Xếp hồ sơ

Trên giá sách có N kẹp hồ sơ (N ≤ 20), mỗi hồ sơ được gán một nhãn là một chữ cái. Không có hồ sơ nào trùng nhãn. Khi di chuyển sang địa điểm mới, các hồ sơ được lấy ra, xếp thành chồng và bị xáo trộn thứ tự. Theo yêu cầu của trưởng phòng, các hồ sơ phải được xếp lại theo đúng trình tự từ trái sang phải như cũ. Đáng tiếc, cô văn thư chỉ nhớ có M quan hệ vị trí (M ≤ 50), mỗi quan hệ có dạng *"x ở bên trái y"*, tức là cặp hồ sơ nhãn x nằm bên trái so với cặp hồ sơ nhãn y (có thể không nhất thiết cạnh nhau).

Yêu cầu: Hãy chỉ ra các trình tự ban đầu có thể có của các hồ sơ, biết rằng với M quan hệ đã biết sẽ có *không quá 300* trình tự có thể ban đầu.

Dữ liệu: Vào từ file văn bản HOSO.INP có cấu trúc như sau:

- Dòng đầu của file chứa xâu ký tự độ dài N, mỗi ký tự là nhãn một hồ sơ;
- Dòng thứ 2 là xâu ký tự độ dài 2*M, trong đó mỗi cặp 2 ký tự thứ 2*i -1 và 2*i (i = 1, 2, . . . , M) thể hiện một quan hệ vị trí.

Kết quả: Ghi ra file HOSO.OUT, mỗi dòng một xâu N ký tự, ứng với một trình tự có thể có của tập hồ sơ.

Ví dụ: File HOSO.INP và HOSO.OUT có thể như sau:

HOSO.INP	HOSO.OUT
ABFG	ABFG
ABBF	ABGF
	AGBF
	GABF

Mạng truyền tin

Có N máy nhắn tin (N ≤ 2000), đánh số từ 1 tới N và được trang bị cho N người, mỗi người một máy. Mỗi máy được xác lập chế độ chỉ nhắn tới một máy nhất định hoặc chưa định hướng tới máy khác. Khi nhận được thông báo bất kỳ, mỗi người lại truyền đến người khác theo máy của mình. Hãy kiểm tra xem, hệ thống có đảm bảo được để một thông báo tới người nào đó thì cũng sẽ tới được tất cả mọi người hay không? Nếu tính chất này không đảm bảo được thì hãy chỉ ra một cách chỉnh lại chế độ làm việc của một số ít nhất các máy nhắn tin để từ một người bất kỳ có thể truyền thông báo đến mọi người còn lại.

Dữ liệu: vào từ file văn bản COMMUN.INP, theo quy cách sau:

- Dòng đầu tiên: số nguyên N.
- Các dòng sau: mỗi dòng 2 số nguyên I, J, cho biết máy I nhắn được tới máy J.
- Dòng cuối cùng chứa 2 số 0.

Các số trên một dòng cách nhau ít nhất một dấu cách, những máy chưa xác lập chế độ nhắn được coi là tự nhắn tới mình và có thể không được nêu trong danh sách trên.

Kết quả: Ghi ra file văn bản COMMUN.OUT theo qui cách sau

- Dòng đầu tiên là số nguyên M - số máy cần chỉnh.
- M dòng tiếp theo: mỗi dòng 3 số nguyên I J K, ứng với một máy cần chỉnh, trong đó I máy cần chỉnh, J - máy nhận cũ, K - máy nhận mới.

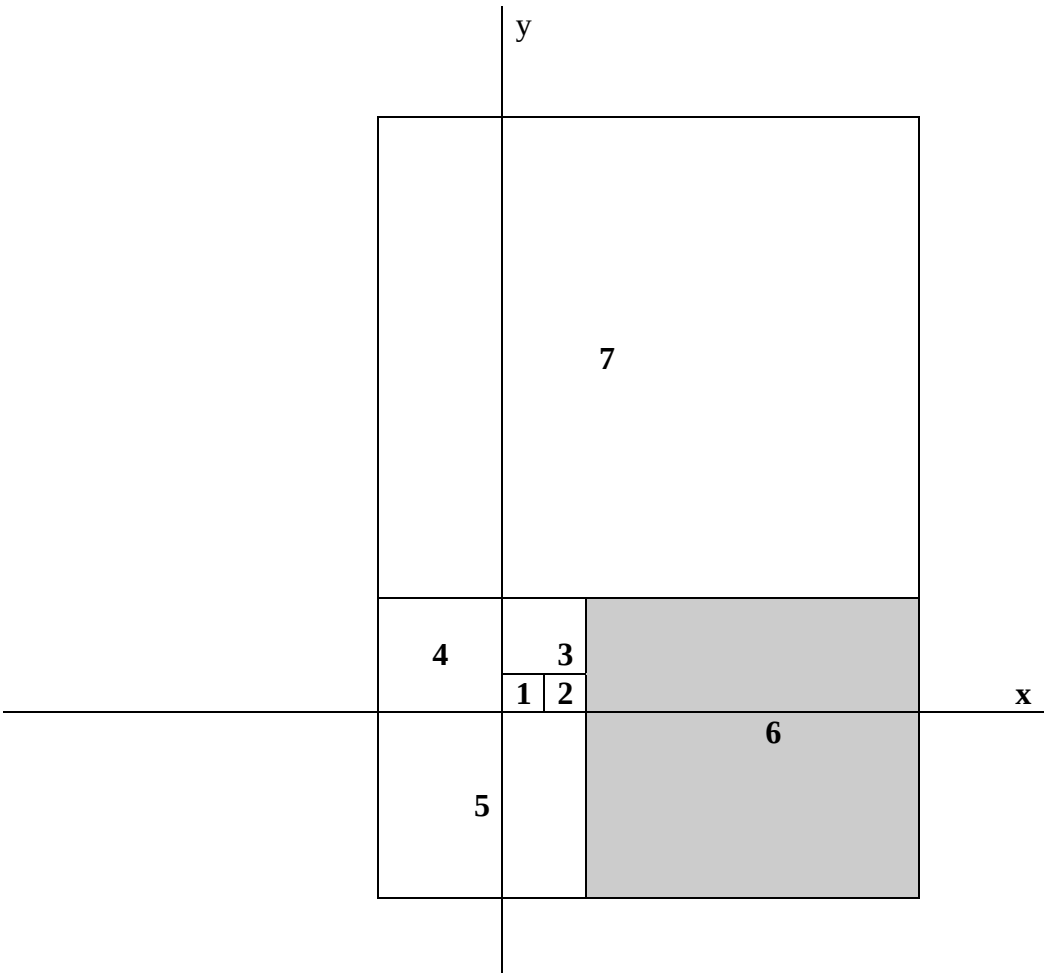
Ví dụ: File dữ liệu và kết quả có thể như sau:

COMMUN.INP
8 6 3 8 4 3 5 5 8 4 3 7 3

COMMUN.OUT
3 7 3 6 4 3 2 2 2 7

Đường đi qua các hình vuông

Trên mặt phẳng toạ độ xây dựng N hình vuông (đánh số từ 1 đến N) có các cạnh song song với các trục toạ độ. Hình vuông thứ 1 có đỉnh góc trái dưới trùng với gốc toạ độ và độ dài cạnh bằng 1. Hình vuông thứ 2 có đỉnh góc trái dưới trùng với đỉnh góc dưới phải của hình vuông thứ 1 và độ dài cạnh là 1. Hình vuông thứ i (i ≥ 3) có độ dài cạnh bằng tổng độ dài các cạnh của hai hình vuông thứ i-1 và i-2 và được vẽ theo qui tắc mô tả trong hình vẽ dưới đây:



Hình 1. Dãy gồm 7 hình vuông

Mỗi hình vuông được tô bởi mầu trắng hoặc đen. Nếu hình vuông được tô màu trắng (đen) thì để đơn giản ta sẽ gọi nó là hình vuông trắng (đen). Từ một hình vuông bất kỳ ta chỉ có thể di chuyển sang hình vuông cùng màu có điểm chung với nó. Cho hai hình vuông trắng với số hiệu là s và t. Ta gọi độ dài của cách di chuyển từ hình vuông s sang hình vuông t là tổng độ dài của các cạnh hình vuông cần phải di chuyển qua. Hãy tìm cách di chuyển từ hình vuông s đến hình vuông t có độ dài là nhỏ nhất, trong trường hợp có nhiều cách di chuyển có cùng độ dài nhỏ nhất cần đưa ra cách di chuyển qua nhiều hình vuông nhất.

Dữ liệu: Vào từ file văn bản SQUARE.INP có cấu trúc như sau

- Dòng đầu tiên ghi số hình vuông N (3 < N < 101) và hai số s và t;
- Dòng tiếp theo ghi N số m_1, m_2, ..., m_N, trong đó m_i = 1 nếu hình vuông thứ i có màu đen và m_i = 0 nếu hình vuông thứ i có màu trắng.

(Các số trên cùng dòng được ghi cách nhau bởi dấu cách).

Kết quả: Ghi ra file văn bản SQUARE.OUT theo qui cách sau:

- Dòng đầu tiên ghi tổng độ dài của cách di chuyển tìm được hoặc ghi số 0 nếu không tìm được cách di chuyển;
- Trong trường hợp tìm được cách di chuyển, dòng tiếp theo ghi *dãy* chỉ số các hình vuông tương ứng với cách di chuyển tìm được bắt đầu từ s và kết thúc bởi t.

Ví dụ. Các file dữ liệu vào ra có thể có dạng

SQUARE.INP	SQUARE.OUT
7 7 5	3
0 0 0 0 0 1 0	7 3 1 5

Tam giác thần bí

Cho một lưới ô vuông gồm n×n ô và số nguyên dương k. Tìm cách điền các số tự nhiên từ 1 đến 3n-3 vào các ô ở cột đầu tiên, dòng cuối cùng và đường chéo chính sao cho tổng các số điền trong cột đầu tiên, dòng cuối cùng và đường chéo chính của lưới đều bằng k.

Ví dụ: Với n = 5, k=35 ta có cách điền sau:

11				
10	2			
9		3		
1			7	
4	5	6	8	12

Dữ liệu: File văn bản MAGICT.INP chứa hai số nguyên dương n, k ($3 \le n \le 100$).

Kết quả: Ghi ra file văn bản MAGICT.OUT theo cấu trúc sau: dòng thứ i ghi các số điền trong dòng thứ i của lưới (i = 1, 2, ..., n).

Ví dụ: Các file dữ liệu vào ra có thể có dạng:

MAGICT.INP
5 35

MAGICT.OUT
11
10 2
9 3
1 7
4 5 6 8 12

Tìm vị trí

Một Uỷ ban bầu cử cần tìm k vị trí đặt thùng phiếu trong một khu vực dân cư gồm n địa điểm đánh số từ 1 đến n. Mỗi thùng phiếu cần đặt tại một trong số n địa điểm dân cư. Có m tuyến đường giao thông (đánh số từ 1 đến m), mỗi tuyến nối một cặp địa điểm dân cư nào đó. Cần tìm k địa điểm đặt thùng phiếu sao cho tổng độ dài đường đi của các cử tri (mỗi cử tri đến bỏ phiếu ở nơi đặt thùng phiếu gần địa điểm mình ở nhất) là nhỏ nhất.

Dữ liệu: Vào từ file văn bản LOCATION.INP có cấu trúc như sau:

- Dòng đầu tiên ghi 3 số n, k, m;
- Dòng tiếp theo ghi cóc số nguyờn dương c_1, c_2,..., c_n (c_i - số cử tri ở địa điểm i, i = 1, 2, ..., n);
- m dòng tiếp theo mỗi dũng chứa 3 số nguyên dương u_j, v_j, d_j theo thứ tự là điểm đầu, điểm cuối và độ dài của tuyến đường j, j = 1, 2, ..., m;

(Qui ước: các dữ liệu số được ghi cách nhau bởi dấu cách).

Kết quả: Ghi ra file văn bản LOCATION.OUT dưới dạng:

- Dòng đầu tiên ghi tổng độ dài đường đi của tất cả các cử tri;
- Dòng tiếp theo ghi k số p_1, p_2,..., p_k là các vị trí đặt thùng phiếu.

Xếp lịch

Một trung tâm máy tính cần bố trí việc thực hiện n chương trình (đánh số từ 1 đến n) của khách hàng trên m máy tính với cấu hình giống nhau đánh số từ 1 đến m. Chương trình thứ i đòi hỏi t_i đơn vị thời gian để hoàn thành nó trên một trong số m máy nói trên. Việc thực hiện các chương trình trên máy cần được tiến hành liên tục, không được phép ngắt quãng. Thời gian mà máy chuyển từ việc thực hiện chương trình này sang thực hiện chương trình tiếp theo là không đáng kể. Giả sử thời điểm bắt đầu việc thực hiện các chương trình là 0, hãy tìm cách bố trí việc thực hiện các chương trình sao cho thời điểm hoàn thành tất cả các chương trình là càng sớm càng tốt.

Dữ liệu vào: File văn bản SCHEDULE.INP có cấu trúc như sau:

- Dòng đầu tiên chứa hai số nguyên dương n, m ($1 \leq m < n \leq 100$);
- Dòng tiếp theo chứa các số nguyên dương $t_1, t_2, ..., t_n$ ($1 \leq t_i \leq 30000$).

Kết quả: Ghi ra file văn bản SCHEDULE.OUT theo cấu trúc:

- Dòng đầu tiên ghi thời điểm hoàn thành việc thực hiện các chương trình;
- Mỗi dòng thứ j trong số m dòng tiếp theo ghi các chương trình được bố trí thực hiện trên máy j, j = 1, 2, ..., m.

Vớ dụ: Các file dữ liệu vào ra có thể có dạng

SCHEDULE.INP
6 3
1 2 4 3 5 6

SCHEDULE.OUT
7
1 6
2 5
3 4

Đặt tâm phục vụ

Có n địa điểm dân cư đánh số từ 1 đến n. Giữa m cặp địa điểm trong số n địa điểm nói trên có tuyến đường nối chúng. Biết độ dài của các tuyến đường này. Người ta cần xây dựng một trung tâm dịch vụ tổng hợp tại một địa điểm hoặc là trùng với một trong số các địa điểm dân cư nói trên, hoặc là nằm trên tuyến đường nối hai địa điểm nào đó, sao cho tổng khoảng cách từ trung tâm dịch vụ đến n địa điểm dân cư là nhỏ nhất (ta gọi khoảng cách giữa hai địa điểm là độ dài đường đi ngắn nhất nối chúng).

Dữ liệu: Vào từ file văn bản có tên TAMPV.INP có cấu trúc như sau:

- Dòng đầu tiên ghi hai số m, n cách nhau bởi dấu cách,
- Dòng thứ i trong số m dòng tiếp theo chứa 3 số mô tả thông tin về tuyến đường thứ i: hai số đầu là chỉ số của hai địa điểm dân cư được nối với nhau bởi tuyến đường này còn số thứ 3 là độ dài của tuyến đường (giả thiết là số nguyên dương).

Kết quả: Ghi ra file văn bản TAMPV.OUT thông tin về vị trí đặt trung tâm dịch vụ theo qui cách sau:

- Dòng đầu tiên ghi chỉ số của địa điểm cần đặt trung tâm dịch vụ hoặc nếu vị trí tìm được nằm trên tuyến đường thì đưa ra hai đầu của tuyến đường cùng khoảng cách từ vị trí tìm được đến đầu thứ nhất,
- Dòng thứ hai ghi tổng khoảng cách từ vị trí tìm được đến N địa điểm dân cư.

Ví dụ:

TAMPV.INP
7 5 1 2 10 1 5 15 2 3 5 2 5 7 3 4 8 5 4 9

TAMPV.OUT
3 31

Nối mạng máy tính

Người ta muốn nối n máy vi tính đánh số từ 1 đến n thành một mạng bằng cách lắp đặt các kênh nối giữa một số cặp máy tính sao cho:

- Hai máy bất kỳ trong mạng đều có thể trao đổi thông tin với nhau hoặc là trực tiếp hoặc là thông qua một số máy trung gian;
- Máy thứ i được nối với đúng $d_i \geq 2$ máy khác, i=1,2,...,n.

Dữ liệu: Vào từ file văn bản NET.INP có cấu trúc như sau:

- Dòng đầu tiên ghi số n,
- Dòng tiếp theo ghi các số $d_1, d_2, \ldots, d_n$.

Kết quả: Ghi ra file văn bản NET.OUT gồm n dòng: dòng thứ i ghi danh sách gồm d_i số là chỉ số các máy được nối với máy i (i=1,2,...,n).

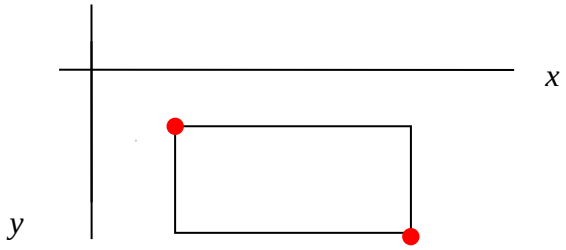
Ví dụ:

NET.INP
5 3 3 3 2 3

NET.OUT
2 3 5 1 3 5 1 2 4 3 5 1 2 4

Phân hoạch nhỏ nhất

Cho N hình chữ nhật khác nhau có các cạnh song song với các trục tọa độ (N ≤ 500). Mỗi hình chữ nhật được xác định bởi đỉnh ở góc trên bên trái và đỉnh ở góc dưới bên phải của nó (xem hình vẽ). Giả thiết các hình chữ nhật được đánh số từ 1 đến N và tọa độ các đỉnh của chúng đều là các số nguyên.



Yêu cầu: Hãy tìm cách phân tập các hình chữ nhật ra thành một số ít nhất các tập con đôi một rời nhau sao cho trong mỗi tập con không tìm được hai hình chữ nhật mà hình này là nằm trong hình kia (có nghĩa là mọi điểm của hình này đều là điểm của hình kia).

Dữ liệu vào: File văn bản RECT.INP có cấu trúc như sau:

- Dòng đầu tiên ghi số N;
- Mỗi dòng thứ i trong số N dòng tiếp theo mô tả hình chữ nhật thứ i gồm 4 số nguyên: 2 số đầu là tọa độ x và tọa độ y của đỉnh góc trên bên trái, 2 số tiếp theo là tọa độ x và tọa độ y của đỉnh góc dưới bên phải của nó.

Kết quả: Ghi ra file văn bản RECT.OUT theo cấu trúc

- Dòng đầu ghi M là số tập con tìm được,
- Dòng thứ i trong số M dòng tiếp theo ghi các số hiệu của các hình chữ nhật thuộc tập con thứ i (i=1,2,...,M).

Ví dụ: File RECT.INP và RECT.OUT có thể

RECT.INP	RECT.OUT
7 1 1 4 4 2 3 6 7 4 5 6 7 1 1 4 9 4 1 8 9 6 7 8 9 8 1 10 4	2 1 3 6 7 2 5 4

Hình vuông latinh

Hình vuông la tinh cấp n ($n \le 26$) là ma trận vuông kích thước n×n mà mỗi dòng và mỗi cột của nó đều là một hoán vị của n chữ cái đầu tiên trong dãy chữ A, B,..., Z. Hai hình vuông la tinh được gọi là tương đương nếu từ hình này ta có thể thu được hình kia nhờ sử dụng các phép biến đổi sau: 1) đổi chỗ hai dòng; 2) đổi chỗ hai cột; 3) đổi tên hai chữ cái.

Ví dụ: Hai hình vuông latinh cấp 4

$$\Omega_1 = \begin{vmatrix} A & B & C & D \\ B & D & A & C \\ C & A & D & B \\ D & C & B & A \end{vmatrix} \qquad \Omega_2 = \begin{vmatrix} A & B & C & D \\ B & C & D & A \\ C & D & A & B \\ D & A & B & C \end{vmatrix}$$

là tương đương, bởi vì đổi chỗ dòng 3 và 4 của Ω_1 ta thu được

$$\begin{vmatrix} A & B & C & D \\ B & D & A & C \\ D & C & B & A \\ C & A & D & B \end{vmatrix},$$

tiếp đến đổi chỗ cột 3 và 4 ta thu được

$$\begin{vmatrix} A & B & D & C \\ B & D & C & A \\ D & C & A & B \\ C & A & B & D \end{vmatrix},$$

cuối cùng, đổi tên hai chữ C và D cho nhau (nghĩa là thay C bởi D và thay D bởi C) ta thu được Ω_2.

Yêu cầu: Cho Ω_1, Ω_2 là hai hình vuông la tinh cấp n. Hãy xác định xem hai hình vuông đã cho có tương đương hay không?

Dữ liệu vào: file văn bản LATIN.INP có cấu trúc như sau

- Dòng đầu tiên chứa số n ($n \le 26$);
- n dòng tiếp theo chứa các dòng của hình vuông Ω_1;
- n dòng cuối cùng chứa các dòng của hình vuông Ω_2;

(các phần tử trong một dòng được viết liền nhau);

Kết quả: Ghi ra file văn bản LATIN.OUT dưới dạng sau:

- Dòng đầu tiên ghi số lượng phép biến đổi k (k = 0, nếu hai hình vuông là không tương đương);
- Các dòng tiếp theo ghi dãy các phép biến đổi cần áp dụng để từ Ω_1 thu được Ω_2: thông tin về một phép biến đổi bao gồm chỉ số của phép biến đổi, chỉ số hai dòng (cột) cần đổi chỗ hoặc hai chữ cái cần đổi tên cho nhau.

Ví dụ: các file dữ liệu và kết quả có thể:

LATIN.INP	LATIN.OUT
4	3
ABCD	1 3 4
BDAC	2 3 4
CADB	3 C D
DCBA	
ABCD	
BCDA	
CDAB	
DABC	

Chuyển kênh

Công ty truyền hình cáp CPN là chủ sở hữu của một loạt kênh truyền hình. Công ty muốn xác định thời điểm phát các chương trình sao cho khán giả có thể chuyển kênh mà không phải bỏ dở phần cuối của chương trình này hoặc không xem được phần đầu của chương trình khác. Để làm điều đó Công ty xác định một số thời điểm gọi là "***điểm dóng hàng***", là thời điểm lý tưởng để một chương trình nào đó kết thúc còn một chương trình khác bắt đầu. Có một số điểm dóng hàng là quan trọng hơn so với một số điểm dóng hàng khác. Chẳng hạn, nhiều khán giả muốn theo dõi chương trình thời sự, vì vậy họ sẽ không muốn theo dõi chương trình của hãng mà thời điểm kết thúc, hoặc thời điểm bắt đầu của nó nằm trong khoảng thời gian phát chương trình thời sự (vì muốn theo dõi chương trình của hãng họ sẽ không theo dõi được trọn vẹn chương trình thời sự). Như vậy, mỗi điểm dóng hàng sẽ có một ***mức độ quan trọng*** được biểu diễn bởi một số nguyên dương q (q càng nhỏ thì điểm dóng càng quan trọng). Cần xác định ***trình tự tốt nhất*** mà theo đó sẽ phát các chương trình trên ***một kênh*** của hãng.

Để so sánh hai trình tự phát sóng (chương trình) ta cần đưa vào một số khái niệm. Ta gọi ***thời gian thiếu hụt*** của một điểm dóng hàng là trị tuyệt đối của hiệu giữa thời điểm của điểm dóng hàng và thời điểm của điểm bắt đầu hoặc kết thúc của chương trình gần nhất. ***Thời gian thiếu hụt tổng cộng*** của mức độ quan trọng q là tổng thời gian thiếu hụt của các điểm dóng hàng có mức độ quan trọng q. Trình tự phát sóng (A) là ***tốt hơn*** trình tự phát sóng (B), nếu như thời gian thiếu hụt tổng cộng ở một mức độ quan trọng nào đó của (A) là bé hơn của (B); còn ở các mức độ quan trọng cao hơn (A) và (B) có thời gian thiếu hụt tổng cộng là như nhau.

Dữ liệu vào: File văn bản SWIT.INP có cấu trúc như sau:
- dòng đầu tiên chứa số chương trình cần sắp thứ tự p ($0 < p \le 20$);
- dòng thứ hai chứa p số nguyên dương là độ dài của p chương trình cần phát (tính bằng phút);
- dòng thứ ba chứa số điểm dóng hàng n ($0< n \le 20$);
- dòng thứ i trong số n dòng tiếp theo mỗi dòng chứa hai số nguyên dương q_i, t_i là mức độ quan trọng và thời điểm của điểm dóng hàng i (i=1,2,...,n).

Kết quả: Ghi ra file văn bản SWIT.OUT theo qui cách sau
- dòng đầu tiên ghi dãy các độ dài của các chương trình theo thứ tự phát sóng tìm được;
- dòng tiếp theo ghi tổng thời gian thiếu hụt của tất cả các điểm dóng hàng theo trình tự tìm được.

Ví dụ: Các file dữ liệu và kết quả có thể

SWIT.INP
```
6
10 15 13 18 25 33
4
1  30
2  15
2  45
1  60
```

SWIT.OUT
```
15  13  33  25  18  10
19
```

Sắp xếp dãy số

Cho dãy số thực

$a_1, a_2, \ldots, a_n$ $(n \le 100)$

và hai số nguyên dương k ($k \ge 2$), p. Hãy tìm cách sắp xếp lại các phần tử của dãy số sao cho nếu lấy ra bất cứ nhóm gồm k phần tử liên tiếp nào của dãy thì trị tuyệt đối của hiệu hai số bất kỳ trong nhóm cũng không vượt quá p.

Dữ liệu: Vào từ file văn bản BL1.INP:

- Dòng đầu: n k p;
- Dòng tiếp theo: $a_1, a_2, \ldots, a_n$.

Kết quả: Đưa ra file văn bản BL1.INP:

- Dòng đầu ghi các phần tử của dãy số đã xếp lại
- Dòng thứ hai ghi chỉ số trong dãy đã cho của các phần tử của dãy đã xếp lại.

Ví dụ:

BL1.INP
10 3 4
1 16 5 5 9 17 12 14 21 9

BL2.OUT
1 5 5 9

Biến đổi bảng số

Cho lưới ô vuông A kích thước M*N, M≤100, N ≤ 15. Ở mỗi ô (i,j) của lưới có một giá trị A_{ij} nguyên. Bằng cách đổi chỗ các hàng của lưới A và đổi chỗ các phần tử trong hàng, người ta có thể nhận được lưới B với tính chất:

$$B_{i-1,\,N} = B_{1,i}\ ,\ i=2,3,\ ...,\ K.$$

Yêu cầu: Hãy xác định B sao cho K là lớn nhất.

Dữ liệu: Vào từ file văn bản BANGSO.INP có cấu trúc như sau

- Dòng đầu chứa hai số M, N.
- Dòng thứ i trong nhóm M dòng tiếp theo chứa N số $A_{i,1}, A_{i,2}, ... , A_{i,N}$.

Kết quả: Ghi ra file văn bản BANGSO.OUT theo qui cách sau

- Dòng đầu ghi số K,
- Dòng thứ i trong nhóm M dòng tiếp theo ghi N số $B_{i,1}, B_{i,2}, ... , B_{i,N}$.

Ví dụ: File dữ liệu và kết quả có thể có dạng

BANGSO.INP

```
7  6
1      8      8      8      8      4
7      7      4      7      7      7
5      6      5      3      6      5
4      4      4      4      4      4
1      2      0      8      3      1
9      9      4      2      9      9
0      4      0      4      0      4
```

BANGSO.OUT

```
5
5      5      6      5      6      3
3      8      2      1      0      1
1      8      8      8      8      4
4      4      4      4      4      4
4      9      9      9      2      9
7      7      7      7      4      7
0      4      0      4      0      4
```

Mạng chẵn lẻ

Có N máy tính đánh số từ 1 đến N được nối với nhau thành một mạng bởi các đoạn nối trực tiếp giữa một số cặp máy nào đó (gọi là kênh). Mạng được gọi là chẵn-lẻ nếu như trong nó tìm được hai máy có thể trao đổi thông tin với nhau thông qua một số chẵn kênh cũng như thông qua một số lẻ kênh.

Yêu cầu: Kiểm tra xem mạng đã cho có phải chẵn-lẻ hay không? Nếu câu trả lời là phủ định hãy tìm tập con X các máy có lực lượng lớn nhất thoả mãn điều kiện: đối với hai máy bất kỳ trong X nếu chúng có thể trao đổi thông tin với nhau thì bao giờ cũng phải thông qua một số chẵn kênh.

Dữ liệu: Vào từ file văn bản CHANLE.INP có cấu trúc như sau

- Dòng đầu tiên chứa số N (N<300),
- Mỗi dòng trong số các dòng tiếp theo chứa một cặp số i, j cho biết máy i được nối trực tiếp với máy j.

Kết quả: Ghi ra file văn bản CHANLE.OUT theo qui cách sau
Nếu mạng là chẵn-lẻ thì ghi vào dòng đầu tiên thông báo: MANG LA CHAN LE, ngược lại kết quả cần ghi theo qui cách sau:

- Dòng đầu tiên ghi số phần tử của tập X
- Trong các dòng tiếp theo ghi các chỉ số của các máy trong tập X, các chỉ số được ghi cách nhau bởi ít nhất một dấu cách.

Ví dụ:

Nếu file CHANLE.INP có dạng

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

thì kết quả đưa ra file CHANLE.OUT sẽ là:

```
MANG LA CHAN LE
```

Nếu file CHANLE.INP có dạng

```
3
1 2
```

thì kết quả đưa ra file CHANLE.OUT sẽ là:

```
2
2 3
```

Xếp hàng

Trong giải vô địch bóng chuyền có n đội đánh số từ 1 đến n thi đấu vòng tròn một lượt. Kết quả thi đấu được cho bởi một bảng số a_{ij} (i, j = 1, 2,..., n), trong đó $a_{ij} = 1$ nếu đội i thắng trong trận gặp đội j, và $a_{ij} = 0$ nếu đội i thua trong trận gặp đội j. Kết thúc giải Ban tổ chức muốn mời các đội trưởng của các đội bóng ra xếp thành một hàng ngang để chụp ảnh. Hãy tìm cách xếp các đội trưởng thành một hàng ngang sao cho, ngoại trừ hai người đứng ở hai mép hàng, mỗi người trong hàng đều đứng cạnh một đội trưởng của đội thắng, một đội trưởng của đội thua mình trong trận đấu của giải.

Dữ liệu: Vào từ file văn bản có tên XEPHANG.INP:

- Dòng đầu tiên chứa số n,
- Các dòng tiếp theo chứa các số a_{11}, a_{12},..., a_{1n}, a_{21}, a_{22},..., a_{2n},..., a_{n1}, a_{n2},..., a_{nn} (các số ghi cách nhau bởi dấu cách hoặc dấu xuống dòng).

Kết quả: Ghi ra file văn bản XEPHANG.OUT dưới dạng dãy gồm n số hiệu của các đội tương ứng với thứ tự đứng của các đội trưởng trong hàng ngang tìm được.

Ví dụ:

XEPHANG.INP

```
6
0 1 1 1 1 1
0 0 1 1 1 1
0 0 0 1 1 1
0 0 0 0 0 1
0 0 0 1 0 0
0 0 0 0 1 0
```

XEPHANG.OUT

```
1 2 3 5 4 6
```

Xếp việc

Có m thợ đánh số từ 1 đến m, và có n công việc đánh số từ 1 đến n. Cần tìm cách phân công thợ thực hiện công việc sao cho:

- Mỗi thợ phải tham gia vào việc thực hiện a_i công việc (i=1,2,...,m);
- Mỗi công việc phải có b_j thợ tham gia thực hiện nó (j=1,2,...,n).

Dữ liệu: Vào từ file văn bản XEPVIEC.INP có cấu trúc như sau

- Dòng đầu tiên chứa hai số m, n (m, n < 100);
- Dòng thứ hai: a_1, a_2,..., a_m;
- Dòng thứ ba: b_1, b_2,..., b_n.

Kết quả: Ghi ra file văn bản XEPVIEC.OUT: dòng thứ i ghi chỉ số các công việc mà thợ i phải tham gia thực hiện.

Ví dụ:

XEPVIEC.INP
```
5  6
4    3    3    4    4
3    3    3    2    3    4
```

XEPVIEC.OUT
```
1    2    3    6
2    5    6
1    3    4
1    3    5    6
2    4    5    6
```

Đổi tiền (Making change)

Có k loại tiền đánh số từ 1 đến k với k mệnh giá khác nhau, trong đó giả thiết là đồng tiền loại 1 có mệnh giá là 1. Mệnh giá của các đồng tiền khác được qui đổi qua mệnh giá của đồng tiền loại 1 (k<10). Cần liệt kê các cách đổi một giá trị là N ra thành các loại tiền khác nhau.

Dữ liệu: Vào từ file văn bản CHANGE.INP có cấu trúc như sau

- Dòng đầu tiên chứa số k và N;
- Dòng thứ i trong số k-1 dòng tiếp theo ghi mệnh giá của loại tiền thứ i+1 qua giá trị đồng tiền loại 1.

Kết quả: Ghi ra file CHANGE.OUT theo qui cách sau

- Dòng đầu tiên ghi số cách đổi tiền M;
- Mỗi dòng thứ i trong số M dòng tiếp theo ghi cách qui đổi tiền thứ i bao gồm K số nguyên không âm q_1 q_2 ... q_k chỉ rõ số lượng đồng tiền các loại trong cách đổi tiền.

Ví dụ: Các file dữ liệu và kết quả có thể

CHANGE.INP
4 23
5
10
20

CHANGE.OUT
10
23 0 0 0
18 1 0 0
13 2 0 0
8 3 0 0
3 4 0 0
13 0 1 0
8 1 1 0
3 2 1 0
3 0 2 0
3 0 0 1

Sự lựa chọn

Trong một kỳ thi dự báo kết quả thi đấu trận chung kết tranh giải vô địch bóng đá SEAGAMES có N (N<101) người có kết quả dự đoán đúng. Để lựa chọn người được nhận giải thưởng, Ban Tổ chức quyết định xếp hàng N người vào một vòng tròn, đánh số từ 1 đến N theo chiều ngược chiều kim đồng hồ (tức là người đánh số N đứng bên trái người đứng số 1), sau đó lựa chọn một cách ngẫu nhiên 4 số nguyên dương m, k, p, q, trong đó m, k < N, còn p, q là số nguyên dương tuỳ ý. Tiếp theo người được nhận giải thưởng sẽ được xác định nhờ thủ tục loại bỏ dần sau đây: Ban Tổ chức cử ra 2 nhân viên, nhân viên thứ nhất bắt đầu đếm từ người đứng ở vị trí m theo chiều ngược với kim đồng hồ từ 1 đến p, còn nhân viên thứ hai bắt đầu đếm từ người đứng ở vị trí k theo chiều kim đồng hồ từ 1 đến q. Hai người đứng ở vị trí kết thúc đếm của hai nhân viên sẽ dời khỏi hàng. Đến đây kết thúc một lượt đếm. Lưu ý rằng, nhân viên thứ hai đếm đồng thời với nhân viên 1, do đó anh ta có thể đếm cả người ở vị trí kết thúc đếm của nhân viên 1, ngoài ra, nếu vị trí kết thúc đếm của hai nhân viên là trùng nhau thì chỉ có một người phải rời khỏi hàng. Sau khi những người không may mắn đã rời khỏi hàng, mỗi nhân viên lại tiếp tục lượt đếm mới bắt đầu từ người đứng cạnh người vừa bị loại theo vòng đếm của họ. Người phải dời khỏi hàng cuối cùng theo thủ tục lựa chọn trên là người được nhận giải. Nếu ở lượt đếm cuối cùng, có hai người phải đồng thời dời khỏi hàng thì cả hai người này cùng được nhận giải.

Dữ liệu: Vào từ file văn bản có tên SELECT.INP trong đó ghi 5 số N, m, k, p , q cách nhau bởi ít nhất một dấu cách.

Kết quả: Ghi ra file văn bản SELECT.OUT có cấu trúc như sau:

- Dòng đầu tiên ghi số L là số lần thực hiện việc đếm,
- Mỗi dòng thứ i trong số L dòng tiếp theo ghi chỉ số của những người dời khỏi hàng ở lượt đếm thứ i (i=1,2,...,L).

Ví dụ. Nội dung file SELECT.INP và file SELECT .OUT có thể như sau:

SELECT.INP
```
10  1  10  4  3
```

SELECT.OUT
```
6
4  8
9  5
3  1
2  6
10
7
```

Chi phí đi đường

Một Công ty vận tải đường bộ muốn xác định chi phí nhỏ nhất để di chuyển từ một thành phố này đến một thành phố khác. Công ty có danh sách những trạm xăng trên đoạn đường di chuyển giữa hai thành phố. Danh sách bao gồm vị trí của các trạm xăng và giá bán mỗi lít xăng ở các trạm.

Để đơn giản cho cách tính chi phí, công ty sử dụng các qui tắc sau về hành vi của người lái xe:

- Lái xe không dừng lại trạm xăng để nạp xăng cho xe chừng nào xăng trong bình xăng của xe vẫn còn nhiều hơn một nửa dung tích của bình và vẫn đủ để đạt trạm xăng tiếp theo.
- Mỗi khi dừng xe để nạp xăng, lái xe luôn nạp đầy bình xăng.
- Mỗi khi dừng xe nạp xăng ở trạm xăng lái xe luôn uống nước mất 2 USD.
- Ở điểm xuất phát bình xăng luôn được đổ đầy.

Cần viết chương trình tính chi phí tối thiểu để lái xe mua xăng và uống nước trên dọc đường đi.

Dữ liệu: Vào từ file văn bản MINBT.INP:

Thông tin về đoạn đường cần đi bao gồm các dòng sau:

Dòng 1: chứa một số thực là khoảng cách (km) giữa điểm xuất phát và đích.

Dòng 2: gồm 3 số thực và một số nguyên:

V - dung tích của bình xăng (lít);

C - khoảng cách xe có thể đi nhờ sử dụng 1 lít xăng (km);

P - chi phí đổ đầy xăng ở điểm xuất phát (USD);

n - số lượng trạm xăng trên tuyến đường ($n < 101$).

Mỗi dòng thứ i trong số n dòng tiếp theo chứa hai số thực

d_i - khoảng cách từ điểm xuất phát đến trạm xăng thứ i,

p_i - giá một lít xăng (đơn vị tính: cent = 0.01 USD), i = 1, 2, ..., n.

Giả thiết rằng các trạm xăng được sắp xếp theo thứ tự tăng dần của khoảng cách từ điểm xuất phát đến chúng và các trạm xăng được phân bố ở các vị trí thích hợp để xe có thể đạt đích mà không thiếu nhiên liệu.

Kết quả: Ghi ra file MINBT.OUT tổng chi phí (USD) (có tính cả chi phí đổ xăng ở điểm xuất phát). Qui ước làm tròn chi phí mỗi lần chi ở trạm xăng đến cent.

Ví dụ: File MINBT.INP và MINBT.OUT có thể

MINBT.INP
475.6
11.9 27.4 14.98 6
102.0 99.9
220.0 132.9
256.3 147.9
275.0 102.9
277.6 112.9
381.8 100.9

MINBT.OUT
27.31 USD

Xóa số

Cho một số tự nhiên n chữ số a= $a_1 a_2 ... a_n$ ($a_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, i = 1, 2, ..., n; (n có thể đạt tới giá trị 1000000). Hãy tìm cách xoá bỏ m chữ số của a, sao cho số thu được sau khi xoá bỏ m chữ số đó là nhỏ nhất.

Dữ liệu: Vào từ file văn bản XOASO.INP có cấu trúc:

- Dòng đầu ghi giá trị m và n cách nhau ít nhất một dấu cách.
- n dòng tiếp theo ghi các chữ số của a theo thứ tự từ trái qua phải.

Kết quả: Ghi ra file XOASO.OUT gồm m dòng, mỗi dòng chứa chữ số bị xoá và chỉ số của nó trong dãy số gốc, được phân cách bởi ít nhất một dấu cách.

Ví dụ: với m=2, n=5, a= 41325 thì file XOASO.INP gồm 6 dòng sau:

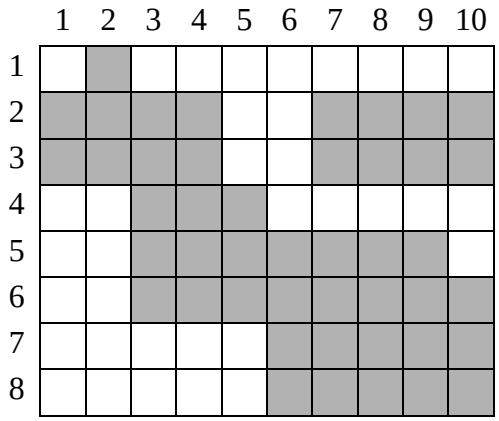
```
2 5
4
1
3
2
5
```

và file XOASO.OUT gồm hai dòng sau:

```
4 1
3 3
```

Hình vuông lớn nhất

Cho lưới ô vuông kích thước m×n. Các ô của lưới được đánh số như chỉ ra trong hình vẽ dưới đây. Mỗi ô của lưới được sơn bởi một trong hai màu đen hoặc trắng.



Cần tìm hình vuông gồm toàn ô đen có kích thước lớn nhất.

Dữ liệu: Vào từ file văn bản SQUARE.INP có cấu trúc như sau:

- Dòng đầu tiên ghi hai số m, n (0 < m, n < 5000);
- Dòng thứ i trong số m dòng tiếp theo chứa n số a_{i1} a_{i2} . . . a_{in}, trong đó a_{ij} = 1 nếu ô (i, j) có màu đen và a_{ij} = 0 nếu ô (i, j) có màu trắng.

Kết quả: Ghi ra file văn bản SQUARE.OUT theo qui cách sau:

- Dòng thứ 1 ghi số k là kích thước của hình vuông tìm được;
- Dòng thứ hai ghi 4 số p, q, r, s, trong đó (p,q) là tọa độ của ô ở góc trên trái còn (r,s) là tọa độ của ô ở góc dưới phải của hình vuông tìm được.

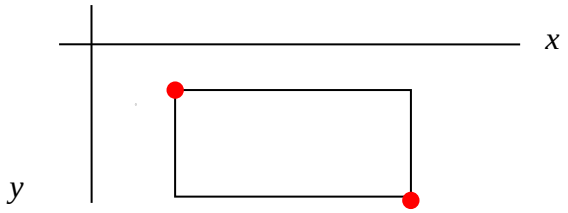
Ví dụ: Với lưới cho trong hình vẽ, các file SQUARE.INP và SQUARE.OUT tương ứng là

SQUARE.INP
8 10
0 1 0 0 0 0 0 0 0 0
1 1 1 1 0 0 1 1 1 1
1 1 1 1 0 0 1 1 1 1
0 0 1 1 1 0 0 0 0 0
0 0 1 1 1 1 1 1 1 0
0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 1 1 1 1 1
0 0 0 0 0 1 1 1 1 1

SQUARE.OUT
4
5 6 8 9

Hình chữ nhật rời nhau

Cho N hình chữ nhật có các cạnh song song với các trục tọa độ (N < 100). Mỗi hình chữ nhật được xác định bởi đỉnh ở góc trên bên trái và đỉnh ở góc dưới bên phải của nó (xem hình vẽ). Giả thiết các hình chữ nhật được đánh số từ 1 đến N và tọa độ các đỉnh của chúng đều là các số nguyên.



Yêu cầu*:* Hãy xác định tập hợp các hình chữ nhật rời nhau từng đôi một có nhiều hình chữ nhật nhất (hai hình chữ nhật gọi là rời nhau nếu chúng không có điểm chung nào).

Dữ liệu vào: File văn bản RECT.INP có cấu trúc như sau:

- Dòng đầu tiên ghi số N;
- Mỗi dòng thứ i trong số N dòng tiếp theo mô tả hình chữ nhật thứ i gồm 4 số nguyên: 2 số đầu là tọa độ x và tọa độ y của đỉnh góc trên bên trái, 2 số tiếp theo là tọa độ x và tọa độ y của đỉnh góc dưới bên phải của nó. Các số trên một dòng được ghi cách nhau bởi ít nhất một dấu trắng.

Kết quả: Ghi ra file văn bản RECT.OUT theo cấu trúc

- Dòng đầu ghi M là số hình chữ nhật nhiều nhất tìm được,
- Mỗi dòng trong số M dòng tiếp theo ghi các số hiệu của các hình chữ nhật này.

Ví dụ: File RECT.INP và RECT.OUT có thể

RECT.INP	RECT.OUT
10 364 152 480 370 264 296 628 370 597 221 639 432 115 290 608 459 94 156 141 374 363 207 550 425 125 361 565 445 481 342 611 396 287 45 457 196 394 290 533 459	4 3 5 6 9

Phân phối kênh

Công ty dịch vụ mạng máy tính cần phân phối kênh hoạt động phục vụ n (n < 1000) yêu cầu của khách hàng đánh số từ 1 đến n. Với mỗi khách hàng thứ i ta biết khoảng thời gian yêu cầu sử dụng kênh là (s_i, t_i), i = 1, 2,..., n (khách hàng sẽ sử dụng kênh từ thời điểm s_i đến thời điểm t_i). Thời gian chuyển giao quyền sử dụng kênh từ khách hàng này cho khách hàng khác là không đáng kể. Như vậy, nếu hai khách hàng nào đó được bố trí làm việc trên cùng một kênh thì các khoảng thời gian sử dụng của họ chỉ có thể có nhiều nhất một điểm chung.

Yêu cầu: Hãy tìm cách phân phối sử dụng ít kênh nhất.

Dữ liệu: Vào từ file văn bản CHANEL.INP có cấu trúc như sau:
- Dòng đầu tiên ghi số n;
- Dòng thứ i trong số n dòng tiếp theo ghi 2 số nguyên dương s_i, t_i, i =1, 2,..., n.

Kết quả: Ghi ra file văn bản với tên CHANEL.OUT theo cấu trúc sau:
- Dòng đầu tiên ghi số lượng kênh cần sử dụng p;
- Mỗi dòng thứ i trong số p dòng tiếp theo chứa các chỉ số của các khách hàng sử dụng kênh thứ i, i = 1, 2,..., p.

Ví dụ: File CHANEL.INP và CHANEL.OUT tương ứng có thể có dạng:

CHANEL.INP
```
7
0       3
3       5
6       8
0       7
7       8
0       2
2       6
```

CHANEL.OUT
```
3
1       2       3
6       7       5
4
```

Nhân ma trận

Như đã biết, tích của ma trận $A = (a_{ik})$ kích thước $p \times n$ với ma trận $B = (b_{kj})$ kích thước $n \times q$ là ma trận $C = (c_{ij})$ kích thước $p \times q$ với các phần tử được tính theo công thức

$$c_{ij} = \sum_{k=1}^{n} a_{ik}\ b_{kj}, \quad 1 \le i \le p, \quad 1 \le j \le q.$$

Rõ ràng, ta phải thực hiện tất cả pqn phép nhân để tính tích của hai ma trận.

Giả sử ta phải tính tích của n ma trận:

$$M = M_1 M_2 \ldots M_n.$$

Do phép nhân ma trận có tính kết hợp, ta có thể tính tích của các ma trận đã cho theo nhiều cách khác nhau, chẳng hạn

$$M = (\ldots ((M_1 M_2)M_3) \ldots M_n) = (M_1(M_2(M_3 \ldots (M_{n-1} M_n) \ldots)))$$
$$= (\ldots ((M_1 M_2) (M_3 M_4)) \ldots),$$

v.v... Mặt khác, do tích ma trận không có tính chất giao hoán, nên ta không được thay đổi thứ tự của các ma trận trong biểu thức đã cho.

Hãy tìm cách tính tích của n ma trận đã cho sao cho tổng số phép nhân phải thực hiện là ít nhất.

Dữ liệu: Vào từ file văn bản MULMAT.INP có cấu trúc như sau:

- Dòng đầu tiên ghi n là số thừa số trong tích ma trận cần tính;
- Dòng thứ i trong số $n+1$ dòng tiếp theo chứa số nguyên dương d_i là thông tin về kích thước của các ma trận thừa số, trong đó ma trận M_i có kích thước $d_i \times d_{i+1}$.

Kết quả: Ghi ra file văn bản có tên MULMAT.OUT theo cấu trúc:

- Dòng đầu tiên ghi số lượng phép nhân cần thực hiện;
- Dòng tiếp theo ghi biểu thức với dấu ngoặc thể hiện cách tính tìm được.

Ví dụ. File MULMAT.INP và MULMAT.OUT tương ứng có thể có dạng:

MULMAT.INP
4
13
5
89
3
34

MULMAT.OUT
2856
(M1(M2M3))M4

Biến đổi xâu

Cho xâu ký tự $U = u_1u_2... u_n$, trong đó $u_i \in X=\{a, b, c\}$, i = 1, 2,..., n ($n \le 50$). Các phép biến đổi sau đây (đánh số thứ tự từ 1 đến 9) cho phép thay thế hai ký tự liên tiếp nhau trong xâu đã cho bởi một ký tự:

1) $aa \to x_1$; 2) $ab \to x_2$; 3) $ac \to x_3$;
4) $ba \to x_4$; 5) $bb \to x_5$; 6) $bc \to x_6$;
7) $ca \to x_7$; 8) $cb \to x_8$; 9) $cc \to x_9$

trong đó $x_i \in X$, i=1,2,...,9.

Cần tìm cách áp dụng lần lượt các phép biến đổi đã cho để chuyển xâu U về một ký tự định trước $w \in X$.

Dữ liệu: Vào từ file văn bản XAU.INP có cấu trúc như sau:

- Dòng đầu tiên chứa số n là độ dài của xâu U,
- Dòng tiếp theo chứa xâu U ;
- Dòng tiếp theo chứa 9 chữ cái $x_1, x_2,..., x_9$ được viết liền nhau;
- Dòng cuối cùng chứa chữ cái w.

Kết quả: Ghi ra file văn bản XAU.OUT theo qui cách sau:

- Dòng đầu tiên ghi số lượng phép biến đổi M cần áp dụng để chuyển xâu U thành chữ cái w, dòng này ghi số 0 nếu không tìm được phép biến đổi như vậy;
- Nếu có thể biến đổi dãy đã cho về một ký tự w thì dòng thứ i trong số M dòng tiếp theo ghi số thứ tự của phép biến đổi cần sử dụng và các chỉ số của hai phần tử liên tiếp nhau trong dãy đang biến đổi cần phải thay thế bằng một ký tự.

Ví dụ: File XAU.INP và file XAU.OUT tương ứng:

XAU.INP
5
bbbba
bbacbaacca
a

XAU.OUT
4
5 2 3
4 3 4
5 1 2
6 1 2

Dãy biến đổi trong ví dụ có thể mô tả trong sơ đồ sau:

b*bb*ba $\to_5$ bb*ba* $\to_4$ b*bc* $\to_5$ *bc* $\to_6$ a

Đường đi trên lưới

Một mạng đường giao thông gồm n×m nút giao thông có dạng lưới ô vuông gồm các đường phố một chiều ngang dọc nối các nút giao thông. Toạ độ của các nút được đánh số từ trên xuống dưới và từ trái sang phải. Khoảng cách giữa hai nút bất kỳ là như nhau. Mỗi nút giao thông (i, j) nằm ở cao độ a_{ij} mét so với mực nước biển (i=1,...,n; j=1,...,n). Cần tìm đường đi ngắn nhất nối đi từ nút (u_s, v_s) đến nút (u_t, v_t) sao cho trong quá trình di chuyển không khi nào phải di chuyển theo phố mà nút đầu có độ cao thấp hơn p (mét) so với độ cao của nút cuối.

Dữ liệu: Vào từ file văn bản PATH.INP có cấu trúc như sau:

- Dòng đầu tiên ghi ba số n, m, p; (0< n, m <51);
- n dòng tiếp theo mỗi dòng gồm m số nguyên ghi cao độ của các nút giao thông: a_{ij}, j=1, 2,..., m; i=1, 2,..., n;
- Các dòng tiếp theo, mỗi dòng gồm 4 số nguyên dương a_i b_i c_i d_i mô tả một đường một chiều nối nút giao thông (a_i,b_i) với nút (c_i,d_i) trên lưới. Mô tả các đường phố kết thúc bởi dòng gồm 4 số 0;
- Dòng cuối cùng ghi 4 số nguyên: u_s v_s u_t v_t là toạ độ của nút xuất phát (u_s, v_s) và nút kết thúc (u_t, v_t).

Kết quả: Ghi ra file văn bản có tên PATH.OUT đường đi ngắn nhất từ nút xuất phát đến nút kết thúc dưới dạng dãy các toạ độ của các nút cần phải đi qua bắt đầu từ nút xuất phát (u_s, v_s) và kết thúc ở nút (u_t, v_t) hoặc thông báo là không có đường đi nối hai nút.

Ví dụ: File PATH.INP và PATH.OUT có thể

PATH.INP
3 4 10 10 15 20 25 19 30 35 30 10 19 26 20 1 1 1 4 2 1 2 4 3 4 3 3 3 3 1 3 1 4 3 4 2 4 2 1 1 1 2 1 0 0 0 0 1 1 2 2

PATH.OUT
(1,1), (1,2), (1,3), (1,4), (2,4), (2,3), (2,2)

Mạng máy tính

Một hệ thống gồm n máy tính đánh số từ 1 đến n được nối thành mạng. Mạng được gọi là *thông suốt* nếu như hai máy bất kỳ trong mạng có thể truyền thông tin cho nhau hoặc là trực tiếp theo kênh nối giữa chúng hoặc thông qua các máy trung gian. Sơ đồ nối mạng được cho bởi hệ thống gồm m kênh nối giữa các cặp máy trong mạng (u_i, v_i), $i = 1, 2,..., m$. Giả sử mạng là thông suốt. Số nguyên dương p được gọi là *độ liên kết nút* của mạng nếu có thể chỉ ra p máy trong mạng mà sự hỏng hóc của chúng dẫn đến mạng các máy còn lại là không thông suốt đồng thời sự hỏng hóc của ít hơn p máy sẽ không làm mất tính thông suốt của mạng các máy còn lại. Số nguyên dương q được gọi là *độ liên kết cạnh* của mạng nếu có thể chỉ ra q kênh trong mạng mà sự hỏng hóc của chúng dẫn đến mạng các máy còn lại là không thông suốt đồng thời sự hỏng hóc của ít hơn q kênh sẽ không làm mất tính thông suốt của mạng các máy còn lại.

Yêu cầu: Tính độ liên kết nút và độ liên kết cạnh của mạng máy tính.

Dữ liệu: Vào từ file văn bản NET.INP có cấu trúc sau:
- Dòng đầu tiên ghi hai số m, n;
- Trong m dòng tiếp theo mỗi dòng ghi hai số nguyên dương u_i, v_i là hai đầu của kênh nối i ($i = 1, 2,..., m$).

Kết quả: Ghi ra file văn bản có tên NET.OUT theo cấu trúc sau:
- Dòng đầu tiên ghi hai số p, q tương ứng là độ liên kết nút và độ liên kết cạnh của mạng đã cho;
- Dòng thứ hai ghi p số nguyên dương là chỉ số của p máy tính mà sự hỏng hóc của nó dẫn đến mạng không thông suốt;
- Mỗi dòng trong số q dòng tiếp theo ghi cặp số u, v biểu diễn một kênh nối trong số q kênh nối của mạng mà sự hỏng hóc của chúng dẫn đến mạng không thông suốt.

