Mục lục

Bài A. Đổi chỗ

File dữ liệu vào:	`switch.inp`
File kết quả:	`switch.out`
Hạn chế thời gian:	0.1 giây
Hạn chế bộ nhớ:	256 MB

Người ta định ra một quan hệ hai ngôi đối xứng trên tập hợp các chữ số $\{0, 1, \ldots, 9\}$ gọi là quan hệ có thể đổi chỗ được.

Một số nguyên bất kỳ được biến đổi theo cách sau: đổi chỗ chữ số thứ i và chữ số thứ $i + 1$ nếu hai chữ số này thuộc quan hệ đang xét. Phép biến đổi như thế được ký hiệu là i (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 cho trước sao cho thu được một số nguyên lớn nhất.

Dữ liệu vào

Dòng đầu ghi số cặp chữ số có thể đổi chỗ được.

Các dòng tiếp theo, mỗi dòng ghi một cặp chữ số có thể đổi chỗ được, các chữ số ghi cách nhau ít nhất một dấu trắng.

Dòng cuối cùng ghi số nguyên cần biến đổi, các chữ số ghi sát nhau. Số nguyên này có số chữ số không vượt quá 1000 chữ số, có thể có những chữ số 0 ở đầu.

Kết quả

Dòng đầu ghi số phép biến đổi (có thể bằng 0).

Dòng thứ hai ghi số nguyên lớn nhất thu được.

Ví dụ

switch.inp	switch.out
7 1 7 4 3 6 4 5 6 5 4 9 2 0 8 3546	3 4365

Bài B. Thêm cạnh

File dữ liệu vào:	`addedge.inp`
File kết quả:	`addedge.out`
Hạn chế thời gian:	0.1 giây
Hạn chế bộ nhớ:	256 MB

Bình đang quan sát một đồ thị vô hướng. Bình băn khoăn liệu trong đồ thị này có tồn tại hai đỉnh nào mà sau khi nối hai đỉnh đó lại với nhau thì đồ thị sẽ có thêm duy nhất một chu trình đơn mới. Nhắc lại chu trình đơn là một chuỗi các đỉnh bắt đầu và kết thúc tại cùng một đỉnh mà không được phép lặp lại đỉnh ngoài đỉnh đầu và đỉnh cuối.

Yêu cầu: hãy giúp Bình tính số lượng cặp đỉnh thoả mãn điều kiện trên.

Dữ liệu vào

Dòng đầu tiên chứa hai số nguyên dương n, m ($n, m \leq 10^5$) tương ứng là số đỉnh và số cạnh của đồ thị.

m dòng tiếp theo mỗi dòng chứa hai số nguyên dương u, v ($u, v \leq n$) là hai đỉnh của một cạnh.

Kết quả

Ghi ra duy nhất một số là số cặp đỉnh tìm được.

Ví dụ

addedge.inp	addedge.out
5 4 1 2 2 3 3 4 4 5	6
5 5 1 2 2 3 1 3 3 4 4 5	1

Bài C. Thành phần liên thông mạnh

File dữ liệu vào:	`stdin`
File kết quả:	`stdout`
Hạn chế thời gian:	1 giây
Hạn chế bộ nhớ:	256 MB

Bài toán yêu cầu tính số thành phần liên thông mạnh của đồ thị.

Dữ liệu vào

- Dòng đầu chứa ba số nguyên dương $n \leq 10^4$, $m \leq 2 \times 10^4$, $k \leq 10^4$;
- m dòng sau mỗi dòng chứa hai số nguyên dương u và v tương ứng với một cung (u, v);
- k dòng tiếp theo mỗi dòng chứa hai số nguyên dương i và j tương ứng với một thao tác thêm cung (i, j) vào đồ thị. Mỗi dòng là một yêu cầu truy vấn: sau thao tác thêm cung này yêu cầu chương trình của bạn trả lời số thành phần liên thông mạnh của đồ thị.

Kết quả

- Dòng đầu chứa duy nhất 1 số nguyên là số thành phần liên thông mạnh tìm được của đồ thị.
- k dòng tiếp theo mỗi dòng ghi ra số thành phần liên thông mạnh tìm được của đồ thị tương ứng với yêu cầu truy vấn trong file dữ liệu vào.

