The 2020 ICPC Caribbean Finals Qualifier

Real contest problemset

Problem set developers:

Alberto Gonzalez Rosales (Cuba)

Alexander Bestard (Cuba)

Ariel Cruz Cruz (Cuba)

Aurora Gil Pons (Cuba)

Carlos Joa (Dominican Republic)

Ernesto David Peña Herrera (Cuba)

Jose Carlos Gutiérrez Pérez (Cuba)

Marcelo Fornet Fornés (Cuba)

Reynaldo Gil Pons (Cuba)

Roberto Abreu (Dominican Republic)

Rubén Alcolea Núñez (Cuba)

June 26th, 2021

Problem A. Llave perdida

Juan diseñó un algoritmo de encriptación muy poco ortodoxo. Este funciona de la siguiente manera: Dada una cadena $A = a_0a_1a_2 \ldots a_{n-1}$, y una llave $K = k_0k_1k_2 \ldots k_{m-1}$, el algoritmo encripta la cadena A, en una cadena resultante $B = b_0b_1b_2 \ldots b_{n-1}$, donde $b_i = a_i + k_{i \bmod m}$. La suma de dos caracteres x' y y' es igual a la letra que está en la posición z' en el alfabeto, donde z' es el resultado de la suma de la posición de la letra x' en el alfabeto más la posición de la letra y' en el alfabeto módulo 26 (cíclicamente). Todas las posiciones en el alfabeto están indizadas empezando por la posición 0. Ejemplo. Encriptar la cadena $A =$ *"caribe"* con la llave $K =$ *"icpc"* retornará la cadena $B =$ *"kcgkjg"*, a través de los siguientes pasos:

1. La letra **k** está en la posición $10 = (2 + 8) \bmod 26$, **c** está en la posición 2 e **i** está en la posición 8.
2. La letra **c** está en la posición $2 = (0 + 2) \bmod 26$, **a** está en la posición 0 y **c** está en la posición 2.
3. La letra **g** está en la posición $6 = (17 + 15) \bmod 26$, **r** está en la posición 17 y **p** está en la posición 15.
4. La letra **k** está en la posición $10 = (8 + 2) \bmod 26$, **i** está en la posición 8 y **c** está en la posición 2.
5. La letra **j** está en la posición $9 = (1 + 8) \bmod 26$, **b** está en la posición 1 e **i** está en la posición 8.
6. La letra **g** está en la posición $6 = (4 + 2) \bmod 26$, **e** está en la posición 4 y **c** está en la posición 2.

Juan empezó a encriptar un texto ayer y hoy necesita continuar. Sin embargo, ha perdido a K. Ayuda a Juan a encontrar la llave, para que pueda continuar trabajando. Juan te dará una cadena, A, y su correspondiente encriptado, B.

Input

La primera línea de entrada contiene dos cadenas, formadas por letras minúsculas, separadas por un espacio, A y B ($|A| = |B|, 1 \leq |A| \leq 100$). La segunda línea contiene a $m (1 \leq m \leq 4, m \leq |A|)$, el tamaño de la llave olvidada.

Output

Imprima la llave, una cadena formada por letras minúsculas, si pudo encontrarla, o -1, si es imposible conseguir B a partir de A con una llave de tamaño m.

Example

standard input	standard output
caribe kcgkjg 4	icpc
latin lbfop 3	-1
zombie zamniq 2	am

Problem B. Suma Máxima de MCD

Para dos números enteros positivos a y b, $gcd(a,b)$ es el máximo común divisor de a y b, es decir, el entero más grande que divide a ambos a y b (sin dejar residuo). Extendamos la definición de gcd a una secuencia $S = s_1, s_2, ..., s_k$, donde $k > 2$, de manera recursiva como sigue: $gcd(S) = gcd(s_1, gcd(s_2, ..., s_{k-1}, s_k))$. Es decir, gcd de una secuencia es el gcd del primer elemento de la secuencia y el gcd del resto de la sequencia. Para una secuencia S que consiste en al menos dos números enteros, definamos la función $F(S) = |S| \times gcd(S)$, donde $|S|$ es la longitud de la secuencia. Dado un arreglo A, determina el máximo valor posible de $F(S)$ donde S es una subsecuencia de A con al menos dos elementos. Una secuencia S es una subsecuencia de un arreglo A si S se puede obtener de A eliminando algunos de sus elementos (posiblemente ninguno o todos).