Ví dụ: File NET.INP và NET.OUT tương ứng có thể có dạng:

NET.INP
8 12
1 2
1 3
1 4
2 3
2 4
3 4
4 5
4 8
5 6
5 7
6 8
7 8

NET.OUT
1 2
4
5 7
7 8

Truyền tin trên mạng

Trong một mạng gồm n máy tính đánh số từ 1 đến n. Sơ đồ nối mạng được cho bởi hệ thống gồm m kênh nối trực tiếp giữa một số cặp máy trong mạng. Biết chi phí truyền một đơn vị thông tin theo mỗi kênh nối của mạng.
Người ta cần chuyển một bức thông điệp từ máy s đến máy t. Để đảm bảo an toàn, người ta muốn chuyển bức thông điệp này theo hai đường truyền tin khác nhau (tức là không có kênh nào của mạng được sử dụng trong cả hai đường truyền tin). Chi phí của một đường truyền tin được hiểu là tổng chi phí trên các kênh của nó.

Yêu cầu: Giả sử bức thông điệp có độ dài là 1 đơn vị thông tin, hãy tìm cách chuyển thông điệp từ s đến t sao cho tổng chi phí chuyển thông điệp (bằng tổng chi phí theo cả hai đường truyền tin) là nhỏ nhất.

Dữ liệu: Vào từ file văn bản MESS.INP với cấu trúc như sau:

- Dòng đầu tiên ghi bốn số n, m, s, t cách nhau bởi dấu cách ($n \le 100$) ;
- Mỗi dòng thứ i trong số m dòng tiếp theo ghi thông tin về kênh nối thứ i của mạng gồm ba số d_i, c_i, g_i, trong đó d_i, c_i là chỉ số của hai máy tương ứng với kênh này và g_i (g_i nguyên dương) là chi phí để truyền một đơn vị thông tin từ máy d_i đến máy c_i (và ngược lại) theo kênh này (i = 1,2,...,m).

Kết quả: Ghi ra file văn bản MESS.OUT theo cấu trúc sau:

- Dòng đầu tiên ghi chi phí truyền thông điệp theo cách truyền tin tìm được;
- Dòng thứ hai ghi đường truyền tin thứ nhất dưới dạng dãy có thứ tự các máy bắt đầu từ máy s kết thúc ở máy t;
- Dòng thứ ba ghi đường truyền tin thứ hai dưới dạng dãy có thứ tự các máy bắt đầu từ máy s kết thúc ở máy t.

Ví dụ: File MESS.INP và MESS.OUT có thể:

MESS.INP
5 7 1 5 1 2 3 1 4 8 2 3 5 2 4 4 3 5 5 4 3 8 4 5 3

MESS.OUT
24 1 2 3 5 1 4 5

Domino

Bộ bài domino gồm 28 quân đánh số từ 1 đến 28. Mỗi quân bài là một thanh hình chữ nhật được chia làm hai hình vuông bằng nhau, trong đó người ta ghi các số từ 0 (để trống) đến 6 bằng cách trổ các dấu tròn trắng. Dưới đây liệt kê 28 quân bài đôminô:

Số TT	Quân bài	Số TT	Quân bài	Số TT	Quân bài	Số TT	Quân bài
1	0|0	8	1|1	15	2|3	22	3|6
2	0|1	9	1|2	16	2|4	23	4|4
3	0|2	10	1|3	17	2|5	24	4|5
4	0|3	11	1|4	18	2|6	25	4|6
5	0|4	12	1|5	19	3|3	26	5|5
6	0|5	13	1|6	20	3|4	27	5|6
7	0|6	14	2|2	21	3|5	28	6|6

Sắp xếp 28 quân bài đôminô ta có thể tạo ra một hình chữ nhật kích thước 7×8 ô vuông. Mỗi cách xếp như vậy sẽ tạo ra một bản đồ số. Ngược lại, mỗi bản đồ số có thể tương ứng với một số cách xếp.

Ví dụ: Bản đồ số

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

tương ứng với hai cách xếp mô tả bởi bảng số

16	16	24	18	18	20	12	11
6	6	24	10	10	20	12	11
8	15	15	3	3	17	14	14
8	5	5	2	19	17	28	26
23	1	13	2	19	7	28	26
23	1	13	25	25	7	4	4
27	27	22	22	9	9	21	21

và bảng số

16	16	24	18	18	20	12	11
6	6	24	10	10	20	12	11
8	15	15	3	3	17	14	14
8	5	5	2	19	17	28	26
23	1	13	2	19	7	28	26
23	1	13	25	25	7	21	4
27	27	22	22	9	9	21	4

Yêu cầu: Cho trước bản đồ số, hãy liệt kê tất cả các cách xếp có thể tạo được bản đồ số đó.

Dữ liệu: Vào từ file văn bản DOMINO.INP gồm 7 dòng, mỗi dòng chứa các phần tử của dòng tương ứng trong bản đồ số.

Kết quả: ghi ra file văn bản DOMINO.OUT: dòng đầu tiên ghi số lượng cách xếp tìm được p, tiếp theo là p nhóm dòng, mỗi nhóm gồm 7 dòng ghi các dòng của các bảng số tương ứng với một cách xếp tìm được.

Sắp xếp

Liên đoàn cờ quốc tế FIDE quản lý một danh sách gồm N kỳ thủ (N ≤ 32000) đánh số từ 1 đến N. Hàng tháng Liên đoàn đều có bảng hệ số ELLO của các kỳ thủ (hệ số ELLO của một kỳ thủ là một số nguyên dương không vượt quá 5000, đánh giá sức cờ của kỳ thủ). Cuối năm, ông chủ tịch FIDE muốn nhận được bảng tổng hợp theo từng tháng, mỗi tháng gồm dãy các các hệ số ELLO của các kỳ thủ được xếp theo thứ tự không tăng.

Dữ liệu: File văn bản SORTING.INP:

- Dòng đầu tiên ghi số N;
- Dòng thứ i trong số 12 dòng mỗi dòng chứa N số h[i,1], h[i,2], ..., h[i,N] tương ứng là hệ số ELLO của các kỳ thủ 1, 2, ..., N trong tháng i.

Kết quả: Ghi ra file SORTING.OUT: dòng thứ i ghi dãy các hệ số ELLO trong tháng i của các kỳ thủ được xếp theo thứ tự không tăng.

Ví dụ: File dữ liệu và kết quả có thể:

SORTING.INP

```
5
2000 3000 1600 1200 5000
3000 3000 1600 1200 3500
4000 3500 4600 4200 4000
3500 3200 3600 3200 4000
3400 3300 3600 3200 3000
3000 3000 3600 3200 4000
3000 3000 1600 1200 5000
2000 3000 1600 1200 5000
2000 3000 1600 1200 5000
2000 3000 1600 1200 5000
2000 3000 1600 1200 5000
2000 3000 1600 1200 5000
```

SORTING.OUT

```
5000 3000 2000 1600 1200
3500 3000 3000 1600 1200
4600 4200 4000 4000 3500
4000 3600 3500 3200 3200
3600 3400 3300 3200 3000
4000 3600 3200 3000 3000
5000 3000 2000 1600 1200
5000 3000 2000 1600 1200
5000 3000 2000 1600 1200
5000 3000 2000 1600 1200
5000 3000 2000 1600 1200
5000 3000 2000 1600 1200
```

Phân công việc

Có n thợ được đánh số từ 1 đến n và n công việc cũng được đánh số từ 1 đến n. Cần chia khoảng thời gian gồm t ngày ra thành một số giai đoạn làm việc và phân công thợ tham gia thực hiện các công việc sao cho:

1. Trong mỗi giai đoạn: mỗi thợ chỉ tham gia vào việc thực hiện một công việc và mỗi công việc chỉ do một thợ thực thực hiện.
2. Số ngày công mà thợ i tham gia thực hiện công việc j là a_{ij}.
3. Mỗi thợ đều có số ngày công là bằng t.
4. Mỗi công việc đều có số công thợ thực hiện bằng t.

Dữ liệu: Vào từ file văn bản PCONG.INP:

- Dòng đầu tiên chứa hai số nguyên dương n, t ($1< n, t \leq 100$);
- Dòng thứ i trong số n dòng tiếp theo chứa các số $a_{i1}, a_{i2}, ..., a_{in}$.

Kết quả: Ghi ra file văn bản PCONG.OUT:

- Dòng đầu tiên chứa số giai đoạn cần chia m;
- Dòng thứ i trong số m dòng tiếp theo chứa thông tin về giai đoạn thứ i gồm n+1 số nguyên dương: $d_i, v_1(i), v_2(i), ..., v_n(i)$ cho biết giai đoạn i gồm d_i ngày và trong giai đoạn này thợ j phải thực hiện công việc $v_j(i)$.

Ví dụ: File dữ liệu và kết quả có thể:

PCONG.INP
```
3	8
2	3	3
4	2	2
2	3	3
```

PCONG.OUT
```
4
2	1	2	3
3	3	1	2
1	2	1	3
2	2	3	1
```

Bài toán phân công

Một Công ty kinh doanh trên một vùng lãnh thổ gồm N địa điểm dân cư đánh số từ 1 đến N. Biết chi phí đi từ địa điểm i đến địa điểm j là c_{ij} (i, j = 1, 2, ..., N). Công ty cần điều động K nhân viên tiếp thị (đánh số từ 1 đến K) đang hoạt động ở K địa điểm khác nhau trong N địa điểm nói trên đến công tác ở K địa điểm mới, mỗi nhân viên đến công tác tại một trong K địa điểm mới này (2K < N < 150).

Yêu cầu: Tìm cách điều động các nhân viên tiếp thị đến các địa điểm hoạt động mới sao cho tổng chi phí cho việc di chuyển chỗ làm việc của các nhân viên này là nhỏ nhất.

Dữ liệu: Vào từ file văn bản ASIGN.INP có cấu trúc như sau:

- Dòng đầu tiên chứa hai số N và K ghi cách nhau bởi dấu cách;
- Tiếp đến là N dòng, mỗi dòng chứa N số nguyên, dòng thứ i chứa chi phí đi từ địa điểm i đến các địa điểm còn lại: c_{i1}, c_{i2}, . . ., c_{iN} (qui ước $c_{ii} = 0$), i = 1, 2, ..., N;
- Dòng tiếp theo chứa K số nguyên dương, số thứ i là chỉ số của các địa điểm hoạt động của nhân viên thứ i trong số K nhân viên tiếp thị;
- Dòng cuối cùng chứa K số nguyên dương là chỉ số của các địa điểm mới cần điều động K nhân viên tiếp thị nói trên đến hoạt động.

Kết quả: Ghi ra file ASSIGN.OUT, dòng đầu tiên chứa tổng chi phí theo cách điều động tìm được, mỗi dòng thứ i trong số K dòng tiếp theo ghi cách di chuyển của nhân viên thứ i bắt đầu từ địa điểm đang hoạt động đến địa điểm hoạt động mới (có thể phải di chuyển qua một số địa điểm trung gian), i = 1, 2, ..., K.

Ví dụ: Nội dung file ASSIGN.INP và file ASSIGN.OUT có thể như sau:

ASSIGN.INP

```
5  2
 0   20   10   60  100
20    0   30   70   80
10   20    0   20   50
60   70   20    0   20
100  80   50   20    0
1  2
4  5
```

ASSIGN.OUT

```
100
1   3   4   5
2   1   3   4
```

Phân công

Có *m* thợ đánh số từ 1 đến *m*, và *n* công việc được đánh số từ 1 đến *n*. Với mỗi thợ *i* biết tập những công việc mà anh ta có thể thực hiện X_i (*i* = 1, 2, ..., *m*). Cần tìm cách phân công thợ thực hiện công việc sao cho:

- Mỗi thợ thực hiện ít nhất 2 công việc,
- Mỗi việc chỉ do không quá 1 thợ thực hiện,

đồng thời sao cho số công việc được phân công thợ thực hiện là nhiều nhất.

Dữ liệu: Vào từ file văn bản PCONG.INP có cấu trúc như sau:

- Dòng đầu tiên chứa hai số nguyên dương *m, n* ($0< 2\ m \le n \le 100$);
- Dòng thứ *i* trong số *m* dòng tiếp theo chứa chỉ số các công việc mà thợ *i* có thể thực hiện, *i* = 1, 2, ..., *m*.

Kết quả: Ghi ra file văn bản với tên PCONG.OUT theo cấu trúc sau:

- Dòng đầu tiên ghi số lượng công việc được thực hiện theo cách phân công tìm được, hoặc ghi số 0 nếu không tìm được cách phân công thoả mãn điều kiện đặt ra,
- Mỗi dòng thứ *i* trong số *m* dòng tiếp theo chứa chỉ số các công việc phân cho thợ i, *i* = 1, 2, ..., *m*.

Ví dụ: File PCONG.INP và file PCONG.OUT tương ứng có thể có dạng

PCONG.INP
3 8 1 2 3 4 6 1 2 7 3 4 5

PCONG.OUT
7 3 6 1 2 7 4 5

Năng suất dây chuyền

Một dây chuyền sản xuất có N vị trí làm việc đánh số từ 1 đến N. Có N công nhân để xếp vào làm việc trên các vị trí này. Biết s_{ij} là năng suất làm việc của công nhân i trên vị trí làm việc j của dây chuyền (i, j = 1, 2, ..., N). Cho trước một cách bố trí công nhân đứng làm việc trên các vị trí của dây chuyền, ta có thể tính năng suất của dây chuyền theo cách bố trí đã cho như là năng suất nhỏ nhất của công nhân trên dây chuyền.

Yêu cầu: Tìm cách bố trí N công nhân vào làm việc trên N vị trí của một dây chuyền sản suất sao cho năng suất của dây chuyền là lớn nhất.

Dữ liệu: Vào từ file văn bản NANGSUAT.INP:

- Dòng đầu tiên chứa số nguyên dương N (N ≤ 200).
- Dòng thứ i trong số N dòng tiếp theo chứa N số nguyên dương s_{i1} s_{i2} . . . s_{iN} (i = 1, 2, ..., N).

Kết quả: Ghi ra file văn bản NANGSUAT.OUT:

- Dòng đầu tiên ghi năng suất của dây chuyền theo cách bố trí tìm được.
- Dòng thứ i trong số N dòng tiếp theo ghi vị trí làm việc của công nhân i trên dây chuyền theo cách bố trí tìm được.

Ví dụ: File dữ liệu vào - ra có thể:

NANGSUAT.INP
4 9 4 4 12 8 7 8 13 2 2 8 3 6 7 3 7

NANGSUAT.OUT
7 1 4 3 2

Xếp lịch thực hiện chương trình trên máy tính

Cần phải bố trí việc thực hiện N chương trình đánh số từ 1 đến N. Mỗi chương trình đòi hỏi thời gian tính là 1 đơn vị thời gian tính trên một máy tính để hoàn thành nó. Trong số các chương trình có những chương trình phụ thuộc nhau. Ta nói chương trình i phụ thuộc chương trình j, nếu việc thực hiện chương trình i chỉ có thể tiến hành sau khi chương trình j đã được hoàn thành. Tuy nhiên để tiết kiệm thời gian, ta có thể tìm ra những chương trình có thể thực hiện đồng thời.

Yêu cầu: Cho trước số chương trình và sự phụ thuộc của chúng, cần xác định thời gian ít nhất cần thiết để hoàn thành việc thực hiện tất cả các chương trình đã cho với giả thiết là số lượng máy tính để thực hiện các chương trình là không hạn chế. Tiếp theo, cần xác định số lượng máy tính ít nhất cần sử dụng để hoàn thành việc thực hiện tất cả các chương trình với thời gian tìm được.

Dữ liệu vào: File văn bản TASK.INP:

- Dòng đầu tiên chứa hai số nguyên dương N, M, trong đó N là số chương trình, M là số quan hệ phụ thuộc. ($N \leq 200$).
- Dòng thứ i trong số M dòng tiếp theo chứa hai số nguyên dương d_i , c_i cho biết chương trình c_i chỉ được thực hiện sau khi chương trình d_i đã được hoàn thành.

Kết quả: Ghi ra trên một dòng của file văn bản TASK.OUT 2 số T, K, trong đó T là thời gian tính ít nhất tìm được và K là số lượng máy tính ít nhất phải sử dụng.

Ví dụ: File dữ liệu vào - ra có thể:

TASK.INP	TASK.OUT
6 6 1 4 2 5 3 6 4 6 4 5 5 6	4 2

Lập lịch cực tiểu chi phí lớn nhất

Cần tìm trình tự thực hiện n công việc đánh số từ 1 đến n trên một máy. Thời gian cần thiết để thực hiện công việc i là p_i (thời gian để máy chuyển từ việc thực hiện công việc này sang công việc khác là không đáng kể). Trong số các công việc đã cho có một số công việc chỉ được tiến hành sau khi một số công việc nào đó đã hoàn thành. Giả sử thời điểm bắt đầu thực hiện các công việc là 0. Khi đó nếu công việc i được hoàn thành vào thời điểm C_i thì ta phải trả chi phí $f_i C_i + b_i$, trong đó f_i và b_i (được gọi là các hệ số chi phí) là các số nguyên dương cho trước.

Yêu cầu: Tìm lịch thực hiện các công việc sao cho chi phí lớn nhất phải trả cho một công việc $\max \{ f_i C_i + b_i : i = 1, 2, ..., n \}$ là nhỏ nhất.

Dữ liệu: Vào từ file văn bản CMAXMIN.INP:

- Dòng đầu tiên chứa số nguyên dương n ($0 < n \le 100$).
- Dòng thứ 2 chứa n số nguyên dương $p_1, p_2, ..., p_n$.
- Dòng thứ 3 chứa n số nguyên dương $f_1, f_2, ..., f_n$.
- Dòng thứ 4 chứa n số nguyên dương $b_1, b_2, ..., b_n$.
- Dòng thứ i trong số n dòng tiếp theo chứa chỉ số các công việc phải hoàn thành trước khi thực hiện công việc i (qui ước ghi số 0 nếu công việc i có thể thực hiện độc lập).

Kết quả: Ghi ra file văn bản CMAXMIN.OUT

- Dòng đầu tiên ghi chi phí lớn nhất phải trả cho một công việc theo trình tự tìm được.
- Dòng tiếp theo ghi trình tự thực hiện các công việc đã cho.

Ví dụ:

CMAXMIN.INP
5 5 9 7 6 14 2 3 5 2 4 11 12 13 14 17 0 1 3 1 1 2 3 1 2 3

CMAXMIN.OUT
157 1 3 2 5 4

Lập lịch ưu tiên đúng hạn

Có n công việc đánh số từ 1 đến n và một máy để thực hiện chúng. Biết

- p_i là thời gian cần thiết để hoàn thành công việc i;
- d_i là thời hạn hoàn thành công việc i.

  Mỗi công việc cần được thực hiện liên tục từ lúc bắt đầu cho tới khi kết thúc, không cho phép ngắt quãng. Khoảng thời gian thực hiện hai công việc bất kỳ chỉ được có nhiều nhất 1 điểm chung. Giả sử C_i là thời điểm hoàn thành công việc i. Khi đó, nếu $C_i > d_i$ ta nói công việc i bị hoàn thành trễ hạn, còn nếu $C_i \leq d_i$ thì ta nói công việc i được hoàn thành đúng hạn.

  Yêu cầu: Tìm trình tự thực hiện các công việc sao cho số công việc được hoàn thành đúng hạn là lớn nhất.

Dữ liệu: Vào từ file văn bản LICHD.INP:

- Dòng đầu tiên chứa số nguyên dương n ($0 < n \leq 100$).
- Dòng thứ 2 chứa n số nguyên dương $p_1, p_2, ..., p_n$.
- Dòng thứ 3 chứa n số nguyên dương $d_1, d_2, ..., d_n$.

Kết quả: Ghi ra file văn bản LICHD.OUT

- Dòng đầu tiên ghi số lượng công việc được hoàn thành đúng hạn theo trình tự tìm được.
- Dòng tiếp theo ghi trình tự thực hiện các công việc đã cho.

Ví dụ:

LICHD.INP
6 2 4 1 2 3 1 3 5 6 6 7 8

LICHD.OUT
4 1 3 4 6 2 5

Đóng gói sản phẩm

Một nhà máy có một rôbốt ở cuối một băng chuyền đóng gói sản phẩm. Chỉ có đúng 2 thùng chứa hàng được mở đồng thời ở cuối băng chuyền, và rôbốt sẽ xếp các kiện sản phẩm vào một trong hai cái thùng được mở đó. Tất cả các thùng chứa hàng đều giống nhau, và mỗi thùng chỉ có thể chứa một số kiện hàng cho đến một trọng lượng giới hạn cho trước. Rôbốt chỉ có thể thực hiện các thao tác sau:

1. Xếp kiện sản phẩm từ băng chuyền vào thùng 1.
2. Xếp kiện sản phẩm từ băng chuyền vào thùng 2.
3. Đóng thùng 1 và mở thùng rỗng mới tại chỗ thùng vừa đóng.
4. Đóng thùng 2 và mở thùng rỗng mới tại chỗ thùng vừa đóng.

Thao tác xếp kiện sản phẩm vào thùng chỉ có thể thực hiện nếu như kiện sản phẩm xuất hiện ở cuối băng chuyền và trọng lượng của nó cộng với trọng lượng của các kiện sản phẩm đã xếp trong thùng không vượt quá giới hạn trọng lượng cho phép.

Yêu cầu: Cho biết dãy các trọng lượng của các kiện sản phẩm xuất hiện trên dây chuyền, và trọng lượng giới hạn chung cho tất cả các thùng, cần tìm cách xếp các kiện sản phẩm vào các thùng sao cho số thùng cần sử dụng là ít nhất.

Dữ liệu: Vào từ file văn bản PACK.INP

- Dòng đầu tiên chứa trọng lượng giới hạn của mỗi thùng L ($1 \le L \le 100$).
- Dòng thứ hai chứa số kiện sản phẩm sẽ xuất hiện trên băng chuyền N ($1 \le N \le 5000$).
- Dòng thứ i trong số N dòng tiếp theo chứa trọng lượng của kiện sản phẩm thứ i (i đồng thời là thứ tự xuất hiện của kiện sản phẩm ở đầu ra của băng chuyền). Trọng lượng của mỗi kiện là một số nguyên dương không vượt quá L.

Kết quả: Ghi ra file văn bản PACK.OUT

- Dòng đầu tiên ghi số thùng phải sử dụng K.
- Mỗi dòng thứ i trong số K dòng tiếp theo chứa chỉ số của các kiện sản phẩm chứa trong thùng thứ i. Các thùng được đánh số theo thứ tự mà chúng được mở ra.

Ví dụ.

PACK.INP
8 6 4 2 5 3 5 4

PACK.OUT
3 1 6 2 3 4 5

Lập lịch trên hai máy

Cần phải tìm lịch thực hiện N công việc, mỗi công việc phải được thực hiện trên một trong hai máy cùng chức năng, đánh số là 1 và 2. Mỗi công việc đều đòi hỏi 1 đơn vị thời gian để hoàn thành nó. Biết rằng trong số N công việc có những công việc chỉ được thực hiện sau khi một số công việc khác đã hoàn thành.

Yêu cầu: Giả sử thời điểm bắt đầu thực hiện các công việc là 0, hãy tìm lịch thực hiện các công việc trên hai máy sao cho toàn bộ N công việc được hoàn thành ở thời điểm sớm nhất có thể được.

Dữ liệu: Vào từ file văn bản có tên LICH.INP có cấu trúc như sau:

- Dòng đầu tiên ghi số N ($N \le 100$);
- Dòng thứ i trong số N dòng tiếp theo chứa chỉ số các công việc phải hoàn thành trước khi công việc i được thực hiện (qui ước các chỉ số được viết cách nhau ít nhất một dấu cách và sẽ ghi một số 0 nếu công việc i có thể thực hiện không phụ thuộc vào các công việc còn lại).

Kết quả: Ghi ra file văn bản LICH.OUT gồm N dòng: mỗi dòng thứ i ghi hai số k_i , t_i trong đó $k_i \in \{1, 2\}$ là chỉ số máy thực hiện công việc i còn t_i là thời điểm bắt đầu thực hiện công việc i (i = 1, 2, ..., N).

Ví dụ: File LICH.INP và LICH.OUT có thể như sau

LICH.INP
10
2 5 9
0
2 5 9
9
0
4 9
1 2 3 4 5 9
1 2 4 5 9
0
4 9

LICH.OUT
2 2
2 0
1 2
2 3
1 1
1 3
2 5
1 4
1 0
2 4

Biến đổi xâu

Cho hai xâu ký tự S_1 và S_2, mỗi xâu có độ dài không quá 255 ký tự. Cho phép thực hiện các phép biến đổi sau đây đối với xâu ký tự:

1. Thay thế một ký tự nào đó bởi một ký tự khác.
2. Xoá bớt một ký tự.
3. Chèn vào một ký tự.

Yêu cầu: Tìm cách sử dụng một số ít nhất các phép biến đổi trên để biến xâu S_1 thành xâu S_2.

Dữ liệu: Vào từ file văn bản STRING.INP có cấu trúc như sau:

- Dòng đầu tiên chứa xâu S_1.
- Dòng thứ hai chứa xâu S_2.

Kết quả: Ghi ra file văn bản STRING.OUT

- Dòng đầu tiên ghi số lượng phép biến đổi cần sử dụng K.
- Mỗi dòng thứ i trong số K dòng tiếp theo mô tả phép biến đổi được sử dụng ở lần thứ i (i = 1, 2, ..., K): đầu tiên ghi chỉ số của phép biến đổi được sử dụng, tiếp đến:
- Nếu là phép biến đổi 1 cần chỉ ra vị trí của ký tự cần thay thế trong xâu đang biến đổi và ký tự thay thế;
- Nếu là phép biến đổi 2, cần chỉ ra vị trí của ký tự cần xoá trong xâu đang xét.
- Nếu là phép biến đổi 3, cần chỉ ra vị trí của ký tự trong xâu đang xét mà sau nó cần chèn một ký tự và ký tự cần chèn.

Ví dụ:

STRING.INP
aabbbbceffeee
abbbcegfffeeq

STRING.OUT
4
2 2
3 b
3 8 g
1 14 q

Đường đi trong mê cung

Một mê cung gồm M × N ô vuông (M dòng, N cột, M, N ≤ 100). Mỗi ô vuông có thể có từ 0 đến 4 bức tường bao quanh. Cần phải đi từ bên ngoài vào mê cung, bắt đầu từ phía Tây, qua các ô của mê cung và thoát khỏi mê cung về phía Đông. Chỉ được phép di chuyển theo 4 hướng Đông, Tây, Nam, Bắc và đi qua những nơi không có tường chắn.

1. Trong trường hợp có đường đi, hãy tìm đường đi qua ít ô nhất.

2. Trong trường hợp trái lại, hãy tìm cách phá bỏ ít nhất một số bức tường để có đường đi. Nếu có nhiều phương án như vậy, hãy chỉ ra một phương án để đường đi qua ít ô nhất.

Trạng thái của một ô được cho bởi một số nguyên trong khoảng từ 0 đến 15 theo quy tắc: bắt đầu là 0 (không có tường), cộng thêm 1 (nếu có bức tường phía Tây), cộng thêm 2 (nếu có bức tường phía Bắc), cộng thêm 4 (nếu có bức tường phía Đông), cộng thêm 8 (nếu có bức tường phía Nam).

Dữ liệu: Vào từ file văn bản MECUNG.INP bao gồm

- Dòng đầu ghi giá trị M (số dòng) và N (số cột),
- Các dòng tiếp theo ghi ma trận trạng thái của các ô gồm M dòng, N cột, trong đó giá trị ở dòng i cột j mô tả trạng thái của ô [i, j].

Các giá trị trên cùng một dòng ghi cách nhau ít nhất một dấu trắng.

Kết quả: Ghi ra file văn bản MECUNG.OUT

1. Trường hợp tìm thấy đường đi:

- Dòng đầu ghi số ô mà đường đi đi qua,
- Dòng tiếp theo ghi thông tin về đường đi gồm toạ độ dòng xuất phát, sau đó cách một dấu trắng, là xâu ký tự mô tả đường đi (theo hướng) viết liên tiếp nhau, gồm các ký tự D (đông), B (bắc), T (tây), N (nam). Thí dụ *3 DBBDDNTNDD* mô tả đường đi xuất phát từ dòng 3 sau đó lần lượt đi theo các hướng đông, bắc, bắc, đông, đông, nam, tây, nam, đông, đông.

2. Trường hợp không tìm thấy đường đi:

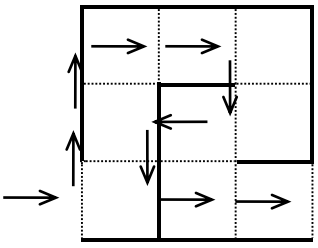
- Dòng đầu ghi số 0,
- Dòng thứ hai ghi số bức tường phải phá,
- Dòng thứ ba ghi số ô mà đường đi đi qua,
- Dòng cuối ghi thông tin về đường đi theo quy cách giống như trường hợp 1.

Ví dụ 1:

MECUNG.INP
3 3 3 10 6 5 3 12 12 9 10

MECUNG.OUT
9 3 DBBDDNTNDD

Hình 1 mô tả một mê cung cho trong ví dụ 1



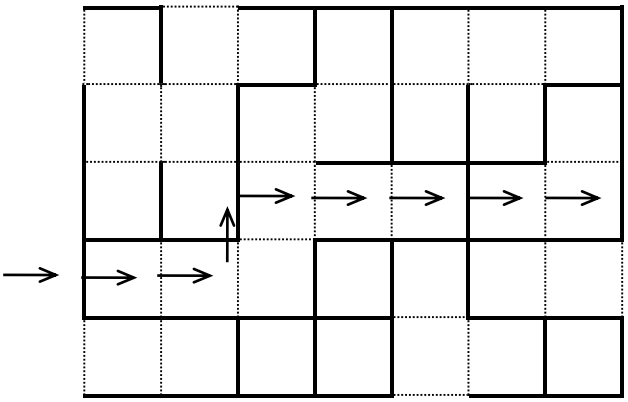
Hình 1

Ví dụ 2:

MECUNG.INP
5 7 6 1 14 7 3 2 14 1 4 11 12 13 13 7 13 13 3 10 14 11 12 11 10 12 15 7 11 10 10 14 15 15 1 14 15

MECUNG.OUT
0 *3* *8* *4 DDDBDDDDD*

Hình 2 mô tả mê cung cho trong ví dụ 2.



Hình 2

Trong ví dụ 2, phải phá ít nhất 3 bức tường để có đường đi. Có nhiều phương án phá 3 bức tường. Phương án nêu trong kết quả có đường đi qua 8 ô (xem hình vẽ) là phương án đi qua ít ô nhất.

Kế hoạch săn tin

Một toà soạn báo Thể thao cần phân công n phóng viên (đánh số từ 1 đến n), mỗi phóng viên đến một trong số n địa điểm khác nhau (cũng được đánh số từ 1 đến n), để săn lùng tin về giải vô địch bóng đá. Mỗi phóng viên được quyền đề xuất một danh sách gồm 3 địa điểm mà họ mong muốn đến công tác xếp theo thứ tự ưa thích nhất, nhì, ba. Một cách phân công được gọi là chấp nhận được nếu như mỗi phóng viên đều được cử đến địa điểm trong danh sách 3 địa điểm mong ước. Nếu có cách phân công chấp nhận được, toà soạn lại quan tâm đến cách phân công chấp nhận được mà trong đó số phóng viên được đến địa điểm mà họ ưa thích nhất là lớn nhất. Cách phân công như vậy gọi là cách phân công phù hợp.

Yêu cầu: Nếu có cách phân công chấp nhận được cần tìm cách phân công phù hợp trong đó số phóng viên được đến địa điểm mà họ ưa thích thứ nhì là lớn nhất. Ngược lại, cần đưa ra cách phân công mà trong đó số phóng viên không được đến địa điểm trong danh sách 3 địa điểm ưa thích của họ là nhỏ nhất.

Dữ liệu: Vào từ file văn bản ASSIGN.INP có cấu trúc như sau:

- Dòng đầu tiên chứa số nguyên dương n ($n \le 200$).
- Dòng thứ i trong số n dòng tiếp theo chứa ba số nguyên dương a_i, b_i, c_i là danh sách 3 địa điểm ưa thích của phóng viên i (i = 1, 2, ..., n).

Kết quả: Ghi ra file văn bản ASSIGN.OUT theo cấu trúc sau:

- Dòng đầu tiên ghi ba số p, q, r, trong đó p là số phóng viên không được đáp ứng nguyện vọng, q là số phóng viên được cử đến địa điểm ưa thích nhất, r là số phóng viên được cử đến địa điểm ưa thích thứ nhì trong cách phân công tìm được.
- Dòng thứ i trong số n dòng tiếp theo ghi địa điểm mà phóng viên i được cử đến theo cách phân công tìm được.

Ví dụ:

ASSIGN.INP
6
1 2 3
1 2 3
1 2 3
2 3 4
4 5 6
3 4 5

ASSIGN.OUT
0 2 1
1
2
3
4
6
5

Dàn đèn mầu

Cho một lưới toạ độ nguyên, hoành độ từ 0 đến N, tung độ từ 0 đến M (M, N ≤ 200). Trên K nút cho trước của lưới, mỗi nút cần đặt một đèn mầu sao cho hai đèn ở hai nút có cùng hoành độ hoặc có cùng tung độ phải có mầu khác nhau. Hãy tìm cách bố trí dàn đèn sao cho số mầu phải dùng là ít nhất. Các mầu đã sử dụng phải được đánh số bởi các số nguyên dương liên tục bắt đầu từ 1.

Dữ liệu vào: file văn bản COLOR.INP:

- Dòng đầu tiên ghi ba số M, N, K;
- Dòng thứ i trong số K dòng tiếp theo ghi hoành độ và tung độ của nút thứ i trong dãy K nút cần đặt đèn (i=1,2,...,K).

Kết quả: ghi ra file văn bản COLOR.OUT:

- Dòng đầu ghi số lượng mầu cần sử dụng P,
- Dòng thứ i trong số K dòng tiếp theo ghi mầu của đèn ở nút thứ i (i=1,2,...,K);

Ví dụ: Các file dữ liệu - kết quả có thể

COLOR.INP		COLOR.OUT
3 5 13		4
1 1		1
1 2		2
1 5		3
3 1		2
4 1		3
3 2		1
2 3		1
3 3		3
4 3		2
2 4		3
4 4		1
2 5		2
4 5		4

Mê cung

Mê cung là một lưới hình chữ nhật có kích thước M × N, được chia thành M × N ô (M dòng, N cột, các dòng và các cột được đánh số bắt đầu từ 1). Các ô thuộc một trong ba loại sau:

1. Ô cấm: không thể đi vào,
2. Ô tự do: được đi vào,
3. Ô cửa: nếu cửa mở thì được đi vào, nếu cửa đóng thì không thể đi vào. Giả thiết có không quá 20 ô cửa.

Trong số các ô tự do, có một ô đặc biệt, gọi là ô năng lượng. Mỗi lần qua ô này, nếu như chưa có năng lượng thì năng lượng sẽ được nạp một lượng vừa đủ mở một cửa (nghĩa là nếu muốn mở tiếp thêm một cửa thì cần phải nạp lại năng lượng). Cửa đã mở thì không đóng lại nữa.

Một bước di chuyển được hiểu là một di chuyển từ một ô đến ô kề cạnh với nó hoặc một di chuyển ra khỏi mê cung.

Một người đang đứng tại một ô tự do (khác ô năng lượng) và chưa có năng lượng. Giả thiết rằng ban đầu mọi ô cửa đều ở trạng thái đóng. Hãy xét xem người đó có thể ra khỏi mê cung được không? Nếu có thể được, hãy tìm cho anh ta một hành trình với số bước di chuyển ít nhất.

Dữ liệu vào: file văn bản BL2.INP:

- Dòng thứ nhất ghi hai số nguyên dương M, N ≤ 50,
- Dòng thứ hai ghi hai số U, V là các chỉ số dòng và chỉ số cột của ô người đang đứng,
- Dòng thứ ba ghi hai số S, T là các chỉ số dòng và chỉ số cột của ô năng lượng.
- Mỗi dòng thứ i trong M dòng tiếp theo ghi N số nguyên thể hiện tình trạng các ô của dòng thứ i của mê cung với ý nghĩa như sau:
    - số -1 có nghĩa ô tương ứng là ô cấm,
    - số 0 có nghĩa ô tương ứng là ô tự do,
    - số 1 có nghĩa ô tương ứng là ô cửa.

Kết quả: ghi ra file văn bản BL2.OUT:

- Nếu không ra khỏi mê cung được, dòng thứ nhất ghi số 0.
- Nếu đi ra khỏi mê cung được, dòng thứ nhất ghi số K là số bước di chuyển để ra khỏi mê cung, K dòng tiếp theo, mỗi dòng ghi một ô theo thứ tự trên hành trình, bắt đầu từ ô (U, V) cho đến ô cuối cùng, với mỗi ô, ghi hai số lần lượt là chỉ số dòng và chỉ số cột của ô đó.

Ví dụ: Các file dữ liệu - kết quả có thể

BL2.INP

```
6       8
2       2
5       5
-1      -1      -1      -1      -1      -1      -1      -1
-1       0       0       0       0       1       0       0
-1       0      -1      -1       1      -1       0      -1
-1       0      -1       0       0      -1       0      -1
-1       0       0       0       0      -1       0      -1
-1      -1      -1      -1      -1      -1       0      -1
```

BL2.OUT

```
17
2       2
3       2
4       2
5       2
5       3
5       4
5       5
4       5
3       5
4       5
5       5
4       5
3       5
2       5
2       6
2       7
2       8
```

Dãy số

Cho dãy gồm n (n≤500) số nguyên dương

$a_1, a_2, ..., a_n$

và số nguyên dương k (k ≤10). Hãy tìm dãy con nhiều phần tử nhất của dãy đã cho có tổng các số hạng chia hết cho k.

Dữ liệu vào: file văn bản bản DAYSO.INP

- Dòng đầu tiên chứa hai số n, k được ghi cách nhau bởi ít nhất một dấu cách;
- Các dòng tiếp theo chứa các số $a_1, a_2, ..., a_n$ được ghi cách nhau bởi ít nhất một dấu cách hoặc dấu xuống dòng.

Kết quả: ghi ra file văn bản DAYSO.OUT

- Dòng đầu tiên ghi k là số phần tử của dãy con tìm được;
- Các dòng tiếp theo ghi dãy k chỉ số các phần tử của dãy đã cho có mặt trong dãy con tìm được.

Ví dụ: Các file dữ liệu-kết quả có thể

DAYSO.INP

```
10  3
2       3       5       7
9       6       12      7
11      15
```

DAYSO.OUT

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

Tạo số lớn nhất

Người ta định trước một quan hệ hai ngôi đối xứng trên tập hợp các chữ số {0, 1,..., 9}, gọi là quan hệ có thể đổi chỗ được, bằng cách chỉ ra từng cặp chữ số thuộc quan hệ này. Có thể biến đổi một số nguyên bất kỳ bằng cách đổi chỗ hai chữ số liền kề nếu hai chữ số này thoả mãn quan hệ đã cho. Phép biến đổi chữ số thứ i và chữ số thứ i+1 như vậy được ký hiệu là [i, i+1] (các chữ số trong một số nguyên được xếp thứ tự 1, 2, ... theo chiều từ trái sang phải).

Hãy tìm dãy biến đổi ngắn nhất trên một số nguyên dương cho trước sao cho thu được một giá trị nguyên lớn nhất. Số chữ số của số nguyên đã cho có thể lên đến 200.

Dữ liệu: Vào từ file văn bản SO.INP có cấu trúc như sau:

- Dòng đầu ghi hai số nguyên dương M, N, trong đó M là số cặp chữ số có thể đổi chỗ được còn N là số chữ số của số nguyên dương cần biến đổi.
- M dòng tiếp theo mỗi dòng ghi cặp chữ số thuộc quan hệ đang xét. Các số viết cách nhau ít nhất một dấu trắng hoặc dấu xuống dòng.
- Dòng thứ i trong số N dòng tiếp theo ghi chữ số thứ i của số cần biến đổi (i = 1, 2, ..., N).

Kết quả: Ghi ra file văn bản SO.OUT theo cấu trúc sau:

- Dòng đầu tiên ghi số lượng phép biến đổi K (K = 0, nếu không cần biến đổi) .
- Dòng thứ i trong số K dòng tiếp theo ghi phép biến đổi thứ i theo qui cách đã nêu ở trên.
- Nếu K>0, thì dòng thứ i trong số N dòng cuối cùng ghi chữ số thứ i của số thu được sau khi biến đổi (i = 1, 2, ..., N);

Ví dụ: File dữ liệu vào ra có thể có dạng:

SO.INP	SO.OUT
7 4 1 7 4 3 6 4 5 6 5 4 9 2 0 8 3 5 4 6	3 [2, 3] [1, 2] [3, 4] 4 3 6 5

SO.INP	SO.OUT
7 6 1 7 4 3 6 4 5 6 5 4 9 2 0 8 3 0 5 4 6 8	2 [4, 5] [3, 4] 3 0 6 5 4 8

SO.INP	SO.OUT
7 11 1 7 4 3 6 4 5 6 5 4 9 2 0 8 2 1 0 7 8 9 5 7 4 8 6	0

Chi phí vận chuyển tối thiểu

Một đại lý vận chuyển đảm bảo việc vận chuyển hàng hoá giữa N thành phố. Giữa một số cặp thành phố có thể có hoặc không có tuyến đường vận chuyển. Công ty phải lập kế hoạch thực hiện M vận đơn của khách hàng. Mỗi vận đơn là một yêu cầu vận chuyển một kiện hàng từ một thành phố nào đó đến một thành phố khác. Chi phí vận chuyển bao gồm hai phần:

1. Chi phí vận chuyển khi đi theo các tuyến đường nối các cặp thành phố, và
2. Chi phí trả cho việc phải đi qua mỗi thành phố trên tuyến đường vận chuyển, ngoại trừ thành phố xuất phát và thành phố kết thúc.

Yêu cầu: Lập trình tìm cách vận chuyển với chi phí vận chuyển nhỏ nhất cho mỗi vận đơn.

Dữ liệu: Vào từ file văn bản TCOST.INP:

- Dòng đầu tiên chứa hai số N, M, trong đó N là số lượng thành phố N ($2 < N < 101$), M là số lượng vận đơn.
- Dòng thứ hai chứa N số nguyên dương $b_1, b_2, ..., b_N$, trong đó b_i là chi phí phải trả khi đi qua thành phố i.
- Dòng thứ i trong số N dòng tiếp theo chứa N số nguyên dương: a_{i1} a_{i2} ..... a_{iN} , trong đó a_{ij} là chi phí đi theo tuyến đường nối thành phố i với thành phố j, qui ước $a_{ij} = -1$ nếu không có tuyến đường nối thành phố i với thành phố j.
- Dòng thứ k trong số M dòng tiếp theo chứa hai số nguyên d_k , c_k là thành phố xuất phát và thành phố kết thúc trong vận đơn thứ k.

Kết quả: Ghi ra file văn bản TCOST.OUT dưới dạng M cặp dòng, cặp dòng thứ i ứng với vận đơn thứ i, trong đó dòng đầu ghi tuyến đường vận chuyển dưới dạng dãy các thành phố trên tuyến đường vận chuyển bắt đầu từ thành phố d_i kết thúc ở thành phố c_i, dòng thứ hai ghi chi phí vận chuyển theo tuyến đường tìm được.

Ví dụ:

TCOST.INP
5 3 5 17 8 3 1 0 3 22 –1 4 3 0 5 –1 -1 22 5 0 9 20 -1 –1 9 0 4 4 –1 20 4 0 1 3 3 5 2 4

TCOST.OUT
1 5 4 3 21 3 4 5 16 2 1 5 4 17

Đếm phần

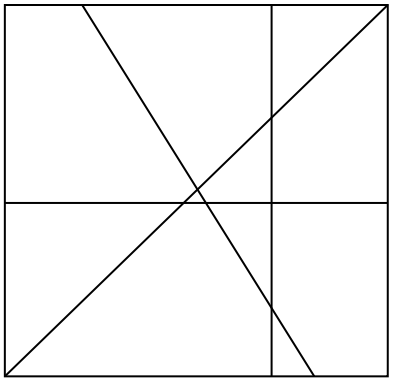
Trên mặt phẳng cho một hình vuông kích thước 1000×1000. Người ta lần lượt vẽ N đường thẳng. Vấn đề đặt ra là cần đếm số phần của hình vuông bị chia bởi N đường thẳng đã vẽ.

Giả thiết là:

1. Sau mỗi lần vẽ độ dài của các cạnh của các phần thu được đều không nhỏ hơn 1.
2. Toạ độ của các đỉnh của hình vuông là (0,0) (0,1000) (1000,1000) (1000,0).
3. Mỗi đường thẳng đều cắt hai cạnh nào đó của hình vuông.

Ví dụ:

Hình vẽ dưới đây cho thấy hình vuông được chia ra làm 10 phần khi vẽ 4 đường thẳng



Dữ liệu: Vào từ file văn bản PART.INP:

- Dòng đầu tiên ghi số đường thẳng được vẽ N (N < 100).
- Dòng thứ i trong số N dòng tiếp theo ghi 4 số nguyên a_i b_i c_i d_i là toạ độ của các giao điểm của đường thẳng thứ i với hai cạnh của hình vuông.

Kết qủa: Ghi ra file văn bản PART.INP: Số phần đếm được.

Ví dụ:

PART.INP
3 0 0 1000 1000 500 0 500 1000 0 500 1000 500

PART.OUT
6

Nói bằng con số

Một nhà sáng chế các trò chơi với các bảng đố điền chữ quyết định xây dựng một tương ứng một-một giữa mỗi từ gồm không quá 20 chữ cái với một số nguyên dương. Phép tương ứng được xây dựng như sau: mỗi từ sẽ được đặt tương ứng với số thứ tự của nó theo cách sắp xếp trước hết theo độ dài sau đó đến thứ tự từ điển. Một đoạn của danh sách tương ứng được chỉ ra dưới đây:

a	1
b	2
. . .	
z	26
aa	27
ab	28
. . .	
snowfall	157118051752
. . .	

Yêu cầu: Cho một từ (hoặc một số) cần đưa ra số (từ) tương ứng với nó.

Dữ liệu: Vào từ file văn bản NUMSPEAK.INP chứa danh sách các từ và số mỗi từ và mỗi số được ghi trên một dòng bắt đầu từ vị trí cột đầu tiên, kết thúc bởi dòng ghi ký tự * ở cột đầu tiên. Mỗi từ gồm không quá 20 ký tự, mỗi ký tự là một chữ cái in thường (từ a đến z). Mỗi số là dãy các chữ số trong phạn vi từ 0 đến 9 được ghi liên tục cho đến dấu kết thúc dòng.

Kết quả: Ghi ra file NUMSPEAK.OUT, mỗi dòng ghi kết quả tương ứng với một dòng dữ liệi của file dữ liệu vào. Nếu kết quả là một từ thì cần ghi liên tục các ký tự của từ bắt đầu từ vị trí cột 1. Nếu kết quả là số thì bắt đầu từ vị trí 1 của dòng tương ứng cần ghi số tìm được cùng với dấu chấm phân cách hàng trăm, nghìn, triệu,...

Ví dụ: Các file dữ liệu – kết quả:

NUMSPEAK.INP
29697684282993 computationally *

NUMSPEAK.OUT
elementary 232.049.592.627.851.629.097

Giai thừa

Theo định nghĩa N! (đọc là N giai thừa) là tích của N số nguyên không âm đầu tiên, trong đó N được giả thiết là số nguyên không âm. Ví dụ

N	N!
0	1
1	1
2	2
3	6
4	24
5	120
10	3628800

Yêu cầu: Tìm chữ số khác không đầu tiên kể từ phải qua trái của N! ($0 < N \leq 10000$).

Dữ liệu vào: File văn bản FACT.INP:

- Dòng đầu tiên ghi số nguyên dương k;
- Dòng thứ i trong số k dòng tiếp theo ghi số nguyên dương N_i (i = 1, 2,..., k).

Kết quả: Ghi ra file FACT.OUT: dòng thứ i ghi chữ số khác không đầu tiên kể từ phải qua trái của N_i! (i = 1, 2, ..., k);

Ví dụ: Các file dữ liệu - kết quả:

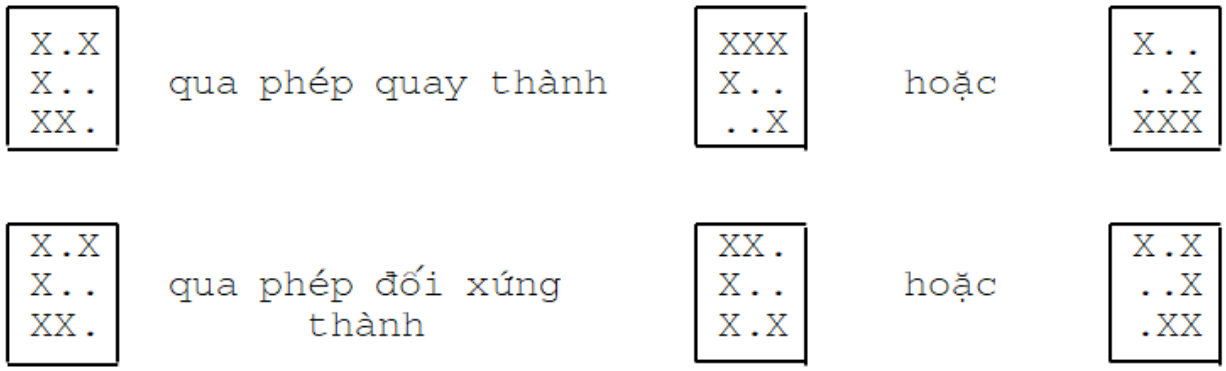
FACT.INP
5 2 26 125 3125 9999

FACT.OUT
2 4 8 2 8

Điệu nhảy của các hình vuông

Hình vuông múa ở đây là một ma trận kích thước n×n, mỗi thành phần chỉ có thể là chữ cái in hoa X hoặc dấu chấm (.). Mỗi hình vuông có thể "nhảy múa" bằng cách quay nó đi một góc 90^o theo chiều thuận hoặc nghịch với chiều kim đồng hồ, hoặc/và lấy đối xứng qua trục tung hoặc hoành gốc ở đỉnh trên phải.

Ví dụ



Yêu cầu: Cho một hình vuông gốc và danh sách gồm m hình vuông đánh số từ 1 đến m, cần xác định xem những hình vuông nào trong danh sách có thể thu được từ hình vuông gốc nhờ quy tắc "múa" đã mô tả ở trên.

Dữ liệu vào: File văn bản DANCE.INP:

- Dòng đầu tiên ghi hai số n, m cách nhau bởi dấu cách (1< n, m ≤ 100);
- Tiếp đến gồm m+1 nhóm dòng, mỗi nhóm gồm n dòng, mỗi dòng gồm n ký tự được ghi bắt đầu từ vị trí cột 1, trong đó nhóm dòng 1 mô tả hình vuông gốc, còn nhóm dòng thứ i (i= 2, 3,..., m+1) mô tả hình vuông thứ i-1 trong danh sách.

Kết quả: Ghi ra file DANCE.OUT, dòng thứ i ghi số 1 nếu hình vuông thứ i có thể thu được từ hình vuông gốc, và ghi số 0 nếu ngược lại.

Ví dụ: Các file dữ liệu - kết quả

DANCE.INP
3 2 X.X X.. XX. XXX ..X X.. XXX ..X X.X

DANCE.OUT
1 0

Sinh số ngẫu nhiên

Khi tiến hành mô phỏng bằng máy tính người ta thường phải sử dụng các số ngẫu nhiên. Một trong các cách sản sinh ra các số giả ngẫu nhiên là sử dụng hàm số có dạng sau

$$f(x+1) = [\, f(x) + h \,] \bmod N \;,\; x = 0, 1, 2, ...$$

trong đó $[\alpha]$ ký hiệu số nguyên lớn nhất còn nhỏ thua hoặc bằng α, **mod** ký hiệu phép toán lấy phần dư (a *mod* b là phần dư trong phép chia số nguyên a cho số nguyên b) .

Hàm số như vậy sẽ sinh cho ta số giả ngẫu nhiên (f(.)) nằm trong khoảng từ 0 đến N-1. Vấn đề nẩy sinh khi sử dụng các hàm số dạng này là chúng sinh ra cùng một đoạn mẫu, lặp lại vô hạn nếu $x \to +\infty$. Để hạn chế tác động xấu này, cần tìm cách lựa chọn các thông số h và N sao cho có thể thu được một phân bố đều của các giá trị nguyên nằm trong khoảng từ 0 đến N-1.

Chẳng hạn, khi h = 3, N = 5, hàm ở trên sẽ sinh ra loạt số giả ngẫu nhiên 0, 3, 1, 4, 2 trong một chu kỳ. Trong ví dụ này, mọi số nguyên từ 0 đến N-1 được sinh ra trong mỗi loạt gồm N bước lặp tính hàm số.

Khi h = 15 và N = 20, hàm ở trên sinh ra loạt số 0, 15, 10, 5 nếu khởi tạo f(0) = 0. Rõ ràng trong ví dụ này, không thể tìm được giá trị nào của f(0) để có thể sinh được tất cả các số trong khoảng từ 0 đến N-1.

Ta nói là hàm số với cách chọn h và N đã cho sinh ra các ***số giả ngẫu nhiên với phân bố đều*** nếu như trong mỗi loạt gồm N bước lặp liên tiếp hàm này luôn sinh ra tất cả các số nguyên trong khoảng từ 0 đến N-1.

Yêu cầu: Với mỗi bộ h, N cho trước cần xác định hàm số ở trên có sinh ra số giả ngẫu nhiên với phân bố đều hay không?

Dữ liệu vào: file văn bản RANDOM.INP gồm 1 dòng chứa hai số nguyên h, N được ghi cách nhau bởi dấu cách (1 Ê h, N ≤ 100000).

Kết quả: Đưa ra màn hình thông báo:

NGON!!!

nếu với các thông số đã cho hàm số cho dãy số giả ngẫu nhiên với phân bố đều, còn nếu ngược lại hãy thông báo

SAI?!

Ví dụ: Với file dữ liệu

RANDOM.INP
63923 99999

Kết quả sẽ là: NGON!!!

RANDOM.INP
15 20

Kết quả sẽ là: SAI?!

Phát hiện chu trình

Một hệ thống gồm N máy tính (đánh số từ 1 đến N) được nối với nhau thành một mạng bởi M kênh nối (đánh số từ 1 đến M) giữa các cặp máy tính. Mỗi kênh chỉ nối hai máy tính nào đó và hai máy tính bất kỳ trong mạng hoặc là không được nối với nhau hoặc được nối với nhau bởi đúng một kênh. Một đường truyền tin trên mạng từ máy u đến máy v là dãy

$$u = w_1, w_2, ..., w_{k-1}, w_k = v,$$

trong đó giữa hai máy w_i và w_{i+1} có kênh nối (i = 1, 2, ..., k-1) . Đường truyền tin nói trên được gọi là đường truyền tin đơn nếu như các máy $w_2, w_3, ..., w_{k-1}$ là khác nhau. Đường truyền tin đơn xuất phát từ máy u lại quay về chính nó được gọi là chu trình.

Yêu cầu: Với mỗi kênh của mạng cần phải xác định xem nó có nằm trên chu trình nào hay không, và nếu câu trả lời là khẳng định cần xác định số lượng chu trình khác nhau đi qua nó.

Dữ liệu: Vào từ file văn bản CYCLE.INP có cấu trúc như sau

- Dòng đầu tiên chứa hai số N, M (0 < N < 21);
- Mỗi dòng thứ i trong số M dòng tiếp theo chứa hai số nguyên dương u_i, v_i là hai máy được nối bởi kênh nối i (i = 1, 2, ..., M).

(Các số trên cùng một dòng được ghi cách nhau bởi ít nhất một dấu cách).