Ví dụ

stdin	stdout
7 5 4 1 2 2 3 3 1 5 1 4 5 3 5 6 7 7 4 4 6	5 4 4 4 2

Bài D. Danh sách móc nối

File dữ liệu vào: `linklist.inp`
File kết quả: `linklist.out`
Hạn chế thời gian: 0.1 giây
Hạn chế bộ nhớ: 256 MB

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

Hàng dưới bảng trên chứa thông tin của mảng A kèm các chỉ số mảng ở hàng trên. Mảng A chứa các con trỏ của một danh sách móc nối tuyến tính. Trong bài toán này, một con trỏ đơn giản chỉ là một giá trị nguyên. Con trỏ tại nốt đầu tiên được lưu tại A[0], có nghĩa là giá trị A[0] chỉ ra vị trí của nốt thứ hai. Con trỏ của nốt thứ hai được lưu tại A[A[0]], rồi con trỏ của nốt thứ ba được lưu tại A[A[A[0]]], ... Con trỏ có giá trị -1 biểu thị điểm cuối của danh sách móc nối.

Trong ví dụ trên, giá trị của A[0] là 2 nghĩa là con trỏ thứ hai được lưu tại A[2]. Giá trị của A[2] là 4 nghĩa là con trỏ thứ 3 được lưu tại A[4]. Giá trị của A[4] là -1 nghĩa là không còn nốt nối sau đó nữa. Như vậy chuỗi con trỏ trong danh sách móc nối trên sẽ gồm 3 nốt như sau: A[0]=2 → A[2]=4 → A[4]=-1.

Cho trước các giá trị của mảng con trỏ A và biết rằng có một vị trí đã bị thay đổi giá trị mới nhưng không được biết thông tin về vị trí và giá trị mới đó. Lưu ý là giá trị mới có thể bằng giá trị cũ, nghĩa là bản chất mảng không bị thay đổi. Đây là một ứng dụng trong hình sự để truy vết tội phạm. Vì vậy để khôi phục lại mảng ban đầu thì cần sửa giá trị tại đúng một vị trí trong mảng mới sao cho mảng được sửa biểu diễn một danh sách móc nối có nhiều nốt nhất. Với ví dụ trên:

- Đổi giá trị A[4] thành 6 sẽ tạo ra danh sách móc nối với 4 nốt.
- Đổi giá trị A[0] thành 7 sẽ tạo ra danh sách móc nối với 4 nốt.
- Đổi giá trị A[0] thành 9 sẽ tạo ra danh sách móc nối sai.
- Đổi giá trị A[2] thành 7 sẽ tạo ra danh sách móc nối với 5 nốt.

Trong số tất cả các cách đổi có thể, kể cả cách không đổi gì, thì danh sách móc nối mới tạo ra với 5 nốt là dài nhất. Do vậy nhiều khả năng giá trị ban đầu của A[2] là 7.

Yêu cầu: tìm cách thay đổi giá trị một nốt để có được mảng biểu diễn danh sách móc nối dài nhất bắt đầu từ vị trí 0.

Dữ liệu vào

Dòng đầu tiên chứa duy nhất một số nguyên N là kích thước của mảng A.

Dòng thứ hai ghi ra N số nguyên là các giá trị trong mảng, bắt đầu từ A[0]. Mỗi số nguyên ≥ -1 và $< N$.

Kết quả

Ghi ra duy nhất một số là số nốt của danh sách móc nối có nhiều nốt nhất tìm được.

Ví dụ

linklist.inp	linklist.out
10 2 5 4 4 -1 1 -1 3 0 8	5

Hạn chế

- Subtask 1 (20%): $1 < N \leq 20$
- Subtask 2 (20%): $20 < N \leq 3000$ và số lượng nốt tối đa của danh sách móc nối tìm được không quá 100.
- Subtask 3 (20%): $20 < N \leq 3000$
- Subtask 4 (40%): $3000 < N \leq 20000$