Permutation Game

Alice và Bob là bạn thời thơ ấu và họ thích chơi các trò chơi trí tuệ. Hôm nay, họ đang chơi một trò chơi mới trên đồ thị.

Bộ trò chơi chứa một đồ thị **liên thông** với m đỉnh, được đánh số từ 0 đến $m-1$ và e cạnh, được đánh số từ 0 đến $e-1$. Cạnh thứ i kết nối hai đỉnh $u[i]$ và $v[i]$.

Bộ trò chơi cũng chứa một hoán vị $p[0], p[1], \ldots, p[n-1]$ có độ dài n, trong đó $m \leq n$. Hoán vị là một mảng trong đó mỗi số từ 0 đến $n-1$ xuất hiện đúng một lần, theo một thứ tự nào đó. **Điểm** của hoán vị p là số lượng các chỉ số i sao cho $p[i] = i$.

Trò chơi sẽ kéo dài tối đa 10^{100} lượt. Trong mỗi lượt, những điều sau đây xảy ra:

1. Nếu Alice quyết định kết thúc trò chơi, trò chơi sẽ dừng lại.
2. Nếu không, Alice chọn **các chỉ số riêng biệt** $t[0], t[1], \ldots, t[m-1]$, trong đó $0 \leq t[i] < n$. Lưu ý rằng, trò chơi **không** yêu cầu $t[0] < t[1] < \ldots < t[m-1]$.
3. Bob chọn một chỉ số $0 \leq j < e$ của các cạnh của đồ thị và hoán đổi $p[t[u[j]]]$ và $p[t[v[j]]]$.

Alice muốn tối đa hóa điểm cuối cùng của hoán vị trong khi Bob muốn cực tiểu hoá điểm cuối cùng của hoán vị.

Nhiệm vụ của bạn là giúp Alice chơi với Bob, trong đó các nước đi của Bob được mô phỏng bởi trình chấm.

Định nghĩa *điểm tối ưu* là điểm cuối cùng của hoán vị nếu cả Alice và Bob đều chơi tối ưu.

Bạn cần xác định điểm tối ưu của hoán vị và sau đó chơi trò chơi với Bob một số lượt để đạt được điểm ít nhất là điểm tối ưu đó.

Lưu ý rằng chiến lược của Alice cần hiệu quả bất kể Bob đi như thế nào, kể cả khi Bob đi nước không tối ưu.

Chi tiết cài đặt

Bạn cần cài đặt hàm sau:

```
int Alice(int m, int e, std::vector<int> u, std::vector<int> v,
          int n, std::vector<int> p)
```

- m: số đỉnh trong đồ thị.
- e: số cạnh trong đồ thị.
- u và v: mảng có độ dài e mô tả các cạnh của đồ thị.
- n: độ dài của hoán vị.
- p: mảng có độ dài n mô tả hoán vị.
- Hàm này này được gọi đúng một lần.
- Hàm này sẽ trả về một số nguyên – điểm tối ưu của trò chơi.

Trong hàm này, bạn có thể gọi hàm sau:

```
int Bob(std::vector<int> t)
```

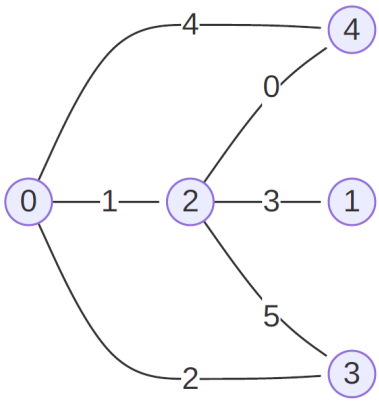
- t: một mảng có kích thước m, chứa các chỉ số phân biệt, trong đó $0 \le t[i] < n$ và $t[i] \ne t[j]$ với mọi $i \ne j$.
- Hàm này trả về một số nguyên j duy nhất thỏa mãn $0 \le j < e$.
- Hàm này có thể được gọi nhiều lần.

