

# Sorting

Aizhan possui uma sequência de inteiros  $S[0], S[1], \dots, S[N - 1]$ . A sequência consiste de números distintos de  $0$  a  $N - 1$ . Ela está tentando ordenar essa sequência em ordem crescente, trocando alguns pares de elementos. Seu amigo Ermek também vai trocar alguns pares de elementos — não necessariamente de forma a ajudar Aizhan.

Ermek e Aizhan irão modificar a sequência em uma série de turnos. Em cada turno, primeiro Ermek executa uma troca e então Aizhan executa outra troca. Mais precisamente, a pessoa executando a troca escolhe dois índices válidos e troca os elementos correspondentes a esses índices. Note que os dois índices não precisam ser distintos. Se eles são iguais, a pessoa corrente troca um elemento com ele mesmo, o que não altera a sequência.

Aizhan sabe que Ermek não se importa em ordenar a sequência  $S$ . Ela também conhece os índices exatos que Ermek irá escolher. Ermek planeja tomar parte em  $M$  turnos de trocas. Vamos numerar esses turnos de  $0$  a  $M - 1$ . Para cada  $i$  entre  $0$  e  $M - 1$  inclusive, Ermek escolherá os índices  $X[i]$  e  $Y[i]$  no turno  $i$ .

Aizhan deseja ordenar a sequência  $S$ . Antes de cada turno, se Aizhan notar que a sequência já está ordenada, ela terminará o processo. Dados a sequência original  $S$  e os índices que Ermek irá escolher, sua tarefa é encontrar uma sequência de trocas que Aizhan pode usar para ordenar a sequência  $S$ . Adicionalmente, em algumas sub-tarefas você deve também encontrar uma sequência de trocas que seja tão curta quanto possível. Você pode considerar que é possível ordenar a sequência  $S$  em  $M$  ou menos turnos.

Note que se Aizhan notar que a sequência  $S$  está ordenada após uma troca de Ermek, ela pode escolher trocar dois índices iguais (por exemplo  $0$  e  $0$ ). Como resultado a sequência  $S$  também estará ordenada após o término do turno, de forma que Aizhan atinge seu objetivo. Note também que se a sequência original  $S$  já está ordenada, o número mínimo de turnos necessários para ordená-la é  $0$ .

## Exemplo 1

Considere que:

- A sequência inicial é  $S = 4, 3, 2, 1, 0$ .
- Ermek se dispõe a fazer  $M = 6$  trocas.
- As sequências  $X$  e  $Y$  que descrevem os índices que Ermek vai escolher são  $X = 0, 1, 2, 3, 0, 1$  e  $Y = 1, 2, 3, 4, 1, 2$ . Em outras palavras, os pares de índices que Ermek planeja escolher são  $(0, 1)$ ,  $(1, 2)$ ,  $(2, 3)$ ,  $(3, 4)$ ,  $(0, 1)$  e  $(1, 2)$ .

Nesse cenário Aizhan pode ordenar a sequência  $S$  na ordem  $0, 1, 2, 3, 4$  em três turnos. Ela pode fazê-lo escolhendo os pares de índices  $(0, 4)$ ,  $(1, 3)$  e então  $(3, 4)$ .

A seguinte tabela mostra como Ermek e Aizhan modificam a sequência.

Turno	Jogador	Par de índices para troca	Sequência
início			4, 3, 2, 1, 0
0	Ermek	(0, 1)	3, 4, 2, 1, 0
0	Aizhan	(0, 4)	0, 4, 2, 1, 3
1	Ermek	(1, 2)	0, 2, 4, 1, 3
1	Aizhan	(1, 3)	0, 1, 4, 2, 3
2	Ermek	(2, 3)	0, 1, 2, 4, 3
2	Aizhan	(3, 4)	0, 1, 2, 3, 4

## Exemplo 2

Considere que:

- A sequência inicial é  $S = 3, 0, 4, 2, 1$ .
- Ermek se dispõe a fazer  $M = 5$  trocas.
- Os pares de índices que Ermek planeja escolher são  $(1, 1)$ ,  $(4, 0)$ ,  $(2, 3)$ ,  $(1, 4)$ , e  $(0, 4)$ .

