Problem Engineers

It is time for the annual server maintenance at *Vianu* . Some of the maintenance can be done by humans, but the cables connecting the network are routed through a series of underground pipes that are too small to access! Luckily, a brilliant engineer came up with the idea of letting trained rats check the computers instead.

Each rat can enter the network at one computer, travel along the cables through the underground pipes, and exit at another computer, checking all computers along the way. Because the rats are quite lazy, each rat will only follow a single simple path between its entry and exit point before getting tired.

The maintenance crew wants to have rats check some computers so that among the remaining unchecked ones, the configuration of security codes is "balanced enough". Your task is to determine the minimum number of rats needed.

Problem

There are N computers, numbered from 0 to $N-1$, linked through $N-1$ cables. The network is a tree (connected and acyclic).

A single rat can check every computer on the unique simple path between the computer where it enters the network and the computer where it exits. Formally, if a rat enters at computer S and exits at computer T ($S, T \in \{0, \ldots, N-1\}$), then the rat will check all computers $v_1, v_2, \ldots, v_k$ such that:

- $v_1 = S$ and $v_k = T$,
- for all $1 \leq i < k$, computers v_i and v_{i+1} are directly connected by a cable,
- k is minimal.

Rats may reuse the same cables and computers, and a computer can be checked by multiple rats.

Each computer $i \in \{0, \ldots, N-1\}$ has an associated positive integer code $C_i \geq 1$. The maintenance crew fix a maximum acceptable difference D. After all rats have done their work, let the set of remaining unchecked computers be $R \subseteq \{0, \ldots, N-1\}$. The crew want to ensure that $|C_i - C_j| \leq D$ for any pair of $i, j \in R$.

In other words, the difference between the maximum and minimum code among *remaining* computers must be at most D.

Each rat can only check one path before getting tired. You want to minimize the number of rats used.

Implementation

You should implement the function

```
int solve(int N, int D, std::vector<int> C, std::vector<int> P, std::vector<int> Q)
```

This function receives as parameters: N, the number of computers, D, the maximum acceptable difference, C, the codes of the computers, and two vectors of length $N-1$, P and Q, meaning there is a cable between computers P_i and Q_i, for all $0 \leq i \leq N-1$.

The function should return the minimum number of rats needed so that after all rats have checked their paths, the remaining unchecked computers satisfy

$$\max_{i \in R} C_i - \min_{i \in R} C_i \leq D.$$

Constraints

- $1 \le N \le 200\,000$.
- $1 \le C_i \le 1\,000\,000\,000$ for all $0 \le i \le N-1$.
- $1 \le D \le 1\,000\,000\,000$.
- $0 \le p_i, q_i < N$, $p_i \ne q_i$, and all pairs (p_i, q_i) are distinct.

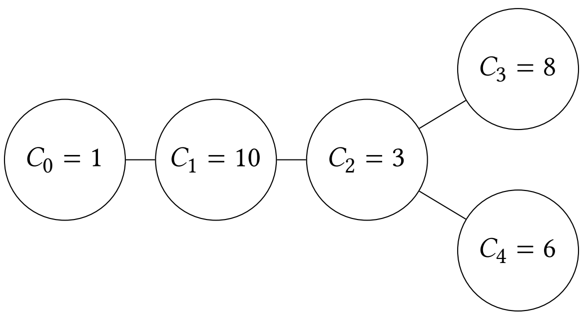
Subtasks

#	Points	Constraints
1	7	$N \le 20$ and $1 \le C_i \le 50$ for all $0 \le i \le N-1$.
2	6	$N \le 1000$ and $1 \le C_i \le 1000$ for all $0 \le i \le N-1$.
3	11	$N \le 1000$.
4	16	$p_i = 0, q_i = i+1$ for all $0 \le i < N-1$.
5	26	$N \le 50000$.
6	34	No additional constraints.

Examples

Input data	Output data
5 3 1 10 3 8 6 0 1 1 2 2 3 2 4	1
20 30 13 36 11 35 4 9 42 9 1 4 11 3 15 31 → 46 41 31 17 11 12 19 5 19 0 19 13 19 9 19 4 19 10 5 1 19 18 0 7 5 8 19 12 5 17 13 16 5 14 13 3 19 6 5 15 5 2 4 11	3

In the first example, there are $N = 5$ computers and $D = 3$. The network can be visualized as:



One possible strategy is to send a single rat on a path that starts at computer 0 and exits at computer 3, passing along the path $0 - 1 - 2 - 3$. After such a path is checked, the network is safe.