Steiner

Alice is working on the "Steiner tree problem".

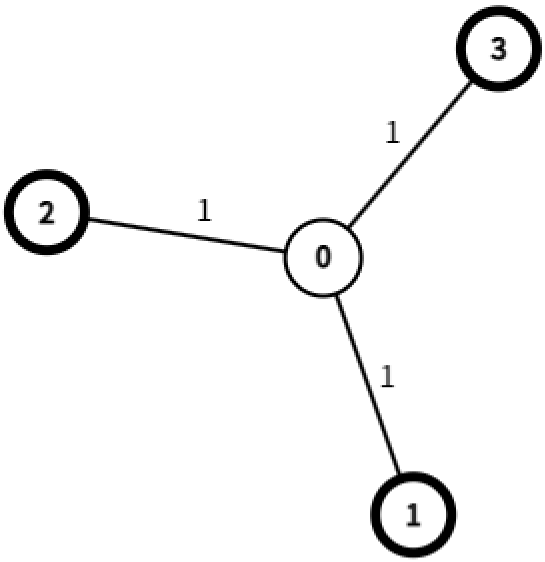
The Steiner tree problem is a graph theory problem named after the Swiss mathematician Steiner (1796-1863):

Given an undirected graph $G = (V, E)$, edges have non-negative integer weights. Then give a subset $S \subseteq V$ of the point set. Define the Steiner tree of S in G as the edge weight of G and the smallest subgraph $G' = (V', E')$ satisfying $S \subseteq V'$ and G' connected. It can be proved that there exists an optimal subgraph G' that is a tree, which is why this problem is called the Steiner tree problem.

The Steiner tree problem is one of the well-known NP-complete problems, and it is difficult to find an exact value in polynomial time. So, Alice came up with the following approximation algorithm (named "Alice's algorithm"):

- Input: undirected graph $G = (V, E)$ with non-negative edge weights and a point set S. Output: The sum of edge weights of the Steiner tree of S in G.
- Construct a complete graph $K = (S, E')$, where for two different points u, v in S, they are connected by an edge in K, and the weight of the edge is $dis_G(u, v)$. where $dis_G(u, v)$ represents the length of the shortest path connecting point u and point v in G.
- Find the minimum spanning tree for K. Output the edge weight sum of the spanning tree.

Obviously, this algorithm is not always able to find the exact value of the edge weight sum of the Steiner tree. For the following example, the edge weight sum of the Steiner tree is 3, but the edge weight sum given by the Alice algorithm is 4.



The subset S contains the points 1, 2, and 3. In the complete graph, the weight of each edge is 2, so the edge weight sum of the minimum spanning tree is 4.

With this particular input, the result of Alice's algorithm is $4/3$ times the exact value. At this point it is said to have achieved an approximate ratio of $4/3$.

Alice proved that the result of Alice's algorithm is always greater than or equal to the exact value, and can achieve an approximate ratio of less than or equal to 2 under any input.

Next, Alice becomes interested in how Alice's algorithm performs when the graph is a tree.

Given a tree of N vertices with non-negative edge weights. Points are numbered 0 through $N-1$ and edges are numbered 0 through $N-2$.

For each i satisfying $0 \leq i \leq N-2$, the edge numbered i connects $U[i]$ and $V[i]$, and its weight is $W[i]$.

Initially, let S be the empty set. Next, S will undergo Q updates. Each update is described by an integer P ($0 \leq P \leq N-1$), which means that if the point P is already in S, the point P will be removed from S, Otherwise, point P is added to S. That is, whether the state of point P in S changes before and after the update. After each update, whether or not the point P is in S remains in the new state until it changes state again in a subsequent update.

Although S is initially an empty set, after each update, S is guaranteed not to be an empty set.

Your task is to determine whether Alice's algorithm can achieve an approximate ratio of 1 after each update. That is to say, is it possible to find the exact value of the edge weight sum of the Steiner tree.

Implementation Details

You need to implement the following function:

```
void init(int N, int[] U, int[] V, int[] W)
```

- N : The number of tree points
- U, V, W : Three arrays of length $N-1$ giving information about each edge
- This function will be called exactly once, before any calls to the function `flip`.

```
int flip(int P)
```

- P: The state of the point numbered P will change. If it was in S before, it will be removed. Otherwise it will be added to S.
- This function shall first perform the specified update. Then, if the Alice algorithm can find the exact value of the edge weight sum of the Steiner tree, it returns 1, otherwise it returns 0.
- Ensure that S is not an empty set after the update.
- This function will be called exactly Q times.

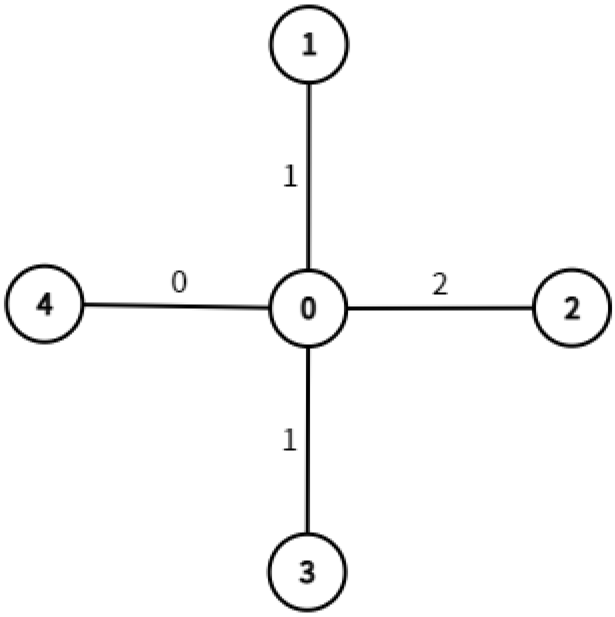
Examples

Example 1

Consider the following call:

```
init(5, [0, 4, 0, 3], [2, 0, 1, 0], [2, 0, 1, 1])
```

The corresponding graph is




```
flip(1)
```

This call adds the point 1 to S. At this time $S = 1$. The exact value of the edge weight sum of a Steiner tree is 0.