Input

La primera línea de la entrada contiene el número entero n ($2 \leq n \leq 10^5$), representando la longitud del arreglo A. La segunda línea contiene n números enteros positivos a_1, a_2, ..., a_n que no exceden 10^6, los cuales corresponden a los elementos del arreglo A.

Output

Imprime una sola línea con el máximo valor posible de $F(S)$ donde S es una subsecuencia del arreglo A con al menos dos elementos.

Example

standard input	standard output
7 10 6 1 30 5 42 6	24
2 3 10	2
6 1 20 30 50 70 5	40

Problem C. Encuentra el punto

Considere una matriz infinita de 2 dimensiones con algunas casillas marcadas. Se desea saber el valor de la mínima suma de las distancias Manhattan de todas las casillas marcadas a cualquier casilla no marcada. Se define como la distancia Manhattan entre dos casillas con coordenadas a_1, b_1 y a_2, b_2 como $|a_1 - a_2| + |b_1 - b_2|$.

Input

La primera línea de la entrada contiene un entero $n (1 \leq n \leq 100000)$, la cantidad de casillas marcadas. Las próximas n líneas contienen las coordenadas a_i y b_i de las casillas marcadas en la matriz $(-2^{30} \leq a_i, b_i \leq 2^{30})$.

Output

Deberá imprimir una sola línea con la mínima suma de distancias Manhattan.

Example

standard input	standard output
4 1 -3 0 1 -2 1 1 -1	10
5 2 2 0 2 4 2 2 0 4 0	11
9 0 0 0 1 0 2 1 0 1 1 1 2 2 0 2 1 2 2	24

Problem D. Secuencia de Paréntesis

Dadas dos secuencias de paréntesis a y b de longitud n, tal que a no es mayor que b, diga el número secuencias balanceadas de paréntesis de longitud n no menores que a y no mayores que b. Asuma que las comparaciones se realizan de forma lexicográfica. Definimos una secuencia balanceada de paréntesis de la siguiente forma:

- La secuencia vacía es una secuencia balanceada de paréntesis.
- Si s es una sequencia balanceada de paréntesis, (s) también lo es.
- Si s y t son secuencias balanceadas de paréntesis, st también lo es.

Dadas dos secuencias de paréntesis x, y de longitud n, decimos que x es menor lexicográficamente que y si tienen un prefijo común de tamaño $t \in [0, n-1]$ y $x[t+1] < y[t+1]$. Asuma que el caracter '(' es menor que el caracter ')'.

Input

La primera línea de la entrada contiene el entero n ($1 \leq n \leq 10^3$), el tamaño de las secuencias de paréntesis. Las próximas dos líneas contienen las cadenas a y b respectivamente, compuestas solamente por caracteres '(' y ')'. Se garantiza que b no es lexicográficamente menor que a.

Output

Imprima una línea con la respuesta. Dado que este número puede ser muy grande imprima el resto que deja al dividirlo por $10^9 + 7$.

Example

standard input	standard output
4 ((((()))	2
3 ()))()	0
12 ((()))(()()) ((()))(()())	1
38 (((((((((((((((((((((((((((((((((((((( ))))))))))))))))))))))))))))))))))))))	767263183

Problem E. Contando Múltiplos

Dado un arreglo A con n enteros, se desea aplicar q consultas sobre este arreglo. Las consultas pueden ser de dos tipos:

- **q i j:** Calcular la cantidad de múltiplos **distintos** de 2 y de 3, pero no de ambos, en el intervalo $[i, j]$.
- **u p v:** Actualizar el valor del arreglo en la posición p con el valor v, es decir, $A[p] = v$.

