Ordinamento

Aizhan possiede una sequenza di N interi $S[0], S[1], \ldots, S[N-1]$ composta da numeri distinti da 0 a $N-1$, e sta provando ad ordinarla in ordine crescente tramite scambi di coppie di elementi. Anche il suo amico Ermek scambierà alcune coppie di elementi — ma non necessariamente in un modo utile alla nostra protagonista.

Ermek e Aizhan modificheranno la sequenza in una serie di round. In ciascun round prima Ermek farà uno scambio, e poi Aizhan ne farà un altro. Più precisamente, la persona che sta facendo lo scambio sceglierà due indici validi e scambierà gli elementi situati in quegli indici. I due indici non devono necessariamente essere distinti: se sono uguali, un elemento verrà scambiato con sé stesso, il che non modifica la sequenza.

Aizhan sa che a Ermek non importa di ordinare la sequenza S, però sa esattamente fin dall'inizio quali sono gli indici che sceglierà. Ermek ha intenzione di partecipare ad M round di scambi, numerati da 0 a $M-1$, e per ogni round i Ermek sceglierà gli indici $X[i]$ e $Y[i]$.

Dato che Aizhan vuole ordinare la sequenza, se si accorge che è già ordinata all'inizio di un round terminerà l'intero processo. Data la sequenza originale S e gli indici che verranno scelti da Ermek, il tuo compito è trovare una sequenza di scambi che Aizhan può usare per ordinare la sequenza S. Inoltre, in alcuni subtask è richiesto di trovare la sequenza di scambi *più corta* possibile. Si può assumere che sia sempre possibile ordinare la sequenza S in M round o meno.

Nota: se Aizhan vede che la sequenza S è ordinata dopo lo scambio di Ermek, può scegliere di scambiare due indici uguali (per esempio, 0 e 0), facendo sì che la sequenza risulti ordinata anche dopo l'intero round, e raggiungendo così il suo obiettivo. Inoltre, se la sequenza iniziale è già ordinata, il minimo numero di round necessari per ordinarla è 0.

Esempio 1

Supponiamo che:

- La sequenza iniziale sia $S = 4, 3, 2, 1, 0$.
- Ermek partecipi a $M = 6$ scambi.
- Le sequenze X e Y che descrivono gli indici scelti da Ermek siano $X = 0, 1, 2, 3, 0, 1$ e $Y = 1, 2, 3, 4, 1, 2$. In altre parole, le coppie di indici che Ermek intende scegliere sono $(0, 1)$, $(1, 2)$, $(2, 3)$, $(3, 4)$, $(0, 1)$, e $(1, 2)$.

In questo caso Aizhan può riordinare la sequenza S nella sequenza $0, 1, 2, 3, 4$ in tre round, scegliendo gli indici $(0, 4)$, $(1, 3)$, e $(3, 4)$.

La seguente tabella mostra come Ermek e Aizhan modificano la sequenza.

Round	Giocatore	Indici scambiati	Sequenza
inizio			$4, 3, 2, 1, 0$
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	$(3, 4)$	$0, 1, 2, 3, 4$

Esempio 2

Supponiamo che:

- La sequenza iniziale sia $S = 3, 0, 4, 2, 1$.
- Ermek partecipi a $M = 5$ scambi.
- Le coppie di indici scelte da Ermek siano $(1, 1)$, $(4, 0)$, $(2, 3)$, $(1, 4)$, e $(0, 4)$.

Anche in questo caso Aizhan può riordinare la sequenza S in tre round, ad esempio scegliendo le coppie di indici $(1, 4)$, $(4, 2)$, e $(2, 2)$, come descritto nella tabella seguente.

Round	Giocatore	Indici scambiati	Sequenza
inizio			$3, 0, 4, 2, 1$
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	$(2, 2)$	$0, 1, 2, 3, 4$

Implementazione

Ti verrà fornita la sequenza S, il numero M e le sequenze di indici X e Y. Calcola una sequenza di scambi che Aizhan può usare per ordinare la sequenza S. Inoltre, nei subtask 5 e 6 la sequenza trovata dovrà essere la più corta possibile.

Dovrai implementare la funzione `findSwapPairs`:

- `findSwapPairs(N, S, M, X, Y, P, Q)` — Questa funzione verrà chiamata dal grader esattamente una volta.
  - `N`: la lunghezza della sequenza S.
  - `S`: un array contenente la sequenza iniziale S.

- `M`: il numero di scambi che Ermek ha intenzione di fare.
- `X, Y`: due array di interi di lunghezza M, tali che per $0 \le i \le M-1$, nel round i Ermek scambierà i numeri posti negli indici $X[i]$ e $Y[i]$.
- `P, Q`: due array di interi, in cui dovrai inserire una possibile sequenza di scambi che Aizhan può fare per ordinare la sequenza S. Se R è la lunghezza della sequenza che il tuo programma ha trovato, per ogni $0 \le i \le M-1$ gli indici che Aizhan dovrebbe scegliere nel round i devono essere memorizzati in $P[i]$ e $Q[i]$. Puoi assumere che gli array `P` e `Q` siano già stati allocati a M elementi ciascuno.
- Questa funzione dovrà restituire il valore R (definito sopra).

Subtask

subtask	punti	N	M	vincoli su X, Y	vincoli su R
1	8	$1 \le N \le 5$	$M = N^2$	$X[i] = Y[i] = 0$ per ogni i	$R \le M$
2	12	$1 \le N \le 100$	$M = 30N$	$X[i] = Y[i] = 0$ per ogni i	$R \le M$
3	16	$1 \le N \le 100$	$M = 30N$	$X[i] = 0, Y[i] = 1$ per ogni i	$R \le M$
4	18	$1 \le N \le 500$	$M = 30N$	nessuno	$R \le M$
5	20	$6 \le N \le 2000$	$M = 3N$	nessuno	minimo possibile
6	26	$6 \le N \le 200\,000$	$M = 3N$	nessuno	minimo possibile

Puoi assumere che esista sempre una soluzione che richiede M o meno round.

Grader di prova

Il grader di prova legge l'input dal file `sorting.in` nel formato seguente:

- riga 1: `N`
- riga 2: `S[0] ... S[N - 1]`
- riga 3: `M`
- righe $4, \ldots, M+3$: `X[i] Y[i]`

Il grader di prova stampa il seguente output:

- riga 1: il valore R restituito da `findSwapPairs`
- riga $2+i$, per $0 \le i < R$: `P[i] Q[i]`