Nesse cenário Aizhan pode ordenar a sequência  $S$  em três turnos, por exemplo escolhendo os pares de índices  $(1, 4)$ ,  $(4, 2)$ , e então  $(2, 2)$ .

A seguinte tabela mostra como Ermek e Aizhan modificam a sequência.

Turno	Jogador	Par de índices para troca	Sequência
início			3, 0, 4, 2, 1
0	Ermek	(1, 1)	3, 0, 4, 2, 1
0	Aizhan	(1, 4)	3, 1, 4, 2, 0
1	Ermek	(4, 0)	0, 1, 4, 2, 3
1	Aizhan	(4, 2)	0, 1, 3, 2, 4
2	Ermek	(2, 3)	0, 1, 2, 3, 4
2	Aizhan	(2, 2)	0, 1, 2, 3, 4

## Tarefa

Você receberá a sequência  $S$ , o número  $M$  e as sequências de índices  $X$  e  $Y$ . Compute a sequência de trocas que Aizhan pode usar para ordenar a sequência  $S$ . Nas sub-tarefas 5 e 6 você deve computar a menor sequência de tarefas possível.

Você deve implementar a função `findSwapPairs`:

- `findSwapPairs(N, S, M, X, Y, P, Q)` — Essa função será chamada pelo avaliador exatamente uma vez.
  - $N$ : o comprimento da sequência  $S$ .
  - $S$ : um vetor de inteiros contendo a sequência inicial  $S$ .

- $M$ : o número de trocas que Ermek planeja fazer.
- $X, Y$ : vetores de inteiros de comprimento  $M$ . Para  $0 \leq i \leq M - 1$ , no turno  $i$  Ermek planeja trocar os números de índices  $X[i]$  e  $Y[i]$ .
- $P, Q$ : vetores de inteiros. Use esses vetores para informar uma possível sequência de trocas que Aizhan pode fazer para ordenar a sequência  $S$ . Denote por  $R$  o comprimento da sequência de trocas que seu programa encontrou. Para cada  $i$  entre  $0$  e  $R - 1$  inclusive, os índices que Aizhan deve escolher no turno  $i$  devem ser armazenados em  $P[i]$  e  $Q[i]$ . Você pode considerar que os vetores  $P$  e  $Q$  já foram alocados com  $M$  elementos cada.
- Esta função deve retornar o valor de  $R$  (definido acima).

## Sub-tarefas

sub-tarefa	pontos	$N$	$M$	restrições extras para $X, Y$	requisitos em $R$
1	8	$1 \leq N \leq 5$	$M = N^2$	$X[i] = Y[i] = 0$ para todo $i$	$R \leq M$
2	12	$1 \leq N \leq 100$	$M = 30N$	$X[i] = Y[i] = 0$ para todo $i$	$R \leq M$
3	16	$1 \leq N \leq 100$	$M = 30N$	$X[i] = 0, Y[i] = 1$ para todo $i$	$R \leq M$
4	18	$1 \leq N \leq 500$	$M = 30N$	nenhuma	$R \leq M$
5	20	$6 \leq N \leq 2,000$	$M = 3N$	nenhuma	mínimo possível
6	26	$6 \leq N \leq 200,000$	$M = 3N$	nenhuma	mínimo possível

Você pode considerar que existe uma solução que requer  $M$  ou menos turnos.

### Avaliador exemplo

O avaliador exemplo lê a entrada do arquivo `sorting.in`, no seguinte formato:

- linha 1:  $N$
- linha 2:  $S[0] \dots S[N - 1]$
- linha 3:  $M$
- linhas 4, ...,  $M + 3$ :  $X[i] Y[i]$

O avaliador exemplo escreve na saída:

- linha 1: o valor  $R$  retornado por `findSwapPairs`
- linha  $2+i$ , para  $0 \leq i < R$ :  $P[i] Q[i]$