And the result of Alice's algorithm is also 0: the minimum spanning tree edge weight sum of a graph with only 1 points is 0. Therefore, the function should return 1.

```
flip(1)
```

This call adds the point 2 to S. At this time $S = 1, 2$. The exact value of the edge weight sum of the Steiner tree is 3: the point set of the subgraph is $0, 1, 2$ and contains the edges $(0, 1)$ and $(0, 2)$.

And the result of Alice's algorithm is also 3: the edge weight between point 1 and 2 is 3. Therefore, the function should return 1.

```
flip(3)
```

This call adds the point 3 to S. At this time $S = 1, 2, 3$. The exact value of the edge weight sum of the Steiner tree is 4: the point set of the subgraph is $0, 1, 2, 3$ and contains the edges $(0, 1)$, $(0, 2)$, and $(0, 3)$.

And the result of Alice’s algorithm is 5: the edge weight between points 1 and 2 is 3, the edge weight between points 1 and 3 is 2, and the edge weight between points 2 and 3 is 3 , the edge weight sum of the minimum spanning tree is 5. Therefore, the function should return 0.

```
flip(4)
```

This call adds the point 4 to S. At this time $S = 1, 2, 3, 4$, and the exact value of the edge weight sum of the Steiner tree is 4: the point set of the subgraph is $0, 1, 2, 3, 4$ and contains edges $(0, 1)$, $(0, 2)$, $(0, 3)$, and $(0, 4)$.

And the result of Alice's algorithm is 4: the edge weight between points 1 and 2 is 3, the edge weight between points 1 and 3 is 2, and the edge weight between points 1 and 4 is 1 , the edge weight between points 2 and 3 is 3, the edge weight between points 2 and 4 is 2, the edge weight between points 3 and 4 is 1, and the edge weight sum of the minimum spanning tree is 4 (select edges $(1, 4)$, $(2, 4)$, and $(3, 4)$). Therefore, the function should return 1.

```
flip(2)
```

This call moves the point 2 out of S. At this time $S = 1, 3, 4$, and the exact value of the edge weight sum of the Steiner tree is 2: the point set of the subgraph is $0, 1, 3, 4$ And contains the edges $(0, 1)$, $(0, 3)$, and $(0, 4)$.

And the result of Alice's algorithm is 2: the edge weight between points 1 and 3 is 2, the edge weight between points 1 and 4 is 1, and the edge weight between points 3 and 4 is 1 , the edge weight sum of the minimum spanning tree is 2. Therefore, the function should return 1.

```
flip(4)
```

This call moves the point 4 out of S. At this time $S = 1, 3$, and the exact value of the edge weight sum of the Steiner tree is 2: the point set of the subgraph is $0, 1, 3$ and contains the edge$(0, 1)$ and $(0, 3)$.

And the result of Alice's algorithm is 2: the edge weight between point 1 and 3 is 2. Therefore, the function should return 1.

Constraints

- $2 \leq N \leq 3 \times 10^5$;
- $1 \leq Q \leq 3 \times 10^5$;
- $0 \leq U[i], V[i] \leq N - 1$ (for each i such that $0 \leq i \leq N - 2$);
- $0 \leq W[i] \leq 10^6$ (for each i such that $0 \leq i \leq N - 2$);
- The graph formed by all $(U[i], V[i])$ (for each i such that $0 \leq i \leq N - 2$) is a tree.

For each call to `flip`:

- $0 \leq P \leq N - 1$;
- After an update, S is guaranteed not to be an empty set.

Subtasks

1. (7 points) $N \le 10$, $W[i] \le 10$(for each i such that $0 \le i \le N-2$)。
2. (2 points) $U[i] = i$, $V[i] = i + 1$(for each i such that $0 \le i \le N-2$)。
3. (7 points) $U[i] = 0$, $V[i] = i + 1$(for each i such that $0 \le i \le N-2$)。
4. (10 points) $N \le 200$, $Q \le 200$。
5. (20 points) $N \le 2\,000$, $Q \le 2\,000$。
6. (14 points) $W[i] \ge 1$(for each i such that $0 \le i \le N-2$)。
7. (22 points) $N \le 50\,000$, $Q \le 50\,000$。
8. (18 points) No additional constraints.

Sample grader

The sample grader reads input in the following format:

- Line 1: N
- line $2+i$ ($0 \le i \le N-2$): $U[i]\ \ V[i]\ \ W[i]$
- Line $N+1$: Q
- Row $N+2+k$ ($0 \le k \le Q-1$): P for the kth `flip`

The sample grader prints your answers in the following format:

- line $1+k$ ($0 \le k \le Q-1$): the return value of the kth call to `flip`