Debes ejecutar cada consulta en el orden dado de entrada. En particular, cada consulta del tipo **'q'** debe considerar todas las actualizaciones del tipo **'u'** ejecutadas antes de dicha consulta. Escriba un programa que permita resolver de forma eficiente este problema.

Input

La primera línea contiene dos enteros n y q ($1 \leq n \leq 10^5$, $1 \leq q \leq 10^5$), la cantidad de elementos del arreglo y la cantidad de consultas a resolver respectivamente. La segunda línea contiene n enteros que representan la información inicial del arreglo. Cada valor $A[i]$ del arreglo cumple $1 \leq A[i] \leq 10^9$. Las siguientes q líneas contienen un caracter **'q'** o **'u'** y dos enteros separados por un espacio que representan la información de las consultas.

- Si el caracter es **'q'**, la consulta tiene el formato **q i j**, donde $1 \leq i \leq j \leq N$.
- Si el caracter es **'u'**, la consulta tiene el formato **u p v**, donde $1 \leq p \leq N$ y $1 \leq v \leq 10^9$.

Output

Para cada consulta del tipo **'q'**, imprima una línea con la respuesta. Debes responder las consultas en el orden de entrada.

Example

standard input	standard output
5 6 12 4 2 3 8 q 1 2 u 3 30 q 3 5 u 1 10 u 3 9 q 1 5	1 2 5
5 9 10 2 32 18 54 q 1 5 u 1 5 q 1 5 u 2 4 q 1 5 u 3 3 q 1 5 u 4 2 q 1 5	3 2 2 2 3

Problem F. Antipalíndromos

Una cadena binaria se considera antipalíndroma si no es vacía y se cumple para todo i que $s_i \neq s_{n-i+1}$. Por ejemplo, 01, 0011, 101010 y 011001 son antipalíndromas, pero 0, 1110, 11, no lo son. Dada una cadena binaria, determina cuántas subcadenas son antipalíndromas. Si la cantidad de cadenas antipalíndromas es mayor que 10^5, imprima 10^5 en su lugar.

Input

Una línea con un número $n(1 \leq n \leq 10^5)$, el tamaño de la cadena. En la segunda línea una cadena binaria de n caracteres.

Output

Imprima la cantidad de subcadenas que son antipalíndromas, en caso de ser mayor que 10^5, imprima 100000.

Example

standard input	standard output
4 0101	4
4 1001	2

Problem G. Evolución

Se tiene una lista de n números. Cada número será 0 o 1. Se deben aplicar m iteraciones del mismo procedimiento. Por cada iteración el valor de cada número cambiará de la siguiente forma:

- Si los dos números adyacentes en la iteración anterior son iguales, el valor del número será 0.
- Sino, el valor del número será 1.

Como el primer y el último elemento de la lista solo tienen un número adyacente, considere 0 como el otro número adyacente.

Input

La primera línea contiene 2 números enteros n y m ($1 \leq n \leq 1000$, $1 \leq m \leq 1000$) la cantidad de números de la lista y la cantidad m de iteraciones a aplicar respectivamente. La segunda línea contiene n números separados por espacios.

Output

Imprima la lista de números al cabo de m iteraciones.

Example

standard input	standard output
8 1 1 0 0 0 0 1 0 0	0 1 0 0 1 0 1 0
8 2 1 1 1 0 1 1 1 1	0 0 0 0 0 1 1 0
5 3 0 0 0 1 0	1 0 1 0 0

