



## Отургучтар

Сен эл аралык программалоо мелдешин уюштургуң тик бурчтук залда келет. Залда  $H$  сапта жана  $W$  колонкада тургузулган  $HW$  отургуч бар. Саптар 0дөн  $(H - 1)$ га дейре нумурланган, колонкалар 0дөн  $(W - 1)$  га дейре нумурланган.  $r$ -инчи сапта жана  $c$ -инчи колонкада турган отургуч  $(r, c)$  аркылуу белгиленет. Сен  $HW$  (0дөн  $(HW - 1)$ га дейре нумурланган) катышуучуну чакырдың. Дагы, сен (ар бир)  $i$ -инчи  $(0 \leq i \leq HW - 1)$  катышуучуга көрсөтүлгөн  $(R_i, C_i)$ -отургучтун тизмесин туздүң. Бул тизме ар бир отургучка бир гана катышуучуну жиберет. Эгерде  $0 \leq r_1 \leq r_2 \leq H - 1$ ,  $0 \leq c_1 \leq c_2 \leq W - 1$  анда  $r_1 \leq r \leq r_2$  and  $c_1 \leq c \leq c_2$  шартына канааттандыруучу  $(r, c)$ -отургучтардын  $S$  көптүгү **тик бурчтук** деп аталат. Эгерде  $S$  тик бурчтук көптүгү  $k$  ( $1 \leq k \leq HW$ ) отургучка ээ болсо жана ал отургучтарда отурган катышуучулар 0дөн  $(k - 1)$ га дейре нумурларга ээлер болсо, анда  $S$  **кооз** деп аталат. Тизмедеги кооз тик бурчтуктун саны ал тизменин **кооздугу** деп аталат. Сенин тизмең даярдаган кийин, сен эки катышуучуга көрсөтүлгөн эки отургучту алмаштыруу боюнча бир канча арызды алдың. Тагыраак, анда убакыт боюнча 0дөн  $(Q - 1)$ га дейре нумурланган  $Q$  арыз бар.  $j$ -инчи  $(0 \leq j \leq Q - 1)$  арызда  $A_j$ -инчи жана  $B_j$ -инчи катышуучуларга көрсөтүлгөн отургучтарды алмаштыруу жазылды. Ар бир арызды кабыл алып, сен тизмени ыкчам жаңылайсың. Ар бир жаңылоодон кийин, сенин максатың пайда болгон тизменин кооздугун эсептөө.

## Implementation details

You should implement the following procedure and function:

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

- $H, W$ : the number of rows and the number of columns.
- $R, C$ : arrays of length  $HW$  representing the initial seating chart.
- This procedure is called exactly once, and before any call to `swap_seats`.

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

- This function describes a request to swap two seats.
- $a, b$ : contestants whose seats are to be swapped.
- This function is called  $Q$  times.
- This function should return the beauty of the seating chart after the swap.

## Example

Let  $H = 2$ ,  $W = 3$ ,  $R = [0, 1, 1, 0, 0, 1]$ ,  $C = [0, 0, 1, 1, 2, 2]$ , and  $Q = 2$ .

The grader first calls `give_initial_chart(2, 3, [0, 1, 1, 0, 0, 1], [0, 0, 1, 1, 2, 2])`.

At first, the seating chart is as follows.

0	3	4
1	2	5

Let's say the grader calls `swap_seats(0, 5)`. After the request 0, the seating chart is as follows.

5	3	4
1	2	0

The sets of seats corresponding to the contestants  $\{0\}$ ,  $\{0, 1, 2\}$ , and  $\{0, 1, 2, 3, 4, 5\}$  are rectangular and beautiful. Thus, the beauty of this seating chart is 3, and `swap_seats` should return 3.

Let's say the grader calls `swap_seats(0, 5)` again. After the request 1, the seating chart goes back to the initial state. The sets of seats corresponding to the contestants  $\{0\}$ ,  $\{0, 1\}$ ,  $\{0, 1, 2, 3\}$ , and  $\{0, 1, 2, 3, 4, 5\}$  are rectangular and beautiful. Hence, the beauty of this seating chart is 4, and `swap_seats` should return 4.

The files `sample-01-in.txt` and `sample-01-out.txt` in the zipped attachment package correspond to this example. Other sample inputs/outputs are also available in the package.

## Constraints

- $1 \leq H$
- $1 \leq W$

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

## 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)  $Q \leq 5\,000$ ,  $|a - b| \leq 10\,000$  for any call to `swap_seats`
5. (33 points)  $H = 1$
6. (30 points) No additional constraints

## Sample grader

The sample grader reads the input in the following format:

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

Here,  $A_j$  and  $B_j$  are parameters for the call to `swap_seats` for the request  $j$ .

The sample grader prints your answers in the following format:

- line  $1 + j$  ( $0 \leq j \leq Q - 1$ ): the return value of `swap_seats` for the request  $j$