Diametro Dinamico (diameter)

Giorno 1
Lingua Italiano
Limite di tempo: 5 secondi
Limite di memoria: 1024 megabyte

Ti viene fornito un albero pesato di n vertici con archi non direzionati e una lista di q update. Ogni update cambia il peso di un arco. L'obiettivo è stampare in output il diametro dell'albero dopo ogni update. (La distanza tra due vertici è la somma dei pesi dell'unico percorso che li collega. Il diametro è la più grande di queste distanze.)

File di input

La prima linea contiene tre interi separati da spazi n, q e w ($2 \le n \le 100,000, 1 \le q \le 100,000, 1 \le w \le 20,000,000,000,000$) – il numero di vertici dell'albero, il numero di update e il limite sui pesi degli archi. I vertici sono numerati da 1 a n.

Le successive n-1 righe descrivono l'albero iniziale. I'*i*-esima di queste righe contiene tre interi separati da spazi a_i, b_i, c_i $(1 \le a_i, b_i \le n, 0 \le c_i < w)$ che specificano che inizialmente esiste un arco tra i vertici a_i e b_i con peso c_i . È garantito che queste n-1 righe descrivono un albero.

Infine seguono q righe che descrivono le query. La j-esima di queste righe contiene due interi separati da spazi d_j , e_i $(0 \le d_i < n - 1, 0 \le e_i < w)$. Questi due interi sono trasformati secondo il seguente schema:

- $d'_i = (d_i + last) \mod (n-1)$
- $e'_j = (e_j + last) \mod w$

dove last è il risultato dell'ultima query (inizialmente last = 0). La tupla (d'_j, e'_j) rappresenta una query che prende il $d'_i + 1$ -esimo arco secondo l'ordine di input e cambia il suo peso in e'_i .

File di output

Dovrai stampare q righe. Per ogni i, la riga i dovrà contenere il diametro dell'albero dopo l'i-esimo update.

Assegnazione del punteggio

Subtask 1 (11 punti): $n,q \leq 100$ e $w \leq 10,000$

Subtask 2 (13 punti): $n, q \le 5,000 \text{ e } w \le 10,000$

Subtask 3 (7 punti): $w \le 10,000$ e gli archi dell'albero sono esattamente tutti archi validi nella forma $\{1,i\}$ (In pratica l'albero è una stella centrata al vertice 1.)

Subtask 4 (18 punti): $w \le 10,000$, e gli archi dell'albero sono esattamente tutti archi validi nella forma $\{i,2i\}$ e $\{i,2i+1\}$ (In pratica, se considerassimo il vertice 1 come la radice dell'albero, sarebbe un albero binario bilanciato.)

Subtask 5 (24 punti): è garantito che dopo ogni update il percorso semplice più lungo passa per il vertice 1.

Subtask 6 (27 punti): nessuna limitazione.

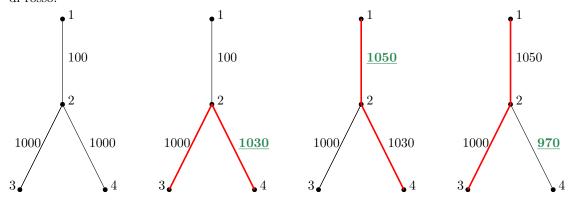


Esempi

standard input	standard output
4 3 2000	2030
1 2 100	2080
2 3 1000	2050
2 4 1000	
2 1030	
1 1020	
1 890	
10 10 10000	6164
1 9 1241	7812
5 6 1630	8385
10 5 1630	6737
2 6 853	6738
10 1 511	7205
5 3 760	6641
8 3 1076	7062
4 10 1483	6581
7 10 40	5155
8 2051	
5 6294	
5 4168	
7 1861	
0 5244	
6 5156	
3 3001	
8 5267	
5 3102	
8 3623	

Note

Il primo esempio è raffigurato nella figura in basso. La figura più a sinistra mostra lo stato iniziale del grafo. Ogni figura che segue raffigura la situazione dopo un update. Il peso dell'arco aggiornato è colorato di verde e il diametro di rosso.



La prima query cambia il peso del terzo edge, i.e. $\{2,4\}$, a 1030. La più grande distanza tra due vertici è 2030 – la distanza tra 3 e 4.

Siccome la distanza calcolata è 2030, la seconda query è

$$d_2' = (1+2030) \bmod 3 = 0$$

$$e_2' = (1020 + 2030) \mod 2000 = 1050$$



Quindi il peso dell'arco $\{1,2\}$ è cambiato a 1050. Questo porta la coppia di vertici $\{1,4\}$ ad essere la coppia a maggiore distanza, con valore 2080.

La terza query è decodificata come:

$$d_3' = (1 + 2080) \mod 3 = 2$$

 $e_3' = (890 + 2080) \mod 2000 = 970$

Dal momento che la distanza dei vertici $\{2,4\}$ diminuisce a 970, la coppia pi distante improvvisamente $\{1,3\}$ con un valore di 2050.