Problem H. Mesa Redonda

Considere $2 \cdot n$ personas con n parejas enemigas. Cada persona tiene **exactamente un** enemigo y la enemistad es bidireccional, es decir, si a es el enemigo de b, entonces b es el enemigo de a. Las parejas enemigas están definidas de la siguiente forma, para todo j $(1 \leq j \leq n)$, son enemigas la persona $2 \cdot j - 1$ y la persona $2 \cdot j$. Se desea calcular de cuántas formas es posible distribuirlos en una gran mesa circular con $2 \cdot n$ asientos, de manera que no haya 2 enemigos sentados uno al lado del otro. Los asientos están numerados de 1 a $2 \cdot n$. Los asientos de las posiciones i e $i + 1$ son adyacentes para todo $1 \leq i < 2 \cdot n$. Los asientos 1 y $2 \cdot n$ también son adyacentes. Dos distribuciones son consideradas diferentes si tienen al menos un asiento ocupado por personas diferentes en cada distribución. Por ejemplo: Tenemos 4 personas y las parejas enemigas son $(1, 2)$ y $(3, 4)$. Existen 8 formas diferentes de ubicarlos alrededor de la mesa:

- $(1, 3, 2, 4)$
- $(2, 3, 1, 4)$
- $(3, 1, 4, 2)$
- $(4, 1, 3, 2)$
- $(1, 4, 2, 3)$
- $(2, 4, 1, 3)$
- $(3, 2, 4, 1)$
- $(4, 2, 3, 1)$

Input

La primera línea de la entrada contiene un entero t $(1 \leq t \leq 10^5)$, el número de casos a procesar. Las próximas t líneas describen los casos de prueba, cada una con un entero n $(1 \leq n \leq 10^5)$, la cantidad de pares de enemigos.

Output

Para cada caso de prueba deberá imprimir una línea con la cantidad de distribuciones posibles. Dado que este número puede ser muy grande, imprima el resto que deja al dividirlo por $10^9 + 7$.

Example

standard input	standard output
4 1 2 4 99999	0 8 11904 720034323

Problem I. Componentes con Peso

Se tiene una matriz X de $n \times m$ casillas donde cada casilla tiene un peso asociado. Un subconjunto S de casillas se considera conexo si, entre todo par de casillas de S, existe un camino de casillas vecinas, también en S, que las una. Dos casillas se consideran vecinas si comparten un lado. El peso de un subconjunto es la suma de los pesos de las casillas que lo conforman. Se desean realizar varias preguntas, cada pregunta tiene un peso k. Para cada pregunta determine si existe un subconjunto conexo cuyo peso se encuentre entre k y $2 \cdot k$ (inclusivo).

Input

La primera línea contiene dos enteros n y m ($1 \leq n \times m \leq 10^5$). Luego n líneas cada una con m enteros. El *j-ésimo* número de la *i-ésima* línea representa el peso de la casilla $X[i, j]$ ($1 \leq X[i, j] \leq 10^9$). Luego una línea con un entero q ($1 \leq q \leq 10^5$), la cantidad de preguntas y a continuación q líneas con un entero k ($1 \leq k \leq 10^{14}$), indicando el peso de cada pregunta.

Output

Por cada pregunta, imprima "YES" en caso de que exista la solución, o "NO" en caso contrario.

Example

standard input	standard output
3 4 1 1 9 2 9 9 9 9 1 1 9 1 10 1 2 3 4 5 6 7 8 9 10	YES YES NO NO YES YES YES YES YES YES
5 5 8 9 1 9 4 941 942 8 4 2 5 939 921 978 914 932 1 8 9 4 7 3 6 5 10 4 450 58 20 10	NO NO YES YES

Problem J. Moneda

Dos jugadores, A y B, compiten sobre un grafo no dirigido. El jugador A debe mover una moneda desde el nodo 1 hasta el nodo N. En cada turno, el jugador B bloquea k nodos, excepto el nodo N y el nodo donde se encuentra actualmente la moneda. Luego el jugador A puede mover la moneda a cualquier nodo vecino (no bloqueado) del nodo donde se encuentra. Los nodos bloqueados se mantienen bloqueados hasta el final del turno y luego se desbloquean. Se desea encontrar el menor k tal que el jugador B puede evitar que A lleve la moneda al nodo N.

