Closing Time

A Hungria é um país com N cidades, numeradas de 0 a $N-1$.

As cidades estão ligadas por $N-1$ ruas *bidirecionais*, numeradas de 0 a $N-2$. Para cada j tal que $0 \leq j \leq N-2$, a rua j liga as cidades $U[j]$ e $V[j]$ e tem comprimento $W[j]$, isto é, a rua permite viajar entre as cidades em $W[j]$ unidades de tempo. Cada rua liga duas cidades diferentes e cada par de cidades está ligada por no máximo uma rua.

Um **caminho** entre duas cidades distintas a e b é uma sequência $p_0, p_1, \ldots, p_t$ de cidades distintas tal que:

- $p_0 = a$,
- $p_t = b$,
- para cada i ($0 \leq i < t$), existe uma rua a ligar as cidades p_i e p_{i+1}.

É possível viajar de qualquer cidade para qualquer outra cidade usando as ruas, isto é, existe um caminho entre quaisquer duas cidades distintas. Pode-se mostrar que este caminho é único para cada par de cidades distintas.

O **comprimento** de um caminho $p_0, p_1, \ldots, p_t$ é a soma dos comprimentos das t ruas que ligam cidades consecutivas ao longo do caminho.

Na Hungria, muitas pessoas viajam para participarem nas festividades do Dia da Fundação em duas das principais cidades. Assim que as celebrações terminarem, estas voltam para as suas casas. O governo quer prevenir que a multidão incomode os locais, pelo que tem planos para encerrar as cidades a certas horas. A cada cidade é atribuída uma **hora de encerramento** não negativa pelo governo. O governo decidiu que a soma de todos os horários de encerramento não deve ser superior a K. Mais precisamente, para cada i entre 0 e $N-1$, inclusive, a hora de encerramento atribuída à cidade i é um inteiro não negativo $c[i]$. A soma de todos os $c[i]$ não pode ser maior do que K.

Considera a cidade a e uma certa atribuição de horas de encerramento. Dizemos que a cidade b é **alcançável** a partir da cidade a se e só se $b = a$, ou o caminho $p_0, \ldots, p_t$ entre estas duas cidades (nota que em particular $p_0 = a$ e $p_t = b$) satisfaz as seguintes condições:

- o comprimento do caminho p_0, p_1 é no máximo $c[p_1]$, e
- o comprimento do caminho p_0, p_1, p_2 é no máximo $c[p_2]$, e
- $\ldots$

- o comprimento do caminho $p_0, p_1, p_2, \ldots, p_t$ é no máximo $c[p_t]$.

Este ano as duas principais festas vão ser nas cidades X e Y. Para cada atribuição de horas de encerramento, a **pontuação de conveniência** é definido como a soma dos seguintes dois números:

- O número de cidades alcançáveis a partir da cidade X.
- O número de cidades alcançáveis a partir da cidade Y.

Nota que se uma cidade é alcançável a partir tanto da cidade X como a partir da cidade Y, esta cidade conta *duas vezes* para a pontuação de conveniência.

A tua tarefa é calcular a pontuação de conveniência máxima que pode ser obtida por alguma atribuição de horas de encerramento.

Detalhes de Implementação

Deves implementar a seguinte função.

```
int max_score(int N, int X, int Y, int64 K, int[] U, int[] V, int[] W)
```

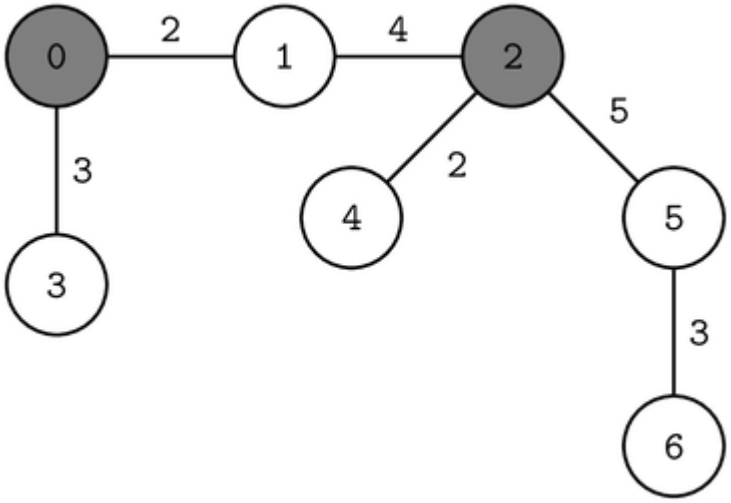
- N: o número de cidades.
- X, Y: as cidades com as principais festas.
- K: o limite da soma das horas de encerramento.
- U, V: arrays de comprimento $N-1$ que descrevem as ligações feitas pelas ruas.
- W: array de comprimento $N-1$ que descreve o comprimento das ruas.
- Esta função deve devolver a pontuação máxima de conveniência que pode ser obtida por alguma atribuição de horas de encerramento.
- Esta função pode ser chamada **várias vezes** em cada caso de teste.