Kết quả: Ghi ra file văn bản CYCLE.OUT:

- Dòng đầu tiên ghi số lượng kênh tham gia vào ít nhất một chu trình trên mạng. (K = 0, nếu mạng không chứa chu trình).
- Dòng thứ i trong số K dòng tiếp theo ghi hai số u_i, s_i cho biết kênh u_i tham gia vào s_i chu trình khác nhau trong mạng (i = 1, 2, ..., K).

Ví dụ 1

CYCLE.INP
6 7
1 2
1 3
2 3
2 4
3 6
4 5
5 6

CYCLE.OUT
7
2
2
2
2
2
2
2

Ví dụ 2

CYCLE.INP
3 3
1 2
2 3
3 4

CYCLE.OUT
0

Bài toán gặp gỡ

Trên một lưới ô vuông kích thước N×N người ta đặt hai rôbốt A và B. Rôbốt A được đặt tại góc trên bên trái, còn rôbốt B được đặt tại ô ở góc dưới bên phải của lưới. Mỗi ô của lưới hoặc là chứa số 0 hoặc là chứa số 1. Tại mỗi bước, mỗi rôbốt chỉ có thể di chuyển lên trên, xuống dưới, sang trái, sang phải (không được di chuyển theo đường chéo) ở những ô chứa số 1. Hai rôbốt sẽ gặp nhau nếu chúng đứng trong cùng một ô. Hai rôbốt bắt đầu di chuyển đồng thời, và mỗi lượt cả hai rôbốt đều phải thực hiện việc di chuyển (nghĩa là không cho phép một rôbốt dừng tại một ô nào đó trong khi rôbốt kia thực hiện bước di chuyển).

Yêu cầu: Tìm số bước di chuyển ít nhất mà hai rôbốt phải thực hiện để có thể gặp nhau.

Chú ý: Phụ thuộc vào lưới đã cho, hai rôbốt có thể không khi nào gặp được nhau.

Dữ liệu: Vào từ file văn bản MEET.INP có cấu trúc như sau:
- Dòng đầu tiên chứa số N ($N \le 20$).
- Dòng thứ i trong số N dòng tiếp theo chứa N số (mỗi số là 0 hoặc 1) được ghi cách nhau bởi dấu cách là các số ghi trong các ô của hàng thứ i của lưới (i=1,2,...,N).

Kết quả: Ghi ra file văn bản MEET.OUT như sau:
- Dòng đầu tiên ghi số K là số bước di chuyển tìm được; (qui ước K=-1 nếu hai rôbốt không thể gặp nhau);
- Trong trường hợp tìm được cách di chuyển để hai rôbốt có thể gặp nhau, hai dòng tiếp theo được dùng để mô tả cách di chuyển của hai rôbốt, dòng đầu tiên ghi cách di chuyển của rôbốt A còn dòng thứ hai ghi cách di chuyển của rôbốt B. Cách di chuyển của rôbốt được biểu diễn bởi một dãy các ký tự mỗi ký tự chỉ có thể là một trong 4 chữ cái T(di chuyển lên trên), B(di chuyển xuống dưới), L(di chuyển sang trái), R (di chuyển sang phải) thể hiện đường đi của rôbốt

Ví dụ:

MEET.INP
5 1 0 0 0 0 1 1 1 0 0 1 1 1 0 1 0 0 0 1 1 0 0 0 0 1
MEET.OUT
-1

MEET.INP
6 1 1 1 1 1 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1 1 1
MEET.OUT
5 DDDDD LLLLL

Chia dừa

Một tàu có N tên cướp biển và một con khỉ ghé vào một hòn đảo hoang vùng Caribê. Trên đảo có rất nhiều dừa. Đám cướp hái được một đống dừa và cho con khỉ đứng canh giữ, còn tất cả đi ngủ. Nữa đêm, một tên cướp thức dậy, đến đống dừa, ném cho con khỉ một quả, còn lại thì chia đều thành N phần và giấu riêng cho mình một phần. Lát sau, tên thứ 2 thức dậy, cũng đến bên đống dừa, ném cho con khỉ một quả, còn lại thì chia đều thành N phần và giấu riêng cho mình một phần. Lần lượt các tên thứ 3, thứ 4, ..., cho đến tên cuối cùng cũng làm như vậy. Đến sáng, cả nhóm họp lại, ném cho con khỉ một quả, còn lại chia đều thành N phần và mỗi tên lấy một phần. Tất cả các lần chia, số dừa đều chia hết cho N.

Hãy xác định tối thiểu mỗi tên cướp được bao nhiêu dừa và tổng số dừa bọn cướp hái được là bao nhiêu?

Dữ liệu: Vào từ file văn bản COCO.INP, chứa số nguyên N ($N \leq 12$).

Kết quả: Ghi ra file văn bản COCO.OUT gồm N+1 dòng: dòng thứ i trong N dòng đầu chứa số dừa của tên cướp thứ i, dòng cuối cùng chứa tổng số dừa bọn cướp hái được.

Ví dụ:

COCO.INP
5

COCO.OUT
4147
3522
3022
2622
2302
15621

Webs

Trong World Wide Web, mỗi trang có thể chứa những móc nối dạng siêu văn bản cho phép bạn có thể di chuyển sang các trang khác. Một trong những câu hỏi của người xây dựng các trang này là: Từ một trang nào đó (gọi là trang gốc) bạn cần xác định xem có thể thâm nhập đến những trang nào bằng cách lần theo các móc nối có thể có trong các trang.

Dữ liệu: Vào từ File văn bản WEB.INP
- Dòng đầu tiên ghi số nguyên dương N là số lượng trang WEB. Các trang WEB sẽ được đánh số từ 1 đến N. ($N \le 100$).
- Dòng tiếp theo ghi số nguyên dương S là chỉ số của trang gốc.
- Dòng thứ i trong số N dòng tiếp theo chứa các chỉ số của các trang mà sử dụng móc nối có trong trang i ta có thể thâm nhập đến được, các chỉ số là các số nguyên trong phạm vi từ 1 đến N được ghi cách nhau bởi dấu cách. (Qui ước: Để dòng rỗng nếu từ trang đang xét không có móc nối).

Kết quả: Ghi ra file WEB.OUT
- Dòng đầu tiên ghi K là số lượng các trang có thể thâm nhập từ trang gốc.
- Dòng tiếp theo chứa K chỉ số của các trang có thể thâm nhập từ trang gốc.

Ví dụ:

WEB.INP
4 1 2 4 1 1 2 4 2

WEB.OUT
3 1 2 4

Gương thần

Hoàng hậu dì ghẻ độc ác hỏi gương thần của mình:

"Gương kia ngự ở trên tường
Thế gian ai đẹp được dường như ta?"

Gương thần trả lời:

" Xưa kia bà đẹp nhất trần
Ngày nay Bạch Tuyết muôn phần đẹp hơn.
Nhà nàng ở khuất núi non
Tại nhà của bảy chú lùn xa xa. "

Hoàng hậu nổi cơn thịnh nộ, cho rằng gương thần nói dối, vì chính mụ đã ra lệnh giết Bạch Tuyết và bà ta đã đập tan gương. Ít lâu sau, Hoàng hậu biết Bạch Tuyết còn sống và hối tiếc đã đập vỡ gương. Bà ra lệnh cho quan Thái sư mua gương khác, thông minh như gương thần. Nhưng tìm đâu một gương như vậy nữa bây giờ? Sau một thời gian suy nghĩ, Thái sư đã tìm ra cách tự cứu được tính mạng của mình: Nối gương với một máy tính chứa chương trình xác định người đẹp nhất thế giới. Thái sư biết cách tính và có mọi dữ liệu cần thiết trong tay. Chỉ đáng tiếc là ông không biết lập trình. Bạn hãy viết giúp quan Thái sư hiền lành lập chương trình tìm người đẹp nhất theo qui tắc sau: Mỗi người thứ i trong danh sách N người đẹp ($0 \le i \le N$) có năm sinh s_i với chỉ số đẹp ban đầu $d_i \ge 0$ lúc mới lọt lòng, sau mỗi năm chỉ số sắc đẹp tăng lên 1 chừng nào tuổi còn nhỏ hơn a_i, từ đó cho đến khi tuổi còn bé hơn b_i chỉ số này không thay đổi, khi tuổi bằng hoặc lớn hơn b_i thì chỉ số này giảm 1 theo từng năm. Khi chỉ số này bằng 0 người đó mất. Mỗi người có một tên riêng, không ai giống ai. Chỉ số càng cao thì càng đẹp.

Dựa vào năm tra cứu và tên của Hoàng hậu, đưa ra một trong 3 câu trả lời sau, phụ thuộc vào hoàn cảnh cụ thể:

- "Tâu Hoàng hậu, không có ai sánh được với Lệnh bà!"
- "Xưa kia bà đẹp nhất trần,
  Ngày nay . . . muôn phần đẹp hơn!" (ở vào vị trí ' . . .' là tên người đẹp nhất),
- "Tất cả mọi người đều đã mất, không còn ai cả!"

Dữ liệu: Vào từ file văn bản GUONG.INP:

- Dòng đầu tiên: số nguyên cho biết năm tra cứu,
- Dòng thứ 2: tên Hoàng hậu;
- Dòng thứ 3: số nguyên N - số người trong danh sách người đẹp ($N \le 1000$),
- Tiếp theo là nhóm N cặp 2 dòng:
  - Dòng đầu trong nhóm thứ i chứa tên người thứ i,
  - Dòng thứ 2 trong nhóm chứa các số nguyên s_i, d_i, a_i, b_i, các số cách nhau ít nhất một dấu cách.

Tên Hoàng hậu và mọi người: xâu không quá 12 ký tự.

Kết quả: Ghi ra file văn bản GUONG.OUT câu trả lời có một trong ba dạng đã nêu (cho phép dùng tiếng Việt không dấu).

Ví dụ:

GUONG.INP
1655 Lida 4 Anna 1640 5 15 20 Lida 1630 6 25 30 Bach Tuyet 1639 16 30 40 Terri 1941 10 20 25

GUONG.OUT
" Xua kia ba dep nhat tran, Ngay nay Bach Tuyet muon phan dep hon!"

Lệnh COPY

Bạn Thuỷ cần sao chép **một số** file từ thư mục gốc của đĩa mềm cắm ở ổ đĩa A vào thư mục hiện tại trên ổ đĩa C. Tên file bao gồm hai phần: phần tên và phần mở rộng. Phần tên là một dãy gồm không quá 8 ký tự, mỗi ký tự chỉ có thể là một chữ cái trong bảng chữ cái tiếng Anh hoặc môt trong các chữ số từ 0 đến 9. Phần mở rộng là một dãy gồm không quá 3 ký tự, mỗi ký tự chỉ có thể là một chữ cái trong bảng chữ cái tiếng Anh hoặc một trong các chữ số từ 0 đến 9. Phần tên được ghi trước, tiếp đến là dấu chấm, cuối cùng là phần mở rộng. Phần mở rộng không nhất thiết phải có mặt. Trong trường hợp tên file không có phần mở rộng, dấu chấm phân cách phần tên và phần mở rộng có thể không có mặt trong tên file. Như đã biết lệnh COPY cho phép sử dụng các ký tự thay thế ? hoặc * để mô tả tên của một hoặc nhiều file cần sao chép.

Yêu cầu: Cho trước danh sách các tên file trên thư mục gốc của đĩa cắm ở ổ đĩa A và danh sách các file cần sao chép, hãy lập trình xác định xem có thể chỉ dùng một lệnh COPY để sao chép chỉ các file trong danh sách các file cần sao chép hay không?

Dữ liệu: Vào từ file COPY.INP có cấu trúc như sau:

- Dòng đầu tiên chứa số N (N < 1000) là số lượng file trên thư mục gốc của đĩa mềm cắm ở ổ đĩa A.
- N dòng tiếp theo mỗi dòng bắt đầu từ dấu + hoặc dấu – tiếp đến là tên file; trong đó dấu cộng cho biết file với tên ghi sau nó cần sao chép, còn dấu – cho biết file với tên ghi sau nó không được sao chép.

Kết quả: Ghi ra file văn bản với tên COPY.OUT :

- Trong trường hợp câu trả lời là khẳng định cần ghi ra lệnh COPY cần thực hiện;
- Ngược lại ghi dòng thông báo: KHONG CO

Ví dụ:

BL1.INP
9 +BTAP.EXE +BINPACK.PAS –TINE.COM +BICH.TXT +BACK.DOC +BIENBAN.DOC –HUNG.PAS –HUONG.PAS +BYE

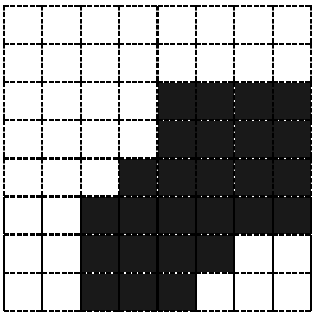
BL1.OUT
COPY A:\B*.*

Cây tứ phân

Cây tứ phân thường được dùng để biểu diễn dữ liệu ảnh trong nhiều hệ xử lý ảnh. Xét ảnh đen trắng có kích thước N*N điểm sáng ($N = 2^k$). Nếu ảnh bao gồm cả điểm đen lẫn điểm trắng thì nó được chia thành 4 phần tư, nếu một phần tư nào đó bao gồm các điểm sáng khác màu, thì nó lại được chia thành 4 phần tư con, . . . Cứ như thế cho đến khi nào mỗi phần tư chỉ bao gồm điểm sáng một màu. Giả thiết điểm trắng được mã hoá bằng 0, còn điểm đen - bằng 1.

Cây tứ phân được xây dựng như sau: điểm gốc tương ứng với toàn ảnh. Nếu ảnh được chia thành 4 phần tư, thì từ gốc có 4 nhánh đi ra, 4 nút ở cuối mỗi nhánh, tính từ trái sang phải tương ứng với phần tư trên trái, phần tư trên phải, phần tư dưới trái và phần tư dưới phải. Nếu phần tư nào đó bị chia thành 4 phần tư con, thì từ nút tương ứng lại có 4 nhánh đi ra, xác định 4 nút ứng với 4 phần tư con . . . Kết quả là ta có một cây, mà từ mỗi nút hoặc không có nhánh nào đi ra hoặc có 4 nhánh. Nút không có nhánh nào đi ra gọi là nút lá và nó tương ứng với một phần tư vuông con một màu. Nếu phần tư con này có màu đen thì ta gọi nút lá đó là nút đen.

Các nhánh rẽ ra từ một nút được đánh số từ trái sang phải bằng các số nguyên 1, 2, 3, 4 (gọi là chỉ số nhánh). Như vậy 1 là chỉ số của nhánh phần tư trên trái, 2 là chỉ số của nhánh phần tư trên phải, 3 là chỉ số của nhánh phần tư dưới trái và 4 là chỉ số của nhánh phần tư dưới phải.



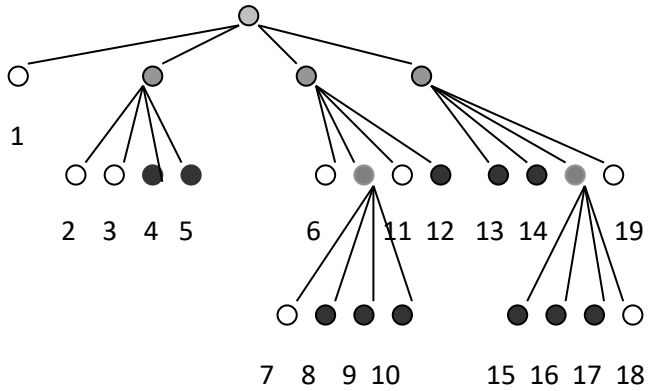
0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 1 1 1 1

0 0 0 0 1 1 1 1

0 0 0 1 1 1 1 1



Mỗi đường đi từ lá tới gốc được xác định bởi một dãy các chỉ số các nhánh phần tư bắt đầu từ nhánh phần tư ứng với lá và kết thúc bởi nhánh phần tư rẽ ra từ gốc. Nếu viết liền các chỉ số này ta thu được một số nguyên dương ở hệ cơ số 5 để biểu diễn đường đi. Ví dụ, đường đi từ nút 4 đến gốc ở hình trên có đường đi là 32_5 hoặc là 17 ở hệ 10. Như vậy mỗi một lá sẽ được đặt tương ứng với một số biểu diễn đường đi từ nó tới gốc. Một ảnh hoàn toàn được xác định khi biết các lá màu đen của cây tứ phân tương ứng, tức là biết dãy số nguyên tương ứng với các lá đen. Trong ví dụ trên, ảnh được xác định bởi dãy số nguyên (hệ 10):

9 14 17 22 23 44 63 69 88 94 113.

Yêu cầu: Hãy lập trình xác định ảnh từ dãy số nguyên cho trước, hoặc ngược lại hãy xác định dãy số nguyên tương ứng với các lá đen từ ảnh cho trước.

Dữ liệu: Vào từ file văn bản QTREE.INP, dòng đầu là số nguyên, có giá trị tuyệt đối bằng N. Nếu số này là dương, thì sau đó là N dòng (N < 64), mỗi dòng có N số nguyên 0, 1 xác định màu các điểm ảnh, các số cách nhau một dấu cách. Nếu số nguyên này là âm, thì ở các dòng tiếp theo là các số nguyên dương (ở hệ 10) ứng với các lá đen. Các số cách nhau ít nhất một dấu cách hoặc nhóm dấu xuống dòng, dấu hiệu kết thúc dãy số là số nguyên -1.

Kết quả: đưa ra file QTREE.OUT:

- Nếu dữ liệu vào là ảnh 0,1 thì đưa ra:
  - Số lượng lá đen của cây tứ phân tương ứng,
  - Ở các dòng tiếp theo: các số nguyên hệ cơ số 10 ứng với các nút lá đen, đưa ra theo thứ tự tăng dần và các số cách nhau một dấu cách hoặc nhóm dấu xuống dòng.
- Nếu dữ liệu vào là dãy số nguyên ứng với các lá đen, thì đưa ra N dòng, mỗi dòng gồm N số 0 hoặc 1, xác định ảnh đen trắng, các số cách nhau một dấu cách.

Ví dụ 1:

QTREE.INP
8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0

QTREE.OUT
11 9 14 17 22 23 44 63 69 88 94 113

Ví dụ 2:

QTREE.INP
-8 9 14 17 22 23 44 63 69 88 94 113 -1

QTREE.OUT
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0

Tập các đoạn có điểm chung

Trên trục số cho tập gồm N đoạn mở ở mút phải ($N \le 5000$). Các đoạn được đánh số từ 1 đến N. Đoạn thứ i được xác định bởi đầu mút trái L_i và đầu mút phải R_i , L_i, R_i - nguyên, $L_i < R_i$, i = 1, 2, ..., N. Một số đoạn có thể có điểm chung với nhau. Một tập con của tập các đoạn thẳng đã cho được gọi là *tập con có điểm chung* nếu tìm được ít nhất một điểm thuộc vào tất cả các đoạn trong tập đã cho.

Yêu cầu: Hãy tìm tập con có điểm chung gồm nhiều đoạn nhất.

Chú ý: Ta gọi ***đoạn mở ở mút phải*** với đầu mút trái a và đầu mút phải b, ký hiệu là [a, b), là tập các điểm x thoả mãn: $a \le x < b$.

Dữ liệu: Vào từ file văn bản PSET.INP:

- Dòng đầu tiên chứa số lượng đoạn N;
- Dòng thứ i trong số N dòng tiếp theo chứa hai số nguyên L_i, R_i là đầu mút trái và đầu mút phải của đoạn thứ i, i = 1, 2, ..., N. ($-10^{-6} < L_i, R_i < 10^6$)

Kết quả: Ghi ra file PSET.OUT

- Số phần tử của tập con tìm được K.
- Dòng thứ i trong số K dòng tiếp theo chứa chỉ số của đoạn thứ i trong tập con tìm được.

Ví dụ 1:

PSET .INP
4 6 8 3 7 1 5 2 4

PSET.OUT
3 2 3 4

Ví dụ 2:

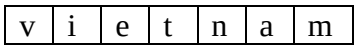
PSET .INP
2 2 4 4 7

PSET.OUT
1 1

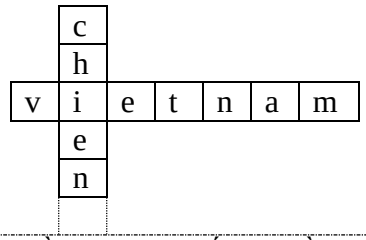
Ô đố chữ

Một phương pháp tạo các ô đố chữ nhờ máy tính được tiến hành như sau: Cho trước một danh sách các từ cùng các ***chữ khớp***, ta sẽ nạp lần lượt các từ trong danh sách vào ô đố chữ.

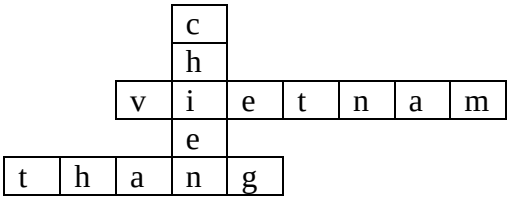
Ví dụ, cho danh sách các từ: **vietnam, chIen, thaNg, tiGercub**. Ta sẽ tạo ô đố chữ như sau: Mỗi từ sẽ được nối vào từ đứng trước nó tại chữ khớp (là chữ cái được in hoa), vì thế ô đố chữ sẽ được tạo theo từng bước như sau:



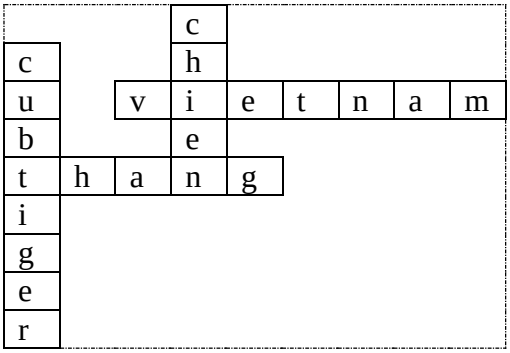
Lần nạp 1: ô đố chữ gồm 1 dòng, 7 cột



Lần nạp 2: ô đố chữ gồm 5 dòng, 7 cột



Lần nạp 3: ô đố chữ gồm 5 dòng, 9 cột



Lần nạp 4: ô đố chữ gồm 9 dòng, 9 cột

Chú ý rằng theo cách nạp vừa mô tả các từ luân phiên được nạp vào bảng theo dòng rồi theo cột, chữ khớp được chỉ ra bởi chữ cái viết hoa trong từ.

Yêu cầu: Lập trình tính kích thước của ô đố chữ tạo được theo phương pháp mô tả ở trên từ danh sách các từ cho trước.

Dữ liệu: Vào từ file văn bản CROSSW.INP:

- Dòng đầu tiên ghi số lượng từ trong danh sách;
- Dòng thứ i trong N dòng tiếp theo ghi từ thứ i trong danh sách (qui ước mỗi từ là dãy gồm không quá 10 chữ cái, trong đó có duy nhất một chữ khớp là chữ cái in hoa).

Kết quả: Ghi ra một dòng của file văn bản CROSSW.OUT hai số P, Q theo thứ tự là số dòng, số cột của ô đố chữ thu được.

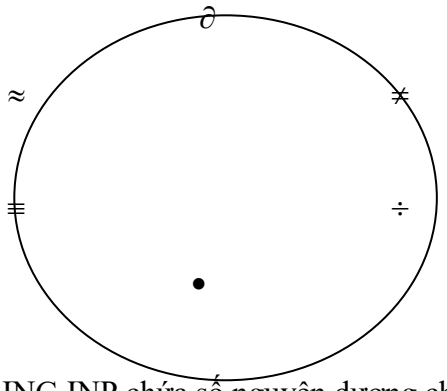
Ví dụ:

CROSSW.INP
4
vietnam chIen thaNg cubTiger

CROSSW.OUT
9 9

Vòng số nguyên tố

Một vòng tròn chứa n (số chẵn) vòng tròn nhỏ (xem hình vẽ). Các vòng tròn nhỏ được đánh số từ 1 đến n theo chiều kim đồng hồ. Cần điền các số tự nhiên từ 1 đến n vào các vòng tròn nhỏ, mỗi số vào một vòng tròn nhỏ sao cho tổng của hai số trên hai vòng tròn nhỏ liên tiếp là số nguyên tố. Số điền ở vòng tròn nhỏ 1 luôn là số 1.



Dữ liệu: Vào từ file văn bản RING.INP chứa số nguyên dương chẵn n (2 < n < 20).

Kết quả: Ghi ra file văn bản RING.OUT :

- Dòng đầu tiên ghi số lượng cách điền số tìm được k;
- Dòng thứ i trong số k dòng tiếp theo ghi các số trong các vòng tròn nhỏ bắt đầu từ vòng tròn nhỏ 1 đọc theo thứ tự của các vòng tròn nhỏ.

Ví dụ:

RING.INP	RING.OUT
6	2 1 4 3 2 5 6 1 6 5 2 3 4

RING.INP	RING.OUT
8	4 1 2 3 8 5 6 7 4 1 2 5 8 3 4 7 6 1 4 7 6 5 8 3 2 1 6 7 4 3 8 5 2

Kế hoạch thuê nhân công

Giám đốc điều hành của một Công ty tin học cần xác định số lượng nhân công cần sử dụng trong mỗi tháng để thực hiện một dự án phát triển tin học. Ông giám đốc nắm được số lượng nhân công tối thiểu cần cho mỗi tháng. Mỗi lần thuê hoặc sa thải một nhân công luôn mất thêm một khoản chi phí. Mỗi khi một thợ nào đó được thuê, anh ta luôn nhận được tiền lương ngay cả khi anh ta không phải làm việc. Giám đốc nắm được chi phí để thuê một nhân công mới, chi phí sa thải một nhân công, lương một tháng của một nhân công. Vấn đề đặt ra cho giám đốc là phải xác định số lượng nhân công cần thuê hoặc sa thải trong mỗi tháng để cho chi phí thực hiện dự án là tối thiểu.

Dữ liệu: Vào từ file văn bản PROJECT.INP:

- Dòng đầu tiên ghi thời gian thực hiện dự án n (đơn vị tính: tháng, $n \le 12$).
- Dòng thứ hai chứa ba số nguyên dương theo thứ tự là chi phí thuê một nhân công mới, lương tháng của một nhân công, chi phí sa thải một nhân công.
- Dòng cuối cùng ghi n số nguyên dương $d_1, d_2, ..., d_n$, trong đó d_i là số lượng nhân công tối thiểu cần cho tháng i.

Kết quả: Ghi ra file PROJECT.OUT

- Dòng đầu tiên ghi chi phí tối thiểu tìm được;
- Mỗi dòng thứ i trong số n dòng tiếp theo ghi số s_i, trong đó nếu $s_i > 0$ thì nó là số lượng nhân công cần thuê thêm ở tháng i, còn nếu $s_i < 0$ thì $|s_i|$ là số lượng nhân công cần sa thải ở tháng i của dự án, $s_i = 0$ nếu không có biến động nhân sự trong tháng i.

Ví dụ:

PROJECT.INP
3 4 5 6 10 9 11

PROJECT.OUT
199 10 0 1

Kim cương hay Rubi

Hai người thợ đào đá quí P và Q săn được n viên đá quí G_1, G_2, ..., G_n thuộc hai loại: kim cương và rubi. Với mỗi viên đá quí G_i biết:

- v_i là trị giá tính bằng USD;
- t_i loại đá: $t_i = 1$ nếu nó là kim cương, $t_i = 0$ nếu nó là rubi.

P nhận trách nhiệm chia phần. Anh ta muốn chia số đá quí ra làm hai phần P, Q sao cho:

- Số lượng viên Rubi trong phần P ***không ít hơn*** số lượng Rubi trong Q;
- Số lượng viên kim cương trong phần P ***không nhiều hơn*** số lượng kim cương trong Q;
- Chênh lệch

$$\delta = \left| \sum_{G_i \in P} v_i - \sum_{G_i \in Q} v_i \right|$$

là nhỏ nhất.

Dữ liệu: Vào từ file văn bản GEMS.INP

- Dòng đầu tiên chứa số lượng đá quí n ($n \le 100$);
- Dòng thứ i trong số n dòng tiếp theo chứa hai số v_i, t_i.

(Tổng giá trị của n viên đá quí không vượt quá 20000 USD)

Kết quả: Ghi ra file GEMS.OUT

- Dòng đầu tiên chứa δ là chênh lệch theo cách chia phần tìm được;
- Dòng thứ i trong số N dòng tiếp theo ghi ký tự P nếu viên đá G_i nằm trong phần P, ghi ký tự Q nếu G_i nằm trong phần Q.

Ví dụ:

GEMS.INP
5 13 1 25 1 20 1 15 0 12 0

GEMS.OUT
5 P Q Q P P

Trò chơi với các con số

Cho tập N = {1, 2, ..., n }, n ≥ 2. Xét trò chơi sau đây: Hai đấu thủ A và B luân phiên thực hiện nước đi. Tại mỗi một nước đi, đấu thủ đến lượt thực hiện nước đi sẽ phải chọn một tập con thực sự của tập N sao cho nó không được chứa bất cứ tập nào trong số các tập con đã bị chọn trước đó. Đấu thủ nào đến lượt mình mà không thể thực hiện được nước đi sẽ bị thua trong trò chơi.

Ví dụ: Khi A = { 1, 2, 3, 4 }, các nước đi của một trận đấu có thể như sau:

	Lượt đi thứ 1		Lượt đi thứ 2		Lượt đi thứ 3		Lượt đi thứ 4	
Đấu thủ	A	B	A	B	A	B	A	B
Nước đi	{1, 2}	{3, 4}	{1, 3}	{2, 4}	{1}	{2}	{3}	{4}

Đến đây đấu thủ A bị thua vì anh ta không còn khả năng thực hiện nước đi.

Yêu cầu: Giả sử trong trò chơi nói trên đấu thủ A là người thực hiện nước đi trước, và hai đối thủ đã thực hiện được m lượt đi đánh số thứ tự từ 1 đến m. Cần lập trình xác định:

- Cuộc chơi còn có thể kéo dài thêm nhiều nhất đến lượt đi thứ bao nhiêu (ký hiệu số thứ tự của lượt đi này là K)?
- Cuộc chơi có thể kết thúc sớm nhất ở lượt đi thứ bao nhiêu (ký hiệu số thứ tự của lượt đi này là L)?
- Nế u như tiếp tục cuộc chơi đến cùng thì đấu thủ nào có cách chơi đảm bảo chắc chắn thắng không phụ thuộc cách chơi của đối thủ của mình?

Dữ liệu: Vào từ file văn bản NUMGAME.INP:

- Dòng đầu tiên chứa hai số n, m (2 ≤ n ≤ 5);
- Tiếp theo là m cặp dòng mô tả m nước đi đã thực hiện của hai đấu thủ, trong đó dòng đầu tiên chứa nước đi của đối thủ A, dòng thứ hai chứa nước đi của đấu thủ B trong lượt đi thứ i. Nước đi của một đấu thủ được mô tả bởi danh sách các số trong tập con được chọn.

Kết quả: Ghi ra file văn bản NUMGAME.OUT

- Dòng đầu tiên ghi hai số K, L;
- Dòng thứ hai ghi tên đấu thủ (A hoặc B) chắc chắn thắng nếu cuộc chơi được tiếp tục đến cùng.

Ví dụ:

NGAME.INP	NGAME.OUT
4 3 1 2 3 4 1 3 2 4 1 2	4 4 B

NGAME.INP	NGAME.OUT
4 4 1 2 3 4 1 3 2 4 1 2 3 4	4 4 B

Các đoạn số nguyên

Giả sử a, b là hai số nguyên, a < b. Ta gọi đoạn số nguyên [a, b] là tập tất cả các số nguyên liên tiếp bắt đầu từ a và kết thúc tại b, nghĩa là [a, b] = {a, a+1, ..., b}. Cho n đoạn số nguyên $[a_i, b_i]$, i = 1, 2, ..., n. Cần tìm tập số nguyên N với lực lượng nhỏ nhất sao cho mỗi đoạn số nguyên đã cho đều chứa ít ra là 2 phần tử của N.

Dữ liệu: vào từ file văn bản INTERVAL.INP:

- Dòng đầu tiên chứa số n ($n \le 10000$);
- Dòng thứ i trong số n dòng tiếp theo chứa hai số nguyên a_i, b_i ($a_i < b_i$) mô tả đoạn số nguyên thứ i, i = 1, 2, ..., n.

Kết quả: Ghi ra file INTERVAL.OUT:

- Dòng đầu tiên chứa số nguyên k là lực lượng của tập N tìm được.
- k dòng tiếp theo mỗi dòng chứa một số nguyên trong tập N.

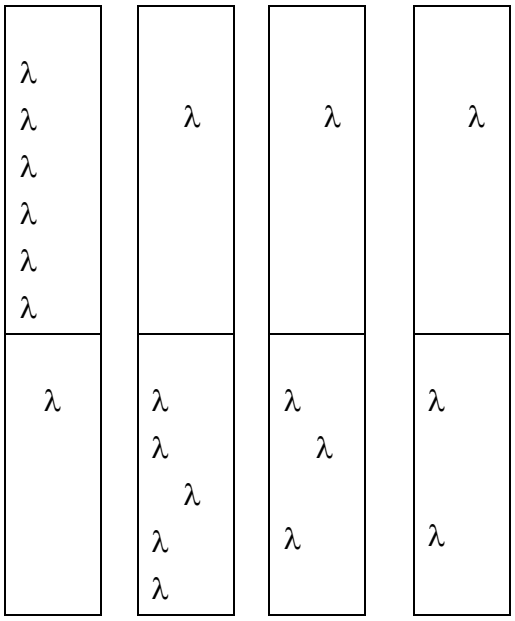
Ví dụ:

INTERVAL.INP
4 3 6 2 4 0 2 4 7

INTERVAL.OUT
4 4 5 1 2

Dãy các quân bài Domino

Domino là một thanh nhựa hình chữ nhật với bề mặt được chia làm hai ô vuông trong mỗi ô vuông hoặc là để trống hoặc là chạm từ 1 đến 6 nốt tròn. Dưới đây cho ta một dãy các thanh domino được xếp cạnh nhau thành một bảng:



Tổng các nốt ở nửa trên của bảng domino trên là 6 + 1 + 1 + 1 = 9, còn tổng các nốt ở nửa dưới là 1 + 5 + 3 + 2 = 11. Chênh lệch giữa nửa trên và nửa dưới là 2. Ta gọi chênh lệch giữa hai nửa trên và dưới là trị tuyệt đối của hiệu hai tổng.

Mỗi quân bài domino được phép đặt đảo ngược trên dưới. Cần tìm cách đảo ngược một số ít nhất quân bài trong một dãy quân bài domino được xếp thành bảng để thu được bảng mới với chênh lệch giữa nửa trên và nửa dưới là nhỏ nhất.

Dữ liệu: vào từ file DOM.INP:

- Dòng đầu tiên chứa số n là số quân bài domino trong bảng ($1 \le n \le 1000$);
- Dòng thứ i trong số n dòng tiếp theo chứa hai số nguyên a_i, b_i theo thứ tự là số nốt tròn ở nửa trên, nửa dưới của quân bài domino thứ i trong bảng. ($0 \le a_i, b_i \le 6$).

Kết quả: Ghi ra file DOM.OUT:

- Dòng đầu tiên ghi số lượng quân bài cần đảo ngược k;
- k dòng tiếp theo mỗi dòng chứa số thứ tự trong dãy của quân bài cần đảo ngược.

Ví dụ:

DOM.INP
4 6 1 1 5 1 3 1 2

DOM.OUT
1 4

Đặt máy theo dõi

Trong đoạn đường từ thành phố A đến thành phố B có n nút giao thông đánh số từ 1 đến n. Cần bố trí máy theo dõi hoạt động giao thông tại các nút này. Mỗi máy theo dõi đặt ở vị trí nào đó có thể theo dõi được hoạt động giao thông trên những điểm ở cách nó không quá r (km). Biết d_i (km) là khoảng cách từ A đến nút giao thông i (i = 1, 2, ..., n). Cần tìm cách đặt một số ít nhất các máy theo dõi để có thể theo dõi được hoạt động giao thông tại tất cả n nút giao thông trên đoạn đường từ A đến B (nghĩa là mỗi nút phải nằm trong tầm theo dõi của ít nhất một trong số các máy theo dõi được bố trí).

Dữ liệu: Vào từ file văn bản WATCH.INP:

- Dòng đầu tiên ghi số nguyên dương n ($n \leq 100$) và số thực r;
- Dòng thứ i trong số n dòng tiếp theo ghi số thực d_i (i = 1, 2, ..., n).

Kết quả: Ghi ra file văn bản WATCH.OUT:

- Dòng đầu tiên ghi k là số lượng máy theo dõi cần đặt ;
- Dòng thứ i trong số k dòng tiếp theo ghi v_i là khoảng cách từ A đến vị trí đặt máy i (các máy được đánh số sao cho: $v_1 < v_2 < ... < v_k$).

Ví dụ:

WATCH.INP		WATCH.OUT
7 0.7 0.5 1.2 1.5 2.5 3.4 2.6 4.7		3 1.2 2.7 4.0

Lập lịch trên m máy

Có n công việc và m máy có tính năng hoàn toàn giống nhau để thực hiện chúng. Các công việc được đánh số từ 1 đến n. Các máy được đánh số từ 1 đến m. Với mỗi công việc j biết:

- Thời gian hoàn thành p_j (là thời gian cần thiết để hoàn thành công việc j).
- Thời điểm sẵn sàng r_j (công việc j không được tiến hành sớm hơn thời điểm r_j);
- Thời hạn hoàn thành d_j (tức là công việc j phải hoàn thành không muộn hơn thời điểm d_j);

Giả thiết là $d_j - r_j \geq p_j$, j = 1, 2, ..., n. Quá trình thực hiện công việc cho phép được ngắt quãng, nghĩa là trong quá trình thực hiện, một máy có thể ngắt việc thực hiện công việc đang tiến hành trên nó để chuyển sang thực hiện một công việc khác. Thời gian để máy chuyển sang thực hiện công việc khác giả thiết bằng 0.

Yêu cầu: Hãy lập lịch để tất cả các công việc được hoàn thành đúng hạn.

Dữ liệu: Vào từ file văn bản JOBS.INP:

- Dòng đầu tiên ghi 2 số n, m ($1 \leq n, m \leq 100$);
- Dòng thứ i trong số n dòng tiếp theo ghi ba số nguyên dương p_i, r_i, d_i (i =1, 2,..., n).

Kết quả: Ghi ra file văn bản JOBS.OUT

- Dòng đầu tiên ghi YES (NO) nếu như có (không có) lịch thoả mãn yêu cầu đặt ra.
- Nếu có lịch thì n nhóm dòng tiếp theo sẽ mô tả lịch tìm được, trong đó nhóm dòng thứ i mô tả lịch thực hiện công việc thứ i:
    1. Dòng đầu tiên ghi k_i ($k_i \geq 0$) là số lần việc thực hiện công việc i bị ngắt quãng;
    2. Dòng thứ j trong số k_i +1 tiếp theo mô tả khoảng thời gian thứ j trong quá trình thực hiện công việc i bao gồm 3 số nguyên dương theo thứ tự là thời điểm bắt đầu, thời điểm dừng và chỉ số của máy thực hiện công việc i.

Ví dụ:

JOBS.INP
4 2 4 2 7 3 3 8 4 3 7 5 1 10

JOBS.OUT
YES 0 2 6 1 1 3 4 2 6 8 1 0 4 7 2 1 1 3 2 7 10 2

Tổ chức thi lại trong ba ngày

Phòng đào tạo của một trường đại học cần lập lịch thi lại n môn học đánh số từ 1 đến n cho m sinh viên được đánh số từ 1 đến m. Với mỗi sinh viên i biết danh sách các môn học trong số n môn học nói trên mà anh ta phải thi lại. Mỗi môn thi sẽ được tổ chức vào một trong ba ngày của tuần lễ thi lại. Để đảm bảo cho mỗi sinh viên có thời gian nghỉ giữa hai môn mà anh ta phải thi lại, lịch thi của các môn phải được bố trí sao cho không một sinh viên nào phải thi lại hai môn trong cùng một ngày. Cần xác định xem có thể bố trí lịch thi đáp ứng yêu cầu đặt ra hay không.

Dữ liệu: Vào từ file văn bản 3DAY.INP:
- Dòng đầu tiên chứa hai số n, m ($1 \le n \le 50$, $1 \le m \le 500$);
- Dòng thứ i trong số m dòng tiếp theo chứa chỉ số các môn học mà sinh viên i phải thi lại (i = 1, 2, ..., m).

Kết quả: Ghi ra file văn bản 3DAY.OUT:
- Dòng đầu tiên ghi YES hoặc NO tuỳ thuộc vào việc có thể bố trí được lịch thi thoả mãn yêu cầu đặt ra hay không.
- Nếu tìm được lịch thoả mãn yêu cầu đặt ra, ba dòng tiếp theo mô tả lịch tìm được:
  1. Dòng đầu tiên trong ba dòng này ghi các chỉ số của các môn thi trong ngày thứ nhất;
  2. Dòng tiếp ghi các chỉ số của các môn thi trong ngày thứ hai;
  3. Dòng cuối cùng ghi các chỉ số của các môn thi trong ngày thứ ba.

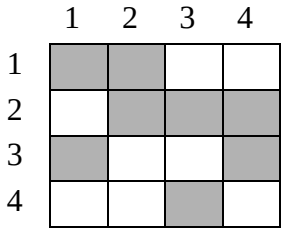
Ví dụ:

3DAY.INP
13 18 1 2 2 3 3 4 4 5 5 6 6 1 1 12 2 11 3 10 4 9 5 8 6 7 7 12 13 12 11 13 11 10 13 10 9 13 9 8 13 8 7 13

3DAY.OUT
YES 2 10 4 12 8 1 5 11 9 3 13 4 6

Hội thi bắn súng

Trong một hội thi bắn súng, bia có dạng một lưới ô vuông kích thước n × n. Các ô vuông trong lưới được sơn bởi hai màu đen và trắng. Trong mỗi cột có đúng 2 ô trắng và n-2 ô đen. Các dòng được đánh số từ 1 đến n từ trên xuống dưới. Các cột được đánh số từ 1 đến n từ trái qua phải (xem hình vẽ). Xạ thủ được thực hiện bài bắn gồm n phát. Bài bắn được gọi là chấp nhận được nếu mỗi phát đều bắn trúng vào ô trắng và không có hai ô trắng bị bắn trúng nào ở trên cùng một dòng hay một cột.



Yêu cầu: Hãy xác định bài bắn chấp nhận được cho xạ thủ.

Dữ liệu: Vào từ file văn bản SHOT.INP:

- Dòng đầu tiên chứa số nguyên n ($2 \le n \le 1000$);
- Dòng thứ i trong số n dòng tiếp chứa 2 số nguyên dương được ghi cách nhau bởi dấu cách là các toạ độ dòng của hai ô trắng trong cột thứ i ($1 \le i \le c$).

Kết quả: Ghi ra file SHOT.OUT:

- Dòng đầu tiên ghi YES nếu tìm được bài bắn chấp nhận được, còn nếu trái lại ghi NO.
- Nếu dòng đầu tiên chứa YES thì dòng thứ i trong số c dòng tiếp theo ghi toạ độ dòng của ô cần bắn trong cột thứ i ($1 \le i \le n$).

Ví dụ:

SHOT.INP	SHOT.OUT
4 2 4 3 4 1 3 1 4	YES 2 3 1 4

SHOT.INP	SHOT.OUT
5 1 5 2 4 3 4 2 4 2 3	NO

Dãy con có tổng lớn nhất

Cho dãy gồm n số nguyên

$a_1, a_2, ..., a_n$

Tìm dãy con gồm một hoặc một số phần tử liên tiếp của dãy đã cho với tổng các phần tử trong dãy là lớn nhất.

Dữ liệu: Vào từ file văn bản SUBSEQ.INP:

- Dòng đầu tiên chứa số nguyên dương n ($n<10^6$);
- Dòng thứ i trong số n dòng tiếp theo chứa số a_i ($|a_i| \le 1000$);

Kết quả: Ghi ra file văn bản SUBSEQ.OUT:

- Dòng đầu tiên ghi vị trí của phần tử đầu tiên của dãy con tìm được;
- Dòng thứ hai ghi vị trí của phần tử cuối cùng của dãy con tìm được;
- Dòng thứ ba ghi tổng các phần tử của dãy con tìm được.

Ví dụ:

SUBSEQ.INP	SUBSEQ.OUT
8 12 -14 1 23 -6 22 -34 13	3 6 40

Các số ở hệ đếm cơ số 16 (Hexadecimal Numbers)

Cơ số của hệ đếm hexa là 16. Để ghi các số ở hệ đếm này người ta sử dụng 16 ký tự 0,1, ...,9, A, B, C, D, E, F. Các chữ cái A..F được sử dụng để thay thế cho các số 10..15 tương ứng. Ví dụ số CF2 trong hệ đếm hexa chính là số $12*16^2 + 15*16 + 2 = 3314$ trong hệ cơ đếm thập phân. Giả sử X là tập các số nguyên dương có biểu diễn trong hệ đếm hexa gồm không quá 8 ký tự và không có ký tự nào bị lặp lại. Ký tự trái nhất trong biểu diễn như vậy không được là 0. Phần tử lớn nhất trong X là FEDCBA98, lớn nhì là FEDCBA97, lớn thứ ba là FEDCBA96, ...

Yêu cầu: Cho số nguyên dương n (không vượt quá số lượng số trong tập X), cần tìm số lớn thứ n trong tập X.

Dữ liệu: Vào từ file HEXA.INP: chứa số nguyên dương n.

Kết quả: Ghi ra file HEXA.OUT số lớn thứ n trong tập X dưới dạng hệ đếm hexa.

Ví dụ:

HEXA.INP	HEXA.OUT
11	FEDCBA87

Kết quả cuộc thi

Trong đợt tổng kết phiếu dự thi đoán kết quả của giải vô địch Bóng đá, Ban tổ chức cuộc thi cần chọn ra k ($k \le 2000$) người có điểm số cao nhất để trao giải thưởng. Danh sách tên và điểm của n ($n \le 10^6$) người dự thi được ghi trên một file văn bản, giả thiết rằng không có hai người nào có cùng số điểm.

Yêu cầu: Đưa ra danh sách k người đạt điểm cao nhất của cuộc thi.

Dữ liệu: Vào từ file văn bản LOTO.INP

- Dòng đầu tiên ghi 2 số nguyên dương n, k;
- Tiếp đến là n nhóm dòng, mỗi nhóm gồm 2 dòng, trong đó dòng đầu ghi họ tên (là dãy gồm không quá 10 ký tự) của người dự thi, dòng thứ hai ghi điểm số mà anh ta đạt được.

Kết quả: Ghi ra file văn bản LOTO.OUT: k nhóm dòng tương ứng với k người có điểm cao nhất của cuộc thi theo thứ tự điểm số giảm dần:

- Dòng đầu tiên trong nhóm ghi họ tên;
- Dòng thứ hai ghi điểm số.

Ví dụ:

LOTO.INP
6 3 A 9 B 7 C 12 D 8 E 5 F 11

LOTO.OUT
C 12 F 11 A 9

Phân bổ tài nguyên

Một Công ty dịch vụ phát triển phần mềm dự định thuê thêm các nhân viên thảo chương và tăng tiền đầu tư cho việc mua sắm các thiết bị cũng như các phần mềm hệ thống mới để tăng năng suất làm việc của các nhóm lập trình trong công ty. Để có thể lượng hoá hiệu quả đầu tư, công ty đánh giá năng suất của mỗi nhóm lập trình bởi số lượng câu lệnh mà nhóm có thể thực hiện được trong một đơn vị thời gian. Khi đó, mỗi phương án đầu tư người và tiền cho mỗi nhóm sẽ được đánh giá bởi số lượng câu lệnh mà nhóm có thể làm tăng thêm khi sử dụng nguồn đầu tư này. Công ty cần phải xác định phương án đầu tư người và tiền cho các nhóm lập trình của mình sao cho tổng năng suất tăng thêm là lớn nhất.

Mỗi nhóm lập trình về phần mình cần xác định các phương án bổ sung người, cũng như các phương án sử dụng kinh phí để tăng năng suất lập trình. Ví dụ, nếu một nhóm nào đó có các phương án sử dụng thêm 0, 3, 5, 6 thảo chương viên, thì điều đó có nghĩa là khi xét bổ sung nhân viên cho nhóm này công ty chỉ có thể tăng thêm số nhân viên cho nhóm theo một trong 4 phương án đó, và như vậy việc bổ sung 1, 2, 4, 7, 8,... nhân viên sẽ không được chấp nhận. Nếu như đối với nhóm này có 3 phương án đầu tư kinh phí 10000, 20000, 40000 (USD), thì tất cả có 12 phương thức đầu tư người-của khác nhau cho nhóm. Hiệu quả của mỗi phương thức đầu tư này sẽ được đánh giá bởi một con số - là số lượng câu lệnh mà nhóm có thể làm tăng thêm khi sử dụng nguồn đầu tư theo phương thức.

Yêu cầu: Viết chương trình tìm kế hoạch đầu tư cho các nhóm lập trình sao cho tổng năng suất tăng thêm của các nhóm là lớn nhất mà không sử dụng quá số lượng nhân viên có thể thuê thêm cũng như không sử dụng quá tổng số vốn mà Công ty có thể đầu tư.

Dữ liệu: Vào từ file văn bản RESALO.INP có cấu trúc như sau. Dòng đầu tiên chứa 3 số nguyên dương d, p, b, trong đó

d - số nhóm lập trình trong Công ty ($0 < d \le 20$);

p - tổng số nhân viên thảo chương mà Công ty có thể thuê thêm;

b - tổng số tiền mà Công ty có thể đầu tư.

Sau dòng này là d nhóm dòng, nhóm dòng thứ i mô tả các phương thức đầu tư cho nhóm lập trình thứ i của Công ty. Mỗi nhóm dòng như vậy được tổ chức như sau:

n - số phương án bổ sung người ($0 < n \le 10$)

$x_1\ x_2\ \ldots\ x_n$ - số lượng người cần bổ sung theo mỗi phương án

k - số phương án sử dụng kinh phí

$b_1\ b_2\ \ldots\ b_k$ - lượng tiền cần cấp theo các phương án sử dụng kinh phí

tiếp đến là bảng gồm n.k số nguyên không âm cho biết hiệu quả của các phương thức đầu tư:

$c_{11}\ c_{12}\ \ldots\ c_{1k}$

$c_{21}\ c_{22}\ \ldots\ c_{2k}$

. . . . . .

$c_{n1}\ c_{n2}\ \ldots\ c_{nk}$

trong đó c_{uv} - hiệu quả của việc sử dụng phương thức đầu tư theo phương án đầu tư người thứ u và phương án sử dụng kinh phí thứ v, (u = 1,2,...,n; v=1,2,...,k).

Chú ý là trong số các phương án đầu tư người hoặc của cho mỗi nhóm luôn có phương án 0 (tức là không bổ sung thêm người hoặc không bổ sung thêm kinh phí).

Kết quả: Ghi ra file có tên RESALO.OUT theo cấu trúc sau

- Dòng đầu tiên ghi tổng số kinh phí cần sử dụng,
- Dòng thứ hai ghi tổng số nhân viên cần thuê thêm,
- Dòng thứ ba ghi tổng năng suất tăng thêm,

tiếp đến là d nhóm dòng mô tả phương thức sử dụng kinh phí của các nhóm lập trình, nhóm dòng thứ i mô tả phương thức của nhóm lập trình thứ i, mỗi nhóm như vậy gồm 3 dòng:

- Dòng đầu tiên ghi lượng kinh phí mà nhóm sử dụng
- Dòng thứ hai ghi số nhân viên bổ sung cho nhóm
- Dòng thứ ba ghi năng suất tăng thêm của nhóm.

Ví dụ:

RESALO.INP	RESALO.OUT
3 10 90000 4 0 2 5 6 4 0 20000 50000 70000 0 10000 20000 50000 60000 20000 10000 40000 20000 10000 30000 40000 30000 10000 40000 30000 5 0 1 3 4 8 3 0 40000 80000 0 50000 30000 50000 40000 60000 20000 30000 50000 80000 90000 50000 30000 40000 70000 3 0 4 6 5 0 20000 30000 40000 50000 0 30000 50000 60000 30000 10000 20000 30000 40000 50000 20000 30000 40000 50000 60000	80000 6 210000 0 2 60000 40000 4 90000 40000 0 60000

Mạng liên thông đơn

Một hệ thống gồm n máy tính (đánh số từ 1 đến n) được kết nối thành mạng bởi các kênh truyền tin một chiều giữa một số cặp máy tính. Ta hiểu một đường truyền tin từ máy u đến máy v trên mạng là dãy

$u = w_1, w_2, ..., w_k = v,$

trong đó có kênh truyền tin từ máy w_i đến w_{i+1} (i = 1, 2, ..., k-1). Đường truyền tin được gọi là đơn nếu như không có máy nào bị lặp lại.

Mạng được gọi là liên thông đơn nếu như có đường truyền tin từ máy u đến máy v thì có không quá một đường truyền tin đơn từ máy u đến máy v.

Yêu cầu: Xác định mạng đã cho có phải là liên thông đơn hay không.

Dữ liệu: Vào từ file văn bản SINGLE.INP:

- Dòng đầu tiên chứa số n ($n \le 200$);
- Mỗi dòng thứ i trong số n dòng tiếp theo chứa chỉ số của các máy mà từ máy i có kênh truyền tin đến chúng, hoặc chứa số 0 nếu từ máy i không có kênh truyền tin đến bất kể máy nào (i = 1, 2,..., n).

Kết quả: Ghi ra file SINGLE.OUT:

- Dòng đầu tiên ghi YES hoặc NO tương ứng với mạng là liên thông đơn hoặc không là liên thông đơn;
- Nếu dòng đầu tiên ghi
    1. YES: thì dòng thứ i trong n-1 dòng tiếp theo ghi độ dài đường truyền tin đơn từ máy 1 đến máy i (ghi số 0 nếu không có đường truyền tin từ máy 1 đến máy i)
    2. NO: Ghi chỉ số của hai máy u, v mà từ u đến v có ít ra là hai đường truyền tin đơn.

Ví dụ:

SINGLE.INP	SINGLE.OUT
6 3 1 2 4 5 6 4 3	YES 2 1 2 3 3

SINGLE.INP	SINGLE.OUT
6 2 6 3 4 5 6 2	NO 1 2

Buôn tiền

Một người làm việc ở một ngân hàng ngoại tệ theo dõi tỷ giá hối đoái phát hiện là nếu khôn khéo thì từ một lượng ngoại tệ ban đầu nhờ chuyển đổi sang các ngoại tệ khác anh ta có thể thu được lợi nhuận đáng kể. Ví dụ: Nếu anh ta có 1 USD và tỷ giá hối đoái giữa các ngoại tệ như sau:

1 USD = 0.7 bảng Anh
1 bảng Anh = 9.5 Franc Pháp
1 franc Pháp = 0.16 USD

Khi đó với 1 USD anh ta có thể mua được 0.7 * 9.5 * 0.16 = 1.064 USD nhờ việc chuyển đổi tiền qua bảng Anh, rồi từ bảng Anh sang franc Pháp, và cuối cùng lại quay về USD. Nhờ đó mỗi USD đã đem lại cho anh ta lợi nhuận là 0.064 USD. Giả sử trong nhà băng quản lý tất cả n loại ngoại tệ đánh số là 1, 2, ..., n. Biết bảng tỷ giá hối đoái R[i,j], i, j = 1,2,...,n (một đơn vị ngoại hối i mua được R[i, j] đơn vị ngoại hối j). Cần xác định xem có cách đổi tiền đem lại lợi nhuận hay không?

