JOI ツアー (JOI Tour)

IOI 国には N 個の街があり，0 から $N-1$ までの番号が付けられている．また，IