

INPUT FORMAT

第 1 行: 两个用空格分开的整数 N ($0 \leq N \leq 200$) 和 M ($2 \leq M \leq 200$). N 是农夫约翰已经挖好的排水沟的数量, M 是排水沟交叉点的数量. 交点 1 是水潭, 交点 M 是小溪.

第二行到第 $N+1$ 行: 每行有三个整数, S_i , E_i , 和 C_i . S_i 和 E_i ($1 \leq S_i, E_i \leq M$) 指明排水沟两端的交点, 雨水从 S_i 流向 E_i . C_i ($0 \leq C_i \leq 10,000,000$) 是这条排水沟的最大容量.

SAMPLE INPUT (file ditch.in)

```
5 4
1 2 40
1 4 20
2 4 20
2 3 30
3 4 10
```

OUTPUT FORMAT

输出一个整数, 即排水的最大流量.

SAMPLE OUTPUT (file ditch.out)

```
50
```

★The Perfect Stall 完美的牛栏

农夫约翰上个星期刚刚建好了他的新牛棚, 他使用了最新的挤奶技术. 不幸的是, 由于工程问题, 每个牛栏都不一样. 第一个星期, 农夫约翰随便地让奶牛们进入牛栏, 但是问题很快地显露出来: 每头奶牛都只愿意在她们喜欢的那些牛栏中产奶. 上个星期, 农夫约翰刚刚收集到了奶牛们的爱好的信息 (每头奶牛喜欢在哪些牛栏产奶). 一个牛栏只能容纳一头奶牛, 当然, 一头奶牛只能在一个牛栏中产奶.

给出奶牛们的爱好的信息, 计算最大分配方案.

PROGRAM NAME: stall4

INPUT FORMAT

第一行 两个整数, N ($0 \leq N \leq 200$) 和 M ($0 \leq M \leq 200$). N 是农夫约翰的奶牛数量, M 是新牛棚的牛栏数量.

第二行到第 N+1 行 一共 N 行, 每行对应一只奶牛. 第一个数字 (S_i) 是这头奶牛愿意在其中产奶的牛栏的数目 ($0 \leq S_i \leq M$). 后面的 S_i 个数表示这些牛栏的编号. 牛栏的编号限定在区间 $(1..M)$ 中, 在同一行, 一个牛栏不会被列出两次.

SAMPLE INPUT (file stall4.in)

```
5 5
2 2 5
3 2 3 4
2 1 5
3 1 2 5
1 2
```

OUTPUT FORMAT

只有一行. 输出一个整数, 表示最多能分配到的牛栏的数量.

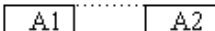
SAMPLE OUTPUT (file stall4.out)


```
4
```

★Job Processing 工序安排

一家工厂的流水线正在生产一种产品, 这需要两种操作: 操作 A 和操作 B. 每个操作只有一些机器能够完成.

Content of the input container: 

Type "A" machines: 

Content of the intermediate container: 

Type "B" machines: 

Content of the output container: 

上图显示了按照下述方式工作的流水线的组织形式. A 型机器从输入库接受工件, 对其施加操作 A, 得到的中间产品存放在缓冲库. B 型机器从缓冲库接受中间产品, 对其施加操作 B, 得到的最终产品存放在输出库. 所有的机器平行并且独立地工作, 每个库的容量没有限制. 每台机器的工作效率可能不同, 一台机器完成一次操作需要一定的时间.

给出每台机器完成一次操作的时间, 计算完成 A 操作的时间总和的最小值, 和完成 B 操作的时间总和的最小值.

PROGRAM NAME: job

INPUT FORMAT

第一行 三个用空格分开的整数:

◇N, 工件数量 ($1 \leq N \leq 1000$).

◇M1, A 型机器的数量 ($1 \leq M1 \leq 30$).

◇M2, B 型机器的数量 ($1 \leq M2 \leq 30$).

第二行...等 M1 个整数 (表示 A 型机器完成一次操作的时间, $1..20$), 接着是 M2 个整数 (B 型机器完成一次操作的时间, $1..20$)

SAMPLE INPUT (file job.in)

```
5 2 3
1 1 3 1 4
```


OUTPUT FORMAT

只有一行. 输出两个整数: 完成所有 A 操作的时间总和的最小值, 和完成所有 B 操作的时间总和的最小值 (A 操作必须在 B 操作之前完成).

SAMPLE OUTPUT (file job.out)

3 5

★Cowcycles 奶牛自行车

[读者请注意, 原题中的一些词语已经被改成了有关奶牛的双关语.]

秀·谢夫(小奶牛)在花花公子杂志上中了大奖, 于是她从农村搬到了城郊的一座别墅中. 可是她还常常怀念乡村的生活, 总想回到原来的农村逛逛. 为了环保, 秀决定骑上为她量身定做的奶牛自行车(特殊的自行车, 专门为牛蹄设计).

秀大约有一吨重. 同样的, 秀在普通的奶牛自行车上, 要想骑得平平稳稳, 也不是一件容易的事. 因此, 调节奶牛自行车的变速器让秀心力交瘁.

帮助秀选择她的奶牛自行车前面 F ($1 \leq F \leq 5$) 个齿轮和后面 R ($1 \leq R \leq 10$) 个齿轮, 使她的 $F \times R$ 奶牛自行车符合下面的标准:

前面齿轮的型号(齿的数量)必须在给定的范围内.

后面齿轮的型号(齿的数量)必须在给定的范围内.

在每一种齿轮组合中, 传动比率就是前面齿轮的齿数除以后面齿轮的齿数所得的商.

最大的传动比率至少是最小的三倍.

齿轮组合(已排好序)相邻两项的差的的方差(见下面的例子) 应该达到最小.

按照下面的公式计算平均数与方差(x_i 代表数据) :

$$\text{平均数} = \frac{1}{n} \sum_{i=1}^n x_i$$

$$\text{方差} = \frac{1}{n} \sum_{i=1}^n (x_i - \text{平均数})^2$$

计算并确定最佳齿轮组合(其中 F 个前齿轮, R 个后齿轮), 使方差最小(传动比率至少是 $3x$).

PROGRAM NAME: cowcycle

INPUT FORMAT

第一行是 F 和 R , 表示前齿轮和后齿轮的数量. 第二行包括 4 个数字: F_1, F_2 ($25 \leq F_1 < F_2 \leq 80$), R_1, R_2 ($5 \leq R_1 < R_2 \leq 40$). 从 F_1 到 F_2 型号的前齿轮都是可用的; 从 R_1 到 R_2 型号的后齿轮都是可用的. 至少会有一组合法的解.

SAMPLE INPUT (file cowcycle.in)

2 5

39 62 12 28

OUTPUT FORMAT

在第一行从小到大输出前齿轮的型号, 用空格分开. 在第二行从小到大输出后齿轮的型号, 同样用空格分开. 当然, 齿轮的齿数一定是整数.

如果有多个解, 输出前齿轮齿数最小的那一个(第一个齿轮齿数最小的, 若第一个齿轮齿数相等, 输出第二个齿轮齿数最小的……依此类推). 如果所有的前齿轮齿数都相等, 照着上面的办法处理后齿轮(其实就是把第一个, 第二个……齿轮分别设为第一, 第二……个关键字来排序).