Dữ liệu: Vào từ file văn bản MONEY.INP:

- Dòng đầu tiên chứa số n (n ≤ 100);
- Dòng thứ i trong số n dòng tiếp theo chứa các số thực (có không quá 2 chữ số sau dấu phảy) R[i,1], R[i,2],. . . R[i,n], i = 1, 2, ..., n.

Kết quả: Ghi ra file văn bản MONEY.OUT:

- Dòng đầu tiên ghi YES hoặc NO tương ứng với việc có hoặc không có cách đổi tiền sinh lợi nhuận;
- Nếu dòng đầu tiên là YES thì dòng thứ hai ghi hai số u, s, trong đó u là loại tiền xuất phát, còn s là lợi nhuận thu được nhờ cách đổi tiền, còn dòng thứ ba ghi trình tự cần tiến hành đổi tiền để thu lại được lợi nhuận bắt đầu từ loại tiền xuất phát.

Ví dụ:

MONEY.INP	MONEY.OUT
3 1 0.7 9.1 1.2 1 9.5 0.15 0.1 1	YES 1 0.064 1 2 3

Xâu nhị phân

Ta gọi xâu nhị phân độ dài n là dãy $x_1\ x_2\ ...\ x_n$, trong đó $x_i \in \{0, 1\}$, i = 1, 2, ..., n. Giả sử **a** = $a_1\ a_2\ ...\ a_n$ và **b** = $b_1\ b_2\ ...\ b_n$ là hai xâu nhị phân độ dài n. Xâu **a** được gọi là lớn hơn xâu **b** và viết là **a** $\geq$ **b** nếu $a_i \geq b_i$, i = 1, 2, ..., n. Cho

$X = \{ x^1, x^2, ..., x^k \}$ và $Y = \{ y^1, y^2, ..., y^p \}$

là hai tập xâu nhị phân độ dài n.

Yêu cầu: Hãy tìm xâu nhị phân z độ dài n sao cho số xâu nhị phân lớn hơn z trong tập X ***ít ra cũng là bằng*** số xâu nhị phân lớn hơn z trong tập Y. Kết quả tìm được đưa ra màn hình.

Dữ liệu: Vào từ file văn bản có tên là BSTR.INP:

- Dòng đầu tiên chứa hai số k và p cách nhau bởi dấu cách,
- k dòng tiếp theo mỗi dòng chứa một xâu nhị phân trong tập X,
- p dòng cuối cùng chứa các xâu nhị phân trong tập Y.

Kết quả: Ghi ra file BSTR.OUT xâu nhị phân z tìm được.

Mạng máy tính với độ tin cậy lớn nhất

Có n máy tính (đánh số từ 1 đến n). Các máy tính này cần được nối với nhau thành một mạng liên thông không có vòng. Với mỗi kênh (i,j) nối máy i với máy j, biết độ tin cậy là một số thực

$$0 < a[i,j] \le 1 \ (a[i,j] = a[j,i], \ 1 \le i, j \le n).$$

Độ tin cậy của đường truyền tin

$$(i_0, i_1), (i_1, i_2), \ \ldots, (i_{k-1}, i_k)$$

được định nghĩa là tích

$$a[i_0, i_1] \times a[i_1, i_2] \times \ \ldots \times a[i_{k-1}, i_k]$$

và độ tin cậy của mạng được định nghĩa là độ tin cậy nhỏ nhất của các đường truyền tin trong mạng.

Hãy lập trình xây dựng mạng nối n máy đã cho sao cho mạng có độ tin cậy là lớn nhất.

Dữ liệu: Vào từ file văn bản có tên NET.INP có cấu trúc như sau:

- Dòng đầu tiên chứa số n;
- Dòng thứ i trong n dòng tiếp theo chứa các số a[i,1], a[i,2],..., a[i,n], các số ghi cách nhau bởi dấu cách.

Kết quả: Đưa ra file NET.OUT:

- Dòng đầu tiên ghi độ tin cậy của mạng xây dựng được;
- mỗi dòng trong số các dòng tiếp theo ghi thông tin về một kênh trong mạng xây dựng gồm chỉ số của hai máy được nối bởi kênh này.

Ví dụ:

NET.INP	NET.OUT
3 0 1 0.3 1 0 0.5 0.3 0.5 0	0.5 1 2 2 3

Phủ nhỏ nhất

Cho ma trận vuông cấp n A = [a_{ij}] với các phần tử là 0 hoặc 1. Thuật ngữ ***hàng*** dùng để chỉ chung dòng hoặc cột của ma trận A. Hãy chọn ra một số ít nhất hàng để cho mỗi phần tử 1 của ma trận A đều nằm trên ít nhất một hàng được chọn.

Dữ liệu: Vào từ file văn bản COVER.INP có cấu trúc sau:
- Dòng đầu tiên ghi số n,
- Dòng thứ i trong số n dòng tiếp theo chứa các phần tử của dòng thứ i của ma trận A (các phần tử được ghi cách nhau bởi dấu cách).

Kết quả: Ghi ra file COVER.OUT:
- Dòng đầu tiên ghi số lượng hàng cần chọn;
- Dòng thứ hai ghi chỉ số của các dòng được chọn;
- Dòng thứ ba ghi chỉ số của các cột được chọn.

Ví dụ:

COVER.INP	COVER.OUT
4 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 0	3 2 1 3

Ban đại diện

Một lớp học sinh gồm m người đánh số từ 1 đến m. Các thành viên của lớp tham gia vào n nhóm Tin học A_1,..., A_n và n nhóm Thể thao B_1,..., B_n (mỗi học sinh có thể tham gia vào nhiều nhóm Tin học cũng như Thể thao). Lớp trưởng cần phải chọn ra một Ban đại diện gồm n người sao cho mỗi nhóm Tin học và mỗi nhóm Thể thao đều có ít nhất 1 thành viên của mình trong Ban đại diện.

Dữ liệu: Vào từ file văn bản REPR.INP có cấu trúc như sau:

- Dòng đầu tiên chứa hai số m, n;
- Dòng thứ i trong số m dòng tiếp theo ghi chỉ số của các nhóm Tin học mà học sinh i tham gia;
- Dòng thứ i trong số m dòng cuối cùng ghi chỉ số của các nhóm Thể thao mà học sinh i tham gia. (Các dữ liệu số được ghi cách nhau bởi dấu cách).

Kết quả: Đưa ra file REPR.OUT:

- Dòng đầu tiên chứa chỉ số của các học sinh được chọn vào Ban đại diện;
- Dòng thứ hai ghi chỉ số của các nhóm Tin học mà các học sinh được chọn là đại diện (theo thứ tự tương ứng với kết quả trong dòng 1);
- Dòng thứ ba ghi chỉ số của các nhóm Thể thao mà các học sinh được chọn là đại diện (theo thứ tự tương ứng với kết quả trong dòng 1);

Tô màu lưới ô vuông

Một lưới ô vuông gồm M dòng và N cột. Gọi G là một tập hợp các ô nào đó của lưới. Cần tô mầu các ô của G bằng 2 mầu xanh, đỏ sao cho nhìn theo bất kỳ dòng nào cũng như cột nào của lưới, ta đều có số ô mầu xanh và mầu đỏ hoặc bằng nhau, hoặc chênh nhau 1.

Dữ liệu: Vào từ file văn bản có tên PCOLOR.INP:

- Dòng đầu ghi các giá trị M, N (cách nhau ít nhất một dấu trắng),
- M dòng tiếp theo, mỗi dòng ghi thông tin một dòng của lưới dưới dạng một xâu nhị phân độ dài N (các ký tự 0, 1 viết liền nhau), với quy ước ký tự 1 ghi nhận vị trí ô của G và ký tự 0 ghi nhận các vị trí còn lại (không phải ô của G).

Kết quả: Đưa ra file văn bản có tên PCOLOR.OUT:

- Dòng đầu ghi các giá trị M, N (cách nhau một dấu trắng),
- M dòng sau, mỗi dòng ghi một xâu độ dài N gồm các ký tự 0, 1, 2 viết liền nhau với quy ước: ký tự 1 ghi nhận vị trí ô xanh, ký tự 2 ghi nhận vị trí ô đỏ, ký tự 0 ghi nhận các vị trí khác (không phải ô của G).

Ví dụ: Lưới và tập hợp G cho trong hình dưới đây:

	*		*			*				*
*		*					*			
	*		*		*					*
							*		*	
*				*						

được mô tả bởi file văn bản PCOLOR.INP sau

PCOLOR.INP
5 11
01010010001
10100001000
01010100001
00000001010
10000100000

PCOLOR.OUT
5 11
01020010002
10200001000
02010200001
00000002010
20000100000

Dãy đơn điệu chung

Cho dãy số nguyên $x_1, x_2, ..., x_n$; ($1 \le n \le 5000$). Đoạn đơn điệu (j, k) của dãy số trên là cặp chỉ số j và k sao cho j < k và dãy x[j], x[j+1],...,x[k] là đơn điệu không tăng hoặc đơn điệu không giảm. Cho hai dãy số

$a_1, a_2, ..., a_n$

và

$b_1, b_2, ..., b_n$.

Hãy tìm đoạn đơn điệu (j, k) chung dài nhất cho cả hai dãy.

Dữ liệu: Vào từ file văn bản có tên là MONO.INP:

- Dòng đầu tiên chứa số n,
- Dòng thứ i trong số n dòng tiếp theo chứa cặp số a_i, b_i, hai số ghi cách nhau bởi dấu cách.

Kết quả: Ghi ra file MONO.OUT hai chỉ số j và k của đoạn đơn điệu (j, k) chung dài nhất cho cả hai dãy tìm được.

Ví dụ:

MONO.INP
13 2 5 4 9 6 11 13 15 2 17 3 18 4 10 6 11 5 17 4 24 3 33 3 33 2 35

MONO.OUT
9 13

Lịch gia công trên hai máy

Cần phải thực hiện việc gia công n sản phẩm khác nhau (đánh số từ 1 đến n), đối với mỗi sản phẩm cần thực hiện lần lượt nguyên công A rồi đến nguyên công B. Có p máy thực hiện nguyên công A (đánh số từ 1 đến p) và q máy thực hiện nguyên công B (đánh số từ 1 đến q). Với mỗi sản phẩm i biết a_i, b_i theo thứ tự là thời gian cần thiết để hoàn thành việc thực hiện nguyên công A, B trên một trong số các máy thực hiện nguyên công A, B tương ứng. Việc thực hiện nguyên công trên máy đối với sản phẩm phải được tiến hành liên tục, không được phép ngắt quãng. Thời gian để chuyển từ việc thực hiện nguyên công này sang nguyên công khác đối với một sản phẩm cũng như thời gian để máy chuyển từ việc gia công sản phẩm này sang gia công sản phẩm khác là không đáng kể. Giả thiết thời điểm bắt đầu việc thực hiện gia công các sản phẩm là 0, hãy tìm lịch thực hiện việc gia công các sản phẩm sao cho thời điểm hoàn thành việc gia công tất cả các sản phẩm là càng sớm càng tốt.

Dữ liệu vào: File văn bản SCHEDULE.INP có cấu trúc như sau:

- Dòng đầu tiên chứa 3 số n, p, q ghi cách nhau bởi dấu cách ($1< n <100$, $1\leq p, q < 31$);
- Dòng thứ i trong số n dòng tiếp theo chứa hai số nguyên dương a_i, b_i được ghi cách nhau bởi dấu cách (i=1,2,...,n).

Kết quả: Ghi ra file văn bản SCHEDULE.OUT theo qui cách sau:

- Dòng đầu tiên ghi thời điểm hoàn thành việc gia công các sản phẩm theo lịch tìm được;
- Dòng thứ i trong số n dòng tiếp theo chứa thông tin về lịch thực hiện việc gia công sản phẩm i bao gồm bốn số m_{Ai}, t_{Ai}, m_{Bi} t_{Bi} , trong đó m_{Ai} là chỉ số máy thực hiện nguyên công A, t_{Ai} là thời điểm thực hiện nguyên công A, m_{Bi} là chỉ số máy thực hiện nguyên công B, t_{Bi} là thời điểm thực hiện nguyên công B.

Ví dụ. Các file dữ liệu vào-ra có thể có dạng

SCHEDULE.OUT
```
5  2   3
5  17
3  9
8  5
4  7
4  9
```

SCHEDULE.OUT
```
25
1   0   1   5
2   0   2   3
2   3   2   12
1   5   3   9
1   9   3   16
```

Các hình vuông

Trong mặt phẳng toạ độ cho N hình vuông có các cạnh song song với các trục toạ độ. Các hình vuông được đánh số từ 1 đến N. Các đỉnh của các hình vuông đều có toạ độ nguyên, và không có hai hình vuông nào có điểm chung.

Một hình vuông được gọi là *nhìn thấy được từ gốc toạ độ* O, O = (0, 0), nếu tìm được hai điểm A và B trên một trong số các cạnh của hình vuông sao cho phần trong của tam giác OAB không có điểm chung với bất cứ hình vuông nào trong các hình vuông còn lại.

Yêu cầu: Đếm số hình vuông nhìn thấy được từ gốc toạ độ.

Dữ liệu: Vào từ file văn bản SQUARES.INP:

- Dòng đầu tiên chứa số nguyên N, $1 \le N \le 1000$, là số lượng hình vuông;
- Dòng thứ i trong số N dòng tiếp theo chứa 3 số nguyên X_i, Y_i, L_i ghi cách nhau bởi dấu trắng, $1 \le X_i, Y_i, L_i \le 1000$, trong đó (X_i, Y_i) là toạ độ của đỉnh ở góc dưới bên trái, còn L_i là độ dài cạnh của hình vuông thứ i (i=1, 2,..., N).

Kết quả: Ghi ra file SQUARES.OUT:

- Dòng đầu tiên ghi số lượng hình vuông nhìn thấy được từ gốc toạ độ;
- Dòng tiếp theo ghi dãy số hiệu của các hình vuông nhìn thấy được từ gốc toạ độ.

Ví dụ:

SQUARES.INP	SQUARES.OUT
3 2 6 3 1 4 1 3 4 1	3 1 2 3

SQUARES.INP	SQUARES.OUT
4 1 2 1 3 1 1 2 4 2 3 7 1	2 1 2

Mời khách dự tiệc

Công ty trách nhiệm hữu hạn "Vui Vẻ" có n cán bộ đánh số từ 1 đến n. Cán bộ i có đánh giá độ vui tính là h_i (i=1, 2,..., n). Ngoại trừ Giám đốc Công ty, mỗi cán bộ có 1 thủ trưởng trực tiếp của mình.

Bạn cần giúp Công ty mời một nhóm cán bộ đến dự dạ tiệc "Vui vẻ" sao cho trong số những người được mời không đồng thời có mặt nhân viên và thủ trưởng trực tiếp và đồng thời tổng đánh giá độ vui tính của những người dự tiệc là lớn nhất.

Gỉa thiết rằng mỗi một thủ trưởng có không quá 20 cán bộ trực tiếp dưới quyền.

Dữ liệu: Vào từ file văn bản GUEST.INP

- Dòng đầu tiên ghi số cán bộ của công ty: n (1< n < 1001);
- Dòng thứ i trong số n dòng tiếp theo ghi hai số nguyên dương t_i , v_i, trong đó t_i là số hiệu của thủ trưởng trực tiếp và v_i là độ vui tính của cán bộ i (i=1, 2,..., n). Quy ước t_i = 0 nếu i là số hiệu của Giám đốc Công ty.

Kết quả: Ghi ra file văn bản GUEST.OUT:

- Dòng đầu tiên ghi hai số M, V, trong đó M là tổng số cán bộ được mời còn V là tổng độ vui tính của các cán bộ được mời dự tiệc;
- Dòng thứ i trong số M dòng tiếp theo ghi số hiệu của cán bộ được mời thứ i (i=1, 2,..., K);

Ví dụ:

GUEST.INP	GUEST.OUT
3 0 3 1 6 2 4	2 7 1 3

GUEST.INP	GUEST.OUT
7 0 1 1 1 1 12 2 50 2 1 3 1 3 1	3 63 3 4 5

Lập lịch hoàn thành sớm nhất các công việc

Có n công việc đánh số từ 1 đến n cần được bố trí thực hiện trên một máy. Biết

- p_i - thời gian cần thiết để thực hiện công việc i ;
- r_i - thời điểm sẵn sàng của công việc i (nghĩa là công việc i không được thực hiện ở thời điểm sớm hơn r_i);

Trong số các công việc đã cho có một số công việc chỉ được tiến hành sau khi một số công việc nào đó đã hoàn thành. Giả sử thời gian để máy chuyển từ việc thực hiện công việc này sang công việc khác là không đáng kể, và thời điểm bắt đầu thực hiện các công việc là 0.

Yêu cầu: Tìm trình tự thực hiện các công việc sao cho việc hoàn thành tất cả các công việc xảy ra ở thời điểm sớm nhất.

Dữ liệu: Vào từ file văn bản ETIME.INP:

- Dòng đầu tiên chứa số nguyên dương n (0 < n Ê 100).
- Dòng thứ 2 chứa n số nguyên dương $p_1, p_2, ..., p_n$.
- Dòng thứ 3 chứa n số nguyên dương $r_1, r_2, ..., r_n$.
- Dòng thứ i trong số n dòng tiếp theo chứa chỉ số các công việc phải hoàn thành trước khi thực hiện công việc i (qui ước ghi số 0 nếu công việc i có thể thực hiện độc lập).

Kết quả: Ghi ra file văn bản ETIME.OUT

- Dòng đầu tiên ghi thời điểm hoàn thành tất cả các công việc
- Dòng tiếp theo ghi trình tự thực hiện các công việc.

Ví dụ:

ETIME.INP
6 5 9 7 6 14 4 2 1 8 26 20 0 0 1 6 1 1 3 1 3 0

ETIME.OUT
45 6 1 2 3 5 4

Xếp các con xe trên bàn cờ quốc tế

Trên bàn cờ quốc tế kích thước n × n người ta đánh dấu một số ô. Vị trí các ô đánh dấu được cho bởi ma trận vuông cấp n: A = [a_{ij}], trong đó a_{ij} =1 nếu ô (i,j) được đánh dấu, và a_{ij} = 0 nếu trái lại. Hãy tìm cách đặt một số nhiều nhất các con xe lên các vị trí được đánh dấu trên bàn cờ sao cho không có hai con nào được ăn nhau (tức là không có hai con xe nào được xếp trên cùng một dòng hay cột của bàn cờ).

Dữ liệu vào: File văn bản CHESS.INP có cấu trúc như sau:

- Dòng đầu tiên chứa số n (n < 251);
- Mỗi dòng thứ i trong số n dòng tiếp theo chứa các phần tử của dòng thứ i của ma trận A (các số trên một dòng được ghi cách nhau bởi dấu cách).

Kết quả: Ghi ra file văn bản CHESS.OUT theo quy cách sau:

- Dòng đầu tiên ghi số lượng con xe xếp được k;
- Dòng thứ i trong số k dòng tiếp theo chứa chỉ số dòng và cột của vị trí đặt con xe thứ i (i=1,2,...,k).

Ví dụ: Các file dữ liệu vào ra có thể:

CHESS.INP	CHESS.OUT
10	9
1 0 0 0 0 0 0 1 1 0	1 1
1 0 0 0 0 0 0 0 1 0	2 9
1 1 1 0 0 0 0 0 0 1	3 2
1 1 1 0 0 0 0 0 0 0	4 3
1 1 1 1 0 0 0 0 0 0	5 4
1 1 1 1 1 0 0 0 0 0	6 5
1 1 0 1 1 1 0 0 0 0	7 6
1 0 0 1 0 0 1 0 0 0	8 7
0 0 0 0 0 0 0 1 1 0	9 8
0 0 0 0 0 0 0 1 1 0	

Tìm số đứng giữa

Cho dãy gồm n số nguyên dương

$a_1, a_2, ..., a_n$

(n < 250001). Cần tìm giá trị của các số đứng ở vị trí [n/2] và [n/2]+1 của dãy sau khi sắp xếp lại theo thứ tự không giảm.

Dữ liệu: Vào từ file văn bản MEDIAN.INP:

- Dòng đầu tiên chứa số n;
- Dòng thứ i trong số n dòng tiếp theo chứa số a_i.

Kết quả: Ghi ra file văn bản MEDIAN.OUT các giá trị của hai số tìm được.

Ví dụ:

MEDIAN.INP	MEDIAN.OUT
8 9 2 4 5 6 7 8 3	5 6

MEDIAN.INP	MEDIAN.OUT
7 2 4 5 6 7 8 3	4 5

Các người lính

Có N người lính, mỗi người đứng ở một vị trí. Mỗi vị trí được xác định bởi cặp (x,y) các toạ độ nguyên. Mỗi người lính có thể di chuyển từng bước, mỗi bước di chuyển lên, xuống, sang phải, sang trái một đơn vị (nghĩa là anh ta có thể thay đổi hoặc toạ độ x, hoặc toạ độ y của vị trí học tăng thêm 1 hoặc giảm bớt 1).

Các anh lính muốn tập trung thành 1 hàng dọc, người này tiếp nối người kia (tức là sao cho vị trí cuối cùng của họ là (x,y), (x+1,y), ..., (x+N-1,y) với x, y nào đó). Các số nguyên x, y cũng như trình tự của các người lính trong hàng là tuỳ ý.

Cần xác định cách di chuyển các người lính sao cho tổng số bước di chuyển của họ là ít nhất.

Tại mỗi thời điểm không được phép có hai hoặc nhiều hơn người lính ở cùng một vị trí.

Dữ liệu: Vào từ file văn bản SOLDIERS.INP:

- Dòng đầu tiên chứa số N ($1 \le N \le 10000$), là số người lính;
- Dòng thứ i trong số N dòng tiếp theo chứa hai số nguyên x_i, y_i là toạ độ của vị trí ban đầu của người lính i ($-10000 \le x_i , y_i \le 10000$).

Kết quả: Ghi ra file SOLDIERS.OUT: Số bước ít nhất cần thực hiện.

Ví dụ:

SOLDIERS.INP
3 1 0 2 4 3 2
SOLDIERS.OUT
4

SOLDIERS.INP
5 1 2 2 2 1 3 3 -2 3 3
SOLDIERS.OUT
8

Dãy con có tổng lớn nhất

Cho ma trận M kích thước n × n (1≤ n ≤100) với các phần tử M_{ij} là các số nguyên (1≤ i, j ≤ n). Ta gọi ***dãy con liên tiếp*** là dãy các phần tử của ma trận sao cho hai phần tử liên tiếp nhau trong dãy là ở trên hai vị trí liên tiếp trên cùng một dòng hay một cột hay một đường chéo, trong đó cho phép đi vòng quanh, đồng thời các phần tử trong dãy phải ở trên cùng một dòng hay một cột hay một đường chéo. Ví dụ, khi n=8 các dãy sau đây là dãy con liên tiếp:

$M_{2,1}$ $M_{2,2}$ $M_{2,3}$ $M_{2,4}$ $M_{2,4}$ $M_{2,6}$ $M_{2,7}$ $M_{2,8}$

$M_{2,2}$ $M_{2,3}$ $M_{2,4}$

$M_{2,6}$ $M_{2,7}$ $M_{2,8}$ $M_{2,1}$ $M_{2,2}$

$M_{4,3}$ $M_{5,3}$ $M_{6,3}$ $M_{7,3}$

$M_{1,2}$ $M_{2,3}$ $M_{3,4}$ $M_{4,5}$

$M_{2,4}$ $M_{3,3}$ $M_{4,2}$ $M_{5,1}$

$M_{3,3}$ $M_{4,2}$ $M_{5,1}$ $M_{1,5}$

$M_{5,6}$

Yêu cầu: Tìm dãy con liên tiếp có tổng các phần tử là lớn nhất.

Dữ liệu: Vào từ file văn bản SEQ.INP

- Dòng đầu tiên chứa số n (1 ≤ n ≤ 100);
- Dòng thứ k trong số n dòng tiếp theo chứa n số nguyên là các phần tử của dòng k trong ma trận M.

Kết quả: Ghi ra file văn bản SEQ.OUT:

- Dòng đầu tiên ghi hai số L, S là độ dài (số phần tử) và tổng của dãy con tìm được;
- Dòng thứ k trong số L dòng tiếp theo ghi hai số nguyên là chỉ số dòng và chỉ số cột trong ma trận M của phần tử thứ k trong dãy tìm được (1 ≤ k ≤ L).

Ví dụ:

SEQ.INP	SEQ.OUT
4 8 6 6 1 -3 4 0 5 4 2 1 9 1 -9 9 -2	3 24 3 4 4 3 1 2

Hội thảo bằng điện thoại

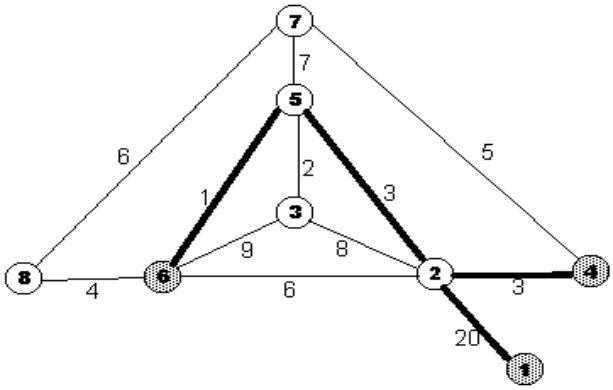
Một Công ty điện thoại cần tổ chức một cuộc hội thảo cho ba khách hàng trên đường truyền thông. Dịch vụ cần đảm bảo cho ba khách hàng có thể trao đổi với nhau đồng thời. Mỗi khách hàng có thể thâm nhập vào mạng truyền thông qua các cổng thâm nhập. Mạng truyền thông bao gồm các cổng thâm nhập được nối với nhau bởi các kênh thoại hai chiều. Ba khách hàng cần thâm nhập vào mạng từ ba cổng thâm nhập khác nhau sao cho tổng chi phí để liên kết họ là nhỏ nhất. Chú ý là được phép liên kết hai cổng thâm nhập bất kỳ. Bạn được biết các kênh nối giữa các cổng cùng với chi phí liên kết chúng. Bạn cần tìm cách liên kết đòi hỏi chi phí ít nhất để nối ba cổng thâm nhập của ba khách hàng tham gia hội thảo.

Dữ liệu: Vào từ file văn bản CONF.INP

- Dòng đầu tiên chứa hai số nguyên N và E, trong đó N (N ≤ 100) là số cổng, E là số kênh nối giữa các cổng.
- Mỗi dòng thứ k trong số E dòng tiếp theo chứa ba số nguyên i, j, c_{ij} cho biết kênh thứ k nối hai cổng i và j và chi phí để liên kết hai cổng này là c_{ij} ($1 \le c_{i,j} \le 100$);
- Dòng cuối cùng chứa ba số nguyên là chỉ số của ba cổng mà từ đó ba khách hàng tham gia hội thảo thâm nhập vào mạng. (Các cổng được đánh số từ 1 đến N.)

Kết quả: Ghi ra file CONF.OUT:

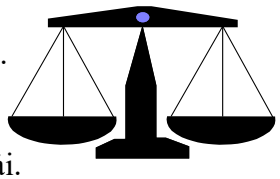
- Dòng đầu tiên chứa S là tổng chi phí theo cách liên kết tìm được và R là số kênh nối cần sử dụng R;
- Mỗi một trong số R dòng tiếp theo chứa hai chỉ số của hai đầu mút của kênh cần sử dụng trong cách liên kết.



Ví dụ

CONF.INP	CONF.OUT
8 12 1 2 20 2 3 8 2 4 3 2 5 3 2 6 6 3 5 2 3 6 9 4 7 5 5 6 1 5 7 7 6 8 4 7 8 6 1 4 6	27 4 1 2 2 4 2 5 5 6

Đồng tiền giả



Có N đồng tiền đánh số từ 1 đến N, trong đó có một đồng tiền giả. Đồng tiền giả có trọng lượng khác với đồng tiền thật. Các đồng tiền thật có trọng lượng giống nhau. Để phát hiện đồng tiền giả người ta sử dụng một cái cân thăng bằng gồm hai đĩa hai bên: đĩa trái và đĩa phải. Người ta tiến hành một loạt lần cân thử. Mỗi lần cân thử người ta đặt vào hai đĩa của cân mỗi đĩa cùng một số lượng đồng tiền và ghi nhận kết quả so sánh trọng lượng của các đồng tiền ở hai đĩa.

Yêu cầu: Dựa vào kết quả cân thử được ghi nhận, cần xác định đồng tiền giả.

Dữ liệu: Vào từ file văn bản CAN.INP:

- Dòng đầu tiên chứa hai số nguyên dương N, K, trong đó N là số lượng đồng tiền, K là số lần cân thử;
- 2K dòng tiếp theo ghi nhận kết quả của các lần cân thử. Thông tin về mỗi lần cân được ghi trong hai dòng liên tiếp:
  - Dòng đầu tiên ghi số nguyên P_i ($1 \le P_i \le N/2$) cho biết số lượng đồng tiền được đặt ở mỗi đĩa, P_i số tiếp theo cho biết chỉ số của các đồng tiền đặt ở đĩa trái, P_i số cuối cùng cho biết chỉ số của các đồng tiền đặt ở đĩa phải;
  - Dòng tiếp theo ghi một trong 3 ký tự <, >, = , ứng với kết quả cân:
    "<" : trọng lượng của các đồng tiền ở đĩa trái là nhẹ hơn trọng lượng của các đồng tiền ở đĩa phải,
    ">" : trọng lượng của các đồng tiền ở đĩa trái là nặng hơn trọng lượng của các đồng tiền ở đĩa phải,
    "=" : trọng lượng của các đồng tiền ở đĩa trái là bằng trọng lượng của các đồng tiền ở đĩa phải.

Kết quả: Ghi ra file văn bản CAN.OUT: chỉ số đồng tiền giả hoặc ghi số 0 nếu kết quả cân thử không cho phép xác định đồng tiền giả.

Ví dụ:

CAN.INP	CAN.OUT
5 3 2 1 2 3 4 < 1 1 4 = 1 2 5 =	3

CAN.INP	CAN.OUT
6 4 3 1 1 3 4 5 6 < 1 1 2 = 2 1 3 4 5 < 2 4 5 2 6 >	0

Lập lịch với tổng chi phí nhỏ nhất

Có n công việc và m máy có tính năng không giống nhau để thực hiện chúng. Các công việc được đánh số từ 1 đến n. Các máy được đánh số từ 1 đến m. Với mỗi công việc j biết thời gian cần thiết để hoàn thành nó trên máy i là p_{ji}, j = 1, 2, ..., n; i = 1, 2,..., m. Quá trình thực hiện công việc trên các máy phải được tiến hành liên tục từ lúc bắt đầu cho đến khi kết thúc, không được phép ngắt quãng. Các công việc đều sẵn sàng chờ thực hiện ở thời điểm bắt đầu lập lịch (giả thiết là 0). Thời gian để máy chuyển từ việc thực hiện công việc này sang thực hiện công việc khác bằng 0. Nếu công việc j được hoàn thành tại thời điểm C_j thì ta phải trả chi phí là C_j.

Yêu cầu: Hãy lập lịch để tổng chi phí phải trả là nhỏ nhất.

Dữ liệu: Vào từ file văn bản JOBS.INP:

- Dòng đầu tiên ghi 2 số n, m ($1 \le n \le 50$, $1 \le m \le 10$);
- Dòng thứ j trong số n dòng tiếp theo ghi các số nguyên dương

  $p_{j1}\ p_{j2} \cdots p_{jm}$ (j =1, 2,..., n).

Kết quả: Ghi ra file văn bản JOBS.OUT

- Dòng đầu tiên ghi T là thời điểm hoàn thành việc thực hiện các công việc theo lịch tìm được
- Dòng thứ i trong số m dòng tiếp theo ghi chỉ số các công việc được bố trí thực hiện trên máy i theo đúng trình thực hiện chúng trên máy này.

Ví dụ:

JOBS.INP
3 2 4 3 2 5 6 2

JOBS.OUT
9 2 3 1

Bảng số lớn nhất

Xét các bảng số A gồm m×n số nguyên khác nhau được xếp thành m dòng n cột. Phần tử nằm trên dòng i cột j của bảng ký hiệu là a_{ij}. Ta nói bảng A có kích thước m×n. Giả sử A=(a_{ij}), B = =($b_{i\,j}$) là hai bảng số kích thước m×n. Ta nói bảng A là nhỏ hơn bảng B nếu so sánh các phần tử của hai ma trận này theo thứ tự lần lượt theo các dòng, ở vị trí khác nhau đầu tiên bảng A chứa phần tử có giá trị nhỏ hơn.

Ví dụ, nếu

$$A=\begin{bmatrix}4 & 3 & 5\\ 2 & 8 & 1\\ 6 & -1 & 7\end{bmatrix} \quad B=\begin{bmatrix}4 & 3 & 5\\ 2 & 8 & 9\\ 1 & 7 & 6\end{bmatrix}$$

thì A là nhỏ hơn B.

Giả sử cho A là một bảng số kích thước m×n. Cho phép thực hiện lần lượt các phép biến đổi sau đây với ma trận A:

- Đổi chỗ hai dòng tuỳ ý của A;
- Đổi chỗ hai cột tuỳ ý của A.

Bằng cách áp dụng các phép biến đổi trên thể thu được từ bảng A một tập S các bảng kích thước m×n.

Yêu cầu: Cần xác định giá trị của phần tử cuối cùng (tức là phần tử ở vị trí dòng m cột n) trong bảng lớn nhất trong tập S.

Dữ liệu: Vào từ file văn bản MATRIX.INP:

- Dòng đầu tiên chứa hai số M, N (1 ≤ M, N ≤ 387);
- Dòng thứ i trong số M dòng tiếp theo mỗi dòng chứa N số nguyên với trị tuyệt đối không vượt quá 2147483647 là các phần tử trong dòng thứ i của bảng A.

Kết quả: Ghi ra file văn bản MATRIX giá trị của phần tử cuối cùng của bảng lớn nhất.

Ví dụ:

MATRIX.INP	MATRIX.OUT
2 3 4 3 5 2 8 1	5

Chương trình nghị sự

Nghị viện của một nước gồm N nghị viên. Theo quy chế hoạt động các nghị viên được chia ra làm các nhóm với số lượng thành viên khác nhau. Mỗi ngày làm việc mỗi nhóm cần cử một đại biểu của mình đến tham gia hoạt động của tiểu ban tư vấn của Nghị viện. Các tiểu ban tư vấn trong hai ngày khác nhau không được có thành phần giống hệt nhau. Nghị viện chỉ có thể tiến hành chương trình nghị sự được khi các nguyên tắc trên được tuân thủ.

Yêu cầu: Cần tìm cách chia các đại biểu ra thành các nhóm sao cho Nghị viện có thể tiến hành nghị sự trong nhiều ngày nhất.

Dữ liệu: Vào từ file văn bản PARL.INP trong đó chứa số nguyên dương N ***(*** $5 \le N \le 1000$***).***

Kết quả: Ghi ra file văn bản PARL.OUT

- Dòng đầu tiên ghi k là số lượng nhóm cần chia
- Dòng tiếp theo ghi số lượng thành viên của k nhóm được sắp xếp theo thứ tự tăng dần

Ví dụ

PARL.INP	PARL.OUT
7	2 3 4

PARL.INP	PARL.OUT
31	6 2 3 5 6 7 8

Cuộc dạo chơi trên bảng số

Trên các ô của lưới ô vuông kích thước m x n người ta ghi các số nguyên dương trong phạm vi từ 1 đến 8. Từ một ô bất kỳ của lưới bạn có thể di chuyển sang ô có chung cạnh với nó nếu hiệu số giữa số ở ô đang xét với số đứng trong ô cần di chuyển sang có giá trị tuyệt đối không lớn hơn 1.

Yêu cầu: Tìm cách di chuyển đi qua được nhiều ô nhất với điều kiện mỗi ô chỉ được đi qua không quá một lần.

Dữ liệu: Vào từ file văn bản WALK.INP:

- Dòng đầu tiên chứa hai số nguyên dương m, n ($1 \le m, n \le 10$);
- Dòng thứ i trong số m dòng tiếp theo ghi n số viết trong dòng thứ i của lưới ô vuông (giả thiết là các dòng của lưới được đánh số từ 1 đến m từ trên xuống dưới, còn các cột được đánh số từ 1 đến n từ trái qua phải).

Kết quả: Ghi ra file WALK.OUT:

- Dòng đầu tiên ghi độ dài của cách di chuyển tìm được (ký hiệu là k);
- Dòng thứ i trong số k dòng tiếp theo ghi toạ độ của ô thứ i trong cách di chuyển tìm được.

1	8	2	4
3	2	1	4
6	6	2	2
5	5	8	3

Hình 1a

1	8	2	4
3	2	1	4
6	6	2	2
5	5	8	3

Hình 1b

Ví dụ: File dữ liệu mô tả lưới ở hình 1a và File kết quả ghi cách di chuyển mô tả trong hình 1b có dạng sau:

WALK.INP	WALK.OUT
4 4 1 8 2 4 3 2 1 4 6 6 2 2 5 5 8 3	6 2 1 2 2 2 3 3 3 3 4 4 4

Di chuyển của xe cắt cỏ

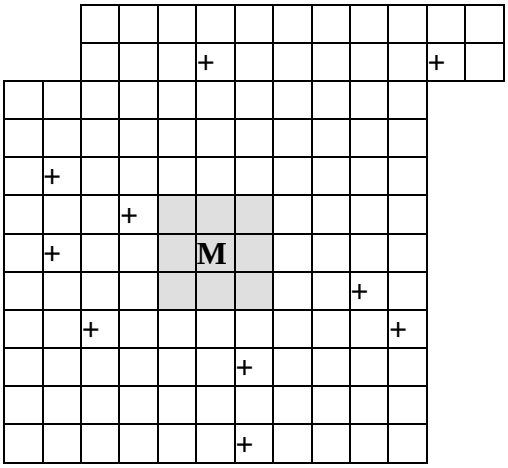
Một khu vườn có dạng hình chữ nhật có chiều dài D và chiều rộng R (mét). Giả sử khu vườn được chia ra làm D x R ô vuông cạnh độ dài 1 mét. Trong một số ô vuông có trồng cây. Một xe cắt cỏ có dạng một hình vuông kích thước 3 x 3 (mét) đặt tại một vị trí nào đó của khu vườn. Xe cắt cỏ có thể di chuyển sang trái, sang phải, lên trên, xuống dưới 1 mét. Xe không thể di chuyển lên các ô có cây. Xe cắt cỏ sẽ dọn dẹp được những ô mà nó đi qua.

Yêu cầu: Xác định các ô mà xe cắt cỏ có thể dọn dẹp được

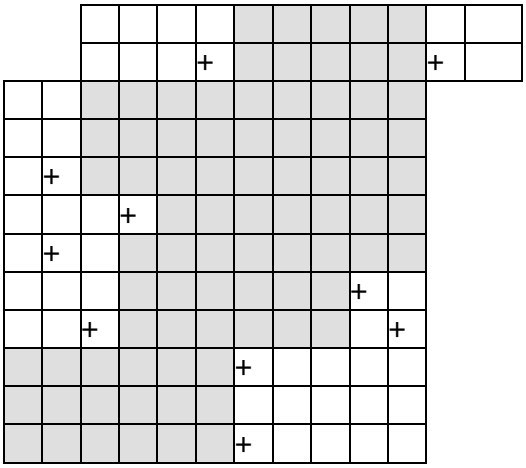
Dữ liệu: Vào từ file văn bản MOWER.INP:

- Dòng đầu tiên ghi 2 số nguyên dương D, R (5 ≤ D, R ≤ 40).
- Dòng thứ i trong số D dòng tiếp theo ghi R ký hiệu mô tả trạng thái của lưới ô vuông: '+ ' cho biết vị trí có cây, '.' cho biết ô vuông rỗng, 'M' cho biết tâm của vị trí của xe cắt cỏ. Các ký hiệu được ghi cách nhau bởi 1 dấu trắng.

Kết quả: Ghi ra file MOWER.OUT gồm D dòng mỗi dòng chứa R ký hiệu mô tả (các ký hiệu mô tả phải ghi cách nhau bởi 1 dấu trắng): ký hiệu '.' cho biết ô không dọn dẹp được bởi xe cắt cỏ, ký hiệu 'O' (chữ O in hoa) cho biết ô dọn dẹp được, ký hiệu '+' cho biết ô có cây.



Hình 1a
Ô bôi đen cho biết vị trí xe cắt cỏ



Hình 1b
Ô bôi đen cho biết các ô dọn dẹp được

Ví dụ: File dữ liệu và kết quả tương ứng với hình vẽ 1a, 1b có dạng

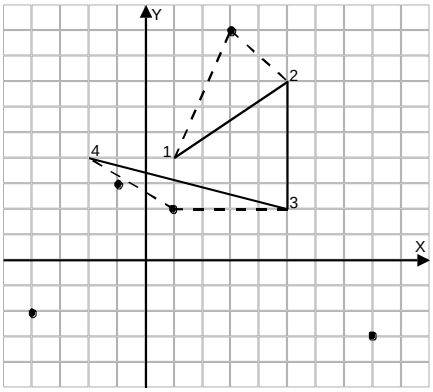
MOWER.INP	MOWER.OUT
12 11 . . . . + ... + ... + ... + ... + . . . M . . . . + . . . + ... + ... + ... + ...	12 11 . . . . O O O O O . . . . . + O O O O O + . . . O O O O O O O O O . . O O O O O O O O O . + O O O O O O O O O . . . + O O O O O O O . + . O O O O O O O O . . . O O O O O O + . . . + O O O O O O . + O O O O O O + ... O O O O O O . . . . . O O O O O O + ...

Thằng Bờm và con Mực

Hàng ngày thằng Bờm thường dạo chơi với con chó của nó có tên là Mực. Bờm đi với tốc độ không đổi và đường đi của nó là một đường gấp khúc (có thể tự cắt) có N điểm gãy. Mỗi điểm gãy được cho bởi cặp hai số nguyên (x_i, y_i) là toạ độ của nó trên mặt phẳng toạ độ. Con Mực chạy chơi với chủ theo con đường riêng của nó nhưng luôn gặp ông chủ tại N điểm gãy nói trên. Con Mực cùng ông chủ của nó cùng bắt đầu dạo chơi từ điểm (x_1, y_1) và kết thúc dạo chơi đồng thời tại điểm (x_N, y_N). Con Mực có thể chạy với tốc độ không vượt quá 2 lần tốc độ của Bờm. Khi Bờm đi theo đường thẳng từ điểm gãy này đến điểm gãy tiếp theo, con Mực có thể chạy đến các *điểm hấp dẫn* nó được cho bởi M cặp số nguyên (x'_j, y'_j). Tuy nhiên, sau khi tách rời ông chủ của nó tại điểm (x_i, y_i) (trong đó $1 \le i < N$) con Mực chỉ có thể thăm không quá 1 điểm hấp dẫn nó để rồi gặp lại chủ của nó tại điểm (x_{i+1}, y_{i+1}).

Yêu cầu: Tìm cách đi của con Mực thoả mãn các điều kiện nêu trên và sao cho số điểm hấp dẫn nó thăm được là nhiều nhất.

Ví dụ, đường đi của thằng Bờm (đường đậm nét), tập các điểm hấp dẫn (các nốt đen) và một trong những đường đi tốt nhất của con Mực (đường đứt nét) được cho trong hình vẽ dưới đây:



Dữ liệu: Vào từ file văn bản DOG.INP:

- Dòng đầu tiên chứa hai số nguyên dương N và M được ghi cách nhau bởi dấu cách ($2 \le N \le 100$, $0 \le M \le 100$);
- Dòng thứ hai chứa N cặp số nguyên $x_1, y_1, ..., x_N, y_N$ được ghi cách nhau bởi dấu cách
- Dòng thứ ba chứa M cặp số nguyên $x'_1, y'_1, ..., x'_M, y'_M$.
- Các điểm trong file dữ liệu là khác nhau từng đôi và toạ độ của chúng là các số nguyên có trị tuyệt đối không vượt quá 1000.

Kết quả: Ghi ra file văn bản DOG.OUT

- Dòng đầu tiên ghi số nguyên dương K là số điểm gãy trên đường đi của con Mực;
- Dòng thứ hai ghi K cặp toạ độ $u_1, v_1, ..., u_K, v_K$ (các toạ độ ghi cách nhau bởi dấu cách) biểu diễn đường đi tìm được.

Ví dụ:

DOG.INP	DOG.OUT
4 5 1 4 5 7 5 2 -2 4 -4 -2 3 9 1 2 -1 3 8 -3	6 1 4 3 9 5 7 5 2 1 2 -2 4

Xếp hàng mua vé

Có N người sắp hàng mua vé dự buổi hoà nhạc. Ta đánh số họ từ 1 đến N theo thứ tự đứng trong hàng. Mỗi người cần mua một vé, song người bán vé được phép bán cho mỗi người tối đa hai vé. Vì thế, một số người có thể rời hàng và nhờ người đứng trước mình mua hộ vé. Biết t_i là thời gian cần thiết để người i mua xong vé cho mình. Nếu người i+1 rời khỏi hàng và nhờ người i mua hộ vé thì thời gian để người thứ i mua được vé cho cả hai người là r_i.

Yêu cầu: Xác định xem những người nào cần rời khỏi hàng và nhờ người đứng trước mua hộ vé để tổng thời gian phục vụ bán vé là nhỏ nhất.

Dữ liệu: Vào từ file văn bản TICK.INP:

- Dòng đầu tiên chứa số N ($1 < N \leq 2000$);
- Dòng thứ 2 ghi N số nguyên dương $t_1, t_2, ..., t_N$;
- Dòng thứ ba ghi N-1 số nguyên dương $r_1, r_2, ..., r_{N-1}$.

Kết quả: Ghi ra file văn bản TICK.OUT

- Dòng đầu tiên ghi tổng thời gian phục vụ;
- Dòng tiếp theo ghi chỉ số của các khách hàng cần rời khỏi hàng (Nếu không có ai cần phải rời khỏi hàng thì qui ước ghi một số 0).

Ví dụ:

TICK.INP	TICK.OUT
5 2 5 7 8 4 4 9 10 10	18 2 4

TICK.INP	TICK.OUT
4 5 7 8 4 50 50 50 50	24 0

Các vị khách sộp

Có N khách sộp dự định đến dùng bữa tại một nhà hàng. Khách i đến nhà hàng vào thời điểm T_i và đem lại lợi nhuận cho nhà hàng là P_i nếu như họ dùng bữa tại nhà hàng. Cánh cửa của nhà hàng có K+1 trạng thái mời chào, mỗi trạng thái được biểu thị bởi một số nguyên trong khoảng từ 0 đến K. Trạng thái mời chào của cánh cửa có thể tăng hoặc giảm 1 mỗi khi đóng/ mở nó. Việc thực hiện một thao tác đóng/mở cửa mất một đơn vị thời gian. Tại thời điểm bắt đầu cánh cửa là đóng và có trạng thái mời chào 0. Mỗi khách i chỉ vào nhà hàng khi tại thời điểm họ đến, cánh cửa là mở và có trạng thái mời chào trùng với con số mà họ yêu thích S_i, bằng không họ sẽ rời khỏi nhà hàng và không bao giờ quay trở lại. Nhà hàng làm việc trong khoảng thời gian [0, T].

Yêu cầu: Bằng cách đóng mở cánh cửa một cách thích hợp tìm cách mời các khách vào nhà hàng dùng bữa sao cho tổng lợi nhuận thu được từ các khách hàng là lớn nhất.

Dữ liệu: Vào từ file văn bản WELCOME.INP:

- Dòng đầu tiên chứa các số N, K, T được ghi cách nhau bởi dấu cách. ($1 \le N \le 100$, $1 \le K \le 100$, $0 \le T \le 30000$).
- Dòng thứ hai ghi các thời điểm mà khách đến nhà hàng $T_1, T_2, ..., T_N$ phân cách nhau bởi dấu cách ($0 \le Ti \le T$, i = 1, 2, ..., N).
- Dòng thứ ba ghi lợi nhuận mà khách hàng đem lại cho nhà hàng nếu như họ vào dùng bữa tại nhà hàng $P_1, P_2, ..., P_N$, phân cách nhau bởi dấu cách ($0 < P_i \le 300$, i = 1, 2, ..., N).
- Dòng thứ tư chứa các số mà các khách hàng ưa thích $S_1, S_2, ..., S_N$, phân cách nhau bởi dấu cách ($1 \le S_i \le K$, i =1, 2, ..., N).

  Tất cả các giá trị số là các số nguyên.

Kết quả: Ghi ra file WELCOME.OUT:

- Dòng đầu tiên ghi tổng lợi nhuận thu được nhờ mời khách vào sử dụng bữa tại nhà hàng. Qui ước ghi số 0 nếu không mời được khách hàng nào vào nhà hàng.
- Nếu tổng lợi nhuận là số khác không thì trong dòng thứ hai ghi chỉ số của các khách mà bạn mời được vào nhà hàng theo thứ tự tăng dần của thời điểm họ đến nhà hàng.

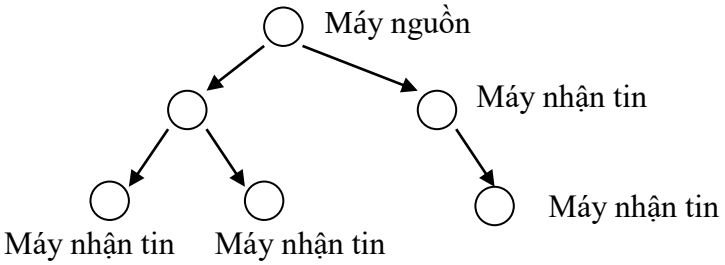
Ví dụ:

WELCOME.INP	WELCOME.OUT
4 10 20 10 16 8 16 10 11 15 1 10 7 1 8	26 3 2

WELCOME.INP	WELCOME.OUT
2 17 100 5 0 50 33 6 1	0

Truyền thông trên mạng

Cùng với giao thức TCP/IP, mà trong đó thông tin được trao đổi giữa hai máy tính thông qua việc xác lập kết nối, trong mạng còn có một kiểu giao thức khác gọi là UDP (user datagram protocol) mà trong đó không cần xác lập kết nối lôgic trực tiếp giữa máy nguồn và máy nhận tin. Điều đó cho phép thực hiện việc truyền thông rộng rãi trong mạng mà không cần truyền các gói dữ liệu cho từng máy nhận tin một. Việc truyền tin được tiến hành theo sơ đồ dạng cây như chỉ ra trong hình sau đây



Giả sử đã biết sơ đồ liên kết của các máy trong mạng dưới dạng bảng các kết nối giữa các máy tính trong mạng, trong đó cho biết máy nguồn và các máy nhận tin. Cần tìm tập gồm một số ít nhất máy tính (kể cả máy nguồn và các máy nhận tin) cần tham gia vào việc truyền thông để toàn bộ các máy nhận tin đều có thể tiếp nhận được thông tin từ máy nguồn.

Dữ liệu: Vào từ file UDP.INP:

- Dòng đều tiên chứa 3 số N, M, b; trong đó N là số máy trong mạng, M - là số máy nhận tin còn b là chỉ số của máy nguồn (Các máy trong mạng được đánh số từ 1 đến N, $1 < N < 11$).
- Dòng thứ i trong số N dòng tiếp theo mô tả thông tin về liên kết của máy i với các máy khác trong mạng: Số đầu tiên K_i là số máy tính trong mạng được nối với máy i, tiếp theo là K_i chỉ số của các máy nối với máy i.
- Dòng cuối cùng ghi chỉ số của các máy nhận tin.

Kết quả: Ghi ra file UDP.OUT:

- Dòng đầu tiên ghi số lượng máy cần chọn P (qui ước P=0, nếu không tìm được tập các máy thoả mãn yêu cầu đặt ra);
- Nếu P>0 thì ghi chỉ số của các máy được chọn.

Ví dụ:

UDP.INP	UDP.OUT
10 3 4 3 2 5 6 3 1 3 5 4 2 4 8 9 3 3 5 8 3 1 2 4 2 1 8 1 8 6 3 4 6 7 9 10 2 3 8 1 8 2 6 7	6 2 4 5 6 7 8

Phân nhóm

Trong một cuộc thăm dò xã hội học về các vấn đề kinh tế xã hội nóng bỏng người ta lập một phiếu gồm n câu hỏi đánh số từ 1 đến n, mà câu trả lời cho mỗi câu hỏi chỉ là đồng ý hoặc không đồng ý. Mỗi người được hỏi ý kiến phải điền vào phiếu thăm dò trả lời cho mỗi câu hỏi. Sau khi lấy ý kiến thăm dò của m người (được đánh số từ 1 đến m), kết quả trả lời của mỗi người được mô tả bởi một xâu nhị phân độ dài n: $b_1b_2...b_n$, trong đó $b_i = 1$ nếu trả lời của câu hỏi i là đồng ý và $b_i = 0$ nếu trả lời của câu hỏi i là không đồng ý. Biết rằng không có hai người nào trả lời hoàn toàn giống nhau.

Yêu cầu: Cần phân tập m người trả lời phiếu thăm dò ra thành một số ít nhất nhóm sao cho hai người bất kỳ trong cùng một nhóm có phiếu trả lời khác nhau ít ra là ra là ở hai câu hỏi.

Dữ liệu: Vào từ file văn bản GROUP.INP:

- Dòng đầu tiên chứa hai số nguyên dương n, m ghi cách nhau bởi dấu cách (n<51, m<201).
- Dòng thứ i trong số m dòng tiếp theo ghi xâu nhị phân độ dài n tương ứng với phiếu trả lời thăm dò của người i, i = 1, 2, ..., m.

Kết quả: Ghi ra file văn bản GROUP.OUT:

- Dòng đầu tiên ghi số nhóm tìm được k;
- Dòng thứ i trong số k dòng tiếp theo chứa chỉ số của những người trong nhóm i, i = 1, 2, ..., k.

Ví dụ:

GROUP.INP	GROUP.OUT
3 3 1 0 0 0 1 0 1 1 0	2 1 2 3

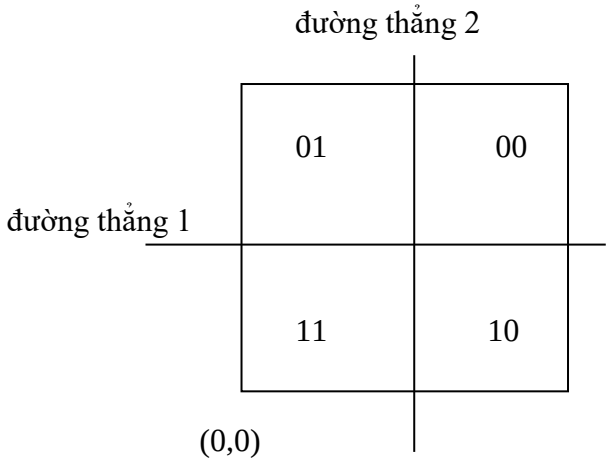
Tô màu

Trên mặt phẳng cho một hình vuông kích thước 1000×1000. Người ta lần lượt vẽ N đường thẳng, đánh số từ 1 đến N. Giả thiết là:

1. Sau mỗi lần vẽ độ dài của các cạnh của các phần thu được đều không nhỏ hơn 1.
2. Toạ độ của các đỉnh của hình vuông là (0,0) (0,1000) (1000,1000) (1000,0).
3. Mỗi đường thẳng đều cắt hai cạnh nào đó của hình vuông.
4. Không có đường nào đi qua đỉnh.

