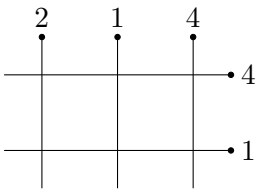
Political cost (cost)

Nella città in cui vivi ci sono N strade che vanno da Est a Ovest (dalla strada 0 alla strada $(N-1)$) e M vie che vanno da Nord a Sud (dalla via 0 alla via $(M-1)$). Ogni strada o via ha un *peso politico*, che è l'importanza del cittadino più importante cittadino che ci abita. Rappresentiamo i pesi politici come due array $A[0 \ldots N-1]$ e $B[0 \ldots M-1]$ di interi da 1 a K. Il seguente disegno rappresenta una tale città con 2 strade e 3 vie, con pesi politici $A = [1, 4]$ e $B = [2, 1, 4]$, rispettivamente.



Il sindaco vuole organizzare una parata attraverso la città. Se la parata passa per l'intersezione tra la strada x e la via y, il traffico di entrambe le strade sarà disturbato, e il sindaco incorrerà in un *costo politico* pari a $\max(A[x], B[y])$. Se la parata passa per più intersezioni, il costo politico sarà il *massimo* dei costi politici di ogni intersezione. Nota che i costi non vengono sommati: ciò che conta non è quante persone la parata disturba, ma quanto è importante la persona più importante che la parata disturba.

La *distanza politica* tra due intersezioni è il minimo *costo politico* di una parata che parte dalla prima intersezione e arriva alla seconda intersezione. Il tuo compito è calcolare la somma delle distanze politiche tra tutte le coppie di intersezioni della città.

Implementazione

Dovrai sottoporre un unico file sorgente, con estensione `.cpp`.

> ☞ Tra gli allegati a questo task troverai un template `cost.cpp` con un esempio di implementazione.

Dovrai implementare la seguente funzione:

C++
```cpp
int solve(int N, int M, int K, vector<int> A, vector<int> B);
```

- L'intero N rappresenta il numero di strade da Est a Ovest.
- L'intero M rappresenta il numero di vie da Nord a Sud.
- L'array A, indicizzato da 0 a $N-1$, contiene i valori A_0, A_1, …, A_{N-1}, dove A_i è il peso politico della i-esima strada da Est a Ovest.
- L'array B, indicizzato da 0 a $M-1$, contiene i valori B_0, B_1, …, B_{N-1}, dove B_i è il peso politico della i-esima via da Nord a Sud.
- La funzione dovrà restituire la somma delle distanze politiche tra tutte le coppie di intersezioni possibili, **modulo** 1000003.

Il grader chiamerà la funzione `solve` e stamperà il valore restituito nel file di output.

Grader di prova

La cartella del problema contiene una versione semplificata del grader usato durante la correzione, che potete usare per testare le vostre soluzioni in locale. Il grader semplificato legge i dati di input da `stdin`, chiama la funzione che deve essere implementata e scrive il risultato su `stdout`.

L'input è composto da 3 righe, contenenti:

- Riga 1: gli interi N, M e K.
- Riga 2: gli interi A_i, separati da spazio.
- Riga 3: gli interi B_i, separati da spazio.

L'output è composto da una sola riga, contenente il valore restituito dalla funzione `solve`.

Assunzioni

- $1 \leq N \leq 3 \times 10^5$.
- $1 \leq M \leq 3 \times 10^5$.
- $1 \leq K \leq N + M$.
- $1 \leq A_i \leq K$ per $i = 0 \dots N - 1$.
- $1 \leq B_i \leq K$ per $i = 0 \dots M - 1$.

Assegnazione del punteggio

Il tuo programma verrà testato su un insieme di test ragruppati per subtask. Per ottenere il punteggio relativo ad un subtask, è necessario risolvere correttamente tutti i test che lo compongono.

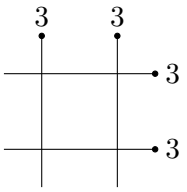
- **Subtask 1 [0 punti]**: Casi d'esempio.
- **Subtask 2 [10 punti]**: $N \leq 10^1, M \leq 10^1$.
- **Subtask 3 [10 punti]**: $N \leq 10^2, M \leq 10^2$.
- **Subtask 4 [10 punti]**: $N = 1, M \leq 10^4$.
- **Subtask 5 [10 punti]**: $N = 1, M \leq 10^5$.
- **Subtask 6 [10 punti]**: $N \leq 10^3, M \leq 10^3$.
- **Subtask 7 [10 punti]**: $N \leq 10^4, M \leq 10^4$.
- **Subtask 8 [10 punti]**: $N \leq 10^5, M \leq 10^5$ e i vettori A e B sono non-decrescenti, cioè se $i < j$ allora $A_i \leq A_j$ e $B_i \leq B_j$.
- **Subtask 9 [10 punti]**: $N \leq 10^5, M \leq 10^5, K \leq 10^1$.
- **Subtask 10[10 punti]**: $N \leq 10^5, M \leq 10^5$.
- **Subtask 11[10 punti]**: Nessuna limitazione aggiuntiva.

Esempi di input/output

stdin	stdout
2 2 4 3 3 3 3	48
1 3 4 2 2 3 1	25
2 3 5 1 4 2 1 4	135

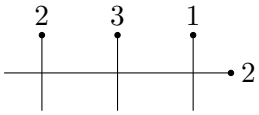
Spiegazione

Nel **primo caso di esempio**, abbiamo una città con 2 strade e 2 vie, tutte con peso politico 3:



Ci sono 16 differenti coppie di intersezioni. Poiché la distanza politica tra ogni coppia di intersezioni è 3, la soluzione è $3 \cdot 16 = 48$.

Nel **secondo caso di esempio**, abbiamo una città con 1 strada e 3 vie, con pesi politici $A = [2]$ e $B = [2, 3, 1]$, rispettivamente:



Ci sono 9 coppie di intersezioni. Tre di queste coppie iniziano e terminano nella stessa intersezione, e hanno distanza politica 2, 3 e 2 rispettivamente (la strada più a destra ha peso politico 1, ma l'unica via ha peso politico 2, quindi la distanza politica di ogni parata è almeno 2). Per ogni altra coppia di intersezioni, la parata che le unisce deve attraversare la via centrale, e quindi deve avere distanza politica 3. Quindi la somma totale è $2 + 3 + 2 + 6 \cdot 3 = 25$.

Il **terzo caso di esempio** corrisponde all'esempio dato nel testo del problema. Qui, ci sono 2 strade e 3 vie. Puoi verificare, con un po' di pazienza, che la somma delle distanze politiche delle 36 coppie di intersezioni è 135.