Ví dụ

Xét lời gọi sau:

```
Alice(5, 6, [4, 0, 3, 1, 4, 2], [2, 2, 0, 2, 0, 3],
      10, [8, 2, 7, 6, 1, 5, 0, 9, 3, 4])
```

Đồ thị như sau:



và p ban đầu là $[8, 2, 7, 6, 1, 5, 0, 9, 3, 4]$.

Với những ràng buộc trên, ta có thể chứng minh được rằng điểm tối ưu của hoán vị là 1.

Giả sử, Alice thực hiện 4 nước đi sau:

Đối số của t cho `Bob`	Giá trị trả về của `Bob`	Chỉ số tương ứng của p	p sau khi hoán đổi bởi Bob
$[3,1,5,2,0]$	5	$5,2$	$[8,2,5,6,1,7,0,9,3,4]$
$[9,3,7,2,1]$	0	$1,7$	$[8,9,5,6,1,7,0,2,3,4]$
$[5,6,7,8,9]$	1	$5,7$	$[8,9,5,6,1,2,0,7,3,4]$
$[7,5,2,3,6]$	3	$5,2$	$[8,9,2,6,1,5,0,7,3,4]$

Lưu ý rằng Alice và Bob không nhất thiết phải đi các nước tối ưu. Các nước đi ở đây được hiển thị chỉ nhằm mục đích minh họa. Cũng cần lưu ý rằng Alice có thể kết thúc trò chơi ngay lập tức, nếu điểm ban đầu của hoán vị đã là 1 rồi.

Sau khi Alice thực hiện tất cả các nước đi ở trên, điểm hiện tại của hoán vị là 3 ($p[2]=2, p[5]=5, p[7]=7$).

Cuối cùng, hàm `Alice()` sẽ trả về 1 – điểm tối ưu của hoán vị.

Lưu ý rằng mặc dù Alice đã đạt được điểm 3 khi chơi với Bob, bạn sẽ nhận được 0 điểm nếu giá trị trả về của `Alice()` là 3 thay vì 1.

Ràng buộc

- $2 \le m \le 400$
- $m-1 \le e \le 400$
- $0 \le u[i], v[i] < m$
- $m \le n \le 400$
- $0 \le p[i] < n$
- Đồ thị liên thông, không có khuyên hoặc cạnh lặp.
- p là một hoán vị, nghĩa là $p[i] \ne p[j]$ với mọi $i \ne j$.

Subtask

1. (6 điểm) $m = 2$
2. (6 điểm) $e > m$
3. (10 điểm) $e = m - 1$
4. (24 điểm) $e = m = 3$
5. (24 điểm) $e = m = 4$
6. (30 điểm) $e = m$

Đối với mỗi subtask, bạn có thể nhận được điểm một phần. Giả sử r là tỷ lệ lớn nhất của $\frac{k}{n}$ trong số tất cả các trường hợp thử nghiệm của một subtask, trong đó k là số lượt (nghĩa là các cuộc gọi đến `Bob()`). Như vậy, điểm của bạn cho subtask đó được nhân với số sau:

Điều kiện	Hệ số nhân
$12 \leq r$	0
$3 < r < 12$	$1 - \log_{10}(r-2)$
$r \leq 3$	1

Cụ thể, nếu chương trình của bạn chạy trong vòng $3n$ lượt, bạn sẽ nhận được điểm tối đa cho subtask đó. Sử dụng hơn $12n$ lượt sẽ dẫn đến việc nhận được 0 điểm cho subtask đó (hiển thị là `Output isn't correct`).

Trình chấm mẫu

Trình chấm mẫu đọc dữ liệu vào theo định dạng sau:

- dòng 1: $m\ e$
- dòng $2+i$ $(0 \leq i \leq e-1)$: $u[i]\ v[i]$
- dòng $2+e$: n
- dòng $3+e$: $p[0]\ p[1]\ \ldots\ p[n-1]$

Trình chấm mẫu ghi kết quả ra theo định dạng sau:

- dòng 1: hoán vị cuối cùng p
- dòng 2: giá trị trả về của `Alice()`
- dòng 3: điểm hiện tại của hoán vị cuối cùng
- dòng 4: số lượng lượt