Như vậy, đường thẳng thứ i chia hình vuông ra làm hai phần: P_i và Q_i trong đó P_i là phần chứa đỉnh (0,0), còn Q_i không chứa đỉnh (0, 0). N đường thẳng đã vẽ sẽ chia hình vuông ra thành một số phần. Mỗi phần như vậy sẽ được gán cho một số hiệu là số nguyên không âm có biểu diễn nhị phân là $b_1b_2...b_n$, trong đó b_i =1 nếu phần này nằm trong P_i , và b_i = 0 nếu phần này nằm trong Q_i.
Yêu cầu: Tìm cách tô màu các phần của hình vuông sao cho hai phần có chung cạnh không được tô bởi cùng một màu và số màu cần sử dụng là ít nhất.

Ví dụ: Hình vẽ cho thấy hình vuông được chia ra làm 4 phần khi vẽ 2 đường thẳng:



Dữ liệu: Vào từ file văn bản PART.INP:

- Dòng đầu tiên ghi số đường thẳng được vẽ N (N < 21).
- Dòng thứ i trong số N dòng tiếp theo ghi 4 số nguyên a_i b_i c_i d_i là toạ độ của các giao điểm của đường thẳng thứ i với hai cạnh của hình vuông.

Kết quả: Ghi ra file văn bản PART.INP:

- Số màu cần sử dụng.
- Mỗi một trong số các dòng tiếp theo ghi số hiệu của một phần (được ghi ở hệ cơ đếm 10) và số hiệu của màu tô phần này.

Ví dụ:

PART.INP
3 0 500 1000 500 500 0 500 1000

PART.OUT
2 0 1 1 2 2 2 3 1

Tam giác

Trên mặt phẳng cho N ≤ 100 tam giác được xác định bởi toạ độ của ba đỉnh: (X_{1i}, Y_{1i}), (X_{2i}, Y_{2i}), (X_{3i}, Y_{3i}) (i = 1, 2, ..., N). Tất cả các toạ độ đều là những số nguyên có trị tuyệt đối không vượt quá 3000. Cần tìm diện tích của phần mặt phẳng bị phủ bởi các tam giác này. Giả thiết rằng tất cả các tam giác đã cho là không suy biến (nghĩa là 3 đỉnh của tam giác là ba điểm phân biệt và không nằm trên cùng một đường thẳng).

Dữ liệu: Vào từ file văn bản TAMGIAC.INP:

- Dòng đầu tiên chứa số N;
- Dòng thứ i trong số N dòng tiếp theo chứa 6 số X_{1i}, Y_{1i}, X_{2i}, Y_{2i}, X_{3i}, Y_{3i} xác định tam giác thứ i (i = 1, 2, ..., N).

Các số trên cùng dòng được ghi cách nhau bởi ít nhất một dấu cách.

Kết quả: Ghi ra file văn bản TAMGIAC.OUT duy nhất một số là giá trị của diện tích cần tìm với 4 chữ số sau dấu phẩy.

Ví dụ:

TAMGIAC.INP	TAMGIAC.OUT
2 0 0 2 0 1 2 0 0 2 0 1 1	2.0000

Dãy con lớn nhất

Trên một vòng tròn người ta đánh dấu N vị trí. Các vị trí được đánh số thứ tự từ 1 đến N theo chiều kim đồng hồ. Tại vị trí thứ i người ta ghi số nguyên a_i, i = 1, 2, ..., N. Cần tìm cách chọn ra dãy con gồm một số số liên tiếp nhau trên vòng tròn có tổng các số hạng là lớn nhất.

Dữ liệu: Vào từ file văn bản DAY.INP:

- Dòng đầu tiên ghi số nguyên dương N ($N \le 10^7$);
- Dòng thứ i trong số N dòng tiếp theo ghi số a_i ($|a_i| \le 10000$), i = 1, 2, ..., N.

Kết quả: Ghi ra file văn bản DAY.OUT:

- Dòng đầu tiên ghi K là số lượng phần tử của dãy con được chọn ;
- Dòng thứ j trong số K dòng tiếp theo ghi số thứ tự trên vòng tròn của số hạng thứ j của dãy con được chọn.

Ví dụ:

DAY.INP	DAY.OUT
7 2 -4 1 -7 4 6 -1	4 5 6 7 1

DAY.INP	DAY.OUT
11 -10 -1 -3 5 6 -4 -9 20 -5 6 -3	3 8 9 10

Dãy con chung lớn nhất

Cho 3 dãy ký tự

x[1], x[2], ..., x[m];
y[1], y[2], ..., y[n];
z[1], z[2], ..., z[k].

Dãy ký tự

w[1], w[2], ..., w[p]

được gọi là dãy con chung độ dài p của 3 dãy đã cho nếu tìm được các dãy các chỉ số

$ix_1, ix_2, ..., ix_p$; $iy_1, iy_2, ..., iy_p$; $iz_1, iz_2, ..., iz_p$

sao cho

1. $0 < ix_1 < ix_2 < ... < ix_p \le m$,
2. $0 < iy_1 < iy_2 < ... < iy_p \le n$,
3. $0 < iz_1 < iz_2 < ... < iz_p \le k$,
4. $x[ix_j] = y[iy_j] = z[iz_j] = w[j]$, j = 1, 2,..., p.

Yêu cầu: Tìm dãy con chung có độ dài lớn nhất của 3 dãy đã cho.

Dữ liệu: Vào từ file văn bản LCS.INP

- Dòng đầu tiên ghi 3 số nguyên dương m, n, k (m, n, k ≤ 60);
- Dòng thứ 2 ghi m ký tự của xâu thứ 1;
- Dòng thứ 3 ghi m ký tự của xâu thứ 2;
- Dòng thứ 4 ghi m ký tự của xâu thứ 3;

(Các ký tự trong các dòng 2, 3, 4 được ghi liên tiếp nhau).

Kết quả: Ghi ra file văn bản LCS.OUT :

- Dòng đầu tiên ghi p là độ dài của dãy con chung tìm được;
- Dòng thứ hai ghi dãy chỉ số $ix_1, ix_2, ..., ix_p$.
- Dòng thứ hai ghi dãy chỉ số $iy_1, iy_2, ..., iy_p$.
- Dòng thứ hai ghi dãy chỉ số $iz_1, iz_2, ..., iz_p$.

Qui ước: Nếu không tìm được dãy con chung của 3 dãy thì file LCS.OUT chỉ chứa một số 0.

Ví dụ:

LCS.INP	LCS.OUT
3 4 5 abc acac babcd	2 1 3 1 2 2 4

LCS.INP	LCS.OUT
3 5 4 abc defgh opqz	0

Lập lịch với hiệu quả lớn nhất

Có n công việc đánh số từ 1 đến n cần được bố trí thực hiện trên một máy. Mỗi công việc đều đòi hỏi thời gian hoàn thành là 1. Việc thực hiện mỗi công việc phải được tiến hành liên tục không cho phép ngắt quãng. Biết

- w_i - mức độ quan trọng của công việc i ;
- d_i - thời hạn hoàn thành công việc i, i = 1, 2, ..., n.

Giả sử C_i là thời điểm hoàn thành công việc i, khi đó

- Nếu $C_i < d_i$ thì tiền thưởng thu được từ việc thực hiện công việc i sớm hạn là $w_i(d_i - C_i)$,
- Nếu $C_i > d_i$ thì tiền phạt do việc thực hiện công việc i trễ hạn là $w_i(C_i - d_i)$,
- Nếu $C_i = d_i$ lượng phạt và thưởng đều bằng 0.

Giả thiết rằng:

- Thời gian để máy chuyển từ việc thực hiện công việc này sang công việc khác là không đáng kể
- Thời điểm bắt đầu thực hiện các công việc là 0.

Ta gọi hiệu quả của một lịch thực hiện các công việc đã cho là hiệu số giữa tổng tiền thưởng và tổng tiền phạt từ việc thực hiện các công việc

Yêu cầu: Tìm lịch thực hiện với hiệu quả lớn nhất.

Dữ liệu: Vào từ file văn bản JOBS.INP:

- Dòng đầu tiên chứa số nguyên dương n (0 < n Ê 100).
- Dòng thứ 2 chứa n số nguyên dương $w_1, w_2, ..., w_n$.
- Dòng thứ 3 chứa n số nguyên dương $d_1, d_2, ..., d_n$.

Kết quả: Ghi ra file văn bản JOBS.OUT

- Dòng đầu tiên ghi hiệu quả của lịch tìm được;
- Dòng thứ i trong số n dòng tiếp theo ghi thời điểm bắt đầu thực hiện công việc i, i=1, 2, ..., n.

Ví dụ:

JOBS.INP
2 3 2 2 1

JOBS.OUT
1 0 1

Lập lịch với hiệu quả lớn nhất II

Có n công việc đánh số từ 1 đến n cần được bố trí thực hiện trên một máy. Mỗi công việc đều đòi hỏi thời gian hoàn thành là 1. Việc thực hiện mỗi công việc phải được tiến hành liên tục không cho phép ngắt quãng. Biết

- t_i - hệ số thưởng của công việc i ;
- p_i - hệ số phạt của công việc i ;
- d_i - thời hạn hoàn thành công việc i, i = 1, 2, ..., n.

Giả sử C_i là thời điểm hoàn thành công việc i, khi đó

- Nếu $C_i < d_i$ thì tiền thưởng thu được từ việc thực hiện công việc i sớm hạn là $t_i(d_i - C_i)$,
- Nếu $C_i > d_i$ thì tiền phạt do việc thực hiện công việc i trễ hạn là $p_i(C_i - d_i)$,
- Nếu $C_i = d_i$ lượng phạt và thưởng đều bằng 0.

Giả thiết rằng:

- Thời gian để máy chuyển từ việc thực hiện công việc này sang công việc khác là không đáng kể
- Thời điểm bắt đầu thực hiện các công việc là 0.

Ta gọi hiệu quả của một lịch thực hiện các công việc đã cho là hiệu số giữa tổng tiền thưởng và tổng tiền phạt từ việc thực hiện các công việc

Yêu cầu: Tìm lịch thực hiện với hiệu quả lớn nhất.

Dữ liệu: Vào từ file văn bản JOBS.INP:

- Dòng đầu tiên chứa số nguyên dương n (0 < n Ê 100).
- Dòng thứ 2 chứa n số nguyên dương $t_1, t_2, ..., t_n$.
- Dòng thứ 3 chứa n số nguyên dương $p_1, p_2, ..., p_n$.
- Dòng thứ 4 chứa n số nguyên dương $d_1, d_2, ..., d_n$.

Kết quả: Ghi ra file văn bản JOBS.OUT

- Dòng đầu tiên ghi hiệu quả của lịch tìm được;
- Dòng thứ i trong số n dòng tiếp theo ghi thời điểm bắt đầu thực hiện công việc i, i=1, 2, ..., n.

Ví dụ:

JOBS.INP
2
3 2
1 5
2 1

JOBS.OUT
0
1
0

Câu trả lời nào không có?

Trong một cuộc thăm dò xã hội học về các vấn đề kinh tế xã hội nóng bỏng người ta lập một phiếu gồm k câu hỏi đánh số từ 1 đến k, mà câu trả lời cho mỗi câu hỏi chỉ là đồng ý hoặc không đồng ý. Mỗi người được hỏi ý kiến phải điền vào phiếu thăm dò trả lời cho mỗi câu hỏi. Sau khi lấy ý kiến thăm dò của n người (được đánh số từ 1 đến n), kết quả trả lời của mỗi người được mô tả bởi một xâu nhị phân độ dài k: $b_1b_2...b_k$, trong đó $b_i = 1$ nếu trả lời của câu hỏi i là đồng ý và $b_i = 0$ nếu trả lời của câu hỏi i là không đồng ý. Biết rằng không có hai người nào trả lời giống nhau.

Yêu cầu: Hãy chỉ ra một cách trả lời không giống với bất cứ cách trả lời ở phiếu thăm dò nào.

Dữ liệu: Vào từ file văn bản ANS.INP:

- Dòng đầu tiên chứa 2 số nguyên dương n, k ($2^{k-1}+1 < n < 2^k$, $k <21$).
- Dòng thứ i trong số n dòng tiếp theo ghi xâu nhị phân độ dài k tương ứng với phiếu trả lời thăm dò của người i, i = 1, 2, ..., n.

Kết quả: Ghi ra file văn bản ANS.OUT:

- Dòng đầu tiên ghi YES (NO) nếu tìm được (không tìm được) câu trả lời thoả mãn yêu cầu;
- Nếu dòng đầu ghi YES, thì dòng thứ 2 ghi xâu nhị phân độ dài k tương ứng với cách trả lời tìm được

Ví dụ:

ANS.INP	ANS.OUT
5 100 010 110 101 011	YES 111

Hệ thống đèn

Khu vực đặt các bể xăng của một Tổng Đại lý Xăng dầu có dạng một hình chữ nhật được chia thành m × n ô vuông. Các ô vuông được đánh tọa độ như chỉ ra trong hình vẽ dưới đây:

	1	...	j	...	n
1					
...					
i					
...					
m					

Tại k ô của lưới có đặt các bể xăng. Người ta cần xây dựng một hệ thống đèn pha chiếu sáng, mỗi đèn chỉ chiếu dọc theo hoặc là hàng hoặc là cột của lưới ô vuông sao cho mỗi bể chứa phải được chiếu sáng bởi ít nhất một đèn pha chiếu dọc theo hàng hoặc cột chứa nó. Biết:

- a_i là chi phí xây dựng đèn chiếu sáng dọc theo hàng i (i = 1, 2,..., m);
- b_j là chi phí xây dựng đèn chiếu sáng dọc theo cột j (j = 1, 2,..., n).

Yêu cầu: Tìm cách xây dựng hệ thống đèn với tổng chi phí xây dựng là nhỏ nhất.

Dữ liệu: Vào từ file văn bản LIGHT.INP:

- Dòng đầu tiên chứa ba số nguyên dương m, n, k (m, n < 100);
- Dòng thứ hai chứa m số nguyên dương a_1, a_2, ..., a_m;
- Dòng thứ ba chứa n số nguyên dương b_1, b_2, ..., b_n;
- Dòng thứ i trong số k dòng tiếp theo chứa toạ độ của bể xăng thứ i (i = 1,2,...,k).

Kết quả: Ghi ra file văn bản LIGHT.OUT:

- Dòng đầu tiên ghi tổng chi phí theo cách xây dựng tìm được;
- Dòng thứ hai ghi hai số nguyên P và Q theo thứ tự là số lượng đèn chiếu dọc theo hàng và cột (ghi số 0 nếu không có);
- P+Q dòng tiếp theo chứa lần lượt các toạ độ của các hàng rồi đến các toạ độ của các cột có đặt đèn chiếu sáng, mỗi toạ độ ghi trên một dòng.

Ví dụ:

LIGHT.INP	LIGHT.OUT
2 3 4 10 5 12 4 2 1 2 1 3 2 1 2 3	11 1 2 2 2 3

LIGHT.INP	LIGHT.OUT
2 3 4 15 17 2 4 6 1 1 2 2 2 3 2 1	12 0 3 1 2 3

Hệ thống đường một chiều

Sơ đồ giao thông của thành phố gồm N nút giao thông đánh số từ 1 đến N và M đường phố đánh số từ 1 đến M. Mỗi đường phố nối hai nút giao thông. Để góp phần giảm tình trạng ách tắc giao thông trong thành phố người ta muốn qui định tất cả các đường phố thành các đường một chiều sao cho vẫn đảm bảo giữa hai nút giao thông bất kỳ luôn có ít nhất một đường đi nối chúng. Cần lập chương trình kiểm tra xem có thể đáp ứng yêu cầu đặt ra hay không.

Dữ liệu: Vào từ file văn bản ONEWAY.INP:

- Dòng đầu tiên chứa hai số nguyên dương N, M (N < 101);
- Dòng thứ i trong số M dòng tiếp theo chứa hai số nguyên dương tương ứng là chỉ số của các nút giao thông là các đầu mút của đường phố i, i = 1, 2, ..., M.

Kết quả: Ghi ra file ONEWAY.OUT:

- Dòng đầu tiên ghi số 1 nếu tìm được cách qui định hướng của các đường phố, ghi 0 nếu ngược lại.
- Nếu có cách định hướng của các đường phố thì dòng thứ i trong số M dòng tiếp theo ghi hai số nguyên dương theo thứ tự là chỉ số của nút đầu và nút cuối của đường phố i (i = 1, ..., M).

Ví dụ:

ONEWAY.INP	ONEWAY.OUT
5 6 1 2 2 3 4 3 5 4 5 1 3 5	1 1 2 2 3 3 4 4 5 5 1 3 5

ONEWAY.INP	ONEWAY.OUT
4 4 1 2 1 3 1 4 2 3	0

Tập trùng hợp lớn nhất

Có n công văn đánh số từ 1 đến n (n < 500). Mỗi công văn được gửi cho một trong số n địa chỉ cũng được đánh số từ 1 đến n. Cần tìm tập hợp các công văn trong số các công văn đã cho, có số công văn lớn nhất, thoả mãn tính chất: tập số thứ tự của các công văn này trùng với tập số thứ tự của các địa chỉ mà chúng được gửi đến.

Dữ liệu: Vào từ file văn bản CV.INP:

- Dòng đầu tiên ghi số n;
- Dòng thứ i trong số n dòng tiếp theo ghi số thứ tự của địa chỉ mà công văn i được gửi đến (i = 1, 2, ..., n).

Kết quả: Ghi ra file văn bản CV.OUT:

- Dòng đầu ghi số k là số công văn của tập hợp tìm được;
- k dòng tiếp theo, mỗi dòng ghi số thứ tự của một công văn trong tập hợp tìm được.

Ví dụ:

CV.INP	CV.OUT
6 3 1 2 2 4 5	3 1 2 3

Các miền trên bảng

Cho một bảng chữ nhật được chia thành M × N ô vuông (M dòng, N cột). Mỗi ô vuông ghi một số nguyên dương (trong khoảng từ 1 đến 255). Một miền của bảng là tập hợp tất cả các ô có cùng giá trị số sao cho chúng đi được sang nhau bằng cách đi qua các ô có chung cạnh và có cùng giá trị số đang xét. Địa chỉ của một miền là toạ độ [dòng, cột] của ô đầu tiên thuộc miền theo thứ tự duyệt từ trái sang phải và từ trên xuống dưới. Diện tích của một miền là số ô thuộc miền đó.
Ví dụ: Bảng

1	1	2	2	2
1	2	2	1	2
3	1	1	1	2

có 4 miền, miền tô mầu xám (giá trị các ô là 2) có địa chỉ là [1, 3] và diện tích là 7.

Cần xác định:

- số miền của mảng,
- miền có diện tích lớn nhất (chỉ rõ giá trị diện tích và địa chỉ của miền).

Dữ liệu: Vào từ file văn bản MIEN.INP có dạng:

M N
A[1, 1] A[1, 2] ... A[1, N]
A[2, 1] A[2, 2] ... A[2, N]
.....
A[M, 1] A[M, 2] ... A[M, N]

trong đó A[i, j] là giá trị số của ô [i, j], các số trên cùng một dòng ghi cách nhau ít nhất một dấu trắng.

Giới hạn: M, N < 101.

Kết quả: Ghi ra file MIEN.OUT:

- Dòng đầu tiên ghi số miền của mảng;
- Dòng thứ hai ghi ba số S, x, y, trong đó S là diện tích còn (i, j) là địa chỉ của miền lớn nhất.

Ví dụ:

MIEN.INP	MIEN.OUT
3 5 1 1 2 2 2 1 2 2 1 2 3 1 1 1 2	4 7 1 3

Cân hoa quả

Có một thùng táo gồm N quả. Một bà nội trợ cần mua một kg táo. Hãy tìm cách giúp bà nội trợ chọn những quả táo nào trong thùng để có thể mua đúng 1 kg táo.

Dữ liệu: Cho trong file văn bản FRUITS.IN:
- Dòng đầu tiên ghi số nguyến dương N (N < 1001);
- Dòng thứ i trong số N dòng tiếp theo ghi trọng lượng (đơn vị tính là gam) của quả táo thứ i trong thùng, i = 1, 2, ..., N.

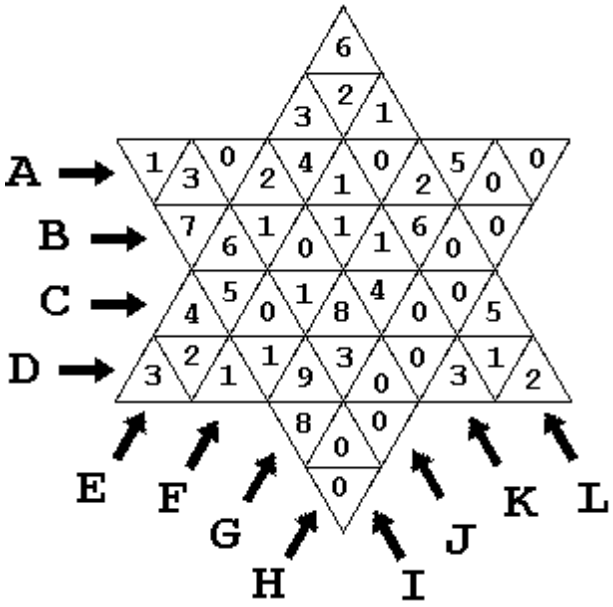
Kết quả: Ghi ra file văn bản FRUITS.OUT:
- Nếu tìm được cách mua thoả mãn yêu cầu thì ghi trọng lượng của các quả táo cần mua, mỗi trọng lượng ghi trên một dòng.
- Nếu không tìm được cách chọn thoả mãn yêu cầu đầu bài ghi thông báo: NO SOLUTION

Ví dụ:

FRUITS.IN	FRUITS.OUT
5 200 300 200 400 500	300 200 500

Hình ông sao (30 điểm)

Cho một lưới tam giác gồm 48 ô. Trong mỗi ô người ta viết một chữ số từ 0 đến 9. Mỗi một ô của lưới sẽ nằm trên một, hai hoặc ba đường thẳng được đánh nhãn bởi các chữ cái từ A đến L. Một trong những cách điền số vào các ô của lưới được cho trong hình 1. Trong hình 1, ô chứa số 9 nằm trên các đường D, G và I; còn ô chứa số 7 nằm trên các đường B và I.
Với mỗi đường ta đều biết số lớn nhất trên các ô thuộc nó. Chẳng hạn, trong ví dụ ở hình 1, số lớn nhất trong đường A là 5, đường B: 7, đường E: 6, đường H: 0, đường J: 8,... Như vậy mỗi bảng số đều có một bộ giá trị của các số lớn nhất trên các đường của nó. Ví dụ: đối với cách điền số trong hình 1 bộ số đó là: 5 7 8 9 6 1 9 0 9 8 4 6.



Hình 1

Yêu cầu: Hãy viết chương trình dựa vào bộ giá trị của các số lớn nhất trên các đường của bảng số, tìm giá trị nhỏ nhất và giá trị lớn nhất của tổng các số trên bảng số.

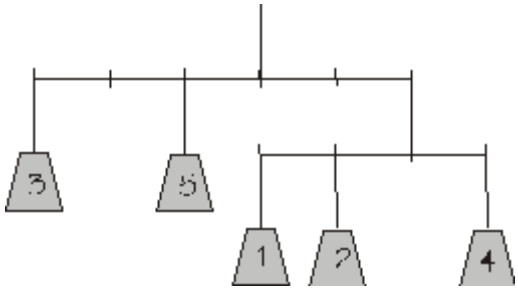
Dữ liệu: File văn bản STAR.IN chứa 12 số lớn nhất trong các dòng theo thứ tự từ A đến L của bảng số.

Kết quả: Ghi ra file văn bản STAR.OUT hai số theo thứ tự là giá trị nhỏ nhất và giá trị lớn nhất của tổng các chữ số trong bảng hoặc ghi dòng thông báo: NO SOLUTION, nếu không tìm được cách điền các chữ số vào bảng với bộ giá trị của các số lớn nhất trên các đường đã cho.

Ví dụ:

STAR.IN	STAR.OUT
5 7 8 9 6 1 9 0 9 8 4 6	40 172

Cân thăng bằng (40 điểm)



Hình 1.

Trên hình 1 bạn có thể thấy một dạng phức tạp của hệ cân thăng bằng hoàn hảo (nghĩa là mỗi cân con trong nó cũng là thăng bằng). Trọng lượng của các quả cân là 1, 2, ..., 5 kg. Khoảng cách giữa hai vạch mức là 1 mét.

Bạn có thể kiểm tra tính cân bằng của hệ cân thông qua tính toán sau:
-3*3 + (-1)*5 + 2*(1+2+4) = 0 (Tay đòn mức 1) ,
-2*1 + (-1)*2 + 1*4=0 (Tay đòn mức 2) .

Cấu trúc tay đòn của hệ cân thăng bằng hoàn hảo được xác định bởi bộ các toạ độ tương đối của các vị trí treo quả cân. Hệ trong hình 1 có cấu trúc tay đòn được mô tả bởi dòng thông tin sau:

(-3,-1,2(-2,-1,1))

Cấu trúc trọng lượng của hệ cân thăng bằng hoàn hảo được xác định bởi bộ trọng lượng của các quả cân. Ví dụ, hệ cân trong hình 1 có cấu trúc trọng lượng được mô tả bởi dòng thông tin sau:

(3,5,(1,2,4))

Yêu cầu: Hãy viết chương trình dựa vào cấu trúc tay đòn cho trước, tìm cấu trúc trọng lượng của hệ cân thăng bằng hoàn hảo.

Hạn chế kỹ thuật:

- Hệ cân gồm N < 18 quả cân.
- Chỉ được sử dụng các quả cân có trọng lượng từ 1 đến N kg, mỗi quả không quá một lần.
- Mỗi tay đòn có không quá 7 điểm chịu tải trọng. Hệ trong hình 1 có 2 tay đòn, mỗi tay đòn có 3 điểm chịu tải trọng.
- Giả thiết rằng tất cả các bộ dữ liệu đều có lời giải, và chỉ cần tìm một lời giải, khi có nhiều lời giải.

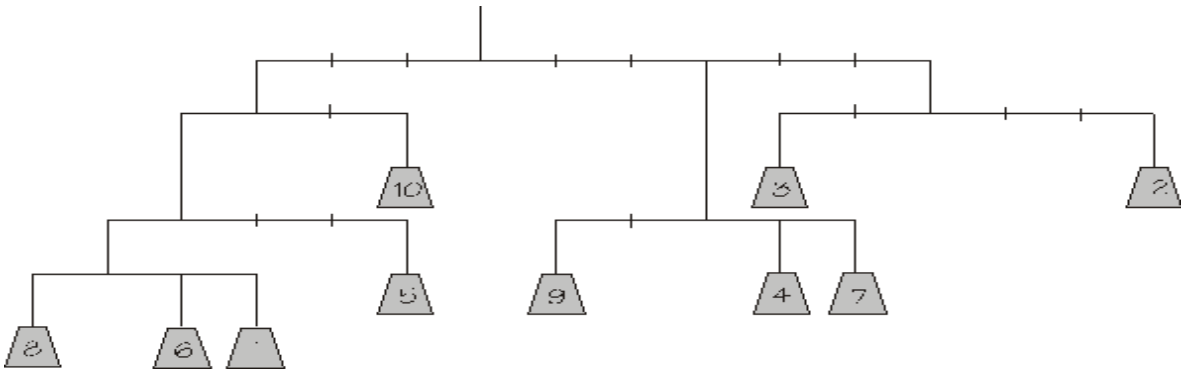
Dữ liệu: Vào từ file văn bản MOBILE.IN gồm một dòng duy nhất chứa cấu trúc tay đòn của hệ cân, các số trong dòng là số nguyên trong khoảng từ -50 đến 50.

Kết quả: Ghi ra một dòng của file MOBILE.OUT cấu trúc trọng lượng của hệ cân không chứa dấu cách.

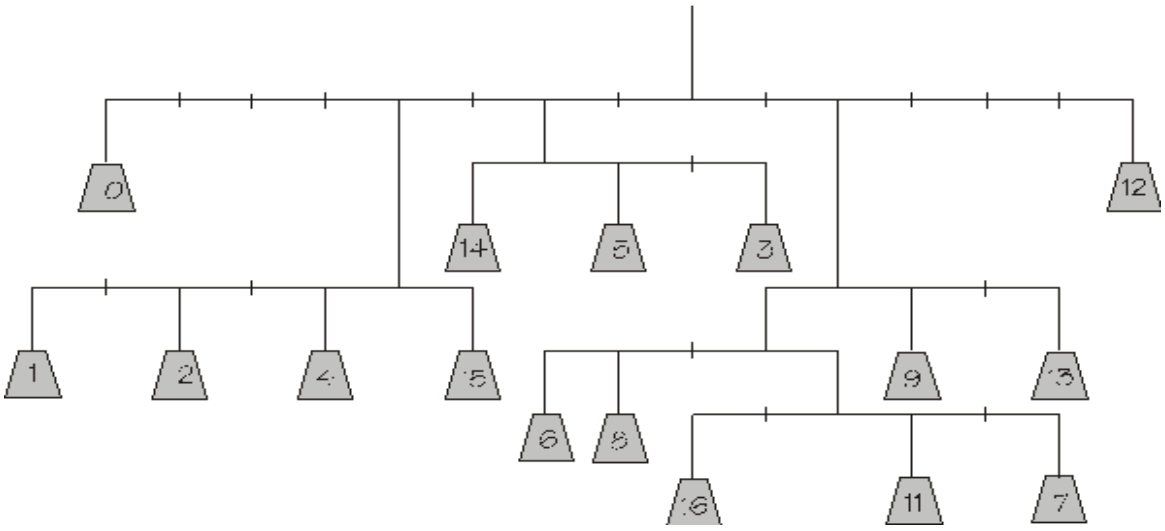
Ví dụ:

MOBILE.IN	MOBILE.OUT
(-3,-1,2(-2,-1,1))	(3,5,(1,2,4))

Một số ví dụ:



MOBILE.IN	MOBILE.OUT
(-3(-1(-1(-1,1,2),3),2),3(-2,1,2),6(-2,3))	((((8,6,1),5),10),(9,4,7),(3,2))



MOBILE.IN	MOBILE.OUT
(-8,-4(-5,-3,-1,1),-2(-1,1,3),2(-1(-3,-2,1(-2,1,3)),1,3),6)	(10,(1,2,4,15),(14,5,3),((6,8,(16,11,7)),9,13),12)

Chọn môn học

Sinh viên theo học các trường đại học theo tín chỉ thường rất bối rối bởi các quy tắc phức tạp về hoàn thành chương trình học. Việc hoàn thành chương trình học có một số đòi hỏi kiến thức về một một số lĩnh vực nhất định. Mỗi một đòi hỏi có thể được đáp ứng bởi nhiều môn học. Một môn học lại có thể đáp ứng được đồng thời nhiều đòi hỏi khác nhau. Các quy tắc này thường được phổ biến cho sinh viên ngay từ khi họ vào trường. Thông thường mỗi sinh viên đều muốn phải theo học ít môn nhất mà vẫn đáp ứng được các đòi hỏi về hoàn thành chương trình học.

Có M môn học được đánh số từ 1 đến M. Có N đòi hỏi được đánh số từ 1 đến N. Với mỗi môn học cho biết danh sách các đòi hỏi được thoả mãn bởi nó. Bạn cần viết chương trình tìm ra một số ít nhất các môn học mà một sinh viên cần học để có thể hoàn thành chương trình học (nghĩa là các môn được chọn đáp ứng toàn bộ N đòi hỏi).

Dữ liệu. Vào từ file văn bản LAZY.IN:

- Dòng đầu tiên chứa hai số nguyên dương M và N ghi cách nhau bởi dấu cách, ($1 \le M \le 200$; $1 \le N \le 100$).
- Dòng thứ i trong số M dòng tiếp theo ghi N số nguyên phân cách nhau bởi dấu cách, số thứ j sẽ là 1 nếu môn học i đáp ứng được đòi hỏi j và sẽ là 0 nếu ngược lại, i = 1, 2, ..., M.

Kết quả. Ghi ra file LAZY.OUT:

- Dòng đầu tiến ghi số lượng môn cần theo học;
- Dòng tiếp theo chứa chỉ số của các môn đó.

Ví dụ:

LAZY.IN	LAZY.OUT
4 3 0 1 0 1 1 0 0 1 1 0 0 1	2 2 3

Đôi bạn

Tuấn và Mai là đôi bạn rất thân. Cứ vào 4 giờ chiều thứ bảy mỗi tuần, Mai đi học đàn ở Câu lạc bộ còn Tuấn thì đi đá bóng ở một Sân vận động. Cả hai đều đi đến đích của mình theo đường đi ngắn nhất. Cho biết sơ đồ giao thông của thành phố gồm N nút giao thông được đánh số từ 1 đến N và M tuyến đường phố (mỗi đường phố nối hai nút giao thông). Vị trí ở của Mai và Tuấn cũng như vị trí của Câu lạc bộ và Sân vận động đều nằm ở các nút giao thông. Cần xác định xem Mai và Tuấn có cách đi nào thoả mãn yêu cầu nêu ở trên đồng thời họ lại có thể gặp nhau tại một nút giao thông trên đường đi đến đích hay không? (Ta nói: Tuấn và Mai có thể gặp nhau ở một nút giao thông nào đó nếu họ đến nút giao thông này tại cùng một thời điểm).

Dữ liệu: Vào từ file văn bản FRIEND.IN:

- Dòng đầu tiên chứa số 2 số nguyên dương N, M ($1 \le N \le 200$);
- Dòng tiếp theo chứa 4 số nguyên dương Sa, Ta, Sb,Tb trong đó Sa là vị trí nhà Tuấn, Ta là vị trí sân vận động, Sb là vị trí nhà Mai, Tb là vị trí Câu lạc bộ.
- Dòng thứ i trong số M dòng tiếp theo chứa ba số nguyên dương A, B, T, trong đó A và B là hai đầu của tuyến đường phố i, còn T là thời gian (tính bằng phút) cần thiết để cả Tuấn và Mai đi từ A đến B cũng như từ B đến A.

Giả thiết là sơ đồ giao thông của thành phố đảm bảo có thể đi từ một nút giao thông bất kỳ đến tất cả các nút còn lại.

Kết quả: Ghi ra file văn bản FRIEND.OUT: Thời điểm sớm nhất (tính bằng phút) kể từ 4 giờ chiều mà Tuấn và Mai có thể gặp nhau, hoặc ghi số -1 nếu họ không thể gặp nhau.

Ví dụ:

FRIEND.IN	FRIEND.OUT	FRIEND.IN	FRIEND.OUT
7 9 1 4 7 6 1 2 10 2 3 10 3 4 10 4 5 15 5 1 15 1 6 10 2 7 5 5 7 15 5 6 10	15	2 1 1 2 2 1 1 2 10	-1

Làm quen

Có n người đánh số từ 1 đến n. Mỗi người i cần phải làm quen với d_i người khác (i=1,2,...,n). Tìm cách làm quen sao cho có thể phân họ ra làm hai nhóm: một nhóm gồm những người đôi một quen nhau, còn một nhóm gồm những người đôi một không quen nhau và đồng thời số thành viên trong nhóm thứ nhất là lớn nhất.

Dữ liệu: Vào từ file văn bản QUEN.INP:

- Dòng đầu tiên chứa số nguyên dương n ($n \le 1000$);
- Dòng thứ hai chứa các số nguyên dương d_1, ..., d_n.

Kết quả: Ghi ra một dòng của file văn bản QUEN.OUT thông báo "NO SOLUTION" nếu không có cách làm quen thoả mãn yêu cầu. Ngược lại cần ghi cách làm quen tìm được: Mỗi dòng ghi 2 số thứ tự của hai người cần làm quen với nhau.

Ví dụ:

QUEN.INP	QUEN.OUT
4 3 2 2 1	1 2 1 3 1 4 2 3

QUEN.INP	QUEN.OUT
8 5 5 5 4 1 2 2 2	1 2 1 3 1 4 2 3 2 4 3 4 2 8 3 8 3 5 1 6 4 6 1 7 2 7

Dãy con chung dài nhất

Cho ba dãy ký tự $A = a_1a_2 ... a_m$, $B = b_1b_2 ... b_n$, $C = c_1c_2 ... c_p$. Hãy tìm dãy các ký tự D có nhiều ký tự nhất là dãy con của cả ba dãy nói trên. (Ta nói dãy D là dãy con của dãy A nếu có thể xoá bớt một số ký tự nào đó trong A để thu được dãy D).

Dữ liệu: Vào từ file văn bản LCS.INP gồm ba dòng, mỗi dòng chứa một trong ba xâu A, B, C đã cho. Giả thiết là độ dài của mỗi dãy không vượt quá 100.

Kết quả: Ghi ra file văn bản LCS.OUT:

- Dòng đầu tiên ghi độ dài của xâu D tìm được;
- Dòng thứ hai chứa xâu D.

Ví dụ:

LCS.INP	LCS.OUT
aaaaaabbbbbccccc aaebbbdccccc ddffaaghbbijkccc	7 aabbccc

Lập lịch thực hiện các công việc trên hai máy

Có n công việc đánh số từ 1 đến n và 2 máy thực hiện đánh số là 1 và 2. Mỗi công việc j cần được bố trí thực hiện trên máy i với thời gian là p_{ji} , j = 1, 2, ..., n; i = 1, 2, (nếu công việc j không cần được thực hiện trên máy i thì đặt $p_{ji} = 0$). Việc thực hiện các công việc trên máy được tiến hành liên tục, không cho phép ngắt quãng. Tại mỗi thời điểm mỗi máy thực hiện không quá 1 công việc và một công việc được thực hiện trên không quá một máy. Giả thiết rằng:

- Thời điểm bắt đầu thực hiện các công việc là 0.
- Thời gian cần thiết để máy chuyển từ việc thực hiện công việc này sang thực hiện công việc khác là 0.

Cần tìm lịch thực hiện các công việc sao cho thời điểm hoàn thành việc thực hiện các công việc là sớm nhất.

Dữ liệu: Vào từ file văn bản OSS.INP:

- Dòng đầu tiên chứa số nguyên dương n ($n \le 100$);
- Dòng thứ hai chứa các số nguyên không âm p_{11} p_{12} ... p_{1n};
- Dòng thứ hai chứa các số nguyên không âm p_{21} p_{22} ... p_{2n}.

Kết quả: Ghi ra file văn bản OSS.OUT:

- Dòng đầu tiên ghi thời điểm hoàn thành theo lịch tìm được;
- Dòng thứ hai ghi trình tự thực hiện các công việc trên máy 1;
- Dòng thứ ba ghi trình tự thực hiện các công việc trên máy 2.

Ví dụ:

OSS.INP	OSS.OUT
4 10 5 4 11 9 8 7 12	36 4 1 3 2 1 2 4 3

Polyblock

Giả sử P = (x_P, y_P) và Q = (x_Q, y_Q) là hai điểm trên mặt phẳng. Ta nói P là trội hơn Q nếu $x_P \geq x_Q$ và $y_P \geq y_Q$. Cho tập X gồm n điểm trên mặt phẳng X = {M_i = (x_i, y_i): i = 1, ... , n}. Một điểm M $\in$ X được gọi là đỉnh của X nếu không tìm được điểm nào trong X\{M} là trội hơn M. Yêu cầu: Xác định các đỉnh của X.

Dữ liệu: Vào từ file văn bản PBLOCK.INP:

- Dòng đầu tiên chứa số nguyên dương n (n ≤ 30000);
- Dòng thứ i trong số n dòng tiếp theo chứa toạ độ x_i, y_i (là các số nguyên dương) của điểm M_i, i = 1, ... , n.

Kết quả: Ghi ra file văn bản PBLOCK.OUT:

- Dòng đầu tiên ghi k là số lượng đỉnh của tập X;
- k dòng tiếp theo mỗi dòng ghi chỉ số của phần tử trong tập X là đỉnh của nó.

Ví dụ:

PBLOCK.INP	PBLOCK.OUT
5 1 4 1 2 2 1 4 2 3 3	3 1 4 5

Khôi phục kết quả

Kết quả của một chương trình là các hệ số $a_1, ..., a_{n+1}$ và x là nghiệm của phương trình bậc n:

$$a_1 x^n + a_2 x^{n-1} + ... + a_n x + a_{n+1} = 0 \qquad (*)$$

($|a_i| \le 1000$, $|x| \le 10$; $1 \le n \le 5$ là các số nguyên). Các số này cần được đưa ra file kết quả theo thứ tự trên (đầu tiên là các hệ số, tiếp theo là nghiệm), các số được ghi cách nhau bởi dấu cách. Rất đáng tiếc là người lập trình do sơ suất đã quên đặt các dấu cách vào các chỗ cần thiết của các câu lệnh writeln. Vì vậy, một kết quả mất nhiều công tìm kiếm đã biến thành một bộ số vô nghĩa.

Yêu cầu: Bạn hãy viết chương trình tìm lại kết quả từ bộ số vô nghĩa này.

Dữ liệu: Vào từ file văn bản SOLPOL.INP:

- Dòng đầu tiên chứa số n;
- Dòng thứ hai chứa bộ số gồm các hệ số của phương trình (*) kể từ số mũ lớn đến số mũ nhỏ, cuối cùng là nghiệm x của nó, các số được viết liền nhau.

Kết quả: Ghi ra file SOLPOL.OUT, mỗi dòng ghi một số theo thứ tự: $a_1, a_2, ..., a_{n+1}, x$.

Ví dụ:

SOLPOL.INP	SOLPOL.OUT
2 12-153	1 2 -15 3

Trình tự xuất hàng

Một thủ kho đã sắp xếp các loại hàng hoá có trong kho theo thứ tự từ điển của các mã hàng hoá. Mỗi loại hàng hoá đều có mã bắt đầu từ chữ cái và chúng được cất giữ trong nhà kho có tên gọi là chữ cái này. Mỗi ngày một thủ kho nhận được một tệp lệnh xuất kho, mỗi lệnh xuất kho chỉ đòi hỏi xuất một loại hàng. Thủ kho phải thực hiện các lệnh xuất kho theo trình tự thời gian xuất hiện của chúng.

Bạn được biết trước tất cả các lệnh xuất kho phải thực hiện bởi thủ kho trong ngày, nhưng không được biết trình tự thời gian xuất hiện của chúng. Hãy liệt kê tất cả các trình tự thực hiện các lệnh xuất kho đó của người thủ kho.

Dữ liệu: Vào từ file ORDERS.IN gồm 1 dòng chứa các nhãn của các hàng hoá được yêu cầu xuất trong ngày. Mỗi loại hàng được biểu diễn bởi chữ cái đầu tiên trong mã của nó. Chỉ sử dụng chữ cái tiếng Anh thường. Số lượng phiếu xuất không quá 200.

Kết quả: Ghi ra file ORDER.OUT tất cả trình tự thực hiện việc xuất hàng của người thủ kho. Mỗi trình tự được biểu diễn bởi dãy tên các nhà kho mà người thủ kho phải lần lượt đi qua. Tên mỗi nhà kho là chữ cái tiếng Anh in thường giống như chữ cái bắt đầu của mã loại hàng chứa trong nó. Mỗi trình tự được ghi trên một dòng. Các trình tự phải được đưa ra theo thứ tự từ điển. Không có file kết quả nào dài quá 2 megabyte.

Ví dụ:

ORDERS.IN	ORDERS.OUT
bbjd	bbdj bbjd bdbj bdjb bjbd bjdb dbbj dbjb djbb jbbd jbdb jdbb

Trò chơi chẵn lẻ

Mời bạn tham gia trò chơi sau đây với một người bạn. Người bạn viết sẵn một dãy số chỉ gồm các số 0 và 1. Bạn chọn một dãy con liên tiếp (chẳng hạn từ số hạng thứ ba đến số hạng thứ năm) và hỏi người bạn xem số lượng số một trong dãy con của dãy số của người bạn đã viết là một số chẵn hay số lẻ. Người bạn phải trả lời câu hỏi này. Bạn lại có thể tiếp tục ra câu hỏi về một dãy con khác, v.v... Bằng cách hỏi như vậy bạn cần đoán ra số của người bạn nghĩ.

Bạn có thể nghi ngờ là bạn mình đưa ra các câu trả lời không đúng, và muốn vạch ra sự không hợp lý của các câu trả lời của người bạn. Vì thế hãy viết chương trình để giúp cho việc phát hiện tính mâu thuẫn trong các câu trả lời của người bạn. Chương trình cần nhận vào các câu hỏi của bạn và các câu trả lời của người bạn, và phải chỉ ra vị trí đầu tiên của câu trả lời mâu thuẫn, nghĩa là luôn tìm được dãy số thoả mãn tất cả các câu trả lời trước đó, nhưng không có dãy số như vậy thoả mãn câu trả lời này.

Dữ liệu: Dòng đầu tiên của file PARITY.IN chứa số nguyên dương là độ dài của dãy số gồm các số 0 và 1. Độ dài này không vượt quá 10^9 . Dòng thứ hai chứa số nguyên dương K (K <= 5000) là số lượng câu hỏi và câu trả lời cho chúng. Dòng thứ i trong số K dòng tiếp theo ghi nội dung câu hỏi thứ i và câu trả lời cho nó bắt đầu bởi con số theo thứ tự là vị trí bắt đầu và vị trí kết thúc của dãy con cần hỏi, tiếp đến hoặc là từ 'even' hoặc là từ 'odd' cho biết tính chẵn hoặc lẻ cuả số lượng số một trong dãy con được hỏi đó, i = 1, 2,..., n.

Kết quả: Ghi ra một dòng của file văn bản PARITY.OUT số nguyên X. Số nguyên X cho biết là tồn tại dãy số gồm các số 0 và 1 thoả mãn X điều kiện chẵn lẻ đầu tiên, nhưng không tồn tại dãy số như vậy thoả mãn X+1 điều kiện đầu tiên. Nếu tìm được dãy số thoả mãn tất cả các điều kiện đã cho, thì X là số lượng tất cả các câu hỏi.

Ví dụ 1:

PARITY.IN

```
10
5
1 2 even
3 4 odd
5 6 even
1 6 even
7 10 odd
```

PARITY.OUT

```
3
```

Ví dụ 2:

PARITY.IN

```
10
5
1 2 even
1 4 even
2 4 odd
1 10 even
3 10 even
```

PARITY.OUT

```
5
```

Chuyến tham quan

Một chi nhánh dịch vụ du lịch ở một thành phố nọ quyết định mời chào các khách hàng của mình ngoài những dịch vụ hấp dẫn khác, một chuyến tham quan miễn phí thành phố. Để có thể thu được càng nhiều lợi nhuận càng tốt, chi nhánh đưa ra quyết định tinh ranh sau: Cần tìm một hành trình ngắn nhất bắt đầu và kết thúc tại cùng một địa điểm nào đó. Nhiệm vụ của bạn là viết chương trình tìm ra hành trình như vậy.

Trong thành phố có N nút giao thông đánh số từ 1 đến N và M đoạn đường phố hai chiều đánh số từ 1 đến M. Hai nút giao thông có thể được nối với nhau bởi một vài đoạn đương phố, nhưng không có đoạn đường phố nào nối một nút giao thông với chính nó. Mỗi một hành trình tham quan là một dãy các chỉ số các đoạn đường phố y_1, ..., y_k , k > 2. Đoạn đường phố y_i (1 < = i < = k-1) nối hai nút giao thông x_i và x_{i+1}, đoạn đường phố y_k nối hai nút giao thông x_k và x_1. Các chỉ số x_1, ..., x_k khác nhau từng đôi.

Độ dài của hành trình tham quan là tổng độ dài của các đoạn đương phố trên hành trình, tức là số $L(y_1) + L(y_2) +...+ L(y_k)$, trong đó $L(y_i)$ là độ dài của đoạn đường phố y_i (1 <= i <= k). Chương trình của bạn cần tìm hành trình tham quan có độ dài nhỏ nhất, hoặc thông báo rằng không thể tìm được, bởi vì không tồn tại hành trình tham quản trong thành phố.

Dữ liệu: Dòng đầu tiên trong file TRIP.IN chứa hai số nguyên dương: số nút giao thông N <= 100 và số đoạn đường phố M <= 10000. Mỗi một trong số M dòng tiếp theo mô tả một đoạn đương phố bao gồm 3 số nguyên dương: chỉ số của nút giao thông đầu, chỉ số của nút giao thông cuối và độ dài của đoạn đường phố (độ dài là số nguyên không vượt quá 500).

Kết quả: Ghi ra một dòng của file TRIP.OUT dòng thông báo 'No solution.' nếu không tồn tại hành trình tham quan, hoặc chỉ số của các nút giao thông trên hành trình tham quan ngắn nhất theo thứ tự hành trình đi qua chúng (tức là các số từ x_1 đến x_k theo định nghĩa hành trình tham quan đã nêu ở trên).

Ví dụ 1:

TRIP.IN

```
5 7
1 4 1
1 3 300
3 1 10
1 2 16
2 3 100
2 5 15
5 3 20
```

TRIP.OUT (một trong các lời giải đúng)

```
1 3 5 2
```

Ví dụ 2:

TRIP.IN

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

TRIP.OUT (duy nhất một lời giải đúng)

```
No solution.
```

Trò chơi trên bàn cờ

Trên bàn cờ kích thước 4x4 có 8 viên đá đen và 8 viên đá trắng, mỗi viên ở một ô của bàn cờ. Một hiện trạng của các viên đá trên bàn cờ như vậy được gọi là một trạng thái của trò chơi. Hai viên đá được gọi là kề nhau nếu chúng nằm ở hai ô có chung cạnh. Điều đó có nghĩa là mỗi viên đá có nhiều nhất là 4 viên kề với nó. Một nước đi đúng luật của trò chơi là đổi chỗ hai viên đá kề nhau nào đó. Bạn cần viết chương trình tìm một dãy ngắn nhất các nước đi đúng luật để chuyển trạng thái ban đầu cho trước của trò chơi về trạng thái đích.

Dữ liệu: Trạng thái ban đầu của trò chơi được mô tả trong 4 dòng của file GAME.IN. Mỗi dòng có 4 ký tự xác định màu của các viên đá đặt trong dòng kể từ trái qua phải. Các dòng mô tả các dòng của bàn cờ theo thứ tự từ trên xuống dưới. Ký tự '0' có nghĩa là viên đá màu trắng, ký tự '1' là viên đá màu đen. Các ký tự được ghi liền nhau. Tiếp đến là một dòng rỗng. 4 dòng tiếp theo mô tả trạng thái đích theo cùng qui cách trên.

Kết quả: Dòng đầu tiên của file kết quả GAME.OUT chứa số lượng nước đi cần thực hiện. Các dòng tiếp theo mô tả dãy các nước đi cần thực hiện trong quá trình chơi. Mỗi dòng mô tả một nước đi và bao gồm 4 số nguyên dương R_1 C_1 R_2 C_2 cách nhau bởi một dấu cách. Đó chính là toạ độ của các ô kề nhau của nước đi, tức là ô [R_1,C_1] và [R_2,C_2], trong đó R_1 (hay R_2) là chỉ số của dòng và C_1 (hay C_2) là chỉ số của cột. Các dòng trên bàn cờ được đánh số từ 1 (dòng trên cùng) đến 4 (dòng dưới cùng), còn các cột được đánh số từ 1 (cột trái nhất) đến 4 (cột phải nhất). Nghĩa là ô ở góc trên bên trái của bàn cờ có toạ độ là [1,1]. Nếu có nhiều lời giải bạn chỉ cần đưa ra một.

Ví dụ:

GAME.IN

```
1111
0000
1110
0010

1010
0101
1010
0101
```

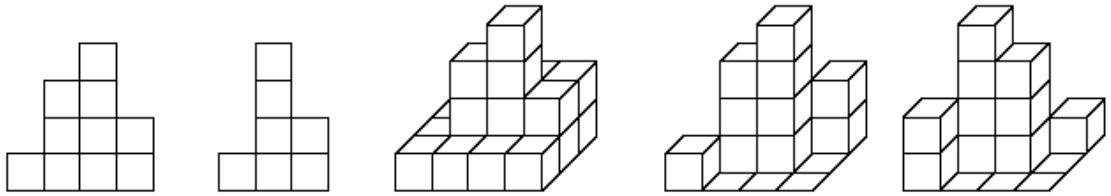
GAME.OUT (một trong các lời giải đúng)

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

Tháp gạch

Trẻ em rất yêu thích trò chơi xếp gạch (là các viên gỗ có dạng hình lập phương). Các em thường xây dựng các toà tháp cao, nhưng cậu bé Johny lại ước mơ về một dự án thiết kế hoàn toàn khác. Cậu ta lại định xây một toà tháp rộng. Ông của cậu bé phải mua cho cậu một bảng hình chữ nhật có chiều rộng là K viên gạch và chiều dài là L viên. Johny quyết định vạch ra một bản thiết kế của một toà tháp như vậy trước khi xây dựng nó. Cậu đã vẽ một lưới có dạng hình vuông trên bảng chứa KxL ô vuông. Cậu muốn đặt các tháp gồm một hoặc nhiều viên gạch lên trên một số ô vuông của lưới đã vẽ; các ô vuông còn lại sẽ được để trống. Vì bảng đã cho là quá rộng, Johny chưa xác định được chính xác cần xếp chồng bao nhiêu viên gạch lên mỗi ô vuông. Cậu chỉ muốn quyết định về mặt trước và bên phải của toà tháp. Cậu vẽ hai mặt chiếu (hình chiếu lên mặt phẳng của tháp) này lên một tờ giấy. Bạn có thể xem ví dụ về các bản vẽ này và hình phối cảnh của toà tháp từ các viên gạch trong hình vẽ sau:

Mặt trước Mặt phải Nhiều gạch nhất ít gạch nhất (mặt trước và mặt sau)



Ông của Johny lo lắng là không có đủ gạch để hoàn thành việc xây dựng tháp do Johny thiết kế.

Bạn cần viết chương trình tính lượng gạch tối thiểu và tối đa cần thiết để xây dựng toà tháp theo thiết kế của Johny. Hơn thế nữa chương trình còn phải cho biết có thể xây dựng tháp thoả mãn các bản vẽ hai mặt chiếu đã cho hay không.

Dữ liệu: Dòng đầu tiên của file TOWN.IN chứa hai số nguyên dương K, L là chiều rộng và chiều dài (biểu diễn bởi số gạch) của bảng. K, L <= 100000 viên gạch. Các dòng tiếp theo mô tả mặt trước của tháp bao gồm dãy các chiều cao của các ngôi nhà nhìn thấy được trên mỗi ô vuông được liệt kê theo thứ tự từ trái sang phải (các chiều cao cũng được đo bằng số lượng gạch). Mỗi dòng chỉ chứa một số, nghĩa là số dòng mô tả mặt trước của tháp là K - chính là chiều rộng của bảng. Tương tự như vậy, L dòng tiếp theo chứa mô tả mặt bên phải của tháp: Các chiều cao của các tháp gạch được liệt kê theo thứ tự từ mặt trước đến mặt sau. Giả thiết rằng không có tháp nào có chiều cao vượt quá 5000 viên gạch. Số lượng gạch lớn nhất cần thiết để xây dựng tháp không vượt quá 2000000000.

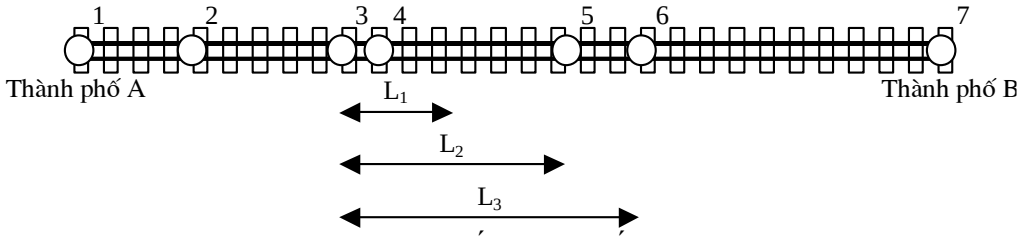
Kết quả: Ghi ra một dòng của file TOWN.OUT. Nếu không thể xây dựng tháp với mặt trước và mặt phải đã cho, thì cần ghi đoạn văn bản 'No solution.' . Ngược lại, ghi hai số nguyên dương theo thứ tự là số lượng gạch tối thiểu và tối đa mà cậu bé Johny cần sử dụng để xây dựng toà tháp của mình theo đúng thiết kế đã vạch ra.