Exemplo

Considera a seguinte chamada:

```
max_score(7, 0, 2, 10,
          [0, 0, 1, 2, 2, 5], [1, 3, 2, 4, 5, 6], [2, 3, 4, 2, 5, 3])
```

Isto corresponde à seguinte rede de ruas:



Supõe que as horas de encerramento são atribuídas da seguinte forma:

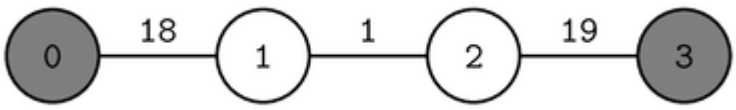
Cidade	0	1	2	3	4	5	6
Hora de encerramento	0	4	0	3	2	0	0

Nota que a soma de todas as horas de encerramento é 9, que não é maior do que $K = 10$. As cidades 0, 1, e 3 são alcançáveis a partir da cidade X ($X = 0$), enquanto que as cidades 1, 2, e 4 são alcançáveis a partir da cidade Y ($Y = 2$). Logo, a pontuação de conveniência é $3 + 3 = 6$. Não existe uma atribuição de horas de encerramento com pontuação de conveniência superior a 6, pelo que a função deve devolver 6.

Considera também a seguinte chamada:

```
max_score(4, 0, 3, 20, [0, 1, 2], [1, 2, 3], [18, 1, 19])
```

Isto corresponde à seguinte rede de ruas:



Supõe que as horas de encerramento são atribuídas da seguinte forma:

Cidade	0	1	2	3
Hora de encerramento	0	1	19	0

A cidade 0 é alcançável a partir da cidade X ($X = 0$), enquanto que as cidades 2 e 3 são alcançáveis a partir da cidade Y ($Y = 3$). Logo, a pontuação de conveniência é $1 + 2 = 3$. Não existe uma atribuição de horas de encerramento com pontuação de conveniência superior a 3, pelo que a função deve devolver 3.

Restrições

- $2 \leq N \leq 200\,000$
- $0 \leq X < Y < N$
- $0 \leq K \leq 10^{18}$
- $0 \leq U[j] < V[j] < N$ (para cada j tal que $0 \leq j \leq N - 2$)
- $1 \leq W[j] \leq 10^6$ (para cada j tal que $0 \leq j \leq N - 2$)
- É possível viajar entre quaisquer duas cidades usando as ruas.
- $S_N \leq 200\,000$, onde S_N é a soma dos valores de N em cada caso de teste entre todas as chamadas de `max_score`.

Subtarefas

Dizemos que uma rede de ruas é **linear** se a rua i liga as cidades i e $i + 1$ (para todo o i tal que $0 \leq i \leq N - 2$).

1. (8 pontos) O comprimento do caminho entre as cidades X e Y é maior que $2K$.
2. (9 pontos) $S_N \leq 50$, a rede de ruas é linear.
3. (12 pontos) $S_N \leq 500$, a rede de ruas é linear.
4. (14 pontos) $S_N \leq 3\,000$, a rede de ruas é linear.
5. (9 pontos) $S_N \leq 20$
6. (11 pontos) $S_N \leq 100$
7. (10 pontos) $S_N \leq 500$
8. (10 pontos) $S_N \leq 3\,000$
9. (17 pontos) Sem restrições adicionais.

Avaliador Exemplo

Seja C o número de cenários diferentes, isto é, o número de chamadas a `max_score`. O avaliador exemplo lê o input no seguinte formato:

- linha 1: C

Descrições dos C cenários seguem-se.

O avaliador exemplo lê a descrição de cada cenário no seguinte formato:

- linha 1: N X Y K
- linha $2 + j$ ($0 \leq j \leq N - 2$): $U[j]$ $V[j]$ $W[j]$

O avaliador exemplo imprime uma única linha para cada cenário, no seguinte formato:

- linha 1: o valor devolvido por `max_score`