C. Flip Brackets

Flippy the Bird lives in a big tree. The tree can be pictured as a tree with N nodes numbered from 0 to N - 1, rooted at node 0. There is a label on each edge of the tree, which is either '(' or ')' (quotes for clarity)

Flippy has a notepad. He can perform two types of operations :

1. Travel from node u to v using the shortest path. For each edge encountered, flip the label of the edge. (so ‘(' becomes ‘)’ and vice versa)

2. Travel from node u to v using the shortest path. For each edge encountered, record the label of the edge in the notepad.

For each operation of type 2, Flippy wants to know the maximum correct bracket subsequence of the brackets currently written in the notepad. After that, Flippy erases everything from his notepad.

Flippy needs your help to answer his question for all operations of type 2. Can you help him?

A subsequence of a string s is a string that can be obtained by deleting some letters of s without changing the order of the letters. For example, "(()(" is a subsequence of "()(()((".

A correct bracket sequence is a bracket sequence that can be transformed into a correct aryphmetic expression by inserting characters "1" and "+" between the characters of the string.

For example, bracket sequences "()()", "(())" are correct (the resulting expressions "(1)+(1)", "((1+1)+1)"), and ")(" and "(" are not.

Task

You need to implement the functions *initialize*, *flipbrackets* and *querypath*:

void initialize(int N, int P[], char C[])

- We will call this function first. Do whatever initialization that is needed here.
- N - number of nodes of the tree
- P[] - P[i] denotes the parent of node i for all 1 <= i <= N - 1.
- C[] - C[i] denotes the label on the edge connecting i and P[i], for all 1 <= i <= N - 1. Note that this means that C[i] is either '(' or ')'.

void flipbrackets(int u, int v)

- Denotes operation 1. You should flip the labels of the edges on the path from u to v.

int querypath(int u, int v)

- Denotes operation 2. You should return the length of the maximum bracket subsequence of the path from u to v.

Subtasks

Subtask	Points	N, Q	Notes
1	8	N, Q <= 1000	No additional limits
2	10	N, Q <= 200000	P[i] = i - 1 for all 1 <= i <= n - 1. For all flip queries, u is guaranteed to be the parent of v.
3	21	N, Q <= 200000	P[i] = i - 1 for all 1 <= i <= n - 1.
4	27	N, Q <= 200000	For all flip queries, u is guaranteed to be the parent of v.
5	34	N, Q <= 200000	No additional limits

Implementation details

You have to submit exactly one file, called flipbrackets.cpp. This file implements the subprogram described above using the following signatures. You also need to include a header file flipbrackets.h.

void initialize(int N, int P[], char C[])
void flipbrackets(int u, int v)
int querypath(int u, int v)

Sample Grader

First line : N, Q

Next N - 1 lines should contain two integers. The i-th line contains p_i and c_i, parent of node i and the character on the edge from node i to p_i. (1 <= i <= N - 1)

Then, Q lines follow. Each line is either

F u v : Flip path from u to v

Q u v : Find max bracket sequence on path from u to v

For example, this is a valid input :

```
9 8
0 (
1 (
0 )
0 )
4 (
4 )
6 (
4 (
Q 5 7
```

F 1 8
Q 2 8
F 4 5
Q 5 7
Q 2 7
F 0 7
Q 2 6

It should provide the following output :
1
2
0
2
1