Ví dụ

TOWN.IN	TOWN.OUT	TOWN.IN	TOWN.OUT
4 3 1 3 4 2 1 4 2	10 21	2 2 4 1 1 3	No solution.

Mua vé tàu hoả

Tuyến đường sắt từ thành phố A đến thành phố B đi qua một số nhà ga. Tuyến đường có thể biểu diễn bởi một đoạn thẳng, các nhà ga là các điểm ở trên nó. Tuyến đường bắt đầu từ A và kết thúc ở B, vì thế các nhà ga sẽ được đánh số bắt đầu từ A (có số hiệu là 1) và B là nhà ga cuối cùng.



Hình 1. Tuyến đường sắt A-B

Giá vé đi lại giữa hai nhà ga chỉ phụ thuộc vào khoảng cách giữa chúng. Cách tính giá vé được cho trong bảng sau đây

Khoảng cách giữa hai nhà ga - X	Giá vé
$0<X<=L_1$	C_1
$L_1<X<=L_2$	C_2
$L_2<X<=L_3$	C_3

Vé để đi thẳng từ nhà ga này đến nhà ga khác chỉ có thể đặt mua nếu khoảng cách giữa chúng không vượt quá L_3. Vì thế nhiều khi để đi từ nhà ga này đến nhà ga khác ta phải đặt mua một số vé.

Ví dụ, trên tuyến đường sắt cho trên hình 1 để đi từ ga 2 đến ga 6 không thể mua vé đi thẳng. Có nhiều cách đặt mua vé để đi từ ga 2 đến ga 8. Chẳng hạn, đặt mua vé từ ga 2 đến ga 3 mất chi phí C_2 và sau đó mua vé đi từ ga 3 đến ga 6 mất chi phí C_3, và chi phí tổng cộng để mua vé đi từ ga 2 đến ga 6 theo cách này là $C_2 + C_3$. Lưu ý là mặc dù khoảng cách từ ga 2 đến ga 6 là $2*L_2$ nhưng không thể đặt mua 2 vé với giá C_2 để đi từ ga 2 đến ga 6 với chi phí tổng cộng là $2*C_2$, vì mỗi vé chỉ có giá trị đi lại giữa hai nhà ga nào đó.

Yêu cầu: Tìm cách đặt mua vé để đi lại giữa hai nhà ga cho trước với chi phí mua vé là nhỏ nhất.

Dữ liệu: Vào từ file văn bản RTICKET.INP:

- Dòng đầu tiên ghi các số nguyên L_1, L_2, L_3, C_1, C_2, C_3 ($1 \le L_1 < L_2 < L_3 \le 10^9$, $1 \le C_1 < C_2 < C_3 \le 10^9$) theo đúng thứ tự vừa liệt kê.
- Dòng thứ hai chứa số lượng nhà ga N ($2 \le N \le 10000$).
- Dòng thứ ba ghi hai số nguyên s, t là các chỉ số của hai nhà ga cần tìm cách đặt mua vé với chi phí nhỏ nhất để đi lại giữa chúng.
- Dòng thứ i trong số N-1 dòng tiếp theo ghi số nguyên là khoảng cách từ nhà ga A(ga 1) đến nhà ga thứ i+1 (i = 1, 2, ..., N-1). Chi phí đi từ nhà ga đầu tiên A đến nhà ga cuối cùng B không vượt quá 10^9.

Kết quả: Ghi ra file RTICKET.OUT chi phí nhỏ nhất tìm được.

Ví dụ:

RTICKET.INP	RTICKET.OUT
3 6 8 20 30 40 7 2 6 3 7 8 13 15 23	70

Săn lùng cổ vật trong mê cung

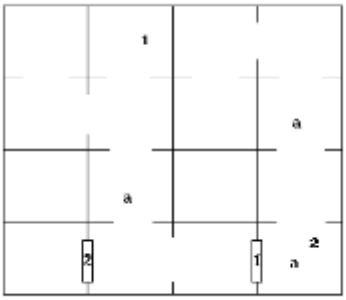
Một mê cung có dạng một lưới ô vuông kích thước n × n (2 ≤ n ≤ 20) và không có lối thoát ra ngoài. Mê cung có n^2 phòng (mỗi phòng tương ứng với một ô vuông của lưới). Các phòng được đánh số từ 1 đến n^2 theo thứ tự từ trái qua phải và từ trên xuống dưới. Giữa hai phòng cạnh nhau (theo chiều ngang hoặc chiều thẳng đứng) có thể là: bức tường đặc, có lối thông, hoặc có cánh cửa bị khoá. Các chìa khoá để mở các cửa bị khoá được cất giấu trong các phòng. Một phòng có thể chứa nhiều chìa khoá. Có k loại chìa khoá khác nhau (k ≤ 9). Chìa khoá loại i (1 ≤ i ≤ k) có thể mở bất cứ khoá nào cùng loại. Sau khi dùng chìa khoá để mở cửa bạn không thể rút được chìa khoá ra nữa nhưng cánh cửa đã mở không bị đóng lại. Số chìa khoá mỗi loại là đúng bằng số cửa có khoá cùng loại. Có m (m ≤ 50) cổ vật được cất giấu trong các phòng của mê cung (trong một phòng có thể có nhiều cổ vật được cất giấu). Một nhà săn lùng cổ vật tại thời điểm bắt đầu ở trong phòng ở góc trên bên trái của mê cung. Ông ta có thể di chuyển sang phòng bên cạnh nếu có lối thông hoặc cánh cửa giữa hai phòng đã được mở. Mỗi khi đi qua phòng nào đó ông ta sẽ lấy toàn bộ cổ vật và chìa khoá trong phòng đó. Nếu có chìa khoá thích hợp ông ta có thể di chuyển qua cửa thông giữa hai phòng. Cuộc săn lùng cổ vật sẽ kết thúc khi nào tất cả cổ vật cuối cùng được nhà săn lùng cổ vật thu nhặt. Mỗi lần nhà săn lùng cổ vật di chuyển từ phòng này sang phòng khác được tính là một bước di chuyển.

Yêu cầu: Tìm cách thu nhặt toàn bộ cổ vật trong mê cung sau số bước di chuyển ít nhất cho người săn lùng cổ vật.

Dữ liệu: Vào từ file ARTIFACT.INP:

- Dòng đầu tiên chứa các số n, k, m;
- n dòng tiếp theo mô tả cấu trúc của mê cung. Dòng thứ i trong số n dòng này mô tả các phòng trong hàng ngang thứ i của mê cung: Mỗi phòng được mô tả bởi 2 ký tự cho biết trạng thái của bức vách ở phía đông và phía nam của phòng theo cách mã hoá sau: W-bức tường đặc, P - có lối thông, chữ số i (1 ≤ i ≤ k) - có cửa với khoá loại i. Các mô tả của các phòng được liệt kê từ trái sang phải và ghi cách nhau bởi dấu phẩy.
- Dòng thứ i trong số k dòng tiếp theo chứa chỉ số các phòng chứa chìa khoá loại i.
- Dòng cuối cùng chứa chỉ số các phòng có cất giấu cổ vật.

Kết quả: Ghi ra file ARTIFACT.OUT: Số bước di chuyển ít nhất cần thực hiện.



Hình 1. Các số trong các phòng cho biết loại chìa khoá cất giấu trong phòng, phòng có chữ a là phòng có cổ vật

Ví dụ: File dữ liệu và file kết quả ứng với sơ đồ mê cung cho trong hình 1:

ARTIFACT.INP	ARTIFACT.OUT
4 2 3 WP,WP,PP,WP PW,WP,WW,WP WW,WP,WW,WP 2W,PW,1W,WW 2 16 8 10 16	10

Xin chữ ký

Giám đốc một công ty trách nhiệm hữu hạn muốn xin chữ ký của ông Kiến trúc sư trưởng thành phố phê duyệt dự án xây dựng trụ sở làm việc của Công ty. Ông kiến trúc sư trưởng chỉ ký vào giấy phép khi bà thư ký của ông ta đã ký duyệt vào giấy phép. Bà thư ký làm việc tại tầng thứ M của toà nhà trụ sở làm việc gồm M tầng của Văn phòng Kiến trúc sư trưởng thành phố. Các tầng của toà nhà được đánh số từ 1 đến M, từ thấp đến cao. Mỗi tầng của toà nhà có N phòng được đánh số từ 1 đến N từ trái qua phải. Trong mỗi phòng chỉ có 1 nhân viên là việc. Giấy phép chỉ được bà thư ký ký duyệt khi có ít nhất một nhân viên ở tầng M đã ký xác nhận. Một nhân viên bất kỳ chỉ ký xác nhận vào giấy phép khi có ít nhất một trong các điều kiện sau được thoả mãn:

a) Nhân viên đó làm việc ở tầng 1;

b) Giấy phép đã được ký xác nhận bởi nhận bởi nhân viên làm việc ở cùng số phòng ở tầng sát dưới;

c) Giấy phép đã được ký xác nhận bởi nhận bởi nhân viên làm việc ở phòng liền kề (hai phòng gọi là liền kề nếu chúng ở cùng tầng và chỉ số phòng sai khác nhau 1).

Mỗi một nhân viên khi ký xác nhận đều đòi một khoản lệ phí. Hãy chỉ ra cách xin được chữ ký của Kiến trúc sư trưởng đòi hỏi tổng lệ phí phải trả cho các nhân viên là nhỏ nhất.

Dữ liệu: Vào từ file văn bản SIGN.INP:

- Dòng đầu tiên chứa hai số M, N ($1 \le M \le 100$; $1 \le N \le 500$);
- Dòng thứ i trong số M dòng tiếp theo chứa N số nguyên dương C_{i1}, C_{i2}, ..., C_{iN} là lệ phí cần trả nhân viên ở các phòng 1, 2, ..., N trên tầng i, (i = 1, 2, ..., M). Giả thiết là $C_{ij} \le 10^9$, i = 1, 2, ..., M; j = 1, 2, ..., N, và tổng chi phí cần trả cũng không vượt quá 10^9.

Kết quả: Ghi ra file văn bản SIGN.OUT:

- Dòng đầu tiên ghi hai số F, K theo thứ tự là chi phí cần trả và số lượng phòng cần đi qua;
- Các dòng tiếp theo ghi chỉ số của các phòng theo thứ tự cần đi qua, mỗi chỉ số ghi trên một dòng.

Ví dụ:

SIGN.INP	SIGN.OUT
3 4 10 10 1 10 2 2 2 10 1 10 10 10	8 5 3 3 2 1 1

Bố trí phòng họp

Có n cuộc họp đánh số từ 1 đến n đăng ký làm việc tạo một phòng hội thảo. Cuộc họp i cần được bắt đầu tại thời điểm s_i và kết thúc tại thời điểm f_i . Hỏi có thể bố trí phòng hội thảo phục vụ được nhiều nhất bao nhiêu cuộc họp sao cho khoảng thời gian làm việc của hai cuộc họp được nhận phục vụ chỉ có thể giao nhau tại đầu mút?

Dữ liệu: Vào từ file văn bản ACTIVITY.INP:

- Dòng đầu tiên chứa số nguyên dương n ($n \le 1000000$);
- Dòng thứ i trong số n dòng tiếp theo chứa hai số nguyên dương s_i, f_i (s_i, $f_i \le 32000$), i = 1, 2, ..., n.

Kết quả: Ghi ra file ACTIVITY.OUT:

- Dòng đầu tiên ghi số K là số lượng cuộc họp được chấp nhận phục vụ;
- Mỗi dòng trong số K dòng tiếp theo ghi chỉ số của một trong K cuộc họp được chấp nhận.

Ví dụ:

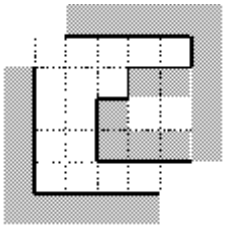
ACTIVITY.INP	ACTIVITY.OUT
5 1 3 2 4 1 6 3 5 7 9	3 1 4 5

Nhà gương cười

Ban quản lý nhà gương cười muốn thay đổi lại toàn bộ các tấm gương phủ tường của nhà gương để phục vụ tốt hơn khách tham quan. Bạn cần giúp ban quản lý tính diện tích gương cần mua để thực hiện công việc đó.

Nhà gương được mô tả bởi bảng ký tự kích thước $N*N$ ($3 <= N <= 33$, (bạn thấy đấy con số '3' quả là thần bí). Một số ô của bảng chứa dấu chấm ('.') để ký hiệu ô trống. Một số ô khác chứa dấu thăng ('#') để ký hiệu ô vuông được bao bọc bởi các bức tường. Tất cả các ô vuông đều có kích thước 3*3 mét.

Người ta xây dựng các bức tường bao bọc chung quanh nhà gương, ngoại trừ ô ở góc trên trái và ô ở góc dưới phải tương ứng với lối vào và ra của nhà gương. Các bức tường cũng được xây dựng chung quanh các ô đánh dấu thăng. Ngoài ra không có bức tường nào khác. Giả thiết rằng ô ở góc trên trái và dưới phải của bảng luôn chứa dấu chấm.



Hình 1. Nhà gương mô tả trong file dữ liệu

Bạn cần tính diện tích của các bức tường ở phía trong của nhà gương, tức là phần nhìn thấy được bởi du khách vào chơi trong nhà gương và cũng chính là diện tích gương cần mua để đổi mới nhà gương. Lưu ý là không có kẽ hở giữa các bức tường để du khách có thể đi hoặc nhìn sang ô có bức tường liền kề. Xem hình vẽ minh họa cho phần bức tường nhìn thấy được từ bên trong nhà gương. Chiều cao của mỗi bức tường đều là 3 mét.

Dữ liệu: Vào từ file MIRROR.INP:

- Dòng đầu tiên chứa số N.
- Dòng thứ i trong số N dòng tiếp theo chứa N ký tự mô tả các ô ở dòng thứ i của bảng ký tự mô tả nhà gương. Các ký tự trên một dòng chỉ là dấu chấm hoặc dấu thăng được ghi liền nhau.

Kết quả: Ghi ra file MIRROR.OUT diện tích của các bức tường mà bạn tìm được.

Ví dụ:

MIRROR.INP	MIRROR.OUT
5 ....## ..#.. ..###	198

Các con hậu chung sống hòa bình

Trên bàn cờ quốc tế kích thước *N*x*N* (*N*<=50) người ta đặt *N* con hậu. Ta nói các con hậu ở trạng thái chung sống hòa bình nếu như không tìm được hai con mà con này có thể tấn công con kia. Bạn cần tìm số lượng trạng thái chung sống hòa bình có thể thu được từ một trạng thái cho trước bằng cách xếp lại vị trí của ***đúng ba*** trong số N con hậu đã xếp.

Dữ liệu: vào từ file văn bản QUEEN.INP:

- Dòng đầu tiên chứa số N;
- N dòng tiếp theo cho biết vị trí N con hậu được xếp trên bàn cờ ở trạng thái chung sống hòa bình. Mỗi dòng chứa 2 số nguyên X, Y ghi cách nhau bởi dấu cách. Các số này cho biết tọa độ theo chiều đứng và theo chiều ngang và nằm trong khoảng từ 1 đến N.

Kết quả: Ghi ra file QUEEN.OUT số lượng trạng thái tìm được.

Chú ý: Các con hậu không được đánh số. Vì vậy nếu bạn chỉ đổi chỗ vòng quanh ba con hậu bất kỳ thì không dẫn đến trạng thái mới.

Ví dụ:

QUEEN.INP	QUEEN.OUT
4 2 1 1 3 3 4 4 2	0

Chuyến tham quan

Một chi nhánh dịch vụ du lịch ở một thành phố nọ quyết định mời chào các khách hàng của mình ngoài những dịch vụ hấp dẫn khác, một chuyến tham quan miễn phí thành phố. Để có thể thu được càng nhiều lợi nhuận càng tốt, chi nhánh đưa ra quyết định tinh ranh sau: Cần tìm một hành trình ngắn nhất bắt đầu và kết thúc tại cùng một địa điểm nào đó. Nhiệm vụ của bạn là viết chương trình tìm ra hành trình như vậy.

Trong thành phố có N nút giao thông đánh số từ 1 đến N và M đoạn đường phố hai chiều đánh số từ 1 đến M. Hai nút giao thông có thể được nối với nhau bởi một vài đoạn đường phố, nhưng không có đoạn đường phố nào nối một nút giao thông với chính nó. Mỗi một hành trình tham quan là một dãy các chỉ số các đoạn đường phố y_1, ..., y_k , k > 2. Đoạn đường phố y_i (1 < = i < = k-1) nối hai nút giao thông x_i và x_{i+1}, đoạn đường phố y_k nối hai nút giao thông x_k và x_1. Các chỉ số x_1, ..., x_k khác nhau từng đôi.

Độ dài của hành trình tham quan là tổng độ dài của các đoạn đường phố trên hành trình, tức là số $L(y_1) + L(y_2) +...+ L(y_k)$, trong đó $L(y_i)$ là độ dài của đoạn đường phố y_i (1 <= i <= k). Chương trình của bạn cần tìm hành trình tham quan có độ dài nhỏ nhất, hoặc thông báo rằng không thể tìm được, bởi vì không tồn tại hành trình tham quan trong thành phố.

Dữ liệu: Dòng đầu tiên trong file TRIP.IN chứa hai số nguyên dương: số nút giao thông N <= 100 và số đoạn đường phố M <= 10000. Mỗi một trong số M dòng tiếp theo mô tả một đoạn đường phố bao gồm 3 số nguyên dương: chỉ số của nút giao thông đầu, chỉ số của nút giao thông cuối và độ dài của đoạn đường phố (độ dài là số nguyên không vượt quá 500).

Kết quả: Ghi ra một dòng của file TRIP.OUT dòng thông báo 'No solution.' nếu không tồn tại hành trình tham quan, hoặc chỉ số của các nút giao thông trên hành trình tham quan ngắn nhất theo thứ tự hành trình đi qua chúng (tức là các số từ x_1 đến x_k theo định nghĩa hành trình tham quan đã nêu ở trên).

Ví dụ:

Ví dụ 1		**Ví dụ 2**	
TRIP.IN	**TRIP.OUT**	**TRIP.IN**	**TRIP.OUT**
5 7 1 4 1 1 3 300 3 1 10 1 2 16 2 3 100 2 5 15 5 3 20	1 3 5 2	4 3 1 2 10 1 3 20 1 4 30	No solution.

Hệ thống đường cao tốc

Hệ thống đường cao tốc hiện tại ở thành phố A mới đảm bảo đi lại giữa một số nút giao thông trọng điểm và còn nhiều nút giao thông trọng điểm chưa có đường cao tốc qua nó. Để giải toả tình trạng ách tắc giao thông trong thành phố, chính quyền thành phố quyết định phát triển hệ thống đường cao tốc của thành phố sao cho có thể đi lại giữa hai nút giao thông trọng điểm bất kỳ. Có N nút giao thông trọng điểm, được đánh số từ 1 đến N. Nút giao thông i được cho bởi toạ độ (x_i, y_i) trong hệ toạ độ Đề các. Mỗi tuyến đường cao tốc nối hai nút giao thông trọng điểm. Tất cả các tuyến đường hiện có cũng như sẽ được phát triển được xây dựng theo đường thẳng nối chúng, vì thế độ dài của mỗi tuyến đường chính là khoảng cách giữa hai điểm tương ứng với hai nút giao thông trọng điểm. Tất cả các tuyến đường cao tốc là hai chiều. Các tuyến đường có thể cắt nhau nhưng người sử dụng phương tiện giao thông chỉ được đổi tuyến đi ở các nút giao thông là đầu mút của các tuyến đường.

Chính quyền thành phố muốn tìm cách xây dựng bổ sung một số tuyến đường cao tốc nối các nút giao thông trọng điểm với chi phí nhỏ nhất đảm bảo sự đi lại giữa mọi nút giao thông trọng điểm. Chi phí xây dựng tỷ lệ thuận với độ dài của tuyến đường, vì vậy bạn có thể tính chi phí xây dựng các tuyến đường bổ sung như là tổng độ dài của các tuyến đường cần xây dựng.

Dữ liệu: Vào từ file văn bản HIGHWAY.IN:

- Dòng đầu tiên chứa số N (N <= 750);
- Dòng thứ i trong số N dòng tiếp theo chứa toạ độ x_i, y_i của nút giao thông trọng điểm i;
- Dòng tiếp theo chứa số M là số tuyến đường cao tốc hiện có (0 <= M <= 1000);
- Dòng thứ j trong số M dòng tiếp theo chứa hai chỉ số của hai nút giao thông là đầu mút của tuyến đường j.

Kết quả: Ghi ra file HIGHWAY.OUT:

- Dòng đầu tiên chứa số K là số lượng tuyến đường cần xây dựng bổ sung;
- Dòng thứ i trong số K dòng tiếp theo chứa hai chỉ số của hai nút giao thông là đầu mút của tuyến đường cần xây dựng bổ sung thứ i.

Ví dụ:

HIGHWAY.IN	HIGHWAY.OUT
9 1 5 0 0 3 2 4 5 5 1 0 4 5 2 1 2 5 3 4 1 3 9 7 1 2 2 3	5 1 6 3 7 4 9 5 7 8 3

Chia hết

Xét một dãy gồm các số nguyên tuỳ ý. Ta có thể đặt các dấu cộng hoặc trừ vào giữa hai số hạng của dãy để thu được các biểu thức số học khác nhau. Chẳng hạn xét dãy số: 17, 5, -21, 15. Có 8 biểu thức khác nhau:

17 + 5 + -21 + 15 = 16
17 + 5 + -21 - 15 = -14
17 + 5 - -21 + 15 = 58
17 + 5 - -21 - 15 = 28
17 - 5 + -21 + 15 = 6
17 - 5 + -21 - 15 = -24
17 - 5 - -21 + 15 = 48
17 - 5 - -21 - 15 = 18

Ta nói dãy số là chia hết cho K nếu có thể đặt các dấu cộng hoặc trừ vào giữa các số hạng của nó để thu được biểu thức số học có giá trị là một số chia hết cho K. Trong ví dụ trên dãy số đã cho là chia hết cho 7 (17+5+-21-15=-14) nhưng nó không chia hết cho 5.

Hãy viết chương trình xác định tính chia hết của một dãy số đã cho.

Dữ liệu: Vào từ file văn bản có tên DIV.IN:
- Dòng đầu tiên chứa hai số nguyên N và K (1 <= N <= 10000, 2 <= K <= 100) ghi cách nhau dấu trắng;
- Các dòng tiếp theo ghi các số hạng của dãy số đã cho, mỗi số hạng của dãy số có giá trị tuyệt đối không quá 10000 được ghi cách nhau bởi dấ trắng hoặc dấu xuống dòng.

Kết quả: Ghi ra file DIV.OUT số 1 nếu dãy đã cho chia hết cho K và số 0 nếu ngược lại

Ghi chú: Bạn chỉ được điểm nếu như trả lời đúng cả câu khẳng định lẫn phủ định.

Ví dụ:

DIV.IN	DIV.OUT
4 7 17 5 -21 15	1

DIV.IN	DIV.OUT
4 5 17 5 -21 15	0

Xây dựng bảng số

Cho một bảng số gồm m dòng n cột ($1 \le m < n \le 100$) có tính chất thú vị sau: "*Mỗi dòng của bảng ghi các số tự nhiên từ 1 đến n sao cho không có cột nào chứa hai số giống nhau*".
Bạn cần tiếp tục bổ sung vào bảng số này k dòng nữa ($k \le n-m$) sao cho bảng thu được vẫn có tính chất giống như bảng ban đầu

Dữ liệu: Vào từ file văn bản LREC.IN:

- Dòng đầu tiên chứa ba số m, n, k;
- Dòng thứ i trong số m dòng tiếp theo chứa các phần tử của dòng thứ i của bảng đã cho.

Kết quả: Ghi ra file văn bản LREC.OUT: Mỗi dòng ghi các phần tử của một trong k dòng bổ sung.

Ví dụ:

LREC.IN	LREC.OUT
2 4 1 1 2 3 4 4 3 2 1	2 1 4 3

San bằng bộ số

Cho bộ gồm k số nguyên không âm (a_1, a_2, ..., a_n). Ta gọi một bước biến đổi đối với bộ số là việc thay nó bởi bộ ($|a_1 - a_2|$, $|a_2 - a_3|$, ... , $|a_n - a_1|$). Bằng cách thực hiện liên tục các phép biến đổi đó ta có thể thu được bộ gồm các số bằng nhau.

Ví dụ: Bắt đầu từ bộ số (0,1,4,11) ta có thể thực hiện biến đổi như sau:

$$(0,1,4,11) \rightarrow (1,3,7,11) \rightarrow (2,4,4,10) \rightarrow (2,0,6,8) \rightarrow (2,6,2,6) \rightarrow (4,4,4,4).$$

Như vậy, sau 5 lần biến đổi ta thu được một bộ số gồm toàn các số giống nhau mà ta sẽ gọi là bộ số san bằng. Bộ số ở ví dụ này có phần tử lớn nhất là 11. Ta có thể bắt đầu từ bộ gồm 4 số với phần tử lớn nhất trong bộ có giá trị nhỏ hơn, nhưng vẫn đòi hỏi 5 phép biến đổi để san bằng. Chẳng hạn:

$$(0,0,1,3) \rightarrow (0,1,2,3) \rightarrow (1,1,1,3) \rightarrow (0,0,2,2) \rightarrow (0,2,0,2) \rightarrow (2,2,2,2).$$

Bộ số này gồm 4 số nguyên không âm, mỗi số đều không vượt quá 3 và đòi hỏi 5 phép biến đổi để san bằng. Dễ thấy, nếu như mọi phần tử của bộ 4 số đều nhỏ hơn 3 thì bộ số san bằng thu được sau ít hơn 5 phép biến đổi.

Yêu cầu: Cho trước số nguyên dương n và k , hãy tìm bộ gồm k số với phần tử lớn nhất là nhỏ nhất đòi hỏi k phép biến đổi để san bằng.

Dữ liệu: Vào từ file văn bản EQUALIZE.IN: chứa số nguyên dương n, k (n, k ≤ 10).

Kết quả: Ghi ra trên một dòng của file văn bản EQUALIZE.OUT: n số của bộ số tìm được

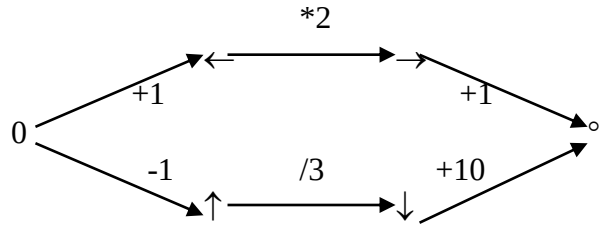
Ví dụ:

EQUALIZE.IN	EQUALIZE.OUT
4 5	0 0 1 3

EQUALIZE.IN	EQUALIZE.OUT
4 3	1 0 0 0

Trò chơi số học

Xét trò chơi bao gồm n phòng và m hành lang nối các phòng. Các phòng được đánh số từ 0 đến n-1. Phòng 0 là điểm bắt đầu của trò chơi, còn phòng n-1 là phòng kết thúc cần đến. Mỗi hành lang được đặt tương ứng với một phép toán số học và một số. Các phép toán số học là cộng (+), trừ (-), nhân (*), chia nguyên (/). Phép chia ở đây chỉ lấy thương số nguyên, ví dụ 3/2 = 1. Thoạt tiên bạn không có một đồng xu dính túi. Mỗi lần di chuyển qua một hành lang bạn phải thực phép toán số học gán cho hành lang này với giá trị số của nó.
Ví dụ: Xét sơ đồ trò chơi cho trong hình vẽ sau:



Đường đi $0 \to 1 \to 3 \to 5$ dẫn đến dãy các phép toán phải làm với túi tiền của bạn sau đây: ((0 + 1) * 2) + 1 = 3. Đường đi của bạn có thể qua một phòng nhiều lần, nhưng chỉ được qua mỗi hành lang không quá một lần. Lưu ý rằng các hành lang chỉ cho phép đi qua theo một chiều nhất định.
Yêu cầu: Tìm đường đi từ phòng 0 đến phòng n-1 sao cho số lượng tiền trong túi bạn khi kết thúc trò chơi có giá trị lớn nhất. Nếu có nhiều đường đi có cùng kết quả bạn cần tìm đường đi qua ít hành lang hơn.

Dữ liệu: Vào từ file văn bản MAZE.IN:

- Dòng đầu tiên chứa hai số nguyên dương n, m ($n \le 50$, $m \le 100$);
- Dòng thứ i trong số m dòng tiếp theo mô tả hành lang thứ i gồm các thông tin sau: d_i, c_i, $\#b_i$ cho biết hành lang này cho phép đi từ phòng d_i đến phòng c_i và phép toán số học # cùng với giá trị số gán cho hành lang này $999 \ge b_i \ge 0$, trong đó # là một trong bốn ký tự '+',' -', '*', '/'. Giữa các số có khoảng trắng, nhưng giữa phép toán với giá trị số gán cho hành lang không có dấu trắng.

Lưu ý: Có thể có nhiều hơn một hành lang cùng bắt đầu từ một phòng và cùng kết thúc tại một phòng khác. Cũng có cả hành lang nối một phòng với chính nó.

Kết quả: Ghi ra file văn bản MAZE.OUT: Nếu tìm được đường đi từ phòng 0 đến phòng n-1:

- Dòng đầu tiên ghi giá trị của lượng tiền trong túi khi kết thúc trò chơi;
- Dòng thứ hai ghi K là số lượng hành lang trên đường đi tìm được;
- Dòng thứ i trong số K dòng tiếp theo ghi chỉ số của phòng mà hành lang thứ i trên đường đi tìm được dẫn đến và giá trị lượng tiền sau khi qua hành lang này.

Nếu không có đường đi từ phòng 0 đến phòng n-1 thì ghi dòng thông báo "no path.'.

Ví dụ:

MAZE.IN	MAZE.OUT
6 6 0 1 +1 0 2 –1 1 3 *2 2 4 /3 3 5 +1 4 5 +10	10 3 2 –1 4 0 5 10

MAZE.IN	MAZE.OUT
2 2 1 0 +999 0 0 -999	no path.

Khoảng cách giữa hai xâu

Cho hai xâu ký tự S_1 và S_2, mỗi xâu có độ dài không quá 255 ký tự. Cho phép thực hiện các phép biến đổi sau đây đối với xâu ký tự:

1. Thay thế một ký tự nào đó bởi một ký tự khác.
2. Đổi chỗ hai ký tự liền nhau.
3. Chèn vào một ký tự.
4. Xoá bớt một ký tự.

Ta gọi khoảng cách giữa hai xâu S_1 và S_2 là số nhỏ nhất các phép biến đổi nêu trên cần áp dụng đối với xâu S_1 để biến nó thành xâu S_2.

Yêu cầu: Tính khoảng cách giữa hai xâu S_1 và S_2 cho trước.

Ví dụ: Giả sử S_1 = 'Barney', S_2 = 'brawny'. Khoảng cách giữa hai xâu S_1 và S_2 là 4. Dãy các phép biến đổi cần thực hiện là:

Thay ký tự 1 của S_1 ('B') bởi 'b';
Đổi chỗ hai ký tự thứ hai ('a') và thứ ba ('r');
Chèn ký tự 'w' vào sau ký tự thứ ba;
Xoá ký tự thứ năm 'e'.

Dãy phép biến đổi có thể mô tả như sau:

'Barney' → 'barney' → 'braney' → 'brawney' →'brawny'

Dữ liệu: Vào từ file văn bản STREDIT.INP có cấu trúc như sau:

- Dòng đầu tiên chứa hai số nguyên dương m, n (m, n ≤ 1000) theo thứ tự là độ dài của xâu S_1 và S_2 .
- Dòng thứ hai chứa xâu S_1.
- Dòng thứ ba chứa xâu S_2.

Kết quả: Ghi ra file văn bản STREDIT.OUT

- Dòng đầu tiên ghi số lượng phép biến đổi cần sử dụng K.
- Mỗi dòng thứ i trong số K dòng tiếp theo mô tả phép biến đổi được sử dụng ở lần thứ i (i = 1, 2, ..., K): đầu tiên ghi chỉ số của phép biến đổi được sử dụng, tiếp đến:
  - Nếu là phép biến đổi 1 cần chỉ ra vị trí của ký tự cần thay thế trong xâu đang biến đổi và ký tự thay thế;
  - Nếu là phép biến đổi 2 cần chỉ ra vị trí (xếp theo thứ tự tăng dần) của hai ký tự cần đổi chỗ;
  - Nếu là phép biến đổi 3, cần chỉ ra vị trí của ký tự trong xâu đang xét mà sau nó cần chèn một ký tự và ký tự cần chèn.
  - Nếu là phép biến đổi 4, cần chỉ ra vị trí của ký tự cần xoá trong xâu đang xét.

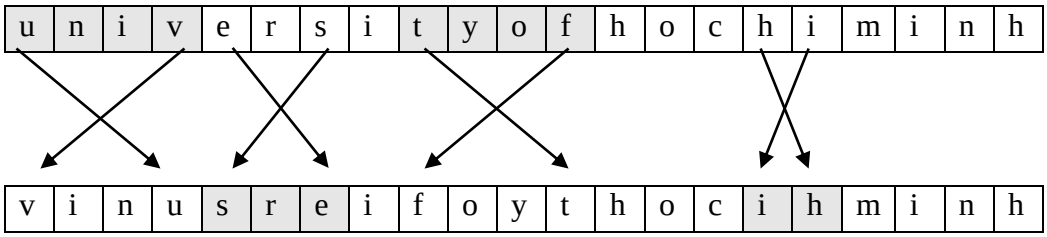
Ví dụ:

STREDIT.INP	STREDIT.OUT
Barney brawny	4 1 1 b 2 2 3 3 3 w 4 6

Đảo ngược xâu

Cho xâu ký tự $X = x_1 x_2 ... x_n$, ta định nghĩa phép đảo ngược xâu con là việc thay thế xâu con $x_i x_{i+1}... x_{j-1} x_j$ bởi xâu đảo ngược của nó $x_j x_{j-1} ... x_{i+1} x_i$. Hai phép đảo ngược xâu con được gọi là độc lập, nếu các xâu con bị biến đổi là không giao nhau.

Ví dụ: Hình vẽ dưới đây mô tả 4 phép đảo ngược xâu con độc lập với nhau:



Yêu cầu: Cho trước hai xâu X và Y có cùng độ dài. Hãy tìm một số ít nhất phép đảo ngược xâu con độc lập để biến đổi xâu X về xâu Y hoặc xác định là không thể thực hiện các phép biến đổi như vậy để biến X thành Y. Giả thiết là độ dài của các xâu không quá 200.

Dữ liệu: Vào từ file văn bản REVSTR.INP có cấu trúc như sau:

- Dòng đầu tiên chứa xâu X.
- Dòng thứ hai chứa xâu Y.

Kết quả: Ghi ra file văn bản REVSTR.OUT

- Dòng đầu tiên ghi số lượng phép biến đảo ngược xâu con độc lập cần sử dụng K. (K=0, nếu không có cách biến đổi). Nếu K>0 thì ghi tiếp:

Mỗi dòng thứ i trong số K dòng tiếp theo mô tả phép đảo ngược xâu con thứ i bao gồm chỉ số bắt đầu và chỉ số kết thúc của xâu con cần đảo ngược.

Ví dụ:

REVSTR.INP	REVSTR.OUT
universityofhochiminh vinusreifoythocihminh	4 1 4 5 7 9 12 16 17

Tìm đường bay

Một máy bay trực thăng làm nhiệm vụ cứu trợ phải thực hiện một phi vụ xuất phát từ sân bay (được đánh số là địa điểm 0) đến thả hàng cứu trợ cho n địa điểm dân cư (các địa điểm dân cư được đánh số từ 1 đến n). Địa điểm i được cho bởi toạ độ (x_i, y_i) trong hệ toạ độ Đề các (i = 0, 1, 2, ..., n). Khi bay từ địa điểm này qua địa điểm khác máy bay luôn bay theo đường thẳng. Thời gian bay từ địa điểm này qua địa điểm khác tỷ lệ thuận với khoảng cách giữa hai địa điểm đó. Vì thế ta có thể coi thời gian bay từ địa điểm này sang địa điểm kia chính bằng khoảng cách giữa chúng.

Máy bay cất cánh từ sân bay phải bay qua tất cả n địa điểm dân cư mỗi địa điểm dân cư đúng một lần rồi lại trở về sân bay.

Cách bay thoả mãn yêu cầu vừa nêu ta gọi là đường bay.

Yêu cầu: Tìm đường bay với tổng thời gian bay là càng nhỏ càng tốt.

Dữ liệu: Vào từ file văn bản FLIGH.IN:

- Dòng đầu tiên chứa số n (n < 750);
- Dòng thứ i trong số n+1 dòng tiếp theo chứa toạ độ x_i, y_i của nút địa điểm i (i = 0, 1, ..., n);

Giả thiết là các toạ độ x_i, y_i là các số nguyên có giá trị tuyệt đối không quá 32000.

Kết quả: Ghi ra file FLIGH.OUT:

- Dòng đầu tiên chứa số K là tổng thời gian bay theo đường bay tìm được (được tính chính xác đến hai chữ số sau dấu phẩy);
- Dòng thứ 2 mô tả đường bay tìm được: ghi chỉ số các địa điểm mà máy bay cần bay qua theo trình tự xuất phát từ sân bay và kết thúc cũng tại sân bay

Ví dụ:

FLIGH.IN	FLIGH.OUT
2 0 0 1 0 0 1	3.14 0 1 2 0

Mạng máy tính

Một hệ thống N máy tính đánh số từ 1 đến N được nối mạng bằng M kênh truyền tin một chiều giữa một số cặp máy. Mạng được gọi là thông suốt nếu như từ một máy u bất kỳ luôn có thể truyền tin đến mỗi máy v trong số các máy còn lại hoặc theo kênh truyền tin từ u đến v, hoặc thông qua một số máy trung gian. Ta gọi một mạng con của mạng đã cho là một mạng gồm một số máy và các kênh nối chúng của mạng đã cho. Trong trường hợp mạng là không thông suốt nó sẽ phân rã ra thành một số mạng con thông suốt. Mạng con thông suốt được gọi là cực đại nếu như không tồn tại một mạng con thông suốt của mạng đã cho chứa nó như là mạng con. Bạn cần xác định số mạng con thông suốt cực đại của mạng đã cho.

Dữ liệu: Vào từ file văn bản MNET.INP:

- Dòng đầu tiên chứa hai số nguyên dương N, M (1<= N <= 1000, 1<= M <= 20000);
- Dòng thứ i trong số M dòng tiếp theo ghi 2 số nguyên dương d_i, c_i cho biết kênh truyền tin thứ i cho phép truyền tin từ máy d_i đến máy c_i.

Kết quả: Ghi ra file MNET.OUT:

- Dòng đầu tiên ghi K là số mạng con thông suốt cực đại của mạng đã cho;
- Nếu K>1 thì mỗi dòng trong số K dòng tiếp theo ghi các chỉ số của các máy của một mạng con thông suốt cực đại.

Ví dụ:

MNET.INP	MNET.OUT
3 3 1 2 2 3 3 1	1

MNET.INP	MNET.OUT
9 14 1 2 1 4 1 7 2 3 2 6 3 1 4 5 5 4 5 7 6 7 7 9 9 6 9 8 8 9	3 1 2 3 4 5 6 7 8

Loại các vectơ thừa

Một vectơ n-chiều là một bộ có thứ tự gồm n số thực $\boldsymbol{x} = (x_1, x_2, ..., x_n)$. Cho $\boldsymbol{x} = (x_1, x_2, ..., x_n)$ và $y = (y_1, y_2, ..., y_n)$ là hai vectơ. Ta nói vectơ x bị phủ bởi vectơ y nếu như $x_i \le y_i$, i = 1, 2, ..., n. Cho một tập S gồm m vectơ. Một vectơ $x \in S$ được gọi là thừa nếu như tìm được vectơ $y \in S$ sao cho x bị phủ bởi y. Nhiệm vụ của bạn là loại bỏ khỏi tập S tất cả các vectơ thừa.

Dữ liệu: Vào từ file văn bản VECTO.IN:
- Dòng đầu tiên chứa hai số nguyên dương n, m ($n \le 20$, $m \le 10000$);
- Dòng thứ i chứa n số thực là các thành phần của vectơ thứ i trong số m vectơ đã cho.

Kết quả: Ghi ra file VECTO.OUT:
- Dòng đầu tiên ghi K là số lượng vectơ thừa cần loại bỏ. (K = 0, nếu tập vectơ đã cho không chứa vectơ thừa).
- Dòng tiếp theo chứa chỉ số của các vectơ cần loại bỏ.

Ví dụ:

VECTO.IN	VECTO.OUT
3 3 1 2 3 2 3 4 3 1 2	1 1

VECTO.IN	VECTO.OUT
4 3 1 2 3 4 5 2 3 1 3 7 2 1	0

Xếp lịch dạy

Trong một trường đại học có M thày giáo đánh số từ 1 đến M và N lớp học đánh số từ 1 đến N. Theo kế hoạch phân công giảng dạy, thày i phải dạy p_{ij} tiết cho lớp j ($0 \le p_{ij} \le 30$). Trong mỗi tiết, mỗi thày không dạy hơn một lớp và mỗi lớp học không hơn một thày. Hãy sắp xếp lịch dạy cho các thày sao cho toàn bộ yêu cầu giảng dạy trên được hoàn thành trong thời gian sớm nhất. Các tiết giảng dạy của lịch được đánh số lần lượt 1, 2, 3,...

Dữ liệu: Vào từ file văn bản TIMETAB.INP:

- Dòng đầu tiên ghi hai số nguyên dương M, N (M, N ≤ 100);
- Dòng thứ i trong số M dòng tiếp theo ghi N số nguyên không âm p_{i1}, p_{i2}, ..., p_{iN}.Các giá trị số trên một dòng được ghi cách nhau bởi ký tự trắng.

Kết quả: Ghi ra file TIMETAB.OUT:

- Dòng đầu tiên ghi thời gian cần thiết để hoàn thành khối lượng giảng dạy theo lịch tìm được K;
- Dòng thứ i trong số K dòng tiếp theo mô tả lịch dạy trong ngày i bao gồm M số L_{1i}, L_{2i}, ..., L_{Mi} , trong đó L_j là chỉ số lớp mà thày j dạy trong tiết i (Qui ước: $L_{ji} = 0$, nếu thày j không được phân dạy trong tiết i), i = 1, 2, ..., K.

Ví dụ:

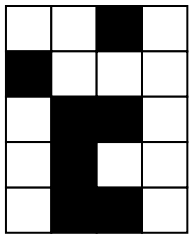
TIMETAB.INP	TIMETAB.OUT
5 4 2 0 0 0 0 1 1 0 1 0 1 0 1 1 1 1 0 0 0 1	4 0 2 3 1 0 1 3 0 2 0 0 0 1 3 4 1 0 0 4 0

Tìm kiếm mẫu

Cho một hình chữ nhật kích thước m × n (m dòng và n cột) được chia thành các ô vuông như mô tả trong hình 1a dưới đây. Mỗi ô vuông được đánh số liên tiếp sao cho ô ở góc trên bên phải được đánh số 1, ô bên phải nó được đánh số 2 và ô ở góc dưới phải được đánh số m×n.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	20

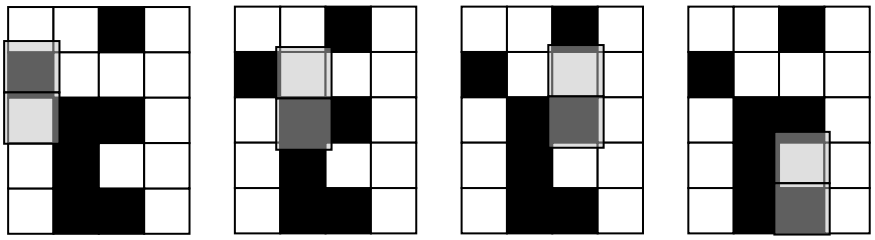
Hình 1a



Hình 1b

Hình 1c

Hình 1b cho thấy lưới ô vuông với một số ô được sơn đen. Xét một hình chữ nhật con với ô ở góc trên trái chính là ô góc trên trái của hình chữ nhật đã cho, còn ô ở góc dưới phải có toạ độ (p, q). Chẳng hạn hình chữ nhật con xác định bởi ô dưới phải (2, 1) của hình chữ nhật cho trong hình 1b có dạng cho trong hình 1c. Hình chữ nhật con này xuất hiện ở 4 vị trí khác nhau trong hình chữ nhật đã cho trong hình 1b (xem các hình dưới đây).



Bạn cần viết chương trình nhập thông tin về hình chữ nhật đã cho và một hình chữ nhật con của nó, sau đó đưa ra số lần xuất hiện của hình chữ nhật con trong hình chữ nhật đã cho. Bạn không cần tính các phép quay của hình chữ nhật con. Lưu ý rằng hình chữ nhật con luôn được chọn ở góc trên trái của hình chữ nhật đã cho.

Dữ liệu: Vào từ file văn bản RPART.INP:

- Dòng đầu tiên chứa bốn số nguyên m ($1 \le m, n \le 20$), p ($1 \le p \le m$) và q ($1 \le q \le n$).
- Dòng thứ hai cho biết số lượng ô được bôi đen x, ($0 \le x \le m.\ n$). Mỗi một trong số x dòng tiếp theo chứa vị trí của một ô được sơn đen (cách đánh số vị trí được mô tả trong hình 1a).

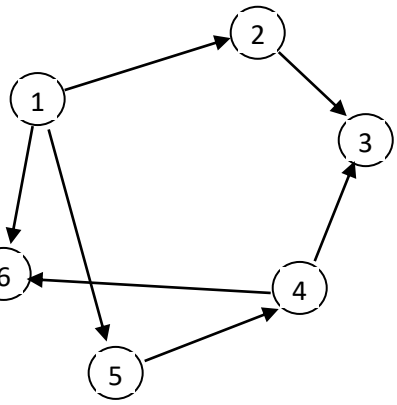
Kết quả: Ghi ra file RPART.OUT số lần xuất hiện của hình chữ nhật con trong hình chữ nhật đã cho (tính cả vị trí ban đầu ở góc trên trái của nó).

Ví dụ:

RPART.INP	RPART.OUT
5 4 2 1 7 3 5 10 11 14 18 19	4

RPART.INP	RPART.OUT
4 5 2 2 2 2 5	2

Hành trình chẵn lẻ



Một mạng giao thông gồm *n* thành phố và các tuyến đường một chiều nối chúng. Các thành phố được đánh số từ 1 đến *n* ($1 \le n \le 100$). Sau mỗi ngày, mỗi tuyến đường của hệ thống đường một chiều này lại đổi chiều, trong các ngày lẻ ta có thể đi theo một chiều nào đó, còn trong các ngày chẵn ta lại chỉ được đi theo chiều ngược lại. Thời gian đi theo một tuyến đường được cho bởi một số nguyên dương (đơn vị thời gian tính bằng giờ).

Cần xác định hành trình nhanh nhất đi từ thành phố A đến thành phố B. Ngày đầu tiên của hành trình được coi là ngày lẻ. Hành trình trong một ngày không được kéo dài quá 12 giờ. Chỉ có thể dừng lại nghỉ đêm ở các thành phố. Hành trình có thể tiếp tục ở ngày kế tiếp.

Dữ liệu: Vào từ file văn bản JOURN.INP:

- Dòng đầu tiên chứa hai số nguyên dương theo thứ tự là chỉ số của hai thành phố A và B.
- Dòng tiếp theo chứa hai số n, k là số thành phố và số tuyến đường giữa các thành phố ($1 \le k \le 1000$).
- Dòng thứ i trong số k dòng tiếp theo chứa ba số nguyên dương theo thứ tự là hai đầu mút và thời gian đi lại của tuyến đường thứ i. Chiều đi lại trong ngày lẻ là hướng đi từ thành phố thứ nhất đến thành phố thứ hai.

Kết quả: Ghi ra file văn bản JOURN.OUT đường đi tìm được theo khuôn dạng sau:

- Dòng đầu tiên ghi độ dài của hành trình tìm được;
- Mỗi dòng chứa thông tin của một chặng đường bao gồm 4 số: thành phố xuất phát, thành phố kết thúc, chỉ số của ngày và độ dài của tuyến đường.

Ví dụ:

JOURN.INP	JOURN.OUT
1 3 6 7 1 2 9 1 6 2 1 5 10 5 4 1 4 6 2 4 3 4 2 3 5	15 1 5 1 10 5 4 1 1 4 3 3 4

JOURN.INP	JOURN.OUT
1 6 6 7 1 2 9 1 6 2 1 5 10 5 4 1 4 6 2 4 3 4 2 3 5	2 1 6 1 2

Di chuyển ghế trong vườn

Trong một khu vườn có dạng một hình vuông độ dài cạnh n được chia ra thành lưới n×n ô vuông có một số cây và một cái ghế băng. Mỗi cây chiếm một ô vuông, còn ghế băng chiếm ba ô vuông liên tiếp. Người ta muốn di chuyển ghế băng từ vị trí ban đầu đến một vị trí mới. Hình 1 dưới đây mô tả khu vườn: Các số 0 mô tả ô rỗng, các số 1 cho biết ô có cây. Vị trí đặt ghế được đánh dấu bởi các chữ 'b', còn vị trí kết thúc được đánh dấu bởi các chữ 'e' (xem hình 2). Giả thiết là luôn có cách di chuyển từ vị trí đầu đến vị trí cuối.

00011	bbb11
00000	00000
00000	00000
11000	11000
00000	eee00
Hình 1	Hình 2

Ta có thể di chuyển ghế sang phải, sang trái, lên trên, xuống dưới, hoặc quay một góc 90 độ nếu như không có cây nằm trên đường di chuyển nó. Các hình 3-8 mô tả các phép di chuyển vừa nêu. Lưu ý là phép quay 90^0 quanh tâm của ghế chỉ có thể thực hiện khi các ô lân cận là rỗng: Trong hình 9 các ô đánh dấu 'x' phải là rỗng mới có thể quay ghế, hình 10 là kết quả của phép quay.

00000	00000	00000	00000	00000	00000	00000	00000
00000	0bbb0	00000	00000	00000	00b00	0xxx0	00b00
0bbb0	00000	00000	00bbb	bbb00	00b00	0bbb0	00b00
00000	00000	0bbb0	00000	00000	00b00	0xxx0	00b00
00000	00000	00000	00000	00000	00000	00000	00000
Hình 3 Ban đầu	Hình 4 Trên	Hình 5 Dưới	Hình 6 Phải	Hình 7 Trái	Hình 8 Quay 90^o	Hình 9 Trước khi quay	Hình 10 Sau khi quay

Cần tìm cách di chuyển với số bước di chuyển là nhỏ nhất.

Dữ liệu: Vào từ file văn bản TMOVE.INP:

- Dòng đầu tiên chứa số *n* là kích thước của vườn ($3 \le n \le 40$).
- *n* dòng tiếp theo mỗi dòng chứa n ký tự mô tả khu vườn. Trong đó có 3 chữ 'b' mô tả vị trí của ghế và 3 chữ 'e' mô tả vị trí mới cần chuyển ghế đến.

Kết quả: Ghi ra file TMOVE.OUT :

- Dòng đầu tiên ghi số bước di chuyển ít nhất cần thực hiện K.
- Dòng thứ 2 ghi K ký tự mô tả các bước di chuyển cần thực hiện. Sử dụng các ký tự L, R, D, U, T theo thứ tự tương ứng để mô tả các phép di chuyển Trái, Phải, Dưới, Trên, Quay.

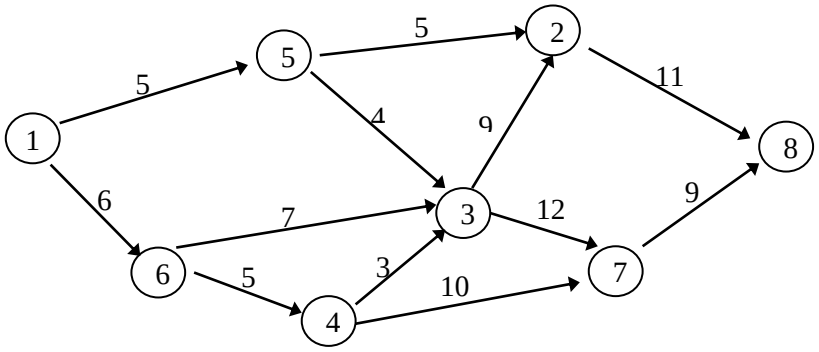
Ví dụ:

TMOVE.INP	TMOVE.OUT
5 bbb11 00000 00000 11000 eee00	8 DDRRDDLL

TMOVE.INP	TMOVE.OUT
6 bbb1e1 0000e0 0000e0 110001 111001 001101	7 DTRDRRU

Cuộc hành quân dã ngoại

Một đoàn học sinh tổ chức hành quân dã ngoại. Có N địa điểm ($2 \le N \le 100$) và thời gian đi lại giữa hai địa điểm (là một số nguyên dương ≤ 100) được thông báo cho mọi học sinh trong đoàn. Các địa điểm được đánh số từ 1 đến N. Sơ đồ dưới đây cho một ví dụ với $N = 8$ địa điểm.



Mỗi học sinh trong đoàn đều cùng xuất phát từ địa điểm 1 và đi theo một đường nào đó đến địa điểm N. Tại mỗi giao lộ, đoàn học sinh lại phải phân ra thành các nhóm, mỗi nhóm đi theo một tuyến đường rẽ nhánh. Chẳng hạn ở địa điểm 1, nhóm được phân thành hai nhóm con, một nhóm đi theo tuyến đương dẫn đến địa điểm 5, còn một nhóm tiến đến địa điểm 6. Nhóm học sinh đến địa điểm 5 lại được chia thành hai nhóm con, một nhóm đi đến địa điểm 2, còn nhóm kia đi đến địa điểm 3. Nghĩa là, tất cả các tuyến đường đều bị khảo sát. Do số lượng tuyến đường là cố định, nên ta giả thiết là tại mỗi giao lộ luôn có đủ học sinh để phân chia thành các nhóm con đòi hỏi để tiếp tục khảo sát.

Để đảm bảo an toàn, nếu một nhóm đến một địa điểm nào đó sớm hơn thì nhóm đó cần phải chờ tất cả các nhóm khác đến đông đủ rồi mới phân chia thành các nhóm con tiếp tục hành trình. Chẳng hạn, giả sử thời điểm bắt đầu cuộc hành quân là 0, nhóm đầu tiên đến địa điểm 3 vào thời điểm 9 (nhóm đi qua địa điểm 5), nhóm này phải chờ hai nhóm đến từ địa điểm 6 và 4. Khi ba nhóm hội đủ, họ mới phân ra thành 2 nhóm tiếp tục đi đến địa điểm 2 và 7.

Chú ý là chỉ có duy nhất một địa điểm xuất phát, đó là địa điểm 1, và chỉ có duy nhất một địa điểm đích là N. Từ địa điểm xuất phát luôn có đường đi đến bất cứ địađiểm nào trong số các địa điểm còn lại, đồng thời từ một địa điểm bất kỳ luôn có đường đi đến địa điểm đích. Giả thiết rằng không có chu trình (nếu không bạn có thể đi quẩn).

