



자리 배치

직사각형의 강당에서 프로그래밍 대회를 열려고 한다. 강당은 H 개의 행과 W 개의 열이 있는 2차원 격자 칸이다. 행들은 0부터 $H - 1$ 까지 번호가 붙어 있다. 열들은 0부터 $W - 1$ 까지 번호가 붙어 있다. 격자에서 r 번째 행, c 번째 열에 있는 자리를 (r, c) 로 표현한다. 초대된 참가자는 HW 명이다. 참가자들은 0부터 $HW - 1$ 까지 번호를 가진다. 당신은 참가자들의 초기 자리 배치를 정해서 i 번 참가자를 (R_i, C_i) 자리에 배치했다. 한 자리에는 한 명의 참가자가 배치되어 있다.

강당에 있는 자리들의 집합 S 에 대해서 다음 조건을 만족하는 정수 r_1, r_2, c_1, c_2 가 존재하면 S 를 **직사각형 집합**이라고 한다.

- $0 \leq r_1 \leq r_2 \leq H - 1$.
- $0 \leq c_1 \leq c_2 \leq W - 1$.
- S 는 $r_1 \leq r \leq r_2$ 와 $c_1 \leq c \leq c_2$ 를 만족하는 자리 (r, c) 들의 집합과 정확히 같다.

k ($1 \leq k \leq HW$)개의 자리로 구성된 직사각형 집합은 다음 조건을 만족할 때 **아름답다**고 한다: 그 직사각형 집합에 배치된 참가자들이 정확히 0번 부터 $k - 1$ 번까지의 번호를 가진다. 자리 배치의 **아름다운 정도**는 존재하는 모든 아름다운 직사각형 집합의 개수이다.

초기 자리 배치를 결정한 후에, 당신은 2명의 위치를 교환해 달라는 요청을 Q 개 받게 된다. 정확히 말하면, 전체 Q 개의 요청이 주어지고, 각 요청은 시간 순서대로 0번부터 $Q - 1$ 번까지 번호가 붙어 있다. 요청들 중 j 번 ($0 \leq j \leq Q - 1$)은 A_j 번 참가자와 B_j 번 참가자의 위치를 교환해 달라는 것이다. 당신은 요청을 받은 즉시 자리 교환을 수행한다. 각 자리 교환을 수행한 다음, 그 때 자리 배치의 아름다운 정도를 계산하려고 한다.

Implementation details

다음 함수를 구현해야 한다.

```
give_initial_chart(int H, int W, int[] R, int[] C)
```

- H, W : 행과 열의 개수
- R, C : 초기 자리 배치가 저장된 크기 HW 인 배열.
- 이 함수는 최초에 정확히 한번만 호출된다. 이 함수가 호출된 이후에 `swap_seats` 함수가 호출된다.

```
int swap_seats(int a, int b)
```

- 한번의 자리 교환에 대한 호출이다.

- a, b : 자리를 교환할 두 참가자의 번호이다.
- 이 함수는 Q 번 호출된다.
- 이 함수는 자리 교환 직후의 아름다운 정도를 리턴해야 한다.

Example

$H = 2, W = 3, R = [0, 1, 1, 0, 0, 1], C = [0, 0, 1, 1, 2, 2], Q = 2$ 라고 하자.

그레이더는 처음에 `give_initial_chart(2, 3, [0, 1, 1, 0, 0, 1], [0, 0, 1, 1, 2, 2])`를 호출한다.

초기 자리 배치는 다음과 같다.

0	3	4
1	2	5

그레이더가 `swap_seats(0, 5)`을 호출한다고 하자. 이 요청 직후에 자리 배치는 다음과 같다.

5	3	4
1	2	0

참가자들 번호 범위가 $\{0\}, \{0, 1, 2\}, \{0, 1, 2, 3, 4, 5\}$ 에 해당하는 자리 집합들이 아름다운 직사각형 집합들이다. 따라서, 현재 자리 배치의 아름다운 정도는 3이다. `swap_seats`는 3을 리턴해야 한다.

그레이더가 `swap_seats(0, 5)`을 다시 호출했다고 하자. 이 요청 직후에 자리 배치는 다시 초기 상태와 같아진다. 참가자들 번호 범위가 $\{0\}, \{0, 1\}, \{0, 1, 2, 3\}, \{0, 1, 2, 3, 4, 5\}$ 에 해당하는 자리 집합들이 아름다운 직사각형 집합이다. 따라서, 현재 자리 배치의 아름다운 정도는 4이다. `swap_seats`는 4를 리턴해야 한다.

압축된 첨부 패키지 파일의 `sample-01-in.txt`와 `sample-01-out.txt`는 이 예제에 대응한다. 다른 입출력 예제도 이 패키지에 포함되어 있다.

Constraints

- $1 \leq H$

- $1 \leq W$
- $HW \leq 1\,000\,000$
- $0 \leq R_i \leq H - 1$ ($0 \leq i \leq HW - 1$)
- $0 \leq C_i \leq W - 1$ ($0 \leq i \leq HW - 1$)
- $(R_i, C_i) \neq (R_j, C_j)$ ($0 \leq i < j \leq HW - 1$)
- $1 \leq Q \leq 50\,000$
- $0 \leq a \leq HW - 1$ (모든 `swap_seats` 호출에서)
- $0 \leq b \leq HW - 1$ (모든 `swap_seats` 호출에서)
- $a \neq b$ (모든 `swap_seats` 호출에서)

Subtasks

1. (5 points) $HW \leq 100, Q \leq 5\,000$
2. (6 points) $HW \leq 10\,000, Q \leq 5\,000$
3. (20 points) $H \leq 1\,000, W \leq 1\,000, Q \leq 5\,000$
4. (6 points) $Q \leq 5\,000, |a - b| \leq 10\,000$ (모든 `swap_seats` 호출에서)
5. (33 points) $H = 1$
6. (30 points) 추가적인 제한이 없음

Sample grader

샘플 그레이더는 다음 형식으로 입력을 받는다.

- line 1: $H\ W\ Q$
- line $2 + i$ ($0 \leq i \leq HW - 1$): $R_i\ C_i$
- line $2 + HW + j$ ($0 \leq j \leq Q - 1$): $A_j\ B_j$

여기서, A_j 와 B_j 는 j 번 `swap_seats` 호출에 주어지는 값들이다. ($0 \leq j \leq Q - 1$)

샘플 그레이더는 다음과 같이 당신의 답을 출력한다.

- line $1 + j$ ($0 \leq j \leq Q - 1$): j 번 `swap_seats` 호출의 리턴 값. ($0 \leq j \leq Q - 1$)