Input

La primera línea contiene dos enteros N y M ($2 \le N \le 10^5, 0 \le M \le 10^5$), la cantidad de nodos y la cantidad de aristas respectivamente. Luego M líneas, cada una con dos enteros u y v ($1 \le u, v \le N$) la descripción de cada arista.

Output

Imprima el valor del k mas pequeño con el cual el B puede evitar que A lleve la moneda al nodo N o imprima -1 si B no puede evitarlo.

Example

standard input	standard output
6 6 1 2 1 3 2 4 3 5 4 6 5 6	1
6 8 1 2 1 3 2 4 3 5 4 6 5 6 3 4 2 5	2
4 6 1 2 1 3 1 4 2 3 2 4 3 4	-1

Problem K. Operaciones con String

Dada una lista A de n cadenas y q operaciones que pueden ser de los siguientes tipos:

- 1 i s: Asignar la cadena ubicada en la posicion i al valor s. Es decir, $A[i] = s$.
- 2 i j s: Responder si alguna cadena ubicada en el intervalo $[i, j]$ es prefijo de la cadena s.
- 3 i j s: Responder si la cadena s es prefijo de alguna de las cadenas ubicadas en el intervalo $[i, j]$.

Para todas las operaciones, se cumple $1 \leq i \leq j \leq n$ y s es una cadena de letras minúsculas en inglés .

Input

La primera línea contiene un entero n ($1 \leq n \leq 10^5$), la cantidad de cadenas de A. Luego, n líneas, donde la *i--ésima* línea corresponde a la *i-ésima* cadena de A. La siguiente línea contiene un entero q ($1 \leq q \leq 10^5$), la cantidad de operaciones a procesar.

Luego, q líneas, cada una con una operación en el formato descrito anteriormente. La suma de las longitudes de las cadenas presentes en el arreglo A y en todas las operaciones no excede $5 \cdot 10^5$.

Output

Por cada operación de tipo 2 o 3 se debe responder 'YES' o 'NO' en dependencia de si se cumple la condición planteada en cada tipo de operación.

Example

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>5
abc
a
ab
a
c
3
2 1 2 a
1 2 b
3 2 3 b</pre></td><td><pre>YES
YES</pre></td></tr>
<tr><td><pre>5
abc
ab
a
xy
wxz
6
2 1 2 abcdario
2 2 2 icpc
3 1 3 ab
3 1 3 x
1 2 xp
3 1 3 x</pre></td><td><pre>YES
NO
YES
NO
YES</pre></td></tr>
</table>

Problem L. Competición

En una competencia de todos contra todos participan n jugadores. Cada pareja de jugadores se enfrentan una sola vez. En cada enfrentamiento hay un solo ganador, (**no hay empates**). Dado un subconjunto de jugadores X y los ganadores de los enfrentamientos efectuados entre estos jugadores, se dice que no se puede determinar el mejor jugador de X si no existe ningún jugador con más victorias que el resto. Se desea encontrar, dados los resultados de todos los enfrentamientos realizados en la competencia, un conjunto de 3 jugadores donde no se pueda determinar el mejor jugador.

Input

La primera línea contiene un entero n ($3 \leq n \leq 2000$), la cantidad de jugadores de la competencia. Las siguientes n líneas representan los resultados del jugador $i-ésimo$ contra los demás jugadores. Si el caracter $j-ésimo$ de la línea $i-ésima$ es 1 indica que el jugador i venció al j en su enfrentamiento, si es 0 indica que perdió. Por cada pareja de jugadores distintos i y j, se garantiza que i derrotó a j ó j derrotó a i (pero no ambas condiciones).

Output

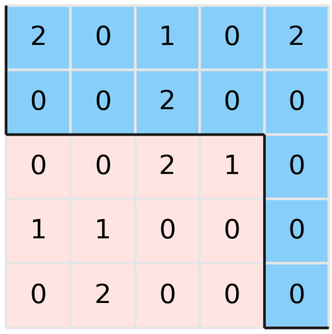
Imprima tres enteros a, b, c ($1 \leq a, b, c \leq n$), los índices de los jugadores que cumplen que no se puede seleccionar un mejor jugador entre ellos. Si hay múltiples respuestas, cualquiera será aceptada. Si no existe solución al problema imprima -1.