SAMPLE OUTPUT (file cowcycle.out)

39 53

12 13 15 23 27

Comment

这个问题最大的挑战就是“读懂题目”。慢慢读，不要想一步登天。如果你读不懂题目，还是得一遍一遍的把它读进去。

问题要我们找出“最佳齿轮组合”，即传动比率最接近平均数的组合。考虑下面的测试数据：

2 5

39 62 12 28

这意味着有 2 个前齿轮，型号范围在 39..62；5 个后齿轮，型号范围在 12..28。程序必须检查所有前齿轮组成的有序对（共 $62-39+1=24$ 种齿轮），和所有后齿轮组成的五元组（共 $28-12+1=17$ 种齿轮）。根据组合数学原理，总共有 $24!/22!/2! \times 17!/5!/12! = 656,880$ 种可能（我是这么认为的）。

对于每一种可能，做下面的计算。举个例子来说，对于枚举到的第一种情况：前齿轮是 39 和 40，后齿轮是 12, 13, 14, 15 和 16。

首先，计算所有可能的传动比率：

$39/12 = 3.25000000000000000000$

$39/13 = 3.00000000000000000000$

$39/14 = 2.78571428571428571428$

$39/15 = 2.60000000000000000000$

$39/16 = 2.43750000000000000000$

$40/12 = 3.33333333333333333333$

$40/13 = 3.07692307692307692307$

$40/14 = 2.85714285714285714285$

$40/15 = 2.66666666666666666666$

$40/16 = 2.50000000000000000000$

然后，对它们进行排序：

$39/16 = 2.43750000000000000000$

$40/16 = 2.50000000000000000000$

$39/15 = 2.60000000000000000000$

$40/15 = 2.66666666666666666666$

$39/14 = 2.78571428571428571428$

$40/14 = 2.85714285714285714285$

$39/13 = 3.00000000000000000000$

$40/13 = 3.07692307692307692307$

$39/12 = 3.25000000000000000000$

$40/12 = 3.33333333333333333333$

然后，计算差的绝对值：

$2.43750000000000000000 - 2.50000000000000000000 = 0.06250000000000000000$

$2.50000000000000000000 - 2.60000000000000000000 = 0.10000000000000000000$

$2.60000000000000000000 - 2.66666666666666666666 = 0.06666666666666666666$

$2.66666666666666666666 - 2.78571428571428571428 = 0.11904761904761904762$

$2.78571428571428571428 - 2.85714285714285714285 = 0.07142857142857142857$

$2.85714285714285714285 - 3.00000000000000000000 = 0.14285714285714285715$

$3.00000000000000000000 - 3.07692307692307692307 = 0.07692307692307692307$

$3.07692307692307692307 - 3.25000000000000000000 = 0.17307692307692307693$

$3.25000000000000000000 - 3.33333333333333333333 = 0.08333333333333333333$

然后计算平均数和方差。平均数是（我是这么认为的）0.099537037037037037036666.. 方差大约是 0.00129798488416722。找出使方差最小的齿轮组合。

Section4.3

★Buy Low, Buy Lower 逢低吸纳

“逢低吸纳”是炒股的一条成功秘诀. 如果你想成为一个成功的投资者, 就要遵守这条秘诀:

“逢低吸纳, 越低越买”

这句话的意思是: 每次你购买股票时的股价一定要比你上次购买时的股价低. 按照这个规则购买股票的次数越多越好, 看看你最多能按这个规则买几次.

给定连续的 N 天中每天的股价, 你可以在任何一天购买一次股票, 但是购买时的股价一定要比你上次购买时的股价低. 写一个程序, 求出最多能买几次股票.

以下面这个表为例, 某几天的股价是:

天数 1 2 3 4 5 6 7 8 9 10 11 12

股价 68 69 54 64 68 64 70 67 78 62 98 87

这个例子中, 聪明的投资者(按上面的定义), 如果每次买股票时的股价都比上一次买时低, 那么他最多能买 4 次股票. 一种买法如下(可能有其他的买法):

天数 2 5 6 10

股价 69 68 64 62

PROGRAM NAME: buylow

INPUT FORMAT

第 1 行: N ($1 \leq N \leq 5000$), 表示能买股票的天数.

第 2 行以下: N 个正整数 (可能分多行), 第 i 个正整数表示第 i 天的股价. 这些正整数大小不会超过 `longint(pascal)/long(c++)`.

SAMPLE INPUT (file buylow.in)

12

68 69 54 64 68 64 70 67

78 62 98 87

OUTPUT FORMAT

只有一行, 输出两个整数:

能够买进股票的天数

长度达到这个值的股票购买方案数量

在计算解的数量时, 如果两个解所组成的字符串相同, 那么这样的两个解被认为是相同的(只能算做一个解). 因此, 两个不同的购买方案可能产生同一个字符串, 这样只能计算一次.

SAMPLE OUTPUT (file buylow.out)

4 2

★The Primes 素数方阵

在下面的方格中, 每行, 每列, 以及两条对角线上的数字可以看作是五位的素数. 方格中的行按照从左到右的顺序组成一个素数, 而列按照从上到下的顺序. 两条对角线也是按照从左到右的顺序来组成.

```

+---+---+---+---+---+
| 1 | 1 | 3 | 5 | 1 |
+---+---+---+---+---+
| 3 | 3 | 2 | 0 | 3 |
+---+---+---+---+---+
| 3 | 0 | 3 | 2 | 3 |
+---+---+---+---+---+
| 1 | 4 | 0 | 3 | 3 |
+---+---+---+---+---+
| 3 | 3 | 3 | 1 | 1 |
+---+---+---+---+---+

```

这些素数各个数位上的和必须相等.

左上角的数字是预先定好的.

一个素数可能在方阵中重复多次.

如果不只有一个解, 将它们全部输出 (按照这 25 个数字组成的 25 位数的大小排序).

一个五位的素数开头不能为 0 (例如: 00003 不是五位素数)

PROGRAM NAME: prime3

INPUT FORMAT

一行包括两个被空格分开的整数: 各个位的数字和 和左上角的数字.

SAMPLE INPUT (file prime3.in)

```
11 1
```

OUTPUT FORMAT

对于每一个找到的方案输出 5 行, 每行 5 个字符, 每行可以转化为一个 5 位的质数. 在两组方案中间输出一个空行. 如果没有解就单独输出一行 "NONE".

SAMPLE OUTPUT (file prime3.out)

上面的例子有三组解.

```
11351
14033
30323
53201
13313
```

```
11351
33203
30323
14033
33311
```

```
13313
13043
32303
50231
13331
```

★Street Race 街道赛跑

图一表示一次街道赛跑的跑道. 可以看出有一些路口 (用 0 到 N 的整数标号), 和连接这些路口的箭头. 路口 0 是跑道的起点, 路口 N 是跑道的终点. 箭头表示单行道. 运动员们可以顺着街道从一个路口移动到另一个路口 (只能按照箭头所指的方向). 当运动员处于路口位置时, 他可以选择任意一条由这个路口引出的街道.

一个良好的跑道具有如下几个特点:

- ◇ 每一个路口都可以由起点到达.
- ◇ 从任意一个路口都可以到达终点.
- ◇ 终点不通往任何路口.

运动员不必经过所有的路口来完成比赛. 有些路口却是选择任意一条路线都必须到达的 (称为 “不可避免” 的). 在上面的例子中, 这些路口是 0, 3, 6, 9. 对于给出的良好的跑道, 你的程序要确定 “不可避免” 的路口的集合, 不包括起点和终点.

假设比赛要分两天进行. 为了达到这个目的, 原来的跑道必须分为两个跑道, 每天使用一个跑道. 第一天, 起点为路口 0, 终点为一个 “中间路口”; 第二天, 起点是那个中间路口, 而终点为路口 N. 对于给出的良好的跑道, 你的程序要确定 “中间路口” 的集合. 如果良好的跑道 C 可以被路口 S 分成两部分, 这两部分都是良好的, 并且 S 不同于起点也不同于终点, 同时被分割的两个部分满足下列条件: (1) 它们之间没有共同的街道 (2) S 为它们唯一的公共点, 并且 S 作为其中一个的终点和另外一个的起点. 那么我们称 S 为 “中间路口”. 在例子中只有路口 3 是中间路口.

PROGRAM NAME: race3

INPUT FORMAT

输入文件包括一个良好的跑道, 最多有 50 个路口, 100 条单行道. 一共有 N+2 行, 前面 N+1 行中第 i 行表示以 i 为起点的街道, 每个数字表示一个终点. 行末用 -2 作为结束. 最后一行只有一个数字 -1.

SAMPLE INPUT (file race3.in)

```
1 2 -2
3 -2
3 -2
5 4 -2
6 4 -2
6 -2
7 8 -2
9 -2
5 9 -2
-2
-1
```

OUTPUT FORMAT

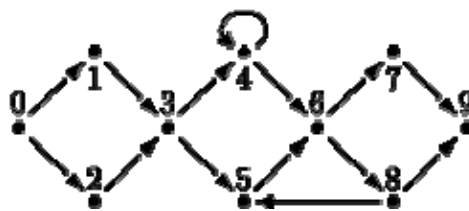
你的程序要有两行输出:

第一行包括: 跑道中 “不可避免的” 路口的数量, 接着是这些路口的序号, 序号按照升序排列.

第二行包括: 跑道中 “中间路口” 的数量, 接着是这些路口的序号, 序号按照升序排列.

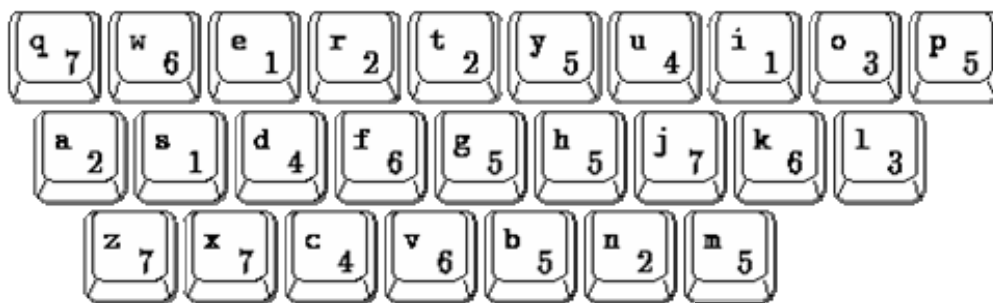
SAMPLE OUTPUT (file race3.out)

```
2 3 6
1 3
```



图一: 有 10 个路口的街道

★Letter Game 字母游戏



在家里用电视机做字母游戏是很流行的, 其中一种玩法是: 每一个字母有一个数值与之对应. 你收集字母组成一个或多个字以得到尽可能高的得分. 除非你已有了 “找字的方法” (“a way with words”), 你会把你知道的字都试一遍. 有时你也许会查阅其拼写, 然后计算得分. 显然可以用计算机更为准确地完成此任务. 上图示出了英文字母及其所对应的值, 当给出英文字(word) 的表列及收集的字母时, 请找出所能形成的得分最高的字或字对(pairs of words).

PROGRAM NAME: lgame

INPUT FORMAT

输入文件 lgame.in 中有一行由小写字母(`A' 到 `Z')组成的字符串, 这就是收集到字母(就是可以使用的字母), 字符串由至少 3 个字母至多 7 个字母(以任意顺序) 组成. 字典文件 lgame.dict 由至多 40,000 行组成, 文件的最后一行有`.` 表示文件的结束. 文件中的字已按字母顺序排序. 其它各行每一行都是由至少 3 个小写字母, 至多 7 个小写字母组成的字符串. 文件中的字已按字母顺序排序.

SAMPLE INPUT (file lgame.in)

prmgroa

SAMPLE INPUT (file lgame.dict)

profile

program

prom

rag

ram

rom

.

OUTPUT FORMAT

在文件 lgame.out 的第一行, 你的程序应写上最高得分(子任务 A), 随后的每一行是由文件 words.txt 中查到的具有这个得分的所有的字和或字对(word pairs)(子任务B). 要利用图中给定的字母的值. 当两个字能够形成一个组合(具有给定的字母)时, 这两个字应该打印到同一行, 两个字中间用一个空格隔开. 不许重复表示字对, 例如'rag prom' 和'prom rag' 是同样的字对, 因此只应该写出其中的一个.

SAMPLE OUTPUT (file lgame.out)

24

program

prom rag

Section4.4

★Shuttle Puzzle 棋盘游戏

大小为 3 的棋盘游戏里有 3 个白色棋子, 3 个黑色棋子, 和一个有 7 个格子一线排开的木盒子. 3 个白棋子被放在一头, 3 个黑棋子被放在另一头, 中间的格子空着.

初始状态: WWW_BBB

目标状态: BBB_WWW

在这个游戏里有两种移动方法是允许的:

1. 你可以把一个棋子移到与它相邻的空格;
2. 你可以把一个棋子跳过一个(仅一个)与它不同色的棋子到达空格.

大小为 N 的棋盘游戏包括 N 个白棋子, N 个黑棋子, 还有有 $2N+1$ 个格子的木盒子.

这里是 3-棋盘游戏的解, 包括初始状态, 中间状态和目标状态:

WWW BBB

WW WBBB

WWBW BB

WWBWB B

WWB BWB

W BWBWB

WBWBWB

BW WBWB

BWBW WB

BWBWBW

BWBWB W

BWB BWW

B BWBWW

BB WBWW

BBBW WW

BBB WWW

请编一个程序解大小为 N 的棋盘游戏($1 \leq N \leq 12$). 要求用最少的移动步数实现.

PROGRAM NAME: shuttle

INPUT FORMAT

一个整数 N.

SAMPLE INPUT (file shuttle.in)

3

OUTPUT FORMAT

用空格在棋盘的位置(位置从左到右依次为 1, 2, ..., $2N+1$)表示棋盘的状态. 输出棋盘的状态变换序列, 每行 20 个数(除了最后一行).

输出的解还应当有最小的字典顺序(即如果有多组移动步数最小的解, 输出第一个数最小的解; 如果还有多组, 输出第二个数最小的解; ...).

SAMPLE OUTPUT (file shuttle.out)

3 5 6 4 2 1 3 5 7 6 4 2 3 5 4

★Pollutant Control 追查坏牛奶

你第一天接手光明牛奶公司就发生了一件倒霉的事情:公司不小心发送了一批坏牛奶.很不幸,你发现这件事的时候,坏牛奶已经进入了送货网.这个送货网很大,而且关系复杂.你知道这批牛奶要发给哪个零售商,但是要把这批牛奶送到他手中有许多种途径.送货网由一些仓库和运输卡车组成,每辆卡车都在各自固定的两个仓库之间单向运输牛奶.在追查这些坏牛奶的时候,有必要保证它不被送到零售商手里,所以必须使某些运输卡车停止运输,但是停止每辆卡车都会有一定的经济损失.你的任务是,在保证坏牛奶不送到零售商的前提下,制定出停止卡车运输的方案,使损失最小.

PROGRAM NAME: milk6

INPUT FORMAT

第一行: 两个整数 $M(0 \leq M \leq 1000)$ 、 $N(2 \leq N \leq 32)$, N 表示仓库的数目, M 表示运输卡车的数量. 仓库 1 代表发货工厂, 仓库 N 代表坏牛奶要发往的零售商.

第 2.. $M+1$ 行: 每行 3 个整数 S_i , E_i , C_i . S_i , E_i 表示这辆卡车的出发仓库, 目的仓库. $C_i(0 \leq C_i \leq 2,000,000)$ 表示让这辆卡车停止运输的损失

SAMPLE INPUT (file milk6.in)

```
4 5
1 3 100
3 2 50
2 4 60
1 2 40
2 3 80
```

OUTPUT FORMAT

第 1 行两个整数 c 、 t , c 表示最小的损失, t 表示要停止的最少卡车数. 接下来 t 行表示你要停止哪几条线路. 如果有多种方案使损失最小, 输出停止的线路最少的方案.

SAMPLE OUTPUT (file milk6.out)

```
60 1 3
```

★Frame Up 重叠的图像

看下面的五张 9×8 的图像:

.....CCC....
EEEEEE..BBBB..	.C.C....
E...E..	DDDDDD..B..B..	.C.C....
E...E..	D...D..B..B..	.CCC....
E...E..	D...D..AAAA	..B..B..
E...E..	D...D..A..A	..BBBB..
E...E..	DDDDDD..A..A
E...E..AAAA
EEEEEE..
1	2	3	4	5

现在, 把这些图像按照 1—5 的编号从下到上重叠, 第 1 张在最下面, 第 5 张在最顶端. 如果一张图像覆盖了另外一张图像, 那么底下的图像的一部分就变得不可见了. 我们得到下面的图像:


```

. CCC. . . .
ECBCBB. .
DCBCDB. .
DCCC. B. .
D. B. ABAA
D. BBBB. A
DDDDAD. A
E. . . AAAA
EEEEEE. .

```

对于这样一张图像, 计算构成这张图像的矩形图像从底部到顶端堆叠的顺序.

下面是这道题目的规则:

◇矩形的边的宽度为 1, 每条边的长度都不小于 3 .

◇矩形的每条边中, 至少有一部分是可见的. 注意, 一个角同时属于两条边.

◇矩形用大写字母表示, 并且每个矩形的表示符号都不相同.

PROGRAM NAME: frameup

INPUT FORMAT

第一行 : 两个用空格分开的整数: 图像高 H ($3 \leq H \leq 30$) 和图像宽 W ($3 \leq W \leq 30$) .

第二行到第 $H+1$ 行: 每行 W 个字母.

SAMPLE INPUT (file frameup.in)

```

9 8
CCC. . . .
ECBCBB. .
DCBCDB. .
DCCC. B. .
D. B. ABAA
D. BBBB. A
DDDDAD. A
E. . . AAAA
EEEEEE. .

```

OUTPUT FORMAT

按照自底向上的顺序输出字母. 如果有不止一种情况, 按照字典顺序输出每一种情况(至少会有一种合法的顺序) .

SAMPLE OUTPUT (file frameup.out)

```

EDABC

```

CHAPTER 5



Section 5.1

★Fencing the Cows 圈奶牛

农夫约翰想要建造一个围栏用来围住他的奶牛,可是他资金匮乏.他建造的围栏必须包括他的奶牛喜欢吃草的所有地点.对于给出的这些地点的坐标,计算最短的能够围住这些点的围栏的长度.

PROGRAM NAME: fc

INPUT FORMAT

输入数据的第一行包括一个整数 N . N ($0 \leq N \leq 10,000$) 表示农夫约翰想要围住的放牧点的数目.接下来 N 行,每行由两个实数组成, X_i 和 Y_i , 对应平面上的放牧点坐标 ($-1,000,000 \leq X_i, Y_i \leq 1,000,000$). 数字用小数表示.

SAMPLE INPUT (file fc.in)

```
4
4 8
4 12
5 9.3
7 8
```

OUTPUT FORMAT

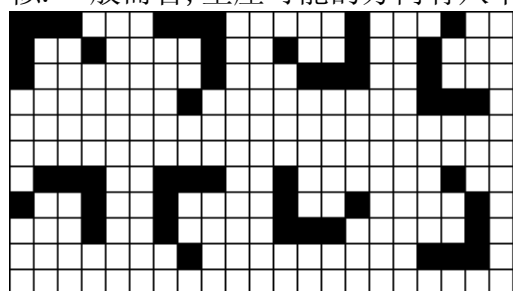
输出必须包括一个实数,表示必须的围栏的长度.答案保留两位小数.

SAMPLE OUTPUT (file fc.out)

```
12.00
```

★Starry Night 夜空繁星

高高的星空,簇簇闪耀的群星形态万千.一个星座(cluster)是一群连通的星组成的非空集合,所谓连通是指水平,垂直或者对角相邻.一个星座不能是另一个更大星座的一部分.星座可以相似(similar).如果两个星座有相同的形状,而且包括相同数目的星体,那么不管其方向性如何,就算相似.一般而言,星座可能的方向有八个,如图所示.



夜空可以表示为一份天体图(sky map),它是一个由字符 0 和 1 组成的二维矩阵,字符 1 表示所在的

位置有一颗星；字符 0 表示该位置上没有星。

任务：给定一份天体图，用同一个小写英文标识(mark)相似的所有星座。相似的星座必须用相同的字母标识为不同的字母。

标识一个星座，就是将其中各星体对应的字符 1 替换为相应的小写字母。

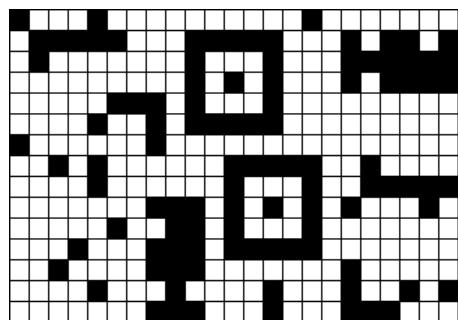
PROGRAM NAME: starry

INPUT FORMAT

文件的前两行分别记录了天体图的宽度 W、深度 H。而天体图则是由接下来的 H 行表示，每行包括 W 个字符。

SAMPLE INPUT (file starry.in)

```
23
15
100010000000000010000000
01111100011111000101101
01000000010001000111111
00000000010101000101111
00000111010001000000000
00001001011111000000000
10000001000000000000000
00101000000111110010000
00001000000100010011111
00000001110101010100010
00000100110100010000000
00010001110111110000000
00100001110000000100000
00001000100001000100101
00000001110001000111000
```



OUTPUT FORMAT

输出文件记录了天体图与文件 starry.in 相似，不同之处在于，各个星座按照“任务”中的要求进行了标识(mark)。

SAMPLE OUTPUT (file starry.out)

```
a000a00000000000b0000000
0aaaaa000cccc000d0dd0d
0a0000000c000c000ddddd
000000000c0b0c000d0ddd
00000eee0c000c000000000
0000e00e0cccc000000000
```

a			a								b							
	a	a	a	a	a			c	c	c	c	c			d	d	d	d
	a							c		c				d	d	d	d	d
								c	b	c				d	d	d	d	d
				e	e	e		c		c								
			e		e		c	c	c	c	c							
b					e													
	b	f						c	c	c	c	c			a			
			f					c			c			a	a	a	a	a
						d	d	d	c	b	c		a			a		
				b		d	d	d	c		c							
		g				d	d	d	c	c	c	c	c					
		g				d	d	d						e				
			b			d				f				e		e		b
						d	d	d		f				e	e	e		

```

b000000e0000000000000000
00b0f000000cccc00a0000
0000f000000c000c00aaaaa
0000000ddd0c0b0c0a000a0
00000b00dd0c000c0000000
000g000ddd0cccc0000000
00g0000ddd0000000e00000
0000b000d0000f000e00e0b
0000000ddd000f000eee000

```

这是上述输入实例的一个可能的结果. 请注意, 该输出文件对应于右上的天空景象.

Constraints

$0 \leq W$ (天体图的宽度) ≤ 100
 $0 \leq H$ (天体图的深度) ≤ 100
 $0 \leq$ 星座的数目 ≤ 500
 $0 \leq$ 不相似的星座数目 ≤ 26 (a..z)
 $1 \leq$ 各星座包含的星体数目 ≤ 160

★Musical Themes 乐曲主题

我们用 N ($1 \leq N \leq 5000$) 个音符的序列来表示一首乐曲, 每个音符都是 1..88 范围内的整数, 每个数表示钢琴上的一个键. 很不幸这种表示旋律的方法忽略了音符的时值, 但这项编程任务是关于音高的, 与时值无关.

许多作曲家围绕一个重复出现的“主题”来构建乐曲. 在我们的乐曲表示法中, “主题”是整个音符序列的一个子序列, 它需要满足如下条件:

长度至少为 5 个音符

在乐曲中重复出现(可能经过转调, 见下)

重复出现的同一主题不能重叠

“转调”的意思是主题序列中每个音符都被加上或减去了同一个整数值.

给定一段乐曲, 计算其中最长主题的长度(即音符数).

本题时限为 1 秒钟!

PROGRAM NAME: theme

INPUT FORMAT

输出文件的第一行包含整数 N . 下面的每一行(最后一行可能除外)包含 20 个整数, 表示音符序列. 最后一行可能少于 20 个音符.

SAMPLE INPUT (file theme.in)

```

30
25 27 30 34 39 45 52 60 69 79 69 60 52 45 39 34 30 26 22 18
82 78 74 70 66 67 64 60 65 80

```

OUTPUT FORMAT

输出文件应只含一个整数, 即最长主题的长度. 如果乐曲中没有主题, 那么输出 0.

SAMPLE OUTPUT (file theme.out)

```

5
(这个长度为 5 的主题是输入文件中第一行的最后 5 个音符和第二行开头 5 个音符)

```

Section5.2

★Snail Trails 蜗牛的旅行

Sally Snail, (蜗牛)喜欢在 $N \times N$ 的棋盘上闲逛 ($1 < n < 120$) .她总是从棋盘的左上角出发. 棋盘上有空的格子 (用 “.” 来表示) 和 B 个路障 (用 “#” 来表示). 下面是这种表示法的示例棋盘:

	A	B	C	D	E	F	G	H
1	S	#	.
2	#	.	.	.
3
4
5	#	.	.
6	#
7
8

萨丽总是垂直 (向上或者向下) 或水平 (向左或者向右) 地走. 她可以从出发地 (总是记作 A1) 向下或者向右走.

一旦萨丽选定了一个方向, 她就会一直走下去. 如果她遇到棋盘边缘或者路障, 她就停下来, 并且转过 90 度. 她不可能离开棋盘, 或者走进路障当中. 并且, 萨丽从不跨过她已经经过的格子. 当她再也不能走的时候, 她就停止散步.

这里是上面的棋盘上的一次散步路线图示:

	A	B	C	D	E	F	G	H
1	S	-----+					#	.
2	#		.	.
3
4	+---	+	
5	#	.	
6	#	
7	+	-----+						
8	+	-----+						

萨丽向右走, 再向下, 向右, 向下, 然后向左, 再向上, 最后向右走. 这时她遇到了一个她已经走过的格子, 她就停下来了. 但是, 如果她在 F5 格遇到路障后选择另外一条路——向我们看来是左边的方向转弯, 情况就不一样了.

任务是计算并输出, 如果萨丽聪明地选择她的路线的话, 她所能够经过的最多格子数.

PROGRAM NAME: snail

INPUT FORMAT

输入的第一行包括 N ——棋盘的大小, 和 B ——路障的数量 ($1 \leq B \leq 200$) . 接下来的 B 行包含着路障的位置信息. 下面的样例输入对应着上面的示例棋盘. 下面的输出文件表示问题的解答. 注意, 当 $N > 26$ 时, 输入文件就不能表示 Z 列以后的路障了.

SAMPLE INPUT (file snail.in) 8 4 E2 A6 G1 F5

OUTPUT FORMAT

输出文件应该只由一行组成, 即萨丽能够经过的最多格子数.

SAMPLE OUTPUT (file snail.out)

33

★Electric Fences 电网

农夫约翰已经决定建造电网. 他已经把他的农田围成一些奇怪的形状, 现在必须找出安放电源的最佳位置.

对于段电网都必须从电源拉出一条电线. 电线可以穿过其他电网或者跨过其他电线. 电线能够以任意角度铺设, 从电源连接到一段电网的任意一点上(也就是, 这段电网的端点上或者在其之间的任意一点上). 这里所说的“一段电网”指的是呈一条线段状的电网, 并不是连在一起的几段电网. 若几段电网连在一起, 那么也要分别给这些电网提供电力.

已知所有的 F ($1 \leq F \leq 150$) 段电网的位置(电网总是和坐标轴平行, 并且端点的坐标总是整数, $0 \leq X, Y \leq 100$). 你的程序要计算连接电源和每段电网所需的电线的最小总长度, 还有电源的最佳坐标.

电源的最佳坐标可能在农夫约翰的农田中的任何一个位置, 并不一定是整数.

PROGRAM NAME: fence3

INPUT FORMAT

第一行包括 F ——电网的数量.

下面的 F 行每行包括两个 X, Y 对, 表示这段电网的两个端点.

SAMPLE INPUT (file fence3.in)

```
3
0 0 0 1
2 0 2 1
0 3 2 3
```

OUTPUT FORMAT

只有一行, 输出三个浮点数, 相邻两个之间留一个空格. 假定你的电脑的输出库会正确地对小数进行四舍五入.

这三个数是:

电源最佳坐标的 X 值,

电源最佳坐标的 Y 值, 和

需要的电线的总长度(要最小).

SAMPLE OUTPUT (file fence3.out)

```
1.6 3.7
```

★Wisconsin Squares 威斯康星州的牧场

威斯康星州的春天来了, 是该把小奶牛们赶到小牧场上并把大奶牛们赶到北纬 40 度的大牧场上的时候了.

农夫约翰的牧场上有五种奶牛: 格恩西奶牛(A), 泽西奶牛(B), 赫里福奶牛(C), 黑安格斯奶牛(D), 朗赫恩奶牛(E). 这些奶牛群放养在一片 16 英亩的牧场上, 每英亩上都有一小群奶牛, 像下面这样排列成 4×4 的格子(用行和列标号):

```

1 2 3 4
+-----
1|A B A C
2|D C D E
3|B E B C
4|C A D E

```

最初牧场上的奶牛群总共有 3 群 A, 3 群 B, 4 群 C, 3 群 D, 3 群 E. 今年的 D 种小奶牛比去年多一群, C 种少一群, 共有 3 群 A, 3 群 B, 3 群 C, 4 群 D, 3 群 E.

农夫约翰对于他牧场上的奶牛群的布局非常小心. 这是因为如果同一种类型的奶牛群靠得太近, 她们就乱来: 她们聚集在栅栏边上抽烟, 喝牛奶. 如果她们在相同的格子上或者在临近的 8 个格子上, 就是靠得太近了.

农夫约翰得用他的棕色旧福特皮卡把他的大奶牛群运出牧场, 并把他的奶牛群运进牧场, 皮卡一次只能运一群奶牛. 他装上一群小奶牛, 开车到小牧场的一个方格中, 卸下这群小奶牛, 再装上这个格子上的那群大奶牛, 开到北纬 40 度的大牧场卸下来. 他重复这样的操作 16 次, 然后开车去杰克商店办理低脂酸奶的交易和家居装修.

帮助农夫约翰. 他必须选择正确的顺序来替换他的奶牛群, 使得他从不把一群小奶牛放入当前被同样类型奶牛占有的方格或者当前被同样类型奶牛占据的方格的临近方格. 当然, 一旦大奶牛走了, 小奶牛来了, 他必须小心以后的情况, 要根据新的排列把奶牛群分开.

非常重要的提示: 农夫约翰从过去的经验知道, 他必须先移动 D 种奶牛群.

找出农夫约翰将他的小奶牛搬迁到她们的新牧场上的办法. 输出 16 个连续的 奶牛群类型/行/列的信息, 使得这样的安排能够符合安全经验.

计算 4 x 4 牧场的最终排列总数 (这应该是原题作者的笔误, 输出样例中并没有说明要输出最终排列总数, 此题只有一个测试数据, 也没有体现这个问题——译者注), 和产生那些排列的方式的总数.

PROGRAM NAME: wissqu

INPUT FORMAT

四行, 每行四个用空格分开的字母, 表示奶牛群 (又是原题作者的笔误了, 字母之间没有空格——译者注).

SAMPLE INPUT (file wissqu.in)

```

ABAC
DCDE
BEBB
CADE

```

OUTPUT FORMAT

16 行, 每行分别由 奶牛群类型/行/列 组成. 如果有多解 (一定有), 那么你应该输出奶牛群类型按照字典序排列在最前面的那个解. 如果不只一个解满足条件, 那么你应该输出行信息按照字典序排列在最前面的那个解. 如果仍然不只一个解满足条件, 那么你应该输出列信息按照字典序排列在最前面的那个解.

最后多输出一行, 包含能够由这个排列方式产生的排列的总数.

SAMPLE OUTPUT (file wissqu.out)

```

D 4 1
C 4 2
A 3 1
A 3 3
B 2 4
B 3 2

```

B 4 4
E 2 1
E 2 3
D 1 4
D 2 2
C 1 1
C 1 3
A 1 2
E 4 3
D 3 4
14925

Section5.3

★Milk Measuring 量取牛奶

农夫约翰要量取 Q ($1 \leq Q \leq 20,000$) 夸脱 (夸脱, quarts, 容积单位——译者注) 他的最好的牛奶, 并把它装入一个大瓶子中卖出. 消费者要多少, 他就给多少, 从不有任何误差.

农夫约翰总是很节约. 他现在在奶牛五金商店购买一些桶, 用来从他的巨大的牛奶池中量出 Q 夸脱的牛奶. 每个桶的价格一样. 你的任务是计算出一个农夫约翰可以购买的最少的桶的集合, 使得能够刚好用这些桶量出 Q 夸脱的牛奶. 另外, 由于农夫约翰必须把这些桶搬回家, 对于给出的两个极小桶集合, 他会选择“更小的”一个, 即: 把这两个集合按升序排序, 比较第一个桶, 选择第一个桶容积较小的一个. 如果第一个桶相同, 比较第二个桶, 也按上面的方法选择. 否则继续这样的工作, 直到相比较的两个桶不一致为止. 例如, 集合 $\{3, 5, 7, 100\}$ 比集合 $\{3, 6, 7, 8\}$ 要好.

为了量出牛奶, 农夫约翰可以从牛奶池把桶装满, 然后倒进瓶子. 他决不把瓶子里的牛奶倒出来或者把桶里的牛奶倒到别处. 用一个容积为 1 夸脱的桶, 农夫约翰可以只用这个桶量出所有可能的夸脱数. 其它的桶的组合没有这么方便.

计算需要购买的最佳桶集, 保证所有的测试数据都至少有一个解.

PROGRAM NAME: milk4

INPUT FORMAT

Line 1: 一个整数 Q

Line 2: 一个整数 P ($1 \leq P \leq 100$), 表示商店里桶的数量

Lines 3..P+2: 每行包括一个桶的容积 ($1 \leq \text{桶的容积} \leq 10000$)

SAMPLE INPUT (file milk4.in)

16 3 3 5 7

OUTPUT FORMAT

输出文件只有一行, 由空格分开的整数组成:

为了量出想要的夸脱数, 需要购买的最少的桶的数量, 接着是:

一个排好序的列表 (从小到大), 表示需要购买的每个桶的容积

SAMPLE OUTPUT (file milk4.out)

2 3 5

★Window Area 窗体面积

你刚刚接手一项窗体界面工程. 窗体界面还算简单, 而且幸运的是, 你不必显示实际的窗体. 有 5 种基本操作:

创建一个新窗体

将窗体置顶

将窗体置底

删除一个窗体

输出窗体可见部分的百分比 (就是, 不被其它窗体覆盖的部分) .

在输入文件中, 操作以如下的格式出现.

创建一个新窗体: `w(I, x, y, X, Y)`

将窗体置顶: `t(I)`

将窗体置底: `b(I)`

删除一个窗体: `d(I)`

输出窗体可见部分的百分比: `s(I)`

`I` 是每个窗体唯一的标识符. 表示符可以是 `'a'..'z'`, `'A'..'Z'` 和 `'0'..'9'` 中的任何一个. 输入文件中没有多余的空格.

`(x, y)` 和 `(X, Y)` 是窗体的对角. 当你创建一个窗体的时候, 它自动被“置顶”. 你不能用已经存在的标识符来创建窗体, 但是你可以删除一个窗体后再用已删除窗体的标识符来创建窗体. 坐标用正整数来表示, 并且所有的窗体面积都不为 0 ($x < X$ 且 $y < Y$). `x` 坐标和 `y` 坐标在 1 —— 32767 的范围内.

PROGRAM NAME: window

INPUT FORMAT

输入文件包含给你的解释程序的一系列命令, 每行一个. 当输入文件结束时, 停止程序.

SAMPLE INPUT (file window.in)

`w(a, 10, 132, 20, 12)`

`w(b, 8, 76, 124, 15)`

`s(a)`

OUTPUT FORMAT

只对于 `s()` 命令进行输出. 当然, 输入文件可能有许多 `s()` 命令, 所以输出文件应该是一个百分比的序列, 每行一个, 百分比是窗体可见部分的百分比. 百分比应该四舍五入到三位小数.

SAMPLE OUTPUT (file window.out)

49.167

★Network of Schools 校园网

一些学校连入一个电脑网络. 那些学校已订立了协议: 每个学校都会给其它的一些学校分发软件 (称作“接受学校”). 注意如果 `B` 在 `A` 学校的分发列表中, 那么 `A` 不必也在 `B` 学校的列表中.

你要写一个程序计算, 根据协议, 为了让网络中所有的学校都用上新软件, 必须接受新软件副本的最少学校数目 (子任务 A). 更进一步, 我们想要确定通过给任意一个学校发送新软件, 这个软件就会分发到网络中的所有学校. 为了完成这个任务, 我们可能必须扩展接收学校列表, 使其加入新成员. 计算最少需要增加几个扩展, 使得不论我们给哪个学校发送新软件, 它都会到达其余所有的学校 (子任务 B). 一个扩展就是在一个学校的接收学校列表中引入一个新成员.

PROGRAM NAME: schlnet

INPUT FORMAT

输入文件的第一行包括一个整数 N : 网络中的学校数目 ($2 \leq N \leq 100$). 学校用前 N 个正整数标识. 接下来 N 行中每行都表示一个接收学校列表 (分发列表). 第 $i+1$ 行包括学校 i 的接收学校的标识符. 每个列表用 0 结束. 空列表只用一个 0 表示.

SAMPLE INPUT (file schlnet.in)

```
5
2 4 3 0
4 5 0
0
0
1 0
```

OUTPUT FORMAT

你的程序应该在输出文件中输出两行. 第一行应该包括一个正整数: 子任务 A 的解. 第二行应该包括子任务 B 的解.

SAMPLE OUTPUT (file schlnet.out)

```
1
2
```

★Big Barn 巨大的牛棚

农夫约翰想要在他的正方形农场上建造一座正方形大牛棚. 他讨厌在他的农场中砍树, 想找一个能够让他有空旷无树的地方修建牛棚的地方. 我们假定, 他的农场划分成 $N \times N$ 的方格. 输入数据中包括有树的方格的列表. 你的任务是计算并输出, 在他的农场中, 不需要砍树却能够修建的最大正方形牛棚. 牛棚的边必须和水平轴或者垂直轴平行.

EXAMPLE

考虑下面的方格, 它表示农夫约翰的农场, ‘.’ 表示没有树的方格, ‘#’ 表示有树的方格

```
  1 2 3 4 5 6 7 8
1 . . . . . . . .
2 . # . . . # . .
3 . . . . . . . .
4 . . . . . . . .
5 . . . . . . . .
6 . . # . . . . .
7 . . . . . . . .
8 . . . . . . . .
```

最大的牛棚是 5×5 的, 可以建造在方格右下角的两个位置其中一个.

PROGRAM NAME: bigbrn

INPUT FORMAT

Line 1: 两个整数: N ($1 \leq N \leq 1000$), 农场的大小, 和 T ($1 \leq T \leq 10,000$) 有树的方格的数量

Lines 2.. $T+1$: 两个整数 ($1 \leq \text{整数} \leq N$), 有树格子的横纵坐标

SAMPLE INPUT (file bigbrn.in)

```
8 3
2 2
2 6
```

6 3

OUTPUT FORMAT

输出文件只由一行组成, 约翰的牛棚的最大边长.

SAMPLE OUTPUT (file bigbrn.out)

5

Section5.4

★All Latin Squares 拉丁正方形

一种正方形的数字编排

1 2 3 4 5

2 1 4 5 3

3 4 5 1 2

4 5 2 3 1

5 3 1 2 4

是一个 5*5 的拉丁正方形, 每个 1 到 5 的整数在每行每列都出现且出现一次.

写个程序计算 $N \times N$ 的拉丁正方形的总数且要求第一行是:

1 2 3 4 5.....N

你的程序应该算称呼任意的从 2 到 7 的 N (Your program should work for any N from 2 to 7)

PROGRAM NAME: latin

INPUT FORMAT

一行包含一个整数 N

SAMPLE INPUT (file latin.in)

5

OUTPUT FORMAT

只有一行没, 表示拉丁正方形的个数, 且拉丁正方形的第一行为 1 2 3 . . . N.

SAMPLE OUTPUT (file latin.out)

1344

★Canada Tour 周游加拿大

你赢得了一场航空公司举办的比赛, 奖品是一张加拿大环游机票. 旅行在这家航空公司开放的最西边的城市开始, 然后一直自西向东旅行, 直到你到达最东边的城市, 再由东向西返回, 直到你回到开始的城市. 每个城市只能访问一次, 除了旅行开始的城市之外, 这个城市必定要被访问两次 (在旅行的开始和结束). 你不允许使用其他公司的航线或者用其他的交通工具.

给出这个航空公司开放的城市列表, 和两两城市之间的直达航线列表. 找出能够访问尽可能多的城市的路线, 这条路线必须满足上述条件, 也就是从列表中的第一个城市开始旅行, 访问到列表中最后一个城市之后再返回第一个城市.

PROGRAM NAME: tour

INPUT FORMAT

Line 1: 航空公司开放的城市数 N 和将要列出的直达航线的数量 V. N 是一个不大于 100 的

正整数. V 是任意的正整数.

Lines 2.. $N+1$: 每行包括一个航空公司开放的城市名称. 城市名称按照自西向东排列. 不会出现两个城市在同一条经线上的情况. 每个城市的名称都是一个字符串, 最多 15 字节, 由拉丁字母表上的字母组成; 城市名称中没有空格.

Lines $N+2$.. $N+2+V-1$: 每行包括两个城市名称(由上面列表中的城市名称组成), 用一个空格分开. 这样就表示两个城市之间的直达双程航线.

SAMPLE INPUT (file tour.in)

```
8 9
Vancouver
Yellowknife
Edmonton
Calgary
Winnipeg
Toronto
Montreal
Halifax
Vancouver Edmonton
Vancouver Calgary
Calgary Winnipeg
Winnipeg Toronto
Toronto Halifax
Montreal Halifax
Edmonton Montreal
Edmonton Yellowknife
Edmonton Calgary
```

OUTPUT FORMAT

Line 1: 按照最佳路线访问的不同城市的数量 M . 如果无法找到路线, 输出 1.

SAMPLE OUTPUT (file tour.out)

```
7
```

也就是: Vancouver, Edmonton, Montreal, Halifax, Toronto, Winnipeg, Calgary, 和 Vancouver (回到开始城市, 但是不算在不同城市之内) .

★Character Recognition 字符识别

这个问题需要你写一个程序完成字符识别的任务.

每个完整的字符图案有 20 行, 20 位. 每个位是 “0” 或 “1”. 图 1a 对应着输入文件中的符号图案.

文件 `font.in` 包括了 27 个字符图案的信息, 以这样的顺序记录:

`_abcdefghijklmnopqrstuvwxyz`

其中 `_` 表示空格字符. 每个完整字符长 20 行.

输入文件包含一个或多个可能损坏的字符图案. 一个字符图案可能以这些方式被损坏.

最多有一行可能被复制了 (就接在原来那一行的下面)

最多有一行可能丢失了

有些 “0” 可能被改成 “1” 有些 “1” 可能被改成 “0”

不会有任何一个字符图案既多余了一行并且又丢失了一行. 在测试数据的任何一个字符图案

中，“0”和“1”的被改变率不超过 30%。
被复制的那一行中，原来的行和多余的行可能都损坏了，而且损坏的部分可能并不相同。
写一个程序，使用 font.in 提供的字体，在输入文件提供的图象中识别出一个或多个的字符序列。
使用提供的字体图象来识别字符的时候，要找出最佳的多余行或遗漏行，使找出的所有“0”和“1”的变化数量最小。在所有可能的多余行中，只按损坏数据最少的那一行计算。你必须确定和输入序列最接近的字符序列（就是损坏数据最少的那一个）。每个测试数据有唯一的最优解。正确的解答必须使用到输入文件中的所有数据。

PROGRAM NAME: char
INPUT FORMAT (both input files)

两个输入文件都以一个整数 N 开始 (19 < N < 1200)，表示接下去的行数。
N

(位 1) (位 2) (位 3) ... (位 20)
(位 1) (位 2) (位 3) ... (位 20)
...

每行有 20 位的宽度。在 0 和 1 之间没有空格分开。
文件 font.in 描述字体。它总有 541 行。它在每个测试数据中可能不同。

SAMPLE INPUT (file char.in)

例:font.in 的开头(不完全的), 有“空格”和“a”	例:char.in 显示了一个损坏的 “a”
font.in	char.in
540	19
00000000000000000000	00000000000000000000
00000000000000000000	00000000000000000000
00000000000000000000	00000000000000000000
00000000000000000000	00000011100000000000
00000000000000000000	00100111011011000000
00000000000000000000	00001111111001100000
00000000000000000000	00001110001100100000
00000000000000000000	00001100001100010000
00000000000000000000	00001100000100010000
00000000000000000000	00000100000100010000
00000000000000000000	00000010000000110000
00000000000000000000	00001111011111110000
00000000000000000000	00001111111111110000
00000000000000000000	00001111111111100000
00000000000000000000	00001000010000000000
00000000000000000000	00000000000000000000
00000000000000000000	00000000000001000000
00000000000000000000	00000000000000000000
00000000000000000000	00000000000000000000
00000000000000000000	
00000000000000000000	
00000000000000000000	
00000000000000000000	
00000011100000000000	

00000111111011000000	
000011111111001100000	
00001110001100100000	
00001100001100010000	
00001100000100010000	
00000100000100010000	
00000010000000110000	
00000001000001110000	
00001111111111110000	
00001111111111110000	
00001111111111000000	
00001000000000000000	
00000000000000000000	
00000000000000000000	
00000000000000000000	
00000000000000000000	
图 1a	图 1b

OUTPUT FORMAT

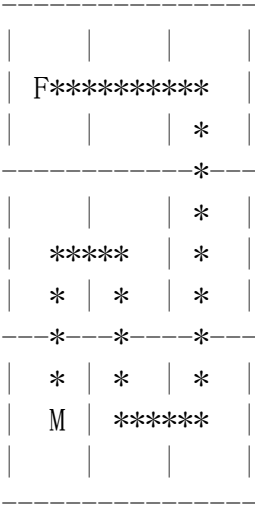
你的程序必须建立一个输出文件, 包含一个识别出来的字符串. 它的格式是一个单行的 ASCII 文本. 输出文件中不应该包含任何分隔符. 如果你的程序没有识别出一个特定的字符, 就必须在适当的位置输出一个“?”.

SAMPLE OUTPUT (file char.out)

a
注意, 上面输出的一行是两个字符: “a” 后面还跟着一个空格. (这是按原文译的, 其实我也不理解这句话, 应该只有一个字符 “a” 吧——译者注)

★Betsy's Tour 漫游小镇

一个正方形的镇区分为 N² 个小方块 (1 ≤ N ≤ 7). 农场位于方格的左上角, 集市位于左下角. 贝茜穿过小镇, 从左上角走到左下角, 刚好经过每个方格一次. 当 N=3 时, 贝茜的漫游路径可能如下图所示:



写一个程序, 对于给出的 N 值, 计算贝茜从农场走到集市有多少种唯一的路径.

PROGRAM NAME: betsy

INPUT FORMAT

行 1: 一个整数 N ($1 \leq N \leq 7$)

SAMPLE INPUT (file betsy.in)

3

OUTPUT FORMAT

只有一行. 输出一个整数表示唯一路径的数量.

SAMPLE OUTPUT (file betsy.out)

2

★Telecommunication 奶牛的电信

农夫约翰的奶牛们喜欢通过电邮保持联系, 于是她们建立了一个奶牛电脑网络, 以便互相交流. 这些机器用如下的方式发送电邮: 如果存在一个由 c 台电脑组成的序列 $a_1, a_2, \dots, a(c)$, 且 a_1 与 a_2 相连, a_2 与 a_3 相连, 等等, 那么电脑 a_1 和 $a(c)$ 就可以互发电邮.

很不幸, 有时候奶牛会不小心踩到电脑上, 农夫约翰的车也可能碾过电脑, 这台倒霉的电脑就会坏掉. 这意味着这台电脑不能再发送电邮了, 于是与这台电脑相关的连接也就不可用了.

有两头奶牛就想: 如果我们两个不能互发电邮, 至少需要坏掉多少台电脑呢? 请编写一个程序为她们计算这个最小值和与之对应的坏掉的电脑集合.

以如下网络为例:

$$\begin{array}{c} 1* \\ / \\ 3 - 2* \end{array}$$

这张图画的是有 2 条连接的 3 台电脑. 我们想要在电脑 1 和 2 之间传送信息. 电脑 1 与 3、2 与 3 直接连通. 如果电脑 3 坏了, 电脑 1 与 2 便不能互发信息了.

PROGRAM NAME: telecow

INPUT FORMAT

第一行: 四个由空格分隔的整数: N, M, c_1, c_2 . N 是电脑总数 ($1 \leq N \leq 100$), 电脑由 1 到 N 编号. M 是电脑之间连接的总数 ($1 \leq M \leq 600$). 最后的两个整数 c_1 和 c_2 是上述两头奶牛使用的电脑编号. 连接没有重复且均为双向的 (即如果 c_1 与 c_2 相连, 那么 c_2 与 c_1 也相连). 两台电脑之间至多有一条连接. 电脑 c_1 和 c_2 不会直接相连.

第 2 到 $M+1$ 行: 接下来的 M 行中, 每行包含两台直接相连的电脑的编号.

SAMPLE INPUT (file telecow.in)

3 2 1 2

1 3

2 3

OUTPUT FORMAT

输出共有两行. 第一行是使电脑 c_1 和 c_2 不能互相通信需要坏掉的电脑数目的最小值. 第二行是排好序的坏掉的电脑的编号列表. 注意 c_1 和 c_2 都不能坏掉. 如果有多种可能情况, 输出第一个数最小的一种, 如果第一个数相同, 则输出第二个数最小的一种, 依此类推.

SAMPLE OUTPUT (file telecow.out)

1

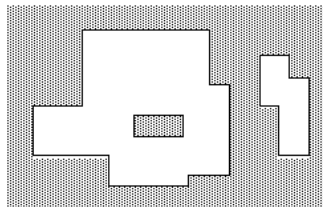
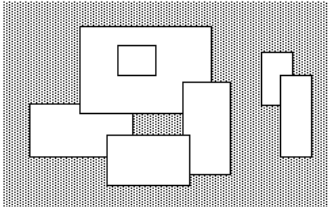
3

Section5.5

★Picture 矩形周长

N 张矩形的海报, 照片和其他同样形状的图片贴在墙上. 它们的边都是垂直的或水平的. 每个矩形可以部分或者全部覆盖其他矩形. 所有的矩形组成的集合的轮廓称为周长. 写一个程序计算周长.

左图是一个有 7 个矩形的例子: 对应的轮廓为右图所示的所有线段的集合:



所有矩形的顶点坐标均为整数. 所有的坐标都在 $[-10000, 10000]$ 的范围内, 并且任何一个矩形面积都为正数. 结果的值可能需要 32 位有符号整数表示.

PROGRAM NAME: picture

INPUT FORMAT

Line 1: N, 张贴在墙上的矩形的数目.

Lines 2..N+1 接下来的 N 行中, 每行都有两个点的坐标, 分别是矩形的左下角坐标和右上角坐标. 每一个坐标由 X 坐标和 Y 坐标组成.

SAMPLE INPUT (file picture.in)

```
7 -15 0 5 10 -5 8 20 25 15 -4 24 14 0 -6 16 4 2 15 10 22 30 10 36 20 34 0 40 16
```

OUTPUT FORMAT

只有一行, 为一个非负整数, 表示输入数据中所有矩形集合的轮廓长度.

SAMPLE OUTPUT (file picture.out)

```
228
```

★Hidden Password 隐藏口令

有时候程序员有很奇怪的方法来隐藏他们的口令. Billy "Hacker" Geits 会选择一个字符串 S (由 L 个小写字母组成, $5 \leq L \leq 100,000$), 然后他把 S 顺时针绕成一个圈, 每次取一个做开头字母并顺时针依次取字母而组成一个字符串. 这样将得到一些字符串, he 把它们排序后取出第一个字符串. 把这个字符串的第一个字母在原字符串中的位置-1 做为口令.

如字符串 alabala, 按操作的到 7 个字符串, 排序后得:

```
aalabal
abalaal
alaalab
alabala
balaala
laalaba
labalaa
```

第一个字符串为 aalabal, 这个 a 在原字符串位置为 7, $7-1=6$, 则 6 为口令.

PROGRAM NAME: hidden

INPUT FORMAT

第一行:一个数:L

第二行:字符串:S

SAMPLE INPUT (file hidden.in)

7

alabala

OUTPUT FORMAT

一行, 为得到的口令

SAMPLE OUTPUT (file hidden.out)

6

★Two Five 二五语言

有一种奇怪的语言叫做“二五语言”. 它的每个单词都由A~Y这25个字母各一个组成. 合法的二五语言单词必须满足这样一个条件: 把它的25个字母排成一个5*5的矩阵, 它的每一行和每一列都必须是递增的. 比如单词ACCEPTBDHQUFJMRWGKNSXILOYVY, 它排成的矩阵如下所示:

A	C	E	P	T
B	D	H	Q	U
F	J	M	R	W
G	K	N	S	X
I	L	O	V	Y

它每行每列都是递增的, 是一个合法的单词. 而YXWVUTSRQPONMLKJIHGFEDCBA则不合法.

由于单词太长存储不便, 需要给每一个单词编一个码. 编码方法如下: 写出单词中字母A在矩阵中的行号和列号, 再写出B的行号和列号……依此类推. 得到的序列叫做行列序列. 如上面的合法单词的行列序列是: 11 21 12 22 13 31 41 23 51 32 42 52 33 43 53 14 24 34 44 15 25 54 35 45 55. 然后把所有合法单词的行列序列排序. 一个单词的行列序列排序后的位置, 就是这个单词的编码. 比如, 单词ABCDEFGH IJKLMNOPQRSTU VWXY的编码为1, 而单词ABCDEFGH IJKLMNOPQRSUTVWXY的编码为2. 现在, 你需要编一个程序, 完成单词与编码间的转换.

PROGRAM NAME: twofive

INPUT FORMAT

第一行为一个字母N或W. N表示把编码转换为单词, W表示把单词转换为编码.

若第一行为N, 则第二行为一个整数, 表示单词的编码. 若第一行为W, 则第二行为一个合法的单词.

SAMPLE INPUT #1 (file twofive.in)

N

2

SAMPLE INPUT #2

W

ABCDEFGH IJKLMNOPQRSTU VWXY

OUTPUT FORMAT

每行一个整数或单词.

SAMPLE OUTPUT #1 (file twofive.out)

ABCDEFGH IJKLMNOPQRSTU VWXY

SAMPLE OUTPUT #2

2

CHAPTER 6

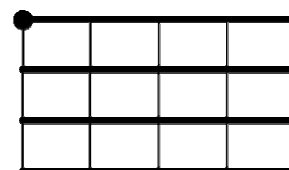
Section 6.1

★Postal Vans 邮政货车

郊区呈矩形, 有四条东西方向的街道和 N ($1 \leq N \leq 1000$) 条南北方向的街道. 在交区的西北角有一个邮局.

如 $N=5$ 时, 郊区如右图所示, 圆点表示邮局, 直线表示街道.

每天邮政卡车从邮局出发, 每个十字路口 (包括边界和四角) 经过且只经过一次. 现在邮局希望知道邮政货车行驶的路线有几种.



PROGRAM NAME: vans

INPUT FORMAT

一行: 一个数值 N

SAMPLE INPUT (file vans.in)

4

OUTPUT FORMAT

一行: 到 INPUT 中给出的街道的路径总数

SAMPLE OUTPUT (file vans.out)

12

★A Rectangular Barn 矩形牛棚

到底是个资本家, Farmer John 想通过买更多的奶牛来扩大它的生意. 它需要给奶牛建造一个新的牛棚.

FJ 买了一个矩形的 R ($1 \leq R \leq 3000$) 行 C ($1 \leq C \leq 3000$) 列的牧场. 不幸的是, 他发现某些 1×1 的区域被损坏了, 所以它不可能在把整个牧场建造成牛棚了.

FJ 数了一下, 发现有 P ($1 \leq p \leq 30000$) 个 1×1 的损坏区域并且请你帮助他找到不包含损坏区域的面积最大的牛棚.

PROGRAM NAME: rectbarn

INPUT FORMAT

• 第 1 行: 三个空格隔开的整数 R , C , and P .

• 第 2.. $P+1$ 行: 每行包含两个空格隔开的整数, r 和 c , 给出一个损坏区域的行号和列号.

SAMPLE INPUT (file rectbarn.in)

3 4 2

1 3

2 1

OUTPUT FORMAT

1 行: 牛棚的最大可能面积

SAMPLE OUTPUT (file rectbarn.out)

6

OUTPUT DETAILS

```
  1 2 3 4
.+++++++
1|  | |X| |
.+++++++
2|X|##|##|
.+++++++
3|  |##|##|
.+++++++
```

标 'X' 的区域是损坏的, 标 '#' 的区域是牛棚.

★Cow XOR 奶牛异或

农民约翰在喂奶牛的时候被另一个问题卡住了. 他的所有 N ($1 \leq N \leq 100,000$) 个奶牛在他面前排成一行 (按序号 $1..N$ 的顺序), 按照它们的社会等级排序. 奶牛 #1 由最高的社会等级, 奶牛 #N 最低. 每个奶牛同时被赋予了一个唯一的数在 $0..2^{21} - 1$ 的范围内.

帮助农民约翰找出应该从那一头奶牛开始喂, 使得从它开始的某一个连续的自序列上的奶牛的数的异或最大. 如果有多个这样的子序列, 选择结尾的奶牛社会等级最高的. 如果还不唯一, 选择最短的.

PROGRAM NAME: cowxor

INPUT FORMAT

- 第 1 行: 一个单独的整数 N .
- 第 2 到 $N + 1$ 行: N 个 $0..2^{21} - 1$ 之间的整数, 代表每头奶牛的被赋予的数. 第 j 行描述了社会等级 $j - 1$ 的奶牛.

SAMPLE INPUT (file cowxor.in)

```
5
1
0
5
4
2
```

OUTPUT FORMAT

- 第 1 行: 3 个空格隔开的整数, 分别为: 最大的异或值, 序列的起始位置、终止位置.

SAMPLE OUTPUT (file cowxor.out)

```
6 4 5
```