A. Desarrollo

3 s., 256 MB

Tomás quiere estudiar cómo fue cambiando el nivel de conectividad de la Ciudad de Buenos Aires a lo largo del tiempo. Para ello, armó un digrafo G donde los n nodos denotan las esquinas de la ciudad, y donde los ejes se corresponden con calles dirigidas que unen estas esquinas. Para estas esquinas conoce el orden cronológico en el que fueron agregadas al trazado de la ciudad.

Para estudiar la conectividad de la ciudad se propone la siguiente tarea: va a ir eliminando las esquinas del grafo en el orden opuesto al que fueron creadas, y cada vez que borra una va a calcular la suma de todas las distancias entre todos los pares de esquinas que todavía están en el mapa. De esta forma, estudiando como aumenta este valor cree poder entender cuáles esquinas fueron las más importantes para aumentar la conectividad de la ciudad entera.

Debemos ayudar a Tomás a completar la tarea que propuso.

Input

La primera línea contiene un entero n $(1 \leq n \leq 500)$ indicando el número de esquinas de la ciudad.

Las siguientes n líneas contienen n enteros cada una. El j-esimo entero en la línea i indica la longitud de la calle que une a i con j. Esta longitud es a lo sumo 100000. La última línea tiene n números distintos, e indican el orden cronológico invertido en el cual fueron agregadas las esquinas. Es decir, la primera esquina de esta lista es la última que fue creada, y por lo tanto la primera que va a quitar Tomás.

Output

Se deben imprimir n enteros. El i-esimo de estos debe indicar la suma de las distancias de los nodos que quedan tras eliminar del grafo las últimas $i-1$ esquinas que fueron creadas.

input
```
1
0
1
```
output
```
0
```

input
```
2
0 5
4 0
1 2
```
output
```
9 0
```

input
```
4
0 3 1 1
6 0 400 1
2 4 0 1
1 1 1 0
4 1 2 3
```
output
```
17 23 404 0
```

B. Aulas sobrecargadas

1 s., 256 MB

Los docentes de Intro se organizaron para tomar el parcial en n aulas distintas, pero debido a una confusión en la distribución de los alumnos ahora las aulas están sobrecargadas. Puntualmente, en el aula i se sentaron a_i estudiantes, cuando en realidad la misma está pensada únicamente para una cantidad b_i.

Se quiere descubrir si es posible reorganizar a los alumnos para que cada aula tenga exactamente b_i estudiantes, teniendo en cuenta que no es posible mover alumnos entre ciertas aulas. Puntualmente, se conocen los pares de aulas (p, q) tales que es posible mover a los alumnos del aula p al aula q. Como el parcial está por empezar, cada alumno puede hacer **a lo sumo un movimiento**. Es decir, un alumno no puede ir del aula p_1 a la p_2 y luego a la p_3.

Input

La primera línea de la entrada tiene dos enteros n y m ($1 \leq n \leq 100$, $0 \leq m \leq 200$) indicando la cantidad de aulas y la cantidad de pares de aulas tales que es posible mover alumnos entre ellas.

La siguiente línea tiene los n valores $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 100$).

La siguiente, los n valores $b_1, b_2, \ldots, b_n$ ($0 \leq b_i \leq 100$).

Finalmente hay m líneas conteniendo cada una dos enteros p y q ($1 \leq p, q \leq n, p \neq q$) indicando que es válido mover alumnos tanto del aula p hacia la q como de la q hacia la p.

Output

Si es imposible reorganizar a los alumnos se debe devolver "`NO`".

Caso contrario, se debe devolver "`YES`" junto a n lineas adicionales, cada una con n enteros. El j-ésimo número en la i-ésima fila debe indicar cuántos estudiantes del aula i deben moverse al aula j (el i-ésimo número de la i-ésima fila debe indicar cuántos tienen que quedarse en esa aula).

Si hay varias respuestas posibles, cualquiera es válida.

input

```
4 4
1 2 6 3
3 5 3 1
1 2
2 3
3 4
4 2
```

output

```
YES
1 0 0 0
2 0 0 0
0 5 1 0
0 0 2 1
```

input

```
2 0
1 2
2 1
```

output

```
NO
```

C. Seguridad

2 s, 512 MB

Tuki está ayudando al Gobierno de la Ciudad a instalar sistemas de seguridad en las distintas calles de la Capital. Esta se modela como un multigrafo no dirigido con n nodos donde estos denotan esquinas de la ciudad mientras que los ejes representan las calles. Para cada calle se conoce su longitud y el costo de colocar un sistema de seguridad en la misma, el cual es exactamente el doble de su longitud. Como cubrir todas las calles es muy caro, el Gobierno se propone solo cubrir aquellas que pertenecen a algún camino mínimo entre el nodo 0 y el $n-1$, y está interesado en descubrir el costo del proyecto.

Debemos ayudar a Tuki a resolver este problema. Para eso, debemos diseñar un algoritmo que dada la descripción de la ciudad encuentre el costo de instalar sistemas de seguridad en todas las calles que pertenecen a algún camino mínimo entre el nodo 0 y el $n-1$.

Input

La entrada consiste de una primera línea con dos enteros n, m ($1 \leq n \leq 10000$, $1 \leq m \leq 300000$) que denotan respectivamente la cantidad de esquinas y la cantidad de calles, seguida de m líneas con 3 enteros v, w y c indicando que hay una calle bidireccional entre los nodos v y w de largo c (con $1 \leq c \leq 1000$). Puede haber más de una calle entre dos esquinas, e incluso calles que conecten una esquina consigo misma (a lo Parque Chas). Se sabe que existe por lo menos una forma de llegar de la esquina 0 a la esquina $n-1$.

Output

Se debe imprimir un único número indicando el costo de instalar un sistema de seguridad en cada calle que pertenece a algún camino mínimo entre las esquinas 0 y $n-1$.

input

```
4 7
0 1 1
0 2 2
0 3 10
0 3 3
1 3 2
2 3 1
1 1 1
```

output

```
18
```

input

```
3 3
0 1 2
1 2 4
2 0 5
```

output

```
10
```