Example

standard input	standard output
4 0110 0000 0101 1100	1 3 4
4 0111 0000 0101 0100	-1

Problem M. Partición pareja

Se tiene una matriz A donde cada casilla tiene un valor 0, 1 o 2. Se desea dividir la matriz en dos partes, desde el vértice en la posición $(0, 0)$ hasta el vértice en (n, m) por una secuencia de aristas, de tal forma que, la arista a precede a la arista b, si b está a la derecha o debajo de a y comparten el vértice de la derecha o de debajo de a. Será posible hacer la división de forma tal que en los dos lados la suma de los valores de las casillas sea la misma?



Ejemplo de caso de prueba 3

Input

La primera línea contiene dos enteros n y m $(1 \leq n, m \leq 1000)$, la cantidad de filas y columnas de la matriz respectivamente. Luego n líneas, cada una con m enteros $A_{i,j}$ $(0 \leq A_{i,j} \leq 2)$, indicando el valor de la casilla j-*ésima* en la fila i-*ésima*.

Output

Imprima 'YES' , si es posible encontrar un camino que cumpla las condiciones, imprima 'NO' en caso contrario. En el caso afirmativo imprima una segunda línea con una cadena de longitud $n + m$, formada por los caracteres 'R' (right) or 'D' (down) que describa un posible camino. Si hay múltiples caminos, cualquier camino será aceptado.

Example

standard input	standard output
2 2 2 2 1 1	YES RDDR
3 3 0 1 1 2 0 0 2 0 0	NO
5 5 2 0 1 0 2 0 0 2 0 0 0 0 2 1 0 1 1 0 0 0 0 2 0 0 0	YES DRRDRRDDDR
2 3 0 0 0 0 0 0	YES DDRRR

Problem N. Media Aritmética

Dados cuatro valores enteros, determina si uno de ellos es igual a la media de otros dos valores. Recuerda que la media de dos valores a y b es $\frac{a+b}{2}$.

Input

La primera línea de la entrada contiene cuatro valores enteros a, b, c y d $(1 \leq a, b, c, d \leq 100)$

Output

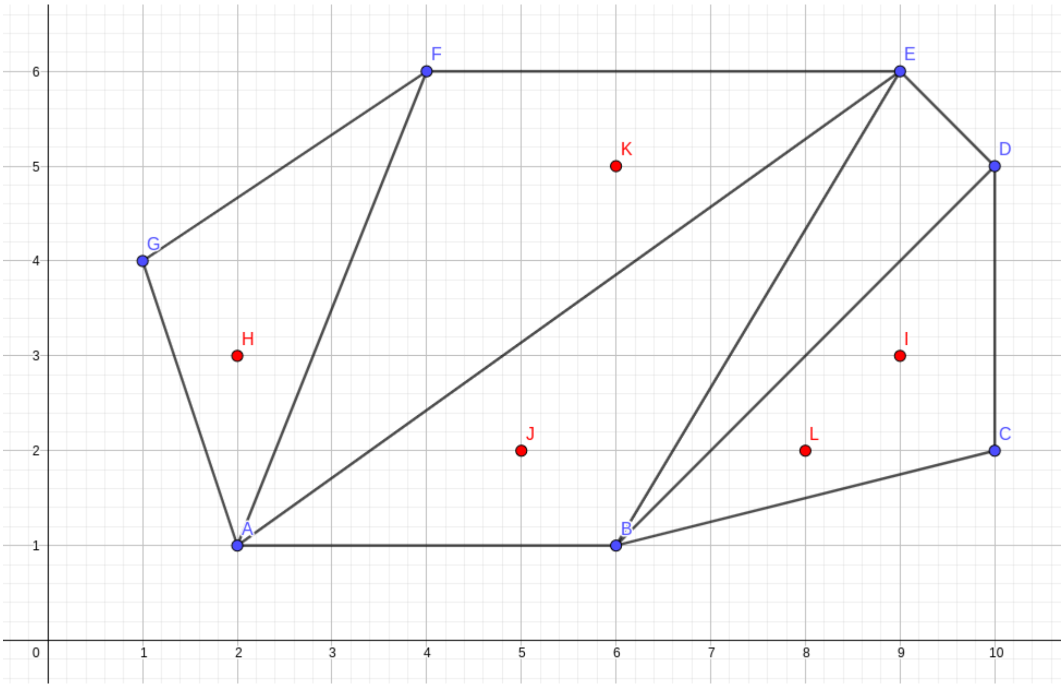
Si existe un número cuyo valor es igual a la media de otros dos valores, entonces imprima `"YES"` sin comillas, sino, imprima `"NO"` sin comillas.

