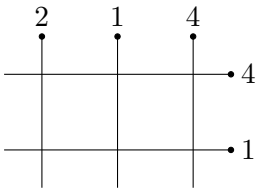
Political cost (cost)

In der Stadt, in der du wohnst, gibt es N Strassen, die von Osten nach Westen verlaufen (von der 0th Street bis zur $(N-1)$th Street) und M Alleen in Nord-Süd-Richtung (von der 0th Avenue bis zur $(M-1)$th Avenue). Jede Strasse oder Allee hat ein *politisches Gewicht*, das der Bedeutung des wichtigsten Bürger, der in ihr lebt. Wir stellen die politischen Gewichte als zwei Arrays dar $A[0 \ldots N-1]$ und $B[0 \ldots M-1]$ aus ganzen Zahlen von 1 bis K. Die folgende Grafik stellt eine solche Stadt mit 2 Strassen und 3 Alleen dar, mit den politischen Gewichten $A = [1, 4]$ bzw. $B = [2, 1, 4]$.



Der Bürgermeister möchte eine Parade durch die Stadt veranstalten. Wenn die Parade über die Kreuzung der xth Street mit der yth Avenue geht, wird der wird der Verkehr auf beiden Strassen gestört, und dem Bürgermeister entstehen *politische Kosten* von $\max(A[x], B[y])$. Wenn die Parade mehrere Kreuzungen überquert, werden die politischen Kosten das *maximum* der politischen Kosten für jede Kreuzung. Beachte, dass die Kosten nicht summiert werden: Es kommt nicht darauf an wie viele Menschen die Parade stört, sondern wie wichtig der der wichtigste Bürger ist, den der Aufmarsch stört.

Der politische Abstand zwischen zwei Kreuzungen entspricht den kleinsten *politischen Kosten* einer Parade, die von der ersten Kreuzung abfährt und an der zweiten Kreuzung ankommt. Deine Aufgabe ist es die Summe der politischen Entfernungen zwischen allen Paaren von Kreuzungen in der Stadt zu berechnen.

Implementierung

Du musst eine einzige `.cpp`-Quelltextdatei einreichen.

> ☞ In den Anhängen zu dieser Aufgabe findst du eine Vorlage `cost.cpp` mit einer Beispielimplementierung.

Du musst die folgende Funktion implementieren:

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

- Die ganze Zahl N steht für die Anzahl der Strassen von Ost nach West.
- Die ganze Zahl M steht für die Anzahl der Nord-Süd-Alleen.
- Das Array A, indiziert von 0 bis $N-1$, enthält die Werte A_0, A_1, …, A_{N-1}, wobei A_i das politische Gewicht der der i-ten Strasse von Ost nach West ist.
- Das Array B, indiziert von 0 bis $M-1$, enthält die Werte B_0, B_1, …, B_{N-1}, wobei B_i das politische Gewicht der der i-ten Nord-Süd-Strasse ist.
- Die Funktion soll die Summe der politischen Entfernungen zwischen allen möglichen Paaren von Kreuzungen **modulo** 1000003 berechnen.

Der Grader ruft die Funktion `solve` auf und schreibt ihren Rückgabewert in die Ausgabedatei.

Beispielgrader

Das Verzeichnis der Aufgabe enthält eine vereinfachte Version des Jury-Graders, die du verwenden kannst, um Ihre Lösung lokal zu testen. Der vereinfachte Grader liest die Eingabedaten aus `stdin`, ruft die Funktionen auf, die du implementieren musst, und schreibt schliesslich die Ausgabe in `stdout`.

Die Eingabe besteht aus 3 Zeilen, die Folgendes enthalten:

- Zeile 1: die ganzen Zahlen N, M und K.
- Zeile 2: die ganzen Zahlen A_i, durch Leerzeichen getrennt.
- Zeile 3: die ganzen Zahlen B_i, durch Leerzeichen getrennt.

Die Ausgabe besteht aus einer einzigen Zeile, die den Wert enthält, der von der Funktion `solve` zurückgegeben wird.

Einschränkungen

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

Punktevergabe

Ihr Programm wird mit einer Reihe von Testfällen getestet, die nach Teilaufgaben gruppiert sind. Um um die einer Teilaufgabe zugeordnete Punktzahl zu erhalten, musst du alle darin enthaltenen Testfälle, die sie enthält, richtig lösen.

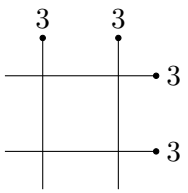
- **Teilaufgabe 1 [0 Punkte]**: Beispiel-Testfälle.
- **Teilaufgabe 2 [10 Punkte]**: $N \leq 10^1, M \leq 10^1$.
- **Teilaufgabe 3 [10 Punkte]**: $N \leq 10^2, M \leq 10^2$.
- **Teilaufgabe 4 [10 Punkte]**: $N = 1, M \leq 10^4$.
- **Teilaufgabe 5 [10 Punkte]**: $N = 1, M \leq 10^5$.
- **Teilaufgabe 6 [10 Punkte]**: $N \leq 10^3, M \leq 10^3$.
- **Teilaufgabe 7 [10 Punkte]**: $N \leq 10^4, M \leq 10^4$.
- **Teilaufgabe 8 [10 Punkte]**: $N \leq 10^5, M \leq 10^5$ und die Arrays A und B sind nicht-abnehmend, d.h. wenn $i < j$, dann $A_i \leq A_j$ und $B_i \leq B_j$.
- **Teilaufgabe 9 [10 Punkte]**: $N \leq 10^5, M \leq 10^5, K \leq 10^1$.
- **Teilaufgabe 10[10 Punkte]**: $N \leq 10^5, M \leq 10^5$.
- **Teilaufgabe 11[10 Punkte]**: Keine zusätzlichen Beschränkungen.

Beispiele

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

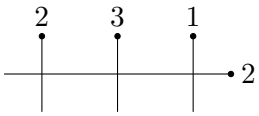
Erklärung

Im **ersten Beispielfall** haben wir eine Stadt mit 2 Strassen und 2 Alleen, die alle politisches Gewicht 3 haben:



Es gibt 16 verschiedene Paare von Schnittpunkten. Da die politische Entfernung zwischen jedem Schnittpunktpaar 3 beträgt, ist die Lösung $3 \cdot 16 = 48$.

Im **zweiten Testfall** gibt es 1 Strasse und 3 Alleen, mit den politischen Gewichten $A = [2]$ bzw. $B = [2, 3, 1]$:



Es gibt 9 Paare von Schnittpunkten. Drei dieser Paare beginnen und enden am gleichen Schnittpunkt und haben politische Abstände von 2, 3 und 2 (die ganz rechte Allee hat ein politisches Gewicht von 1, aber das politische Gewicht der einzigen Strasse ist 2, so dass die politische Entfernung jeder Parade mindestens 2 beträgt). Für jedes verbleibende Paar von Kreuzungen muss der Parade, die sie verbindet, die mittlere Strasse überqueren, und muss daher eine politische Entfernung von 3 haben. Die Gesamtsumme ist also $2 + 3 + 2 + 6 \cdot 3 = 25$.

Der **dritte Testfall** entspricht dem Beispiel aus der der Aufgabenstellung. Hier gibt es 2 Strassen und 3 Alleen. Man kann mit etwas Geduld prüfen, dass die Summe der politischen Abstände der 36-Kreuzungspaare 135 ist.