Yêu cầu: Cần tính thời điểm sớm nhất T khi nhóm cuối cùng về đến đích N, giả thiết là thời điểm bắt đầu cuộc hành quân là 0. Trong ví dụ ở trên $T = 35$. Nghĩa là, thời điểm sớm nhất để nhóm cuối cùng về đến địa điểm 8 là 35.

Do mỗi nhóm khi đến một địa điểm nào đó phải chờ một số nhóm khác đến đông đủ mới có thể tiếp tục hành trình, nên ở đây xuất hiện thời gian chờ đợi. Ta gọi thời gian chờ đợi của một địa điểm là khoảng thời gian từ thời điểm nhóm đầu tiên đến địa điểm này đến thời điểm mà nhóm cuối cùng đến địa điểm này. Ví dụ, tại địa điểm 3, nhóm đầu tiên (nhóm đi qua 5) đến địa diểm 3 tại thời điểm 9, còn nhóm cuối cùng đến địa điểm 3 tại thời điểm 14 (nhóm đi qua 6 và 4). Do dó, thời gian chờ của địa điểm 3 là 5. Tương tự như vậy, thời gian chờ của địa điểm 2 là 13.

Bạn cần tính tổng thời gian chờ của tất cả các địa điểm. Trong ví dụ đang xét, tổng thời gian chờ đợi là 24.

Tại một số địa điểm, các nhóm học sinh khi đã hội họp đủ cũng không nhất thiết phải tiếp tục hành trình ngay tức khắc. Họ có thể cùng nhau vui chơi một thời gian ở địa điểm này mà vẫn có thể đến đích không muộn hơn thời điểm T. Ví dụ, tại địa điểm 5, các nhóm hội đủ ở đây vào thời

điểm 5, và có thể vui chơi cho đến thời điểm 10 mới xuất phát mà vẫn đến đích 8 không muộn hơn thời điểm 35 - thời điểm sớm nhất mà nhóm cuối cùng đạt đến đích. Tương tự, khi các nhóm đã hội đủ tại địa điểm 2, họ có thể vui chơi trong 1 đơn vị thời gian rồi mới tiếp tục hành trình. Tuy nhiên tại tất cả các địa điểm còn lại, các nhóm khi đã tập hợp đông đủ thì phải ngay lập tức phân nhóm tiếp tục hành trình. Như vậy, trong ví dụ đang xét, chỉ có hai địa điểm mà tại đó các nhóm học sinh có thể nghỉ ngơi trước khi tiếp tục hành trình.

Bạn cần xác định số lượng địa điểm mà tại đó các nhóm học sinh có thể nghỉ ngơi trước khi tiếp tục hành trình (gọi là điểm dừng).

Dữ liệu: Vào từ file văn bản HIKE.INP:

- Dòng đầu tiên chứa số nguyên N ($2 \le N \le 100$) là số địa điểm, và số nguyên M ($1 \le M \le 1000$) là số cung đường;
- Dòng thứ i trong số M dòng tiếp theo chứa 3 số nguyên: địa điểm đầu, địa điểm cuối và thời gian đi từ địa điểm đầu đến địa điểm cuối. Các giá trị số trên một dòng được ghi cách nhau bởi dấu trắng. Thời điểm bắt đầu cuộc hành quân là 0. Địa điểm xuất phát là 1, địa điểm kết thúc là N.

Kết quả: Ghi ra trên một dòng của file văn bản HIKE.OUT 3 số nguyên theo thứ tự là thời điểm sớm nhất mà nhóm cuối cùng có thể đến đích, tổng thời gian chờ của các địa điểm, số lượng điểm dừng.

Ví dụ:

HIKE.INP	HIKE.OUT
8 12 3 7 12 5 2 5 6 3 7 1 6 6 4 7 10 2 8 11 1 5 5 5 3 4 6 4 5 7 8 9 4 3 3 3 2 9	35 24 2

Cho thuê máy

Tại thời điểm 0, ông chủ một máy tính hiệu năng cao nhận được đơn đặt hàng thuê sử dụng máy của n khách hàng. Các khách hàng được đánh số từ 1 đến n. Khách hàng i cần sử dụng máy từ thời điểm d_i đến thời điểm c_i (d_i, c_i là các số nguyên và $0< d_i < c_i \le 1000000000$) và sẽ trả tiền sử dụng máy là p_i (p_i nguyên, $0 < p_i \le 10000000$). Bạn cần xác định xem ông chủ cần nhận phục vụ những khách hàng nào sao cho khoảng thời gian sử dụng máy của hai khách được nhận phục vụ bất kỳ không được giao nhau đồng thời tổng tiền thu được từ việc phục vụ họ là lớn nhất.

Dữ liệu: Vào từ file văn bản RENTING.INP:

- Dòng đầu tiên ghi số n ($0 < n \le 1000$);
- dòng thứ i trong số n dòng tiếp theo ghi ba số d_i, c_i, p_i cách nhau bởi dấu trắng (i = 1, 2, ..., n).

Kết quả: Ghi ra file văn bản RENTING.OUT:

- Dòng đầu tiên ghi hai số nguyên dương theo thứ tự là số lượng khách hàng nhận phục vụ và tổng tiền thu được từ việc phục vụ họ.
- Dòng tiếp theo ghi chỉ số của các khách hàng được nhận phục vụ.

Ví dụ:

RENTING.INP	RENTING.OUT
3 150 500 150 1 200 100 400 800 80	2 180 2 3

RENTING.INP	RENTING.OUT
4 400 821 800 200 513 500 100 325 200 600 900 600	2 1100 2 4

Phần tử ngắn nhất

Xét tập Z^n là tập tất cả các bộ có thứ tự gồm n số nguyên. Giả sử $v = (v_1, v_2, ..., v_n) \in Z^n$, ta gọi độ dài của v là số

$$|v| = \sqrt{\sum_{i=1}^{n} v_i^2}.$$

Các phần tử của n sẽ được gọi là các vectơ. Giả sử α, β là các số nguyên, u, $v \in Z^n$, ta định nghĩa các vectơ

- $u = (\alpha u_1, \alpha u_2, ..., \alpha u_n)$
- $u + \beta v = (\alpha u_1 + \beta v_1, \alpha u_2 + \beta v_2, ..., \alpha u_n + \beta v_n)$.

Cho a và b là hai phần tử của Z^n. Xét tập **L** sinh bởi hai vectơ a và b theo qui tắc:

$$\mathbf{L} = \{\alpha a + \beta b: \alpha, \beta \text{ - số nguyên }\}.$$

Một vectơ được gọi là khác không nếu nó có ít nhất một thành phần khác 0. Cho hai vectơ a và b, cần tìm phần tử khác không của L có độ dài nhỏ nhất. Ta sẽ gọi phần tử như vậy là phần tử ngắn nhất. Chẳng hạn, khi $n = 1$, cả hai vectơ a và b đều là các số nguyên. Trong trường hợp này phần tử khác không có độ dài nhỏ nhất trong L chính là ước chung lớn nhất của a và b.

Dữ liệu: Vào từ file văn bản ESHORT.INP:
- Dòng đầu tiên ghi số nguyên dương n ($n < 20$);
- Dòng thứ hai ghi các thành phần của a;
- Dòng thứ ba ghi các thành phần của b.

Giả thiết là độ dài của các vectơ a và b đều nhỏ hơn 2^{31}.

Kết quả: Ghi ra file ESHORT.OUT:
- Dòng thứ nhất ghi độ dài của phần tử ngắn nhất tìm được;
- Dòng thứ hai ghi các thành phần của phần tử ngắn nhất tìm được.

Ví dụ:

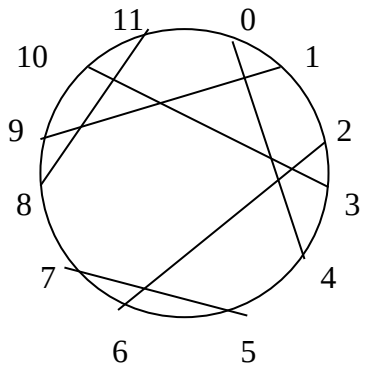
ESHORT.INP	ESHORT.OUT
1 12 8	16 4

ESHORT.INP	ESHORT.OUT
2 1 3 2 5	1 1 0

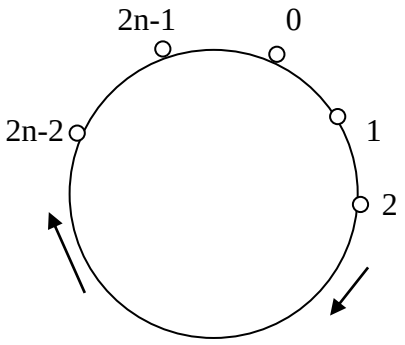
ESHORT.INP	ESHORT.OUT
4 -21735 20335 -8995 6125 20493 -19173 8481 -5775	819876 -621 581 -257 175

Tập con phẳng cực đại

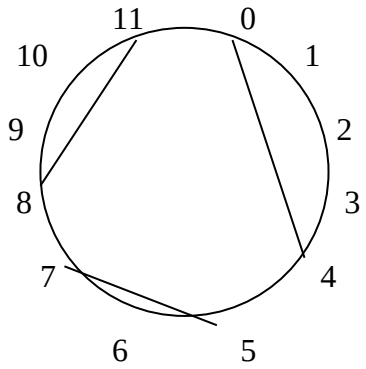
Cho C là tập gồm n dây cung của vòng tròn (xem hình (a)). Giả thiết là không có hai dây cung nào có chung đầu mút. Các đầu mút của các dây cung được đánh số từ 0 đến 2n-1 vòng quanh vòng tròn, theo chiều kim đồng hồ (xem hình (b)). Một tập con X của C được gọi là tập con phẳng nếu như không có hai dây cung nào của X là cắt nhau.



(a) Tập các dây cung C



(b) Cách đánh số đầu mút



(c) Tập con phẳng cực đại

Yêu cầu: Tìm tập con phẳng có lực lượng lớn nhất của C (ta sẽ gọi tập như vậy là tập con phẳng cực đại). Trong ví dụ ở hình (a), tập con phẳng cực đại được cho trong hình (c).

Dữ liệu: Vào từ file văn bản PLANSET.INP:

- Dòng đầu tiên ghi số n ($n \le 100$);
- Dòng thứ i trong số n dòng tiếp theo ghi hai số nguyên a_i, b_i ($0 \le a_i, b_i \le 2n-1$) là các chỉ số của hai đầu mút của dây cung thứ i (i = 1, 2, ..., n).

Kết quả: Ghi ra file văn bản PLANSET.OUT:

- Dòng đầu tiên ghi số nguyên dương k là lực lượng của tập con phẳng cực đại;
- Dòng tiếp theo ghi chỉ số của các dây cung trong tập con phẳng cực đại.

Ví dụ:

PLANSET.INP	PLANSET.OUT
6 0 4 1 9 2 6 3 10 5 7 8 11	3 1 5 12

PLANSET.INP	PLANSET.OUT
2 0 2 1 3	1 1

Mê cung

Mê cung có dạng một hình chữ nhật kích thước m × n được chia ra thành lưới ô vuông cạnh độ dài 1 bằng các đường song song với các cạnh. Mỗi ô vuông của lưới hoặc là ô cấm hoặc là ô tự do. Từ một ô tự do có thể di chuyển sang các ô tự do có chung cạnh với nó. Không được phép di chuyển vượt khỏi biên của mê cung. Mê cung được thiết kế khá đặc biệt: Giữa hai ô tự do bất kỳ chỉ có duy nhất một cách di chuyển từ ô này đến ô kia. Tại tâm của mỗi ô tự do đều có một cái móc. Trong mê cung có hai ô tự do đặc biệt, mà nếu bạn nối được hai cái móc ở hai ô đó bởi một sợi dây thừng, thì cánh cửa bí mật của mê cung sẽ tự mở ra. Vấn đề đặt ra là phải chuẩn bị sợi dây thừng với độ dài ngắn nhất đảm bảo cho dù hai ô đặc biệt có ở vị trí nào trong mê cung bạn vẫn có thể nối được các cái móc ở hai ô đó bởi sợi dây đã chuẩn bị.

Dữ liệu: Vào từ file văn bản LABYR.INP:

- Dòng đầu tiên chứa hai số n, m ($3 \le n, m \le 1000$);
- Các dòng tiếp theo mô tả mê cung. Dòng thứ i trong số m dòng tiếp theo chứa n ký tự, mỗi ký tự chỉ là "#" hoặc ".", trong đó ký tự "#" cho biết ô ở vị trí tương ứng là bị cấm, còn ký tự "." cho biết ô ở vị trí tương ứng là tự do (i= 1, 2, ..., m).

Kết quả: Ghi ra trên một dòng của file văn bản LABYR.OUT độ dài (tính bởi số ô) của sợi dây thừng cần chuẩn bị.

Ví dụ:

LABYR.INP	LABYR.OUT
3 3 ### #.# ###	0

LABYR.INP	LABYR.OUT
7 6 ####### #.#.### #.#.### #.#.#.# #.....# #######	8

Tập các tổng con

Cho N ($1 \le N \le 500$) số nguyên không âm $A_1, A_2, ..., A_N$, ($A_i \le 1000$, i = 1, 2, ..., N). Ký hiệu:

- A là họ các phần tử $A_1, A_2, ..., A_N$;
- 2^N là tập các tập con (kể cả tập rỗng) của tập các chỉ số N = {1, 2, ..., N};
- $Sum(A,S)= \sum_{i \in S} A_i$ là tổng các phần tử trong họ A có chỉ số thuộc tập $S \in 2^N$. Qui ước, $Sum(A,\varnothing) = 0$;
- $Sums(A)= \{Sum(A,S) \mid S \in 2^N\}$ tập các tổng có thể tính được từ các phần tử của họ A với chỉ số thuộc mỗi tập có thể của 2^N.

Yêu cầu: Tính lực lượng của tập Sums(A).

Chẳng hạn, khi N=3, A_1=1, A_2=1, A_3=2, và A = {A_1, A_2, A_3}, ta có

$S \in 2^N$	**∅**	**{1}**	**{2}**	**{3}**	**{1,2}**	**{1,3}**	**{2,3}**	**{1,2,3}**
Sum(A,S)	**0**	**A_1**	**A_2**	**A_3**	**A_1+ A_2**	**A_1+ A_3**	**A_2+ A_3**	**A_1+A_2+ A_3**
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**

Vì vậy, Sums(A) = {0, 1, 2, 3, 4} và giá trị cần tìm là 5.

Dữ liệu: Vào từ file văn bản SUMS.INP chứa các số N A_1 A_2 ... A_N được ghi cách nhau bởi dấu trắng.

Kết quả: Ghi ra file SUMS.OUT lực lượng của tập Sums(A).

Ví dụ:

SUMS.INP	SUMS.OUT
3 1 1 2	5

SUMS.INP	SUMS.OUT
3 1 3 2	7

Dạo chơi bằng xe buýt

Trên một tuyến đường ở thành phố du lịch nổi tiếng X có ô tô buýt công cộng phục vụ việc đi lại của du khách. Bến xe buýt có ở từng km của tuyến đường. Mỗi lần đi qua bến xe đều đỗ lại cho khách lên xuống. Mỗi bến đều có xe xuất phát từ nó, nhưng mỗi xe chỉ chạy không quá b km kể từ bến xuất phát của nó. Hành khách khi đi xe sẽ phải trả tiền cho độ dài đoạn đường mà họ ngồi trên xe. Cước phí cần trả để đi đoạn đường độ dài i là c_i (i = 1, 2, ..., b). Một du khách xuất phát từ một bến nào đó muốn đi dạo L km trên tuyến đường nói trên. Hỏi ông ta phải lên xuống xe như thế nào để tổng số tiền phải trả cho chuyến dạo chơi bằng xe buýt là nhỏ nhất?

Dữ liệu: Vào từ file văn bản BUS.INP:

- Dòng đầu tiên chứa hai số nguyên dương b, L (b≤20; L ≤ 200);
- Dòng thứ hai chứa b số nguyên dương c_1, c_2, ..., c_n được ghi cách nhau bởi dấu trắng.

Kết quả: Ghi ra file văn bản BUS.OUT:

- Dòng đầu tiên ghi chi phí tìm được;
- Dòng thứ hai ghi số lần lên xuống xe k;
- Dòng tiếp theo ghi k số là độ dài của các đoạn đường của k lần ngồi xe.

Ví dụ:

BUS.INP	BUS.OUT
10 15 **12 21 31 40 49 58 65 79 90 101**	**147 3** **3 6 6**

Xếp lại bảng số

Cho bàn cờ quốc tế kích thước 8×8. Các dòng được đánh số từ 1 đến 8, các cột được đánh ký hiệu từ a đến h (xem hình 1). Trên 64 ô của bàn cờ người ta viết các số từ 1 đến 64 theo một thứ tự tuỳ ý. Cho phép đổi chỗ hai số đặt ở hai ô mà từ ô này đến ô kia có thể đạt được bởi một nước đi của con ngựa. Ví dụ số đặt ở vị trí ô b2 có thể đổi chỗ được với các số ở các vị trí a4, c4, d3 và d1. Cần tìm cách đổi chỗ các số sao cho thu được bàn cờ với các số viết trên nó giống như trong hình 2.

	a	b	c	d	e	f	g	h
8	11	2	3	4	5	6	7	8
7	9	10	1	17	13	14	15	16
6	12	18	19	20	21	22	23	24
5	25	26	27	28	29	30	31	32
4	33	34	35	36	37	38	39	40
3	41	42	56	44	45	46	47	48
2	49	50	51	52	53	54	55	43
1	57	58	59	60	61	62	63	64

Hình 1

	a	b	c	d	e	f	g	h
8	1	2	3	4	5	6	7	8
7	9	10	11	12	13	14	15	16
6	17	18	19	20	21	22	23	24
5	25	26	27	28	29	30	31	32
4	33	34	35	36	37	38	39	40
3	41	42	43	44	45	46	47	48
2	49	50	51	52	53	54	55	56
1	57	58	59	60	61	62	63	64

Hình 1

Dữ liệu: Vào từ file văn bản BOARD.INP chứa 64 số từ 1 đến 64 được ghi trên bàn cờ ban đầu. Các số được liệt kê theo thứ tự như ta đánh số các ô trong hình 2. Các số được ghi cách nhau bởi dấu trắng hoặc dấu xuống dòng.

Kết quả: Đưa ra file văn bản BOARD.OUT:

- Dòng đầu tiên chứa số K là số lần thực hiện đổi chỗ;
- Dòng thứ i trong số K dòng tiếp theo chứa toạ độ của hai ô mà các số trong nó cần đổi chỗ cho nhau ở lần thực hiện đổi chỗ thứ i (i = 1, 2, ..., K).

Ví dụ: File dữ liệu tương ứng với hình 1 và file kết quả có thể có dạng sau:

BOARD.INP	BOARD.OUT
11 2 3 4 5 6 7 8 **9 10 1 17 13 14 15 16** **12 18 19 20 21 22 23 24** **25 26 27 28 29 30 31 32** **33 34 35 36 37 38 39 40** **41 42 56 44 45 46 47 48** **49 50 51 52 53 54 55 43** **57 58 59 60 61 62 63 64**	**11** **d7 c5** **c5 a6** **c5 d7** **c3 b1** **b1 d2** **d2 f3** **f3 h2** **f3 d2** **d2 b1** **b1 c3** **c7 a8**

Chu kì của dãy các bộ số

Cho hàm F xác đinh trên tập các số nguyên dương từ 1 đến m. Giá trị của hàm cũng là số nguyên dương từ 1 đến M. Cho một bộ có thư tự gồm n ($1 \le n \le 1000$) số nguyên dương $x_1, x_2, ..., x_n$ ($1 \le x_i \le M$, i = 1, 2, ..., n). Từ bộ số này chúng ta xây dựng một dãy các bộ số mới theo quy tắc sau:

1. $V_0 = x_1, x_2, ..., x_n$
2. $V_1 = F(x_1), F(x_2), ... F(x_n)$
3. $V_2 = F(F(x_1)), F(F(x_2)), ... F(F(x_n))$
4. $V_3 = F(F(F(x_1))), F(F(F(x_2))), ... F(F(F(x_n)))$
5. ...

Do tập các giá trị của hàm F là hữu hạn, nên dãy V_i sẽ bị lặp lại. Bạn cần xác định độ dài của phần không lặp lại và độ dài của chu kỳ. Độ dài của phần không bị lặp lại cũng như độ dài của chu kỳ phải là nhỏ nhất.
Chú ý: Theo định nghĩa, chu kỳ phải là số dương.

Dữ liệu: Vào từ file văn bản PERIOD.INP:

- Dòng đầu tiên chứa số M (M <= 32767);
- Dòng thứ 2 chứa các giá trị F(1), F(2), ..., F(M);
- Dòng thứ 3 chứa số n;
- Dòng thứ tư chứa các số $x_1, x_2, ..., x_n$.

Kết quả: Ghi ra file văn bản PERIOD.OUT đọ dài của phần không bị lặp lại và độ dài của chu kỳ.

Các số trên một dòng được ghi cách nhau bởi dấu cách hoặc dấu xuống dòng.

Ví dụ:

PERIOD.INP	PERIOD.OUT
10 **5 6 4 3 2 5 1 1 5 4** **4** **1 10 8 1**	**2 6**

Chỗ ẩn nấp

Trong phòng trung tâm của Kim tự tháp có rất nhiều cạm bẫy. Các cạm bẫy này được đặt ở các vị trí bí mật trong phòng. Tưởng chừng những cạm bẫy này là vô hại, nhưng khi bị kích hoạt chúng sẽ phun ra những dòng dung nham cực nóng. Rất may là người ta đã biết vị trí của tất cả các cạm bẫy này. Nhà khảo cổ học muốn tìm vị trí trong phòng mà khoảng cách từ đó đến cạm bẫy gần nhất là lớn nhất. Đây là vị trí ẩn nấp đảm bảo an toàn nhất.

Dữ liệu: Vào từ file AWAY.INP:

- Dòng đầu tiên chứa 3 số nguyên dương *X, Y, M* cách nhau bởi dấu trắng, trong đó *X* và *Y* là kích thước của căn phòng trung tâm có dạng một hình chữ nhật có đỉnh tại gốc toạ độ và các cạnh song song với các trục toạ độ ($1 \le X, Y \le 10000$) còn *M* là số cạm bẫy ($1 \le M \le 1000$).
- Dòng thứ *i* trong số *M* dòng tiếp theo chứa hai số U_i , V_i ($0 \le U_i \le X$, $0 \le V_i \le Y$) là toạ độ của cạm bẫy *i*.

Kết quả: Ghi ra trên một dòng của file văn bản AWAY.OUT toạ độ *P, Q* của vị trí ẩn nấp tìm được (các giá trị số làm tròn đến chữ số thứ 1 sau dấu phẩy: 0.05 được làm tròn thành 0.1).

Ví dụ:

AWAY.INP	AWAY.OUT
1000 50 1 10 10	1000.0 50.0

AWAY.INP	AWAY.OUT
100 100 4 10 10 10 90 90 10 90 90	50.0 50.0

AWAY.INP	AWAY.OUT
3000 3000 4 1200 85 63 2500 2700 2650 2990 100	1433.0 1669.8

Thằng Bờm và Phú ông

Thằng Bờm vừa thắng Phú ông trong một cuộc đấu trí. Hôm nay, Bờm đến nhà phú ông để nhận phần thưởng. Phú ông bày ra mặt bàn một dãy các chai có dung tích khác nhau chứa đầy các loại rượu ngon, và nói: "Cho phép nhà ngươi muốn uống bao nhiêu tuỳ thích, nhưng hễ đã mở chai nào thì phải uống hết rượu trong chai đó, trước khi mở chai khác. Sau khi uống hết một chai thì xếp lại chai rỗng vào vị trí ban đầu của nó, và điều quan trọng là nhà ngươi không được uống ba chai liên tiếp nhau trong dãy, điều đó là không may mắn".

Thằng Bờm đứng trước dãy các chai rượu và suy tính làm cách nào có thể uống được nhiều rượu của phú ông nhất. Bạn hãy giúp Bờm lựa chọn uống các chai rượu nào, nếu không cứ tiếp tục suy nghĩ, Bờm sẽ phát điên mất.

Dữ liệu: Vào từ file văn bản BOTTLES.INP:
- Dòng đầu tiên chứa số nguyên dương N ($N \le 10000$) là số lượng chai rượu bày trên bàn;
- Dòng thứ i trong số N dòng tiếp theo chứa số nguyên dương v_i ($v_i \le 32000$) là dung tích của chai thứ i trong dãy.

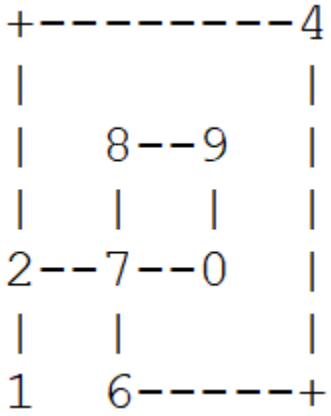
Kết quả: Ghi ra file văn bản BOTTLES.OUT:
- Dòng đầu tiên ghi 2 số k, v theo thứ tự là số lượng chai và tổng lượng rượu mà Bờm uống theo cách tìm được;
- Dòng tiếp theo chứa k chỉ số của các chai rượu cần uống.

Ví dụ:

BOTTLES.INP	BOTTLES.OUT
6 6 10 13 9 8 1	4 33 1 2 4 5

Chuột Mickey

Mickey là tên của con chuột máy mà một nhóm sinh viên vừa tạo ra trong đợt làm bài tập lớn môn *Trí khôn nhân tạo* (Artificial Intelligence). Mickey có thể đi vòng quanh mê cung trong hình vẽ 1. Khi Mickey đi đến vị trí có đánh dấu bởi con số, nó sẽ phải lựa chọn một trong các hướng dịch chuyển. Hành vi của con chuột Mickey khá là kỳ quặc, và điều đó gây ấn tượng rất lớn cho Giáo sư hướng dẫn của nhóm sinh viên.



Hình 1

Các vị trí đánh dấu bởi con số của mê cung ta sẽ gọi là các nút. Mickey chứa một số nguyên X trong bộ nhớ của nó và có thể thực hiện tính toán với số này. Tại mỗi nút (ngoại trừ nút 1) Mickey sẽ lựa chọn hướng dịch chuyển tùy thuộc vào số X, giảm X đi 1 và dịch chuyển đến nút được chọn. Hướng dịch chuyển sẽ được lựa chọn theo qui tắc sau:

Nút 2: Tính X mod 3. Nếu kết quả là 0, 1, 2, Mickey di chuyển đến nút 7, 1, 4 tương ứng.

Nút 4: Gọi Y là số thu được từ X bằng cách viết các chữ số trong hệ đếm thập phân của X theo thứ tự ngược lại. Nếu Y>X thì đi đến nút 6, trái lại đi đến nút 2.

Nút 6: Tính số lượng chữ số của X (trong hệ đếm thập phân). Nếu kết quả là số chẵn thì đi đến nút 4, trái lại đi đến nút 7.

Nút 7: Tính (X*X) mod 7. Nếu kết quả là 0, 1, 2, 4, Mickey di chuyển đến nút 2, 6, 8, 0 tương ứng.

Nút 8: Tính X mod 5. Nếu kết quả là 2 hoặc 3 thì đến nút 7, ngược lại đến nút 9.

Nút 9: Nếu vừa đến nút này từ nút 8 thì sẽ đến nút 0. Nếu vừa đến nút này từ nút 0 thì sẽ đến nút 8.

Nút 0: Gọi Y là chữ số nhỏ thứ ba trong các chữ số của X viết trong hệ đếm thập phân. (nếu X<100 thì Y=0). Nếu Y≤7 thì đến nút 7, trái lại đến nút 9.

Trước khi làm thực nghiệm, người ta đặt chuột tại nút 0 và nói cho nó số X. Con chuột bắt đầu di chuyển. Mickey sẽ hiển thị số X trên màn hình thông báo của nó. Thí nghiệm kết thúc khi chuột di chuyển đến nút 1 và số trên màn hình thông báo của nó sẽ là kết quả của thực nghiệm. Nếu giá trị của X giảm đến 0, thì thực nghiệm bị lỗi và kết quả của thực nghiệm được ghi nhận là -1.

Dữ liệu: Vào từ file văn bản MICKEY.INP chứa số nguyên dương X ($X \leq 10^9$) được nói cho Mickey khi bắt đầu thực nghiệm.

Kết quả: Ghi ra file văn bản MICKEY.OUT kết quả thực nghiệm.

Ví dụ:

MICKEY.INP	MICKEY.OUT	MICKEY.INP	MICKEY.OUT	MICKEY.INP	MICKEY.OUT
30	**-1**	**40**	**9**	**1000**	**789**

Mã xoắn

Lý thuyết mã hoá (*Cryptography*) là lĩnh vực nghiên cứu các phương pháp truyền thông bảo mật cho ta cách biến đổi một bức thông điệp dưới dạng văn bản (*plaintext*) sang một dạng văn bản mã (*ciphertext*) sao cho chỉ có người có chìa khoá giải mã mới có thể đọc được văn bản ban đầu. Việc chuyển từ dạng văn bản ban đầu (plaintext) sang dạng văn bản mã gọi là quá trình mã hoá (encryption); còn việc chuyển từ dạng văn bản mã về văn bản ban đầu gọi là giải mã (decryption). Mã xoắn (twisting) là một phương pháp mã hoá đơn giản đòi hỏi người gửi và người nhận phải thống nhất với nhau một số nguyên dương k dùng làm khoá mã.

Phương pháp mã xoắn sử dụng 4 mảng: *plaintext* và *ciphertext* là các mảng ký tự, còn *plaincode* và *ciphercode* là các mảng số nguyên. Tất cả các mảng đều có độ dài n, trong đó n là độ dài của bức thông điệp cần mã hoá. Các mảng đều bắt đầu từ phần tử 0,vì vậy các phần tử của nó được đánh số từ 0 đến n - 1. Trong bài này, các bức thông điệp chỉ chứa các chữ cái latin thường, dấu chấm '.' và dấu gạch dưới '_' (thay cho dấu cách).

Bức thông điệp cần mã hoá được cất trong mảng *plaintext*. Giả sử cho khoá *k*, khi đó quá trình mã hoá được tiến hành như sau.

- Thoạt tiên chuyển mỗi ký tự trong *plaintext* thành số nguyên để thu được mảng mã *plaincode* theo quy tắc sau: '_' = 0, 'a' = 1, 'b' = 2, ..., 'z' = 26, và '.' = 27.
- Tiếp đến, chuyển mỗi mã trong *plaincode* thành mã trong mảng *ciphercode* theo công thức sau: Với mỗi i = 0, ..., n-1
  ciphercode[*i*] = (*plaincode*[*ki* mod *n*]- *i*) mod 28. (trong đó *x* mod *y* là phần dư dương trong phép chia *x* cho *y*. Ví dụ, 3 mod 7 = 3, 22 mod 8 = 6, và -1 mod 28 = 27).
- Cuối cùng, chuyển mã trong *ciphercode* về ký tự trong mảng *ciphertext* theo quy tắc đã nêu ở trên. Bức thông điệp được mã hoá xoắn sẽ ghi vào mảng *ciphertext*.

Ví dụ, mã xoắn thông điệp "cat" sử dụng khoá 5 được mô tả trong mảng sau:

Mảng	0	1	2
plaintext	'c'	'a'	't'
plaincode	3	1	20
ciphercode	3	19	27
ciphertext	'c'	's'	'.'

Yêu cầu: Cần giải mã các bức thông điệp được mã hoá theo phương pháp mã xoắn nêu trên, khi biết mã khoá k.

Dữ liệu: Vào từ file văn bản UNTWIST:

- Dòng đầu tiên chứa *p* là số bức thông điệp được mã hoá cần giải mã;
- p nhóm dòng tiếp theo mỗi nhóm gồm hai dòng: Dòng đầu tiên chứa khoá mã k là số nguyên dương không vượt quá 300; Dòng thứ hai chứa bức điện được mã hoá xoắn có độ dài không quá 255.

Kết quả: Ghi ra trên p dòng của file văn bản UNTWIST.OUT: mỗi dòng chứa một bức điện đã được giải mã theo thứ tự ghi trong file dữ liệu vào.

Chú ý: *Bạn có thể giả thiết rằng bức điện giải mã là duy nhất. Muốn đạt được điều này chỉ cần chọn mã k sao cho ước chung lớn nhất của k và n (độ dài bức điện) là bằng 1. Điều kiện này được thực hiện ở mọi bộ test.*

Ví dụ:

UNTWIST.INP	UNTWIST.OUT
3 5 cs. 101 thqqxw.lui.qswer 3 b_ylxmhzjsys.virpbkr	cat this_is_a_secret beware._dogs_barking

Tầm nhìn

Trên mặt phẳng cho N đoạn thẳng ($1 \le N \le 1000$). Toạ độ của các đầu mút của các đoạn thẳng là các số nguyên không âm không vượt quá 20000. Các đường thẳng thu được bằng cách kéo dài các đoạn thẳng đã cho luôn cắt hai trục toạ độ và hai giao điểm cùng gốc toạ độ tạo thành một tam giác vuông cân. Không có hai đoạn thẳng nào giao nhau.

Ta nói một đoạn thẳng là nhìn thấy được từ gốc toạ độ O, nếu tìm được điểm X trên nó sao cho đoạn thẳng OX không cắt bất cứ đoạn nào khác trong số các đoạn thẳng đã cho.

Hãy viết chương trình đếm số đoạn thẳng nhìn thấy được từ gốc toạ độ.

Dữ liệu: Vào từ file VPOINT.INP:

- Dòng đầu tiên chứa số đoạn thẳng N;
- Mỗi một trong số N dòng tiếp theo chứa 4 số nguyên không âm X1, Y1, X2 và Y2, phân cách nhau bởi dấu trắng, trong đó (X1, Y1) là toạ độ của đầu mút thứ nhất còn (X2,Y2) là toạ độ của đầu mút thứ hai của đoạn thẳng tương ứng.

Kết quả: Ghi ra trên một dòng của file VPOINT.OUT số đoạn thẳng nhìn thấy được từ gốc toạ độ.

Ví dụ:

VPOINT.INP	VPOINT.OUT
4 3 13 11 5 14 1 10 5 10 14 20 4 5 6 10 1	3

Trò chơi số học

Xét trò chơi sau. Cho đống diêm gồm n que ($0 < n \le 70$) và một số nguyên m ($0 < m \le 20$), hai đấu thủ A và B luân phiên thực hiện nước đi. Đấu thủ A là người đi trước. Tại mỗi nước đi đối thủ đến lượt mình phải tìm số nguyên dương k, $1 \le k \le \min(m,n)$, và đây là số que diêm mà anh ta phải lấy khỏi đống diêm. Không được phép sử dụng hai lần cùng một số nguyên k, tức là nếu một số đã được sử dụng ở một lượt đi nào đó thì nó không được dùng lại, không phụ thuộc vào việc ai là người đã chọn số này trước đó. Trò chơi kết thúc khi không ai có thể thực hiện được nước đi. Người thắng cuộc là người thực hiện nước đi cuối cùng.

Cho trước hai số n, m, bạn cần xác định ai là người thắng trong trò chơi này.

Dữ liệu: Vào từ file văn bản DGAME.INP chứa hai số n và m trên một dòng được ghi cách nhau bởi dấu trắng.

Kết quả: Ghi ra file văn bản DGAME.OUT tên đấu thủ thắng cuộc cùng chiến lược giành phần thắng theo khuôn dạng sau:

- Dòng 1 chứa tên đấu thủ có chiến lược chơi dành phần thắng.
- Các dòng tiếp theo ghi mọi khả năng chơi của A theo thứ tự tăng dần, tiếp đến hoặc là một từ 'winning' hoặc một nước đi đáp lại để dành phần thắng của B .

Ví dụ:

DGAME.INP	DGAME.OUT	DGAME.INP	DGAME.OUT
3 2	B wins 1 2 2 1	7 4	A wins 1 winning 2 winning 3 4 4 3

Đặt trung tâm dịch vụ

Một công ty kinh doanh máy vi tính trên một địa bàn gồm N thành phố ($3 \le N \le 35$). Các thành phố được đánh số từ 1 đến N. Có các tuyến đường hai chiều nối trực tiếp M cặp thành phố. Công ty quyết định xây dựng các trung tâm dịch vụ tại một số thành phố sao cho đối với mỗi thành phố X, hoặc là có trung tâm dịch vụ đặt tại nó hoặc là có trung tâm dịch vụ đặt tại thành phố láng giềng trực tiếp của nó.
Bạn hãy giúp công ty tìm cách đặt một số ít nhất trung tâm mà vẫn đáp ứng được yêu cầu nói trên.

Dữ liệu: Vào từ file văn bản SERVSTAT.INP
- Dòng đầu tiên chứa hai số N, M;
- Mỗi một trong số M dòng tiếp theo chứa cặp gồm hai số nguyên dương là hai đầu mút của tuyến đường tương ứng.

Các số trên một dòng được ghi cách nhau bởi ít nhất một dấu trắng.

Kết quả: Ghi ra file văn bản SERVSTAT.OUT:
- Dòng đầu tiên ghi k là số lượng trung tâm dịch vụ cần đặt;
- Dòng tiếp theo ghi chỉ số của k thành phố cần đặt trung tâm dịch vụ.

Ví dụ:

SERVSTAT.INP	SERVSTAT.OUT
8 12 1 2 1 6 1 8 2 3 2 6 3 4 3 5 4 5 4 7 5 6 6 7 6 8	2 4 6

Các con số và các hàm số

Cho hai số nguyên dương X, Y (X, Y ≤ 32767) và ba hàm số

f1(x) = 2*x, f2(x) = [x/2], f3(x) = [x/3],

trong đó [a/b] là số nguyên lớn nhất không vượt quá a/b. Câu hỏi đặt ra là có thể bằng cách áp dụng lần lượt các hàm số này đối với số X để thu được giá trị Y hay không? Chẳng hạn nếu X=3, Y=8, câu trả lời sẽ là khẳng định. Thực vậy, ta có f1(f1(f1(f3(3)))) = 8 (vì f3(3)=1, f1(1)=2, f1(2)=4, f1(4)=8).

Yêu cầu: Tìm cách áp dụng một số ít nhất lần các hàm trên để từ X thu được Y.

Dữ liệu: Vào từ file văn bản NUMFUNC.INP gồm một dòng chứa hai số X, Y ghi cách nhau bởi dấu trắng.

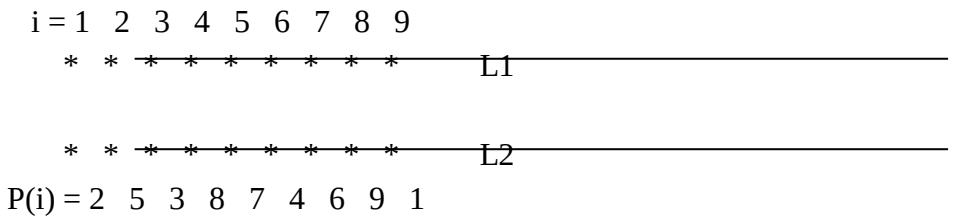
Kết quả: Ghi ra file văn bản NUMFUNC.OUT: Dòng đầu tiên chứa k là số lần áp dụng các hàm số đã cho để từ X thu được Y (k=0, nếu không thể thu được Y từ X bằng cách làm như vậy). Nếu k>0, thì dòng thứ i trong số k dòng tiếp theo ghi chỉ số của hàm cần áp dụng thứ i trong cách tìm được.

Ví dụ:

NUMFUNC.INP	NUMFUNC.OUT
3 8	4 3 1 1 1

Nối điểm

Trên hai đường thẳng song song L1 và L2, người ta đánh dấu trên mỗi đường N điểm. Các điểm trên đường thẳng L1 được đánh số từ 1 đến N từ trái qua phải, còn các điểm trên đường thẳng L2 được đánh số bởi P[1], P[2], ..., P[N] cũng từ trái qua phải, trong đó P[1], P[2], ..., P[N] là một hoán vị của các số 1, 2, ..., N (hình vẽ dưới đây cho một ví dụ, khi N = 9).



Ta gọi các số gán cho các điểm là số hiệu của chúng. Cho phép nối hai điểm trên 2 đường thẳng có cùng số hiệu.

Yêu cầu: Tìm cách nối được nhiều cặp điểm nhất với điều kiện các đoạn nối không được cắt nhau.

Dữ liệu: Vào từ file văn bản WIRES.INP:

- Dòng đầu tiên chứa số nguyên dương N (N ≤ 1000);
- Dòng thứ hai chứa các số P[1] P[2] ... P[N] được ghi cách nhau bởi dấu trắng.

Kết quả: Ghi ra file văn bản WIRES.OUT:

- Dòng đầu tiên chứa k là số lượng đoạn nối tìm được;
- Dòng tiếp theo chứa k số hiệu của các đầu mút của các đoạn nối được ghi theo thứ tự tăng dần.

Ví dụ:

WIRES.INP	WIRES.OUT
9 2 5 3 8 7 4 6 9 1	5 2 3 4 6 9

Số nguyên tố ghép

Xét A là dãy các số nguyên tố

2, 3, 5, 7, 11, 13, 17, 19, ...

và dãy B gồm các số thu được từ dãy A bằng cách ghép hai số liên tiếp trong A:

23, 57, 1113, 1719, ...

Trong dãy B có những phần tử là số nguyên tố. Chẳng hạn 23, 3137, 8389, 157163, ...
Các số nguyên tố trong dãy B ta gọi là số nguyên tố ghép. Dãy con của dãy B bao gồm những số nguyên trong B được giữ nguyên thứ tự của chúng trong B sẽ được gọi là dãy số nguyên tố ghép và ký hiệu dãy này là C.

Yêu cầu: Cho trước số nguyên dương N ≤ 500, hãy tìm số hạng thứ N của dãy C.

Dữ liệu: vào từ file văn bản COPRIME.INP chứa số N

Kết quả: Ghi ra trên một dòng của file văn bản COPRIME.OUT số hạng thứ N của dãy số nguyên tố ghép.

Ví dụ:

COPRIME.INP	COPRIME.OUT
1	23

Theo dõi hoạt động của mạng máy tính

Có n máy tính (đánh số từ 1 đến n). Các máy tính này cần được nối với nhau thành một mạng liên thông không có vòng bởi một số kênh nối giữa một số cặp máy. Người ta muốn cài đặt các phần mềm tại một số máy trong mạng để theo dõi hoạt động của tất cả các kênh truyền tin của mạng. Biết chi phí để cài đặt phần mềm này trên máy j là c_j. Một máy được cài đặt phần mềm sẽ kiểm soát được tất cả các kênh truyền tin trong mạng liên thuộc với nó.

Hãy xác định xem cần phải chọn các máy tính nào trong mạng để cài đặt phần mềm theo dõi hoạt động của các kênh truyền tin sao cho hai máy bất kỳ được chọn không có kênh nối với nhau đồng thời tổng chi phí cài đặt là nhỏ nhất.

Dữ liệu: Vào từ file văn bản có tên CONTROL.INP có cấu trúc như sau:

- Dòng đầu tiên chứa số n ($n \le 500$);
- Dòng thứ hai chứa các số nguyên dương $c_1, c_2, ..., c_n$ được ghi cách nhau bởi dấu trắng;
- Các dòng tiếp theo, mỗi dòng chứa hai số nguyên dương d, c, được ghi cách nhau bởi dấu trắng, là số hiệu của hai đầu mút của một kênh nối trong mạng.

Kết quả: Ghi ra file văn bản CONTROL.OUT:

- Dòng đầu tiên ghi P là tổng chi phí theo cách lựa chọn tìm được;
- Dòng thứ hai chứa K là tổng số máy cần cài đặt phần mềm;
- K dòng tiếp theo mỗi dòng ghi chỉ số của một máy được chọn.

Ví dụ:

CONTROL.INP	CONTROL.OUT
3 1 10 3 1 2 2 3	4 2 1 3

Hình vuông thần bí (Ma phương)

Ma phương bậc N là bảng vuông gồm $N \times N$ số được xếp thành N dòng, N cột sao cho

1. Mỗi số trong bảng là một số nguyên dương trong khoảng từ 1 đến N^2 .
2. Các số trong bảng là đôi một khác nhau.
3. Tổng các số trong N dòng, N cột và hai đường chéo chính của bảng là bằng nhau.

Ví dụ: Bảng trong hình 1 cho ta ma phương bậc 3: Tổng của ba dòng là 8+3+4, 1+5+9, 6+7+2; tổng của ba cột là 8+1+6, 3+5+7, 4+9+2; tổng của hai đương chéo là 8+5+2, 4+5+6. Tất cả các tổng này đều bằng 15.

Yêu cầu: Cho một bảng gồm $N \times N$ ô đã được điền số ở một số ô, cần xác định xem có thể điền tiếp các số vào các ô còn lại để thu được ma phương bậc N hay không?

Ví dụ 1. Bảng 5 × 5 trong hình 2 có thể điền tiếp để thu được ma phương trong hình 3.

8	3	4
1	5	9
6	7	2

Hình 1

	1		24	
		8		
	9			
	10	21		
				11

Hình 2

2	1	18	24	20
25	23	8	4	5
16	9	12	13	15
3	10	21	17	14
19	22	6	7	11

Hình 3

Dữ liệu: Vào từ file văn bản SQMAGIC.INP:

- Dòng đầu tiên chứa số N, M, theo thứ tự là bậc của bảng và số lượng số đã điền trên bảng ($2 \le N \le 8$);
- Mỗi dòng trong số M dòng tiếp theo chứa 3 số nguyên dương r, c, a cho biết ô (r,c) của bảng đã được điền số a.

(Các dòng của bảng được đánh số từ 1 đến N từ trên xuống dưới, các cột được đánh số từ 1 đến N từ trái qua phải, ô nằm trên giao của dòng i, cột j gọi là ô (i, j).)

Kết quả: Ghi ra file văn bản SQMAGIC.OUT:

- Nếu tìm được cách điền thì dòng đầu tiên ghi YES, trái lại ghi NO;
- Nếu dòng đầu tiên ghi YES thì dòng thứ i trong số N dòng tiếp theo ghi N số của dòng i trong bảng tìm được (i = 1, 2, ..., N).

Ví dụ:

SQMAGIC.INP	SQMAGIC.OUT
5 7 1 4 24 4 2 10 5 5 11 2 3 8 3 2 9 1 2 1 4 3 21	YES 2 1 18 24 20 25 23 8 4 5 16 19 12 13 15 3 10 21 17 14 19 22 6 7 11

Đổi tiền

Có N loại đồng tiền. Đồng tiền loại i có giá trị là v_i và trọng lượng là w_i, $1 \le i \le N$. Hai loại đồng tiền khác nhau có thể có cùng giá trị hoặc có cùng trọng lượng nhưng không thể có cả giá trị và trọng lượng giống nhau. Cho trước giá trị V và W, cần tìm một số lượng ít nhất các đồng tiền M sao cho tổng giá trị của các đồng tiền này là V và tổng trọng lượng của chúng là W. Giả thiết là số lượng đồng tiền mỗi loại là không hạn chế.

Dữ liệu: Vào từ file văn bản VWCOIN.INP

- Dòng đầu tiên chứa các số nguyên N, V, W ($1 \le N \le 20$, $1 \le V \le 150$, $1 \le W \le 150$),
- Dòng thứ i trong số N dòng tiếp theo chứa 2 số nguyên v_i ($1 \le v_i \le 150$) và w_i ($1 \le w_i \le 150$), $1 \le i \le N$.

Các số trên một dòng được ghi cách nhau một dấu cách.

Kết quả: Ghi ra file văn bản VWCOIN.OUT:

- Dòng đầu tiên ghi số lượng đồng tiền tìm được M (ghi M = -1, nếu không tìm được lời giải);
- Nếu M>0, thì mỗi dòng trong số M dòng tiếp theo ghi loại tiền của một đồng tiền trong số M đồng tiền cần dùng.

Ví dụ:

VWCOIN.INP	VWCOIN.OUT
8 141 5 1 1 2 1 4 1 8 1 16 1 32 1 64 1 128 1	4 1 3 4 8

VWCOIN.INP	VWCOIN.OUT
4 11 17 12 3 4 7 8 10 21 9	-1

Đếm phần

Trên mặt phẳng toạ độ cho n đường thẳng phân biệt $L_1, L_2, ..., L_n$. Biết rằng đường thẳng L_i đi qua hai điểm $M_{1i}(x_{1i}, y_{1i})$ và $M_{2i}(x_{2i}, y_{2i})$. Hãy đếm số phần của mặt phẳng được chia ra bởi n đường thẳng đã cho.

Dữ liệu: Vào từ file văn bản PARTS.INP:

- Dòng đầu tiên chứa số nguyên dương n ($n \le 3000$);
- Dòng thứ i trong số n dòng tiếp theo chứa 4 số nguyên $x_{1i}, y_{1i}, x_{2i}, y_{2i}$, mỗi số có trị tuyệt đối không vượt quá 10000.

Kết quả: Ghi ra file văn bản PARTS.OUT số phần của mặt phẳng bị chia bởi n đường thẳng đã cho.

Ví dụ:

PARTS.INP	PARTS.OUT
4 5 0 0 5 4 0 4 5 2 4 3 4 1 1 1 5	9

Cứu hoả

Trong một cao ốc N tầng là trụ sở của một nhà băng (N ≤ 150) xảy ra hoả hoạn. Hoả hoạn lan truyền với tốc độ 1 tầng/phút.. Trong cao ốc có một thang máy chuyển động với tốc độ 10 tầng/phút. Nếu thang máy di chuyển qua tầng đang có lửa thì nó cũng bốc cháy. Tại thời điểm bắt đầu hoả hoạn thang máy ở tầng thứ 1. Với mục đích cứu những tài sản quý giá (các hòm với các đồng tiền vàng) được cất giữ trên một số tầng của cao ốc tất cả nhân viên của nhà băng phải dời cao ốc bằng cầu thang còn thang máy được dành cho nhiệm vụ cứu của. Biết vị trí tầng phát sinh hoả hoạn, các tầng có chứa hòm của cùng số lượng đồng tiền vàng trong chúng và thời gian để chuyển chúng vào thang máy là 1.5 phút, bạn cần tìm cách cứu được lượng đồng tiền vàng lớn nhất.

Dữ liệu: Vào từ file văn bản MONET.INP:

- Dòng đầu tiên chứa các số nguyên n, h là số kượng tầng của cao ốc và chỉ số của tầng phát sinh hoả hoạn;
- Dòng thứ i trong số n-1 dòng tiếp theo chứa số lượng đồng tiền vàng có trong hòm của ở tầng i+1 (tầng 1 không có của). Số lượng đồng tiền vàng ở mỗi tầng giả thiết là số nguyên.

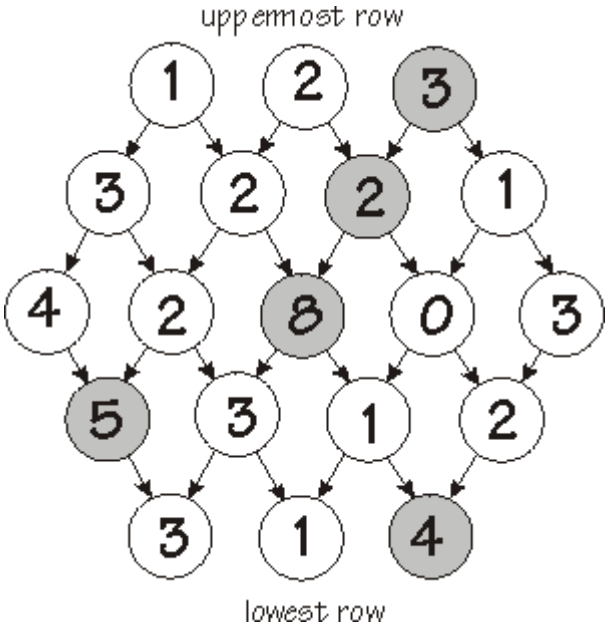
Kết quả: Ghi ra trên một dòng của file văn bản MONET.OUT số lượng đồng tiền có thể cứu được.

Ví dụ:

MONET.INP	MONET.OUT
5 5 100 0 300 1000	100

Lưới tổ ong

Hình 1 cho ta một lưới tổ ong mỗi ô của lưới có ghi một con số (kích thước của lưới là 3). Một đường đi trên lưới là một cách di chuyển bắt đầu từ một ô ở dòng trên cùng và kết thúc tại ô ở dòng cuối cùng. Từ một ô ta chỉ có thể di chuyển theo đường chéo đến một ô ở dòng dưới ở bên trái hoặc bên phải. Khi có một đường đi bạn có thể thực hiện nhiều nhất một lần đổi chỗ hai số trên cùng một hàng ngang của lưới.
Cần phải tìm đường đi có tổng các số trên các ô đi qua là lớn nhất có tính đến khả năng đổi chỗ hai số trên dòng lựa chọn (ta gọi tổng này là độ dài của đương đi).



Hình 1. Lưới tổ ong kích thước 3

Giới hạn: Các số trên lưới là các số nguyên trong khoảng từ 0 đến 99. Kích thước của lưới là số nguyên dương không quá 99.

Dữ liệu: Vào từ file văn bản HON.IN:
- Dòng đầu tiên chứa kích thước của lưới n;
- 2n -1 dòng tiếp theo, mỗi dòng chứa các số trên một hàng ngang của lưới theo thứ tự từ trên xuống dưới.

Kết quả: Ghi ra file văn bản HON.OUT độ dài của đường đi lớn nhất tìm được.

Ví dụ:

HON.IN	HON.OUT
3 1 2 3 3 2 2 1 4 2 8 0 3 5 3 1 2 3 1 4	22

Múi giờ

Một nhà kinh doanh có các bạn hàng trên toàn cầu. Trong một ngày ông ta nhận được dúng một thông điệp từ mỗi một múi giờ kể cả múi giờ mà ông ta đang sống. Mỗi bức thông điệp đều có ghi thời điểm nó được chuyển đi (các thông điệp đến địa chỉ một cách trực tiếp). Không may mỗi người gửi thông điệp đều chỉ ghi giờ địa phương mà không ghi rõ múi giờ nơi người đó sống.

Bạn cần tìm cách xác định múi giờ cho từng bức thông điệp. Giả thiết rằng số lượng giờ, ký hiệu bởi n, trong một ngày là một số trong khoảng từ 5 đến 60. Số lượng múi giờ luôn bằng số lượng giờ và mỗi múi giờ có khoảng phân bố thời gian là một số nguyên giờ.

Các múi giờ được đánh số từ 0 đến n-1. Nhà kinh doanh sống ở múi giờ 0 (giờ GMT) không có khoảng phân bố thời gian. Các múi giờ được tính theo chiều đông-tây. Nghĩa là bạn cần cộng thêm z giờ vào giờ địa phương ở múi giờ z để có được giờ tại múi giờ 0. (Lưu ý là cách tính này không giống cách tính thông thường trên thực tế). Ví dụ: Nếu giờ địa phương tại vùng 2 là 03:15 thì giờ tại vùng 0 sẽ là 05:15 (GMT).

Thông điệp có thể đến tại thời điểm bất kỳ trong ngày (nghĩa là trong khoảng từ 0:00 cho đến (n-1):59), nhưng không có hai thông điệp nào đến cùng thời điểm.

Dữ liệu: Vào từ file văn bản ZON.IN:

- Dòng đầu tiên chứa số giờ trong ngày n ($5 \le n \le 60$) đồng thời là số múi giờ và là số thông điệp nhận được;
- Mỗi dòng trong số n dòng tiếp theo chứa giờ địa phương của một bức thông điệp có dạng hhmm (2 ký tự đầu để ghi giờ, hai ký tự cuối để ghi phút, trong đó $0 \le hh \le n-1$; $0 \le mm \le 59$). Các dòng được sắp xếp theo thứ tự thời gian nhận được, nghĩa là thông điệp đến đầu tiên ở dòng đầu tiên...

Giả thiết là luôn có lời giải duy nhất cho mỗi test.

Kết quả: Ghi ra file ZON.OUT một dòng gồm n số cho biết múi giờ của n thông điệp, số đầu tiên tương ứng với bức thông điệp nhận được đầu tiên...

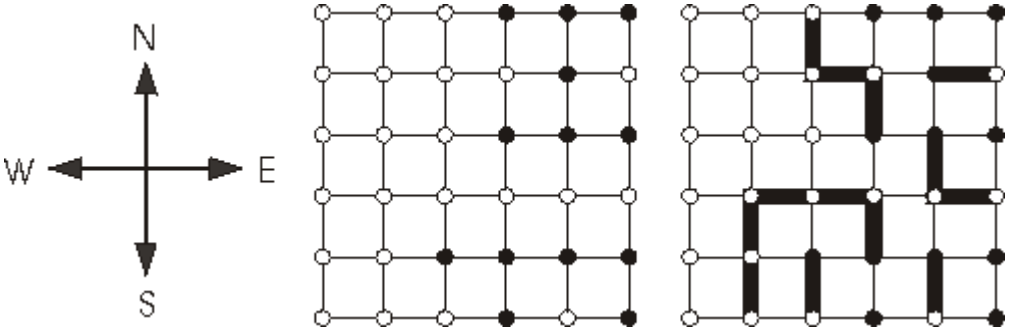
Ví dụ:

ZON.IN	ZON.OUT
5 0017 0250 0400 0201 0002	3 1 0 2 4

Để ý rằng bức thông điệp số 3 phải đến từ múi giờ 0 hoặc phải đến tại thời điểm muộn hơn 4:59 là phút cuối cùng trong ngày gồm 5 giờ.

Bảng điện

Một lưới ô vuông được phủ trên một bảng điện hình vuông. Vị trí nằm trên giao của 2 đường kẻ của lưới sẽ được gọi là nút. Tất cả có **n × n** nút trên lưới.



Hình 2a. *Bảng điện* Hình 2b. *Lời giải*

Có một số nút chứa tiếp điểm. Nhiệm vụ của bạn là cần nối các tiếp điểm với các nút ở trên biên của bảng bởi các đoạn dây dẫn (gọi là các mạch). Các mạch chỉ được chạy dọc theo các đường kẻ của lưới (nghĩa là không được chạy theo đường chéo). Hai mạch không được phép có điểm chung, vì vậy hai mạch bất kỳ không được phép cùng chạy qua cùng một đoạn đường kẻ của lưới cũng như không được chạy qua cùng một nút của lưới. Các mạch cũng không được chạy dọc theo các đoạn kẻ của lưới ở trên biên (mạch phải kết thúc khi nó gặp biên) và cũng không được chạy qua nút chứa tiếp điểm khác.

Ví dụ: Bảng điện và các tiếp điểm được cho trong hình 2a. Nút tô đậm trong hình vẽ thể hiện vị trí các tiếp điểm.

Yêu cầu: Viết chương trình cho phép nối được một số nhiều nhất các tiếp điểm với biên. Các tiếp điểm ở trên biên đã thoả mãn đòi hỏi đạt ra, vì thế không nhất thiết phải thực hiện mạch nối chúng. Nếu như có nhiều lời giải thì chỉ cần đưa ra một trong số chúng.

Dữ liệu: Vào từ file văn bản ELE.INP:

- Dòng đầu tiên chứa số nguyên n ($3 \le n \le 15$).
- Mỗi dòng trong số n dòng tiếp theo chứa n ký tự phân cách nhau bởi một dấu cách. Mỗi ký tự chỉ là 0 hoặc 1. Ký tự 1 thể hiện tiếp điểm, ký tự 0 thể hiện nút không có tiếp điểm trên vị trí tương ứng của lưới. Các nút được đánh số từ 1 đến n x n theo thứ tự từ trái qua phải, từ trên xuống dưới. Chỉ số của nút chứa tiếp điểm sẽ là chỉ số của tiếp điểm.

Kết quả: Ghi ra file Ele.OUT:

- Dòng đầu tiên chứa k là số tiếp điểm lớn nhất có thể nối với biên bởi các mạch.
- Mỗi dòng trong số k dòng tiếp theo mô tả một mạch nối một trong số k tiếp điểm với biên theo qui cách sau: Đầu tiên là chỉ số của tiếp điểm được nối, tiếp đến là dãy các ký tự mô

tả hướng của mạch nối: E: Đông, W: Tây, N: Bắc, S: Nam. Giữa chỉ số và dãy ký tự phải có đúng một dấu cách, còn giữa các ký tự trong dãy ký tự không được có dấu cách.

Kết quả phải được đưa ra theo thứ tự tăng dần của chỉ số tiếp điểm.

Ví dụ:

ELE.IN	ELE.OUT
6 0 0 0 1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 1 0 1	6 11 E 16 NWN 17 SE 27 S 28 NWWSS 29 S

Biểu thức chia

Biểu thức chia là biểu thức số học có dạng sau đây

$x_1/x_2/x_3/.../x_k$

trong đó x_i là số nguyên dương với mọi i ($1 \le i \le k$). Biểu thức chia được tính giá trị theo thứ tự từ trái sang phải. Chẳng hạn giá trị của biểu thức

1/2/1/2

là 1/4. Người ta có thể đặt các dấu ngoặc vào biểu thức để thay đổi giá trị của nó. Ví dụ giá trị của biểu thức

(1/2)/(1/2)

là 1.

Yêu cầu: Cho biểu thức chia E, hỏi có thể đặt các dấu ngoặc vào nó để thu được biểu thức E' có giá trị là một số nguyên hay không?

Dữ liệu: Vào từ file văn bản DIV.INP:

- Dòng đầu tiên chứa số nguyên dương d ($d \le 5$) là số bộ dữ liệu trong file.
- Tiếp đến là các bộ dữ liệu được ghi theo qui cách sau: Dòng đầu tiên của một bộ dữ liệu chứa số nguyên n ($2 \le n \le 10000$) là số lượng số nguyên trong biểu thức. Mỗi dòng trong số n dòng tiếp theo chứa một số nguyên dương không vượt quá 10^9, số ở dòng thứ i tương ứng với số nguyên thứ i trong biểu thức.

Kết quả: Ghi ra file văn bản DIV.OUT: dòng thứ i ($1 \le i \le d$) chứa chữ YES nếu biểu thức thứ i trong file dữ liệu có thể biến đổi thành biểu thức có giá trị nguyên hoặc chứa chữ NO nếu trái lại.

Ví dụ:

DIV.IN	DIV.OUT
2 4 1 2 1 2 3 1 2 3	YES NO

Dán nhãn

Bờm là một cổ động viên nhiệt tình của các cuộc đua ô tô. Vì thế Bờm quyết định tự tạo cho mình một bộ sưu tập các mô hình xe đua. Trong cửa hàng đồ chơi có thể mua các mô hình xe đua đựng trong các hộp kín. Trong mỗi hộp có các bộ phận của một xe đua và bộ các nhãn hiệu chứa hình ảnh và các ký tự. Bộ nhãn trong tất cả các hộp đều như nhau. Bờm quyết định dán nhãn cho các mô hình bởi các số nguyên liên tiếp bắt đầu từ 1. Chẳng hạn, để dán nhãn cho mô hình thứ 2070 cần có 4 nhãn: một nhãn với "2", hai nhãn với "0" và một nhãn với "1".

Bờm lắp ráp các mô hình theo quy trình sau: Đầu tiên nó mở một hộp mới, lắp ráp mô hình và dán nhãn cho mô hình bằng cách sử dụng các nhãn. Để dán nhãn nó có thể sử dụng nhãn có trong hộp vừa mới mở cũng như các hộp đã mở trước đó, nhưng không được phép mở hộp mới để lấy nhãn còn thiếu.

Yêu cầu: Cho trước bộ nhãn trong các hộp, hãy tính xem Bờm có thể dán nhãn được cho bao nhiêu mô hình theo qui trình đã mô tả ở trên.

Dữ liệu: Vào từ file văn bản STI.IN gồm một dòng chứa 10 ký tự số

i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7, i_8, i_9

trong đó i_j là số lượng nhãn dán với chữ số j ($0 \le j \le 9$) trong bộ nhãn của mỗi hộp. Hai ký tự liên tiếp được ghi cách nhau bởi một dấu cách.

Kết quả: Ghi ra trên một dòng của file văn bản **STI.OUT** số lượng mô hình được dán nhãn.

Ví dụ:

STI.IN	STI.OUT
1 1 1 1 1 1 1 1 1 1	199990

STI.IN	STI.OUT
3 4 5 4 3 4 5 4 3 4	499999994999999999499999973

Mutexes

Các ngôn ngữ lập trình tiên tiến cho phép viết các chương trình chứa hàng loạt các cung đoạn tính toán. Có thể hiểu các cung đoạn tính toán này như các chương trình được thực hiện song song trong không gian nhớ như nhau và truy cập đến các biến như nhau. Thông thường các cung đoạn này cần được phối hợp đồng bộ với nhau. Chẳng hạn, một cung đoạn có thể cần chờ một cung đoạn khác hoàn thành một số tính toán nào đó và cất giữ kết quả vào các biến.
Công cụ đơn giản nhất để thực hiện việc đồng bộ hoá là mutex. Mutex là một đối tượng đặc biệt chỉ có hai trạng thái đóng và mở. Một mutex đóng luôn liên quan đến đúng một cung đoạn tính toán. Có hai thao tác mà một cung đoạn tính toán có thể tác động đến mutex đó là: LOCK (Đóng) và UNLOCK (Mở).

Khi một cung đoạn thực hiện thao tác LOCK đến một mutex đang ở trạng thái mở, mutex này sẽ chuyển sang trạng thái đóng và cung đoạn sẽ chủ sở hữu của mutex này. Nếu một cung đoạn thực hiện thao tác LOCK đến một mutex đang ở trạng thái đóng bởi một cung đoạn khác thì cung đoạn sẽ bị đóng cho đến khi mutex được mở.

Khi một cung đoạn thực hiện thao tác UNLOCK đến một mutex bị chiếm dụng bởi nó thì mutex sẽ được chuyển sang trạng thái mở. Nếu có một cung đoạn khác đang chờ LOCK (đóng) mutex, thì một trong số chúng sẽ chiếm dụng mutex này. Nếu có nhiều cung đoạn cùng chờ thì sẽ chọn tuỳ ý một trong số chúng.

Một trong những vấn đề cần xử lý trong các chương trình đa cung đoạn là xử lý xung đột (deadlocks). Xung đột xảy ra khi có hai (hoặc nhiều hơn) cung đoạn cùng chờ việc giải phóng mutex và không có cung đoạn nào có thể tiếp tục thực hiện được. Xung đột cũng xảy ra khi một cung đoạn chờ mutex đang bị đóng bởi một cung đoạn đã chấm dứt thực hiện nhưng không giải phóng mutex.

Yêu cầu: Bạn được biết trước mô tả của một số cung đoạn và cần xác định xem có xảy ra xung đột hay không. Mỗi một cung đoạn là một dãy các chỉ thị dạng
LOCK <mutex>
UNLOCK <mutex>
Đối với các chỉ thị có các giả thiết sau:

- tên của mutex là các chữ cái in hoa từ A đến Z;
- không có cung đoạn nào tìm cách LOCK một mutex đã có chủ sở hữu;
- không có cung đoạn nào tìm cách UNLOCK một mutex không có chủ sở hữu.

Dữ liệu: Vào từ file văn bản MUT.IN: Dòng đầu tiên chứa số cung đoạn M ($1 \le M \le 5$) tiếp đến là M nhóm dòng mô tả các cung đoạn. Dòng đầu tiên trong một nhóm dòng mô tả cung đoạn sẽ chứa số chỉ thị của cung đoạn N_j ($1 \le N_j \le 10$) và tiếp đến là N_j dòng mỗi dòng chứa một chỉ thị. Các chỉ thị không chứa dấu cách thừa.

Kết quả: Ghi ra file văn bản MUT.OUT:

- Dòng đầu tiên ghi một số D (D là 1 nếu xảy ra xung đột và là 0 nếu không xảy ra xung đột)
- Nếu xảy ra xung đột thì dòng thứ hai phải chứa trạng thái của chương trình tại đó xảy ra xung đột. Nếu có nhiều trạng thái như vậy thì chỉ cần đưa ra một trong số chúng. Trạng thái của chương trình được mô tả bởi chỉ số của cung đoạn và chỉ số của chỉ thị đang thực hiện. Các cung đoạn và các chỉ thị của cùng môt cung đoạn được đánh số bởi các số nguyên liên tiếp bắt đầu từ 0 theo thứ tự xuất hiện trong file dữ liệu. Riêng cung đoạn cuối cùng luôn được đánh số bởi -1. Hai chỉ số mô tả trạng thái cần được ghi cách nhau bởi dấu cách.

Ví dụ:

MUT.IN	MUT.OUT
2 1 LOCK X 2 LOCK Y LOCK X	1 -1 1

Bài giải sẽ chỉ được điểm khi trả lời đúng là không có xung đột, nếu như đã trả lời đúng ít nhất một test có xung đột.

Dãy số

Cho dãy số gồm $2N$ số nguyên dương. Biết rằng có thể phân các số trong dãy này thành các cặp số sao cho tổng các số trong mọi cặp đều bằng nhau. Ví dụ, các số trong dãy 99, 23, 77, 1 có thể phân ra thành các cặp: 1 + 99 = 77 + 23. Hỏi rằng dãy số đã cho có thể phân ra thành các cặp sao cho tích các số trong mỗi cặp đều bằng nhau hay không?

Dữ liệu: Vào từ file văn bản SEQ.DAT chứa một dãy các test. Dòng đầu tiên chứa số nguyên dương K là số test trong file. Dòng đầu tiên của mỗi test chứa số $2N$ là số phần tử của dãy số. Trong mỗi dòng của $2N$ dòng tiếp theo chứa một số hạng của dãy số là một số nguyên trong khoảng từ 1 đến 10^9 ($1 \le N \le 50000$).

Kết quả: Ghi ra file văn bản SEQ.OUT: mỗi dòng ghi câu trả lời của một test tương ứng: ghi số 1, nếu câu trả lời là khẳng định, ghi số 0, nếu câu trả lời là phủ định.

Ví dụ:

SEQ.DAT	SEQ.OUT
2 4 99 23 77 1 2 1 10101	0 1

Xe buýt

Một xe buýt của công ty có nhiệm vụ đón nhân viên đến trụ sở làm việc. Trên hành trình xe buýt sẽ tiếp nhận nhân viên đứng chờ ở các điểm hẹn nếu như xe còn chỗ trống. Xe buýt có thể sẽ đỗ lại để chờ những công nhân còn chưa kịp đến điểm hẹn. Cho biết thời điểm mà mỗi nhân viên đến điểm hẹn của mình và thời điểm qua mỗi điểm hẹn của xe buýt. Giả thiết rằng xe buýt đến điểm hẹn đầu tiên tại thời điểm 0, thời gian xếp khách lên xe được coi là bằng 0. Hãy xác định khoảng thời gian ngắn nhất để xe buýt có thể chở một số lượng các nhân viên đến trụ sở làm việc lớn nhất có thể được.

Dữ liệu: Vào từ file BUS.DAT:

- Dòng đầu tiên chứa 2 số nguyên dương *N, M* theo thứ tự là số điểm hẹn và số chỗ ngồi của xe buýt;
- Dòng thứ *i* trong số *N* dòng tiếp theo chứa số nguyên t_i là thời gian cần thiết để xe buýt di chuyển từ điểm hẹn *i* đến điểm hẹn *i*+1 (điểm hẹn thứ *N*+1 sẽ là trụ sở làm việc của công ty), số nguyên *K* là số lượng nhân viên đến điểm hẹn *i* và tiếp đến là *K* số nguyên là các thời điểm đến điểm hẹn của *K* nhân viên.

Giới hạn: $1 \le M \le 2000$, $1 \le N, K \le 200000$.

Kết quả: Ghi ra file văn bản BUS.OUT thời gian ngắn nhất tìm được.

Ví dụ:

BUS.DAT	BUS.OUT
3 5 1 2 0 1 1 1 2 1 4 0 2 3 4	4

Trò chơi với cỗ bài

Trên bàn có dãy gồm N chồng bài, mỗi chồng bài gồm các quân bài có màu sắc khác nhau. Mỗi lượt đi bạn có thể lấy đi tất cả các con bài cùng mầu ở một số tuỳ ý chồng bài cạnh nhau. Hãy tìm cách thực hiện một số ít nhất lượt đi để có thể dọn sạch các đống bài khỏi mặt bàn.

Dữ liệu: Vào từ file văn bản CARDS.DAT:

- Dòng đầu tiên chứa số lượng đống bài $N \geq 2$.
- Dòng thứ i trong số N dòng tiếp theo chứa K_i là số lượng quân bài ở chồng bài i, tiếp đến là K_i số nguyên dương là chỉ số mầu của các quân bài ở chồng bài i được ghi theo thứ tự từ dưới lên. ($1 \leq K_i \times N \leq 10000$), $i = 1, 2, ..., N$.

Kết quả: Ghi ra file CARDS.OUT số lượt đi ít nhất tìm được.

Ví dụ:

CARDS.DAT	CARDS.OUT
2 2 1 2 3 3 1 2	3

Hộp thư điện tử

Một người sử dụng mạng ITERNET đặt yêu cầu nhận thông tin về một số chủ đề khác nhau từ một số địa chỉ truy nhập. Chủ của các địa chỉ truy nhập này sẽ gửi thông tin yêu cầu vào hộp thư của người đặt hàng. Mỗi thông tin nhận được từ địa chỉ truy nhập sẽ được ghi vào một danh mục trong máy của người sử dụng dưới dạng một file mà để ngắn gọn ta sẽ gọi là một thông báo. Để thuận tiện cho việc tra cứu, người sử dụng quyết định xây dựng các cặp tài liệu, mỗi cặp chứa các thông tin về cùng một chủ đề. Trước khi đọc tài liệu người sử dụng sẽ sao chép chúng từ danh mục các thông báo nhận được vào các cặp tài liệu tương ứng.

Chương trình hộp thư được gắn trên máy của người sử dụng cho phép sau "*một thao tác*" chuyển từ danh mục các thông báo vào cặp tài liệu:

- Một thông báo từ danh mục hoặc
- Một dãy các thông báo liên tiếp nhau trong danh mục về cùng một chủ đề.

Việc chuyển thông báo không nhất thiết phải bắt đầu từ đầu danh mục.

Cần tìm cách chuyển các thông báo trong danh mục vào các cặp tài liệu tương ứng đòi hỏi số thao tác phải thực hiện là ít nhất.

Ví dụ: Giả sử người sử dụng muốn thu thập thông tin về các chủ đề: A, B, C, D. Giả sử danh mục các thông báo nhận được theo trình tự thuộc về các chủ đề: (A, C, D, C, B, B, C).
Việc di chuyển vào cặp tài liệu có thể thực hiện như sau: Đầu tiên di chuyển 2 thông báo B. Khi đó danh mục còn lại là (A, C, D, C, C). Tiếp đến thực hiện việc di chuyển thông báo D, rồi thông báo A, và cuối cùng di chuyển nốt 3 thông báo C liền nhau. cách làm này đòi hỏi 4 thao tác.

Dữ liệu: Vào từ file văn bản EMAIL.INP gồm một dòng chứa số nguyên dương N ($0 < N \leq 200$) là số thông báo trong danh mục, tiếp đến là N số nguyên là dãy chỉ số của các chủ đề của dãy các thông báo trong danh mục cần chuyển.

Kết quả: Ghi ra file văn bản EMAIL.OUT số thao tác ít nhất cần thực hiện.

Ví dụ:

EMAIL.INP	EMAIL.OUT
7 1 3 4 3 2 2 3	4

Trò chơi với các băng mầu

Cho một bảng gồm M+1 dòng. M dòng đầu có mầu xanh còn dòng cuối cùng(dòng thứ M+1) có màu đỏ. Mỗi một dòng chứa N số, mỗi số là một số nguyên trong khoảng từ 0 đến P - 1. Cho phép thực hiện phép biến đổi sau đây: Cộng thêm các phần tử của một dòng xanh vào các phần tử tương ứng của dòng đỏ đồng thời các số trong dòng đỏ có giá trị lớn hơn P-1 sẽ bị giảm đi P. Trò chơi gọi là giải được nếu có thể áp dụng các phép biến đổi trên để thu được bảng với các phần tử trên dòng đỏ đều là bằng 0.

Hãy xác định xem trò chơi trên có giải được hay không.

Dữ liệu: Vào từ file văn bản VIRT.INP:

- Dòng đầu tiên chứa số nguyên dương K là số lượng test trong file. K nhóm dòng tiếp theo mỗi nhóm mô tả dữ liệu của một test:
    - Dòng đầu tiên chứa 3 số P, N, M ($1 \le N, M \le 100$, $2 \le P \le 255$);
    - M dòng tiếp theo mỗi dòng chứa N số trên một dòng xanh tương ứng;
    - Dòng cuối cùng chứa N số trên dòng đỏ.

Kết quả: Ghi ra file văn bản VIRT.OUT: mỗi dòng ghi kết quả của một test tương ứng trong file dữ liệu: Nếu trò chơi không giải được thì ghi duy nhất một số 0, nếu giải được thì đầu tiên ghi số 1 tiếp đến là N số nguyên, số thứ i cho biết dòng xanh thứ i được cộng thêm vào dòng đỏ bao nhiêu lần.

Ví dụ:

VIRT.INP	VIRT.OUT
2 4 2 2 2 2 2 2 3 3 3 2 4 1 0 2 0 0 0 0 1 2 1	0 1 1 0 0 2

Bài toán phủ bàn cờ quốc tế bởi các quân bài Domino

Đây là bài toán nổi tiếng trong lý thuyết tổ hợp. Nội dung của nó như sau: Cho bàn cờ quốc tế (lưới ô vuông kích thước 8×8) và một số lượng đủ dùng các quân bài domino. Mỗi quân bài domino đặt lên bàn cờ sẽ phủ kín được đúng hai ô của bàn cờ. Đặt lên bàn cờ hai con tốt ở hai ô ở hai góc đối diện của bàn cờ. Hỏi có thể phủ kín các ô còn lại của bàn cờ bởi các quân bài domino sao cho không có quân bài nào bị chờm ra ngoài bàn cờ, không có hai quân bài nào đè lên nhau hay không. Câu trả lời như đã biết là không thể được. Tuy nhiên khi số con tốt không nhất thiết là 2 thì phụ thuộc vào phân bố của chúng trên bàn cờ, bài toán phủ có thể có lời giải. Bài toán cần giải khi đó là đếm xem có bao nhiêu cách phủ khác nhau.

Dữ liệu: vào từ file văn bản DOMCOVER.INP gồm 8 dòng, mỗi dòng có 8 ký tự, trong đó ký tự '.' để chỉ ô trống, ký tự '#' để chỉ ô có đặt con tốt. Số lượng quân tốt không vượt quá 63.

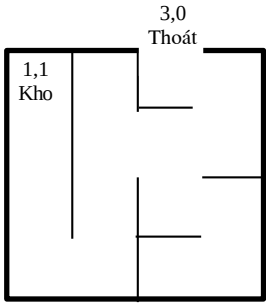
Kết quả: Ghi ra file văn bản DOMCOVER.OUT số lượng cách phủ tìm được.

Ví dụ:

DOMCOVER.INP	DOMCOVER.OUT
#..... ....#	0

DOMCOVER.INP	DOMCOVER.OUT
....#### ....#### ######## ######## ######## ######## ######## ########	5

Tìm đường lấy báu vật



Trong một mê cung kích thước N × N có duy nhất một lối thoát một người săn lùng cổ vật tìm được một kho báu và muốn chuyển nó ra ngoài sau ít bước dịch chuyển nhất. Người săn cổ vật có r quả mìn, mỗi quả có thể phá được một đoạn tường độ dài một đơn vị bên trong mê cung (mìn không thể phá huỷ được tường bao ngoài của mê cung!) Một bước dịch chuyển là việc di chuyển sang một ô kề cạnh cùng với việc phá bức tường ngăn cản việc di chuyển nếu cần thiết. Ô ở góc trên trái được đánh toạ độ là (1, 1).

Dữ liệu: Vào từ file văn bản EXITLAB.INP:

- Dòng đầu tiên chứa các số nguyên N ($2 \le N \le 50$) và r ($0 \le r \le 3$), toạ độ của ô chứa kho báu x, y (x - toạ độ cột, y - toạ độ dòng: $0 < x, y \le N$) và toạ độ u, v của lối thoát ($0 \le u, v \le N+1$);
- Dòng thứ hai chứa số các đoạn tường bên trong mê cung m ($0 \le m \le 2*(N-1)*N$).
- Dòng thứ i trong số m dòng tiếp theo chứa bốn số a_i, b_i, c_i, d_i cho biết đoạn tường thứ i (độ dài 1 đơn vị) ngăn cách ô (a_i, b_i) với ô (c_i, d_i), i = 1, 2, ..., m.

Kết quả: Ghi ra file văn bản EXITLAB.OUT số lượng bước dịch chuyển ít nhất cần thực hiện và số lượng mìn cần sử dụng để thoát mê cung. Nếu có nhiều cách di chuyển như vậy thì hãy chọn trong số chúng cách sử dụng ít mìn nhất để đưa ra. Trong trường hợp không có cách thoát, hãy ghi 0 0.

Ví dụ: (Xem hình vẽ mô tả mê cung)

EXITLAB.INP	EXITLAB.OUT
4 1 1 1 3 0 9 1 1 2 1 2 2 1 2 1 3 2 3 2 1 3 1 3 2 3 1 4 2 4 3 3 3 2 3 3 3 3 4 2 4 3 4	7 1

Điền bảng số

Trò chơi sau đây do Hirofumi Fujiwara đề xuất Trong một lưới ô vuông kích thước 9×9 cần điền các số từ 1 đến 9 sao cho trong mỗi một dòng, mỗi một cột và mỗi một trong số 9 lưới ô vuông con kích thước 3×3 đều có mặt tất cả các số từ 1 đến 9.

Yêu cầu: Cho trước lưới ô vuông trong đó có một số ô đã được điền số, cần xác định xem có thể điền tiếp số vào các ô còn lại để thu được bảng số có tính chất đã nêu hay không.

Dữ liệu: Vào từ file văn bản FUJIWARA.INP gồm 9 dòng mô tả bảng số đã điền bộ phận, mỗi dòng gồm 9 ký tự, ký tự '.' để chỉ ra ô chưa được điền số.

Kết quả: Ghi ra file văn bản FUJIWARA.OUT bảng số tìm được gồm 9 dòng mỗi dòng 9 ký tự, hoặc thông báo 'IMPOSSIBLE' (chữ hoa, không có dấu nháy) nếu không tìm được lời giải.

Ví dụ

FUJIWARA.INP	FUJIWARA.OUT
.....2.3.	185742936
374..6.2.	374916528
.....8.1.	692358714
258.....	258493167
.....	439671852
.....493	716285493
.4.1.....	543167289
.2.8..675	921834675
.6.5.....	867529341

Lưới đèn

Cho một lưới ô vuông kích thước N x M. Mỗi ô của lưới có gắn một bóng đèn màu có hai trạng thái màu xanh và đỏ. Lưới đèn được điều khiển bởi các nút. Mỗi dòng của lưới có một nút đen. Khi ta ấn vào nút đen thì tất cả các bóng đèn trên dòng tương ứng với nút này sẽ chuyển trạng thái màu (tất cả bóng đang ở trạng thái màu xanh sẽ chuyển sang trạng thái màu đỏ và tất cả bóng đang ở trạng thái màu đỏ sẽ chuyển sang trạng thái màu xanh).
Mỗi cột của lưới có một nút trắng. Khi ta ấn đồng thời vào đúng hai nút trắng thì các bóng đèn ở cột thứ nhất sẽ chuyển sang trạng thái của các bóng đèn tương ứng ở cột thứ hai và các bóng đèn ở cột thứ hai sẽ chuyển sang trạng thái của các bóng đèn tương ứng ở cột thứ nhất.
Yêu cầu: Biết trạng thái của lưới đèn ở trạng thái xuất phát. Hỏi có cách ấn các nút điều khiển để đưa lưới đèn về trạng thái đích hay không?

Dữ liệu: Vào từ file văn bản LNET.INP chứa một số bộ dữ liệu:

- Dòng đầu tiên của file LIGHTNET.INP chứa K là số bộ dữ liệu;
- Tiếp đến là K nhóm dòng, mỗi nhóm mô tả một bộ dữ liệu:
    - Dòng đầu tiên của nhóm chứa hai số nguyên dương N, M ($1 \le N, M \le 100$).
    - Tiếp đến là N dòng mô tả trạng thái xuất phát của lưới đèn. Mỗi dòng chứa M từ, mỗi từ hoặc là RED hoặc là BLUE được ghi cách nhau bởi đúng một dấu cách.
    - Cuối cùng là N dòng mô tả trạng thái đích của lưới đèn được ghi theo khuôn dạng giống như đã dùng để mô tả trạng thái xuất phát.

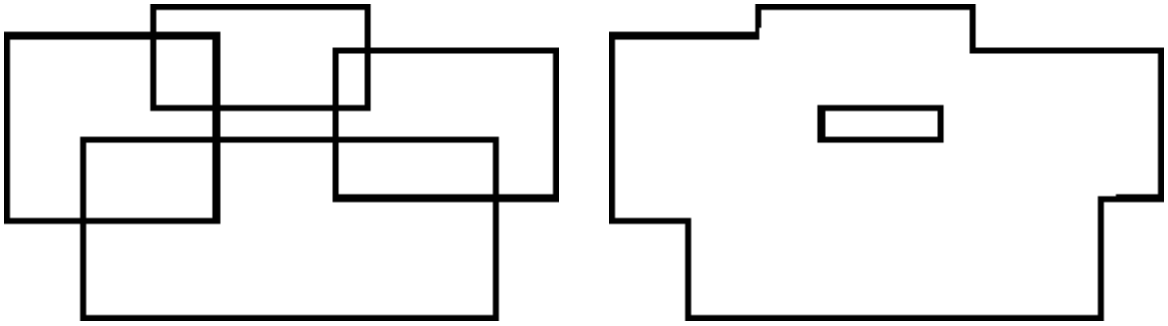
Kết quả: Ghi ra file văn bản LIGHTNET.OUT: dòng thứ i chứa câu trả lời cho bộ dữ liệu thứ i của file dữ liệu (i = 1, 2, ..., K): ghi 'YES' nếu có cách điều khiển về trạng thái đích, ghi 'NO' nếu trái lại.

Ví dụ:

LIGHTNET.INP	LIGHTNET.OUT
2 3 4 BLUE RED BLUE RED RED BLUE BLUE RED BLUE BLUE BLUE BLUE BLUE RED BLUE RED RED RED BLUE BLUE BLUE BLUE BLUE BLUE 2 2 BLUE BLUE BLUE RED RED RED RED RED	YES NO

Các hình chữ nhật

Trên mặt phẳng cho N (N≤ 300) hình chữ nhật có các cạnh song song với các trục toạ độ (xem hình vẽ). Mỗi hình chữ nhật được xác định bởi các cặp toạ độ của hai đỉnh ở hai góc đối diện. Toạ độ của các đỉnh của hình chữ nhật là các số thực với hai chữ số sau dấu phẩy.



Yêu cầu: Tính diện tích của phần mặt phẳng bị phủ bởi các hình chữ nhật đã cho.

Dữ liệu: Vào từ file văn bản RECAREA.INP:

- Dòng đầu tiên chứa số lượng hình chữ nhật N;
- Dòng thứ i trong số N dòng tiếp theo chứa 4 số thực, hai số đầu là toạ độ của đỉnh ở góc dưới trái, hai số sau là toạ độ của đỉnh ở góc trên phải của hình chữ nhật thứ i (i = 1, 2, ..., N).

Kết quả: Ghi ra file văn bản RECAREA.OUT diện tích tìm được chính xác đến 4 chữ số sau dấu phẩy.

Ví dụ:

RECAREA.INP	RECAREA.OUT
4 0 0 10.1 5.2 -3 3 5.36 7 1 6 9 15 8 3 20 8	195.188

Đua mô tô trong thành phố

Trong một cuộc đua mô tô tại thành phố X, người ta chọn đường đua là một đoạn đường phố đi qua N nút giao thông ($1 \le N \le 100$) đánh số từ 1 đến N (điểm xuất phát gọi là nút 0, đích của cuộc đua gọi là nút N+1). Tại mỗi nút giao thông đều có đèn hiệu xanh đỏ, tại thời điểm bắt đầu cuộc đua đèn ở tất cả các nút giao thông là xanh. Mỗi tay đua phải tìm cách đi qua đoạn đường đua với tốc độ không đổi đồng thời không được qua nút giao thông khi đèn tín hiệu là đỏ, ngoại trừ tại thời điểm đèn tín hiệu chuyển đổi từ xanh sang đỏ. Tay đua nào thay đổi tốc độ trong quá trình đua, hoặc chọn tốc độ dưới 5 mét/giây, hoặc qua nút giao thông khi đèn tín hiệu đang là đỏ sẽ bị loại khỏi cuộc thi.

Yêu cầu: Xác định tốc độ của xe để có thể đạt đích với thời gian ngắn nhất.

Dữ liệu: Vào từ file văn bản MOTORACE.INP:

- Dòng đầu tiên chứa số N;
- Dòng thứ i trong số N+1 dòng tiếp theo chứa khoảng cách từ nút giao thông thứ i-1 đến nút i (i = 1, 2, ..., N+1); (khoảng cách là các số nguyên trong khoảng từ 1 đến 10000)
- Dòng thứ j trong số N dòng cuối cùng ghi thời gian của tín hiệu (xanh, đỏ như nhau) của đèn ở nút giao thông thứ j (j = 1, 2, ..., N).

Kết quả: Ghi ra file văn bản MOTORACE.OUT tốc độ tìm được tính chính xác đến 4 chữ số sau dấu phẩy, hoặc ghi số -1 nếu không có cách thực hiện vòng đua.

Ví dụ:

MOTORACE.INP	MOTORACE.OUT
3 200 100 200 200 2 1 9	55.5556

Chợ cá

Bờm câu được N con cá và đem chúng ra chợ cá để bán cho lái buôn. Lái buôn có thể xâu cá thành các xâu với trọng lượng khác nhau. Mỗi xâu có ít nhất một con cá.

Giả sử bạn là người đầu tiên đến mua cá và muốn lựa chọn xâu cá với trọng lượng nào đó. Hỏi có thể tạo được nhiều nhất bao nhiêu loại trọng lượng khác nhau của các xâu cá từ N con cá của Bờm?

Giả sử rằng:

- Số lượng cá N là số nguyên dương và $N \le 500$.
- Trọng lượng của con cá thứ i, ký hiệu là t_i , là số nguyên dương và $t_i \le 1000$.

Dữ liệu: Vào từ file văn bản *MARKET.IN*

- Dòng đầu tiên chứa số N;
- Dòng thứ i trong số N dòng tiếp theo chứa số t_i.

Kết quả: Ghi ra file văn bản *MARKET.OUT* số nguyên K là số các trọng lượng khác nhau.

Ví dụ:

MARKET.IN	**MARKET.OUT**
5 800 200 354 18 182	27

Kích thước fractal

Bờm học được ở trường cách tính kích thước fractal của các cấu hình hình học. Với mục đích này Bờm vẽ một lưới ô vuông kích thước cạnh mỗi ô vuông làmột đơn vị. Tiếp đến nó vẽ cấu hình trên lưới với N đỉnh. N đỉnh này xác định một đa giác không nhất thiết là lồi với các cạnh không tự cắt. Bờm đếm được K1 là số các ô vuông của lưới giao với cấu hình (một ô vuông được coi là có giao với cấu hình nếu nó chứa ít nhất một điểm trong của cấu hình). Sau đó Bờm loại tất cả các đường kẻ đứng của lưới có hoành độ (toạ độ x) là số lẻ và các đường kẻ ngang của lưới với tung độ (toạ độ y) là số lẻ và đếm được K2 ô vuông của lưới thu được có giao với cấu hình. Kích thước fractal sẽ được tính bởi một công thức phụ thuộc vào 2 số K1 và K2 nói trên. Bây giờ Bờm muốn tính kích thước fractal của hồ Tây.

Yêu cầu: Tính hai số K1 và K2.

Giả sử rằng

- Số đỉnh N là số nguyên dương và $2<N<1000$.
- Mỗi đỉnh được cho bởi 2 số nguyên x_i và y_i là toạ độ của nó trong mặt phẳng toạ độ OXY ($-32000 \le x_i, y_i \le 32000$).

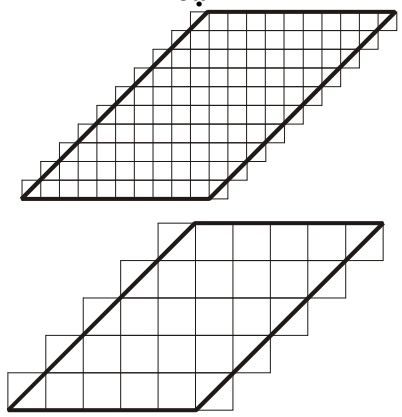
Mỗi đỉnh được nối với đỉnh đứng trước nó và đỉnh đầu tiên được nối với đỉnh cuối cùng.

Dữ liệu: Vào từ file văn bản *FRACTAL.IN*:

Dòng đầu tiên chứa số N. Mỗi dòng trong số N dòng tiếp theo chứa hai số nguyên x_i và y_i.

Kết quả: Ghi ra trên một dòng của file văn bản *FRACTAL.OUT* hai số *K1* và *K2*

Ví dụ:

FRACTAL.IN	**FRACTAL.OUT**	**Hình minh hoạ**
4 0 0 10 0 20 10 10 10	110 30	

Tập lặn

Hàng năm người ta thường tổ chức lớp tập lặn ở hồ Ohrid. Có nhiều bài luyện tập cần phải thực hiện trong quá trình học. Một trong những bài học như vậy là tập lặn từ một bờ đến bờ đối diện củamột bể bơi. Một nhóm gồm *N* học viên chỉ có một bình ôxy để luyện tập. Nhiều nhất chỉ có thể là hai người có thể đồng thời sử dụng chung bình ôxy. Mỗi học viên có một tốc độ lặn, vì thế khi hai người cùng sử dụng chung bình để lặn thì họ sẽ di chuyển với tốc độ của người có tốc độ chậm hơn. Có M cặp học viên không thể bố trí lặn cùng nhau do giữa họ có những điều xích mích đáng ra không nên có.

Yêu cầu: Tìm cách xếp lịch di chuyển tất cả nhóm học viên từ một bờ sang bờ đối diện với thời gian nhanh nhất.
Giả thiết rằng:

- Số học viên *N* là số nguyên dương và $N \le 6000$.
- Các học viên được đánh số bởi các số 1,2, . . .,*N*.
- Số cặp học viên có xích mích *M* là số nguyên dương và $M \le 6000$.
- Thời gian mà học viên *i* có thể lặn từ bờ này sang bờ kia t_i là số nguyên dương.
- Bình ôxy chỉ có thể chuyển từ bờ này sang bờ kia bằng cách lặn. Bình ôxy có dung tích không hạn chế.
- Mỗi dữ liệu vào luôn có ít nhất một lời giải.

Dữ liệu: Vào từ file văn bản *DIVING.IN:*

- Dòng đầu tiên chứa hai số nguyên *N* và *M*.
- Mỗi dòng trong số *N* dòng tiếp theo chứa một số t_i.
- Mỗi một trong số *M* dòng cuối chứa hai chỉ số của hai học viên không thể xếp lặn cùng nhau.

Kết quả: Ghi ra file văn bản *DIVING.OUT*:

- Dòng đầu tiên ghi thời giân ít nhất cần thiết cho việc di chuyển tất cả học viên.
- Các dòng tiếp theo sẽ mô tả lịch di chuyển. Mỗi dòng sẽ mô tả một bước di chuyển bao gồm một hoặc hai số nguyên là chỉ số của các học viên phải lặn với bình ôxy từ bờ này sang bờ kia.

Ví dụ:

DIVING.IN	**DIVING.OUT**
4 2 1 2 1 2 3 4 2 3	6 3 1 1 4 2 3 3 1

Tìm vị trí xây dựng nhà máy bia

Tại đảo H, người ta cần xác định vị trí xây dựng nhà máy bia. N địa điểm dân cư của đảo đều nằm ở rìa biển được nối với nhau bởi một tuyến đường cao tốc chạy vòng quanh đảo. Ban quản lý dự án xây dựng nhà máy, đã khảo sát tình hình thực tế và đã xác định được các thông số sau:

- z_i (thùng)- nhu cầu tiêu thụ bia hàng ngày của địa điểm dân cư i (các địa điểm được đánh số theo thứ tự tăng dần theo một chiều đi vòng quanh tuyến đường cao tốc))
- d_i - khoảng cách từ địa điểm i đến địa điểm tiếp theo trên đường cao tốc (địa điểm tiếp theo của n là 1);
- c - giá cước chuyên chở 1 thùng bia/ km.

Cần xác định vị trí xây dựng nhà máy bia để tổng chi phí chuyên chở bia hàng ngày (bằng tổng chi phí chuyên chở lượng bia tiêu thụ hàng ngày của các địa điểm dân cư từ nhà máy đến chúng) là nhỏ nhất.

Dữ liệu: Vào từ file BRO.IN:

- Dòng đầu tiên chứa số N ($5 \le N \le 10000$);
- Dòng thứ hai chứa giá cước chuyên chở 1 thùng bia/km
- Dòng thứ i trong n dòng tiếp theo chứa hai số z_i d_i ghi cách nhau bởi dấu cách.

Giả thiết là tổng độ dài của tuyến đường cao tốc là không quá 1000000 km.

Kết quả: Ghi ra file BRO.OUT chi phí chuyên chở nhỏ nhất tìm được.

Ví dụ:

BRO.IN	BRO.OUT
6 2 1 2 2 3 1 2 5 2 1 10 2 3	82

Vi rút nhị phân

Khi phát hiện ra rằng một số dãy nhị phân là mã của một số vi rút mới (gọi là vi rút nhị phân), người ta đã tìm cách xác định được tập các mã vi rút. Một dãy nhị phân được gọi là an toàn nếu như mọi đoạn (dãy con gồm các ký tự liên tiếp) của nó đều không là mã của bất cứ vi rút nhị phân nào. Cần xác định xem có tồn tại dãy nhị phân độ dài vô hạn an toàn hay không.

Ví dụ:
Đối với tập vi rút nhị phân {011, 11, 00000}, dãy nhị phân an toàn vô hạn là 010101... . Còn đối với tập vi rút nhị phân {01, 11, 00000} không thể tìm được dãy nhị phân an toàn độ dài vô hạn.

Dữ liệu: Vào từ file WIR.IN:

- Dòng đầu tiên chứa số nguyên n là số lượng vi rút nhị phân;
- Dòng thứ i trong số n dòng tiếp theo chứa mã của vi rút nhị phân thứ i trong tập vi rút nhị phân đang xét là một dãy gồm toàn các số 0 và 1.
- Tổng độ dài của tất cả các mã vi rút không quá 30000.

Kết quả: Ghi ra trên một dòng của file văn bản WIR.OUT thông báo 'YES' nếu tìm được dãy nhị phân độ dài vô hạn an toàn, và 'NO' nếu trái lại.

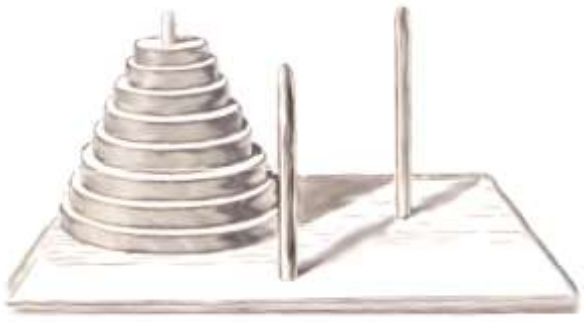
Ví dụ:

WIR.IN	WIR.OUT
3 011 11 00000	NO

WIR.IN	WIR.OUT
3 01 11 00000	YES

Tháp Hà nội

Bài toán tháp Hà nội trở thành nổi tiếng vào năm 1883, sau bài báo của Luca là một nhà toán học người Pháp. Tháp là một cọc đĩa đường kính giảm dần từ dưới lên trên. Bài toán dặt ra là cần chuyển chồng đĩa sang một cọc khác sử dụng một cọc trung gian sao cho trong quá trình chuyển đĩa không có đĩa nào có đường kính lớn hơn lại bị đặt trên đĩa có đường kính nhỏ hơn.



Tháp Hà nội với M=3, N=8

Yêu cầu: Giải bài toán tháp Hà nội tổng quát: Cho M cọc và tháp gồm N đĩa ($3 \le M \le 30$, $1 \le N \le 30$), hãy xác định số lần chuyển đĩa tối thiểu cần thực hiện để chuyển chồng đĩa từ cọc xuất phát sang một cọc đích sử dụng số M-2 cọc còn lại như cọc trung gian.

Dữ liệu: Vào từ file văn bản HANTOWER.INP chứa hai số nguyên N, M được ghi cách nhau theo thứ tự là số đĩa và số cọc trong bài toán tháp Hà nội.

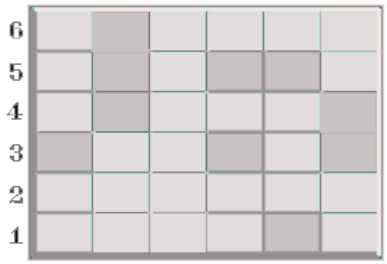
Kết quả: Ghi ra trên một dòng của file văn bản HANTOWER.OUT số lần chuyển đĩa tối thiểu cần thực hiện.

Ví dụ:

HANTOWER.INP	HANTOWER.OUT
5 3	31

Bắn tàu trên biển

Trò chơi Bắn tàu trên biển là trò chơi được nhiều thế hệ học sinh và sinh viên ưa thích. Cho bảng ô vuông kích thước 6×6 có một đội tàu gồm 1 tàu kích thước 3 ô (dãy 3 ô vuông liên tiếp theo chiều dựng đứng hoặc nằm ngang), 2 tàu kích thước 2 ô (dãy 2 ô vuông liên tiếp theo chiều dựng đứng hoặc nằm ngang) và 3 tàu kích thước 1 ô. Các tàu chỉ có thể có chung nhau không quá 1 điểm (nghĩa là không được có cạnh chung).



Bảng trò chơi bắn tàu trên biển với N=6

Yêu cầu: Cho trước một cách xếp một số tàu trên lưới kích thước N×N ($2 \le N \le 15$), cần kiểm tra tính đúng đắn của cách xếp tàu đó. Cách xếp tàu là đúng đắn nếu nó đảm bảo đúng về số lượng và chủng loại các tàu cần xếp và không vi phạm qui tắc xếp tàu. Trong trường hợp cách xếp đã cho là không đúng đắn bạn cần chỉ ra một cách xếp đúng đắn hoặc thông báo là không có cách xếp đúng đắn trên lưới đã cho.

Dữ liệu: Vào từ file SHIP.INP:

- Dòng đầu tiên chứa 2 số N M theo thứ tự là kích thước lưới và số loại tàu cần xếp;
- Dòng thứ i trong số M dòng tiếp theo mô tả thông tin về loại tàu i gồm 2 số nguyên s_i và k_i theo thứ tự là kích thước và số lượng tàu loại i cần xếp.
- N dòng cuối mô tả cách xếp tàu mà bạn cần kiểm tra tính đúng đắn, mỗi dòng gồm N số chỉ là 0 hoặc 1, trong đó 0 để chỉ ra ô rỗng còn 1 cho biết ô đó thuộc vào một con tàu nào đó.

Kết quả: Ghi ra file văn bản SHIP.OUT:

- Nếu cách xếp cho trong file dữ liệu là đúng đắn thì dòng đầu tiên ghi TRUE, ngược lại ghi FALSE;
- Trong trường hợp dòng thứ nhất là FALSE: Nếu không có cách xếp đúng đắn thì dòng thứ hai ghi 'IMPOSSIBLE', ngược lại N dòng tiếp theo ghi thông tin về cách xếp tàu đúng đắn tìm được theo khuôn dạng giống như trong file dữ liệu.

Ví dụ:

SHIP.INP	SHIP.OUT
6 3 3 1 2 2 1 3 0 1 1 0 0 0 0 1 0 1 1 0 0 1 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0	FALSE 0 1 0 0 0 0 0 1 0 1 1 0 0 1 0 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 0

Tìm dòng thông tin đặc biệt

Cho một văn bản gồm N dòng ($1 \le N \le 50000$), mỗi dòng gồm không quá 255 ký tự. Trong file văn bản đã cho có một dòng đặc biệt chỉ xuất hiện trong văn bản đúng một lần, còn các dòng khác đều bị lặp lại và hơn nữa số lần lặp lại là một số lẻ. Bạn cần tìm dòng đặc biệt này.

Dữ liệu: Vào từ file FIRSTROW.INP chứa các dòng của văn bản đã cho, trong đó dòng cuối cùng của file có dấu # đánh dấu kết thúc của văn bản và dòng này không được tính vào văn bản.

Kết quả: Ghi ra trên một dòng của file văn bản FIRSTROW.OUT dòng đặc biệt tìm được.

Ví dụ:

FIRSTROW.INP	FIRSTROW.OUT
- What's your name? - My name's Lan - What's your name? - My name's Lan - I'm from Hanoi, and you? #	- I'm from Hanoi, and you?

Ghi bài hát

Bạn Lan dự định ghi một đĩa CD những bài hát yêu thích từ các chương trình phát thanh của Đài tiếng nói Việt nam. Rất tiếc là Lan chỉ có một máy radio để thực hiện việc thu thanh để ghi đĩa. Cũng may là Lan đã tìm hiểu chương trình phát sóng của Đài để biết được toàn bộ các buổi phát ca nhạc của Đài trong khoảng thời gian dự định ghi đĩa. Từ các bài hát sẽ được phát Lan đã lập ra danh mục các bài hát ưa thích của mình. Máy ghi đĩa CD của Lan có một nhược điểm là không cho phép ngắt quãng khi ghi. Vì thế để ghi được đĩa, mỗi khi ghi xong một bài hát nào đó Lan phải lập tức chuyển sang ghi bài khác.

Yêu cầu: Cho biết thời điểm bắt đầu và kết thúc phát của mỗi một bài hát trong danh mục các bài hát mà Lan đã lựa chọn, bạn hãy giúp Lan tìm cách ghi được nhiều bài hát nhất từ danh mục đó.

Dữ liệu: Vào từ file văn bản CDWRITE.INP chứa N (**$1 \le N \le 5000$**) dòng. Dòng thứ i ghi hai số nguyên dương s_i, t_i theo thứ tự là thời điểm bắt đầu và thời điểm kết thúc việc phát bài hát thứ i trong danh mục của Lan lựa chọn (**$100 \ge t_i - s_i \ge 1$**).

Kết quả: Ghi ra file văn bản CDWRITE.OUT:

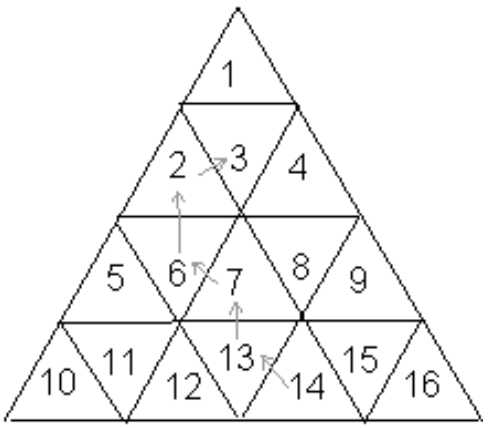
- Dòng đầu tiên ghi k là số bài hát lớn nhất tìm được;
- Dòng thứ i trong số k dòng tiếp theo ghi chỉ số trong danh mục của bài hát thứ i được chọn để ghi. Các bài hát sẽ ghi cần được xếp theo thứ tự thực hiện ghi đĩa.

Ví dụ:

CDWRITE.INP	CDWRITE.OUT
1 3 **1 2** **3 4** 2 4 **5 6** 3 20 **4 5** **2 3**	5 2 8 3 7 5

Lưới tam giác

Lưới tam giác là một tam giác đều được chia thành các tam giác nhỏ bằng cach vẽ các đường thẳng song song với các cạnh và cách đều nhau. Các tam giác con trong lưới được đánh số từ trên xuống dưới, từ trái qua phải bắt đầu từ 1 (xem hình vẽ). Từ một tam giác con bất kỳ chỉ được quyền di chuyển sang tam giác con có chung cạnh với nó. Ta gọi việc di chuyển từ một tam giác con sang tam giác con chung cạnh với nó là một bước di chuyển.



Yêu cầu: Tìm cách di chuyển bắt đầu từ tam giác con với chỉ số N sang tam giác con với chỉ số M sao cho số bước di chuyển cần thực hiện là ít nhất.

Dữ liệu: Vào từ file văn bản TRENET.INP chứa hai số nguyên dương N, M ghi cách nhau bởi dấu cách ($1 \le N, M \le 100000$).

Kết quả: Ghi ra file văn bản TRENET.OUT:

- Dòng đầu tiên ghi số nguyên K là số lượng bước di chuyển ít nhất cần thực hiện.
- K dòng tiếp theo mỗi dòng chứa một chỉ số của tam giác con theo thứ tự trên dường di chuyển tìm được bắt đầu từ chỉ số của tam giác xuất phát và kết thúc bởi chỉ số của tam giác cần đến.

Ví dụ:

TRENET.INP	TRENET.OUT
14 3	5 14 13 7 6 2 3

Đường đi đến số 0

Mỗi một số nguyên dương đều có thể biểu diễn dưới dạng tích của 2 số nguyên dương X, Y sao cho X ≤ Y. Nếu như trong phân tích này ta thay X bởi X-1 còn Y bởi Y+1 thì sau khi tính tích của chúng ta thu được hoặc là một số nguyên dương mới hoặc là số 0.

Ví dụ: số 12 có 3 cách phân tích 1*12, 3*4, 2*6. Cách phân tích thứ nhất cho ta tích mới là 0: (1-1)*(12+1) = 0, cách phân tích thứ hai cho ta tích mới là 10: (3-1)*(4+1) =10, còn cách phân tích thứ ba cho ta 7: (2-1)*(6+1) = 7. Nếu như kết quả là khác không ta lại lặp lại thủ tục này đối với số thu được. Rõ ràng áp dụng liên tiếp thủ tục trên, cuối cùng ta sẽ đến được số 0, không phụ thuộc vào việc ta chọn cách phân tích nào để tiếp tục.

Yêu cầu: Cho trước số nguyên dương N (1 ≤ N ≤ 10000), hãy đưa ra tất cả các số nguyên dương khác nhau có thể gặp trong việc áp dụng thủ tục đã mô tả đối với N.

Dữ liệu: Vào từ file văn bản ZEROPATH.INP chứa số nguyên dương N

Kết quả: Ghi ra file văn bản ZEROPATH.OUT:

- Dòng đầu tiên ghi K là số lượng số tìm được
- Dòng tiếp chứa K số tìm được theo thứ tự tăng dần bắt đầu từ 0.

Lưu ý: Có thể có số xuất hiện trên nhiều đường biến đổi khác nhau, nhưng nó chỉ được tính một lần trong kết quả.

Ví dụ:

ZEROPATH.INP	ZEROPATH.OUT
12	6 0 3 4 6 7 10