Example

standard input	standard output
3 10 1 2	YES
100 100 100 100	YES
10 21 21 15	NO

Problem O. Polígono

Se tiene un polígono **convexo** de n vértices (no hay tres vértices sobre una misma línea) y una triangulación para dicho polígono. Se desea responder q preguntas del tipo x_1, y_1, x_2, y_2. Para cada pregunta deberá decir la menor cantidad de diagonales que es necesario cruzar si camináramos desde el punto (x_1, y_1) al punto (x_2, y_2), sin salirnos del polígono.



La imagen corresponde al ejemplo de entrada

Ejemplo:

1. Si tenemos el punto H en el $\triangle AFG$ y queremos ir hasta el punto I en el $\triangle BCD$, la mejor ruta implica cruzar las 4 diagonales de la entrada ($\overline{AF}$, $\overline{AE}$, $\overline{BE}$ y $\overline{BD}$).

2. Si tenemos el punto J en el $\triangle ABE$ y queremos ir hasta el punto K en el $\triangle AEF$, como los puntos se encuentran en triángulos adyacentes, basta con cruzar la diagonal $\overline{AE}$.

3. Si tenemos el punto L en el $\triangle BCD$ y queremos ir hasta el punto I en el $\triangle BCD$, como ambos se encuentran en el mismo triángulo, la cantidad de diagonales a cruzar es 0.

Input

La primera línea contiene un entero n ($3 \leq n \leq 10^5$), el número de vértices del polígono. A continuación, n líneas, cada una con dos enteros x_i, y_i ($-10^9 \leq x_i, y_i \leq 10^9$), las coordenadas del i-*ésimo* vértice del polígono. Los vértices son dados en el orden contra las manecillas del reloj. Las próximas $n-3$ líneas describen la triangulación del polígono, dando la lista de diagonales que conforman la triangulación. Cada línea contiene dos enteros a_i, b_i ($1 \leq a_i, b_i \leq n$), indicando que hay una diagonal desde el a_i-*ésimo* vértice al b_i-*ésimo*, de acuerdo al orden en que los vértices del polígono fueron dados en la entrada. Se garantiza que la lista de diagonales dada forma una triangulación válida para el polígono. La próxima línea contiene un entero q ($1 \leq q \leq 10^5$), el número de preguntas a procesar. Las próximas q líneas contienen cada una cuatro enteros x_1, y_1, x_2, y_2, cada una describiendo una pregunta a responder. Se garantizan que ambos puntos se encuentran dentro del polígono y no están sobre una diagonal o un lado del polígono.

Output

Para cada pregunta imprima una línea con un entero, la menor cantidad de diagonales que es necesario cruzar.

Example

standard input	standard output
7 2 1 6 1 10 2 10 5 9 6 4 6 1 4 1 6 1 5 2 4 2 5 3 2 3 9 3 5 2 6 5 8 2 9 3	4 1 0
4 100 100 200 100 200 200 100 200 1 3 3 150 140 150 160 190 140 190 160 110 140 110 160	1 0 0