Лагер край езерото

Имало един лагер край Охридското езеро, който е съвсем близо до хотел Мизо. Лагерът се състои от N къщички, номерирани с различни цели числа от 1 до N. Съществуват $N-1$ директни пътища между $N-1$ двойки къщички. Мрежата от пътища е организирана по такъв начин, че вие може да пътувате между къщичките като минавате последователно през (един или повече) отделни директни пътища. Разстоянието между двойка къщички се дефинира чрез броя директни пътища, участващи в последователността от директни пътища между тях. Къщичката, достъпна чрез директен път от друга къщичка се нарича нейна съседна къщичка. Накратко, къщичките са свързани в дървовидна структура.

Във всяка къщичка могат да бъдат инсталирани едно или повече Wi-Fi устройства. Ако устройство с честота f и обхват d е инсталирано в дадена къщичка v, то ще се хваща в другите къщички, които са на разстояние по-малко или равно на d от v.

Определете най-високата честота, която се хваща в дадена къщичка v след инсталиране на дадени Wi-Fi устройства в някои от къщичките.

Ще бъдат изпълнени Q операции. Операциите могат да бъдат два типа:

- **update** v f d - Устройство с честота f е инсталирано в къщичка v, като се хваща от всички къщички, които са на разстояние по-малко или равно на d от къщичката v.
- **query** v - Определя се най-високата честота, която се хваща в къщичката v, т.е., има устройство, което се хваща в къщичката v с тази честота. Ако в къщичката v не се хваща нито едно устройство, отговорът трябва да бъде -1.

Детайли по имплементацията

Включете хедър-файла `camp.h`.

Имплементирайте следните функции:

```
void init(int N, int Q, vector < vector < int > > A);
```

Функцията `init` се извиква само веднъж при стартиране на програмата с брой къщички N, брой операции Q, и вектор A, съдържащ директните пътища между двойки къщички като за всяко i ($0 \leq i < N - 1$) има директен път между $A[i][0]$ и $A[i][1]$ ($1 \leq A[i][j] \leq N$).

```
void update(int v, int f, int d);
```

Функцията `update` може да се извика по всяко време, за да се изпълни операция от тип update с параметри v, f, d, описани по-горе в условието на задачата.

```
int query(int v);
```

Функцията `query` може да се извика по всяко време, за да се изпълни операция от тип query с параметър v, описан по-горе в условието на задачата. Функцията трябва да връща отговора на заявката.

Ограничения

- $3 \leq N \leq 10^5$
- $3 \leq Q \leq 10^5$
- $1 \leq v, d \leq N$
- $1 \leq f \leq 10^9$

Локален грейдър

Локалният грейдър има следния формат на входа:

- **Ред 1**: Цяло число N (броя на къщичките).
- **Ред 2 + i** (за $0 \leq i < N - 1$): Две цели числа, разделени с интервал $A[i][0]$ и $A[i][1]$.
- **Ред N + 1**: Цяло число Q (броя операции, които трябва да бъдат изпълнени).
- **Ред N + 2 + i** (за $0 \leq i < Q$):
    - Цяло число $t[i]$ означаващо типа на операцията, която трябва да бъде изпълнена.
        - Ако $t[i]$ е 1, операцията е от тип update и следват 3 цели числа (параметрите, които ще се използват при извикване на функцията `update`): $v[i]$, $f[i]$, $d[i]$.
        - Ако $t[i]$ е 2, операцията е от тип query и следва 1 цяло число (параметърът, който ще бъде използван при извикване на функцията `query`): $v[i]$.

Подзадачи

- **(7 точки)** $N \leq 1000, Q \leq 1000$.
- **(11 точки)** Съществува двойка къщички, такива че пътят между тях съдържа всичките $N-1$ директни пътища, т.е. всички къщички са разположени на една линия (лагерът представлява дълга линия от къщички с директни пътища между тях).
- **(17 точки)** Ако къщичка има 3 или повече съседи, то само две от тези съседни къщички може да имат повече от 1 съсед.
- **(25 точки)** В лагера има точно една къщичка с повече от два съседа.
- **(18 точки)** Разстоянието между всеки две къщички е не повече от 30.
- **(22 точки)** Няма други ограничения.

Пример

Извикване	Върната стойност

`init(9, 7, [`
`[1, 2]`
`[1, 3]`
`[2, 4]`
`[2, 5]`
`[3, 6]`
`[5, 7]`
`[5, 8]`
`[6, 9]`
`])`	

`query(1)`	-1
`update(3, 5, 2)`	
`update(5, 2, 9)`	
`update(4, 8, 2)`	
`query(1)`	8
`query(6)`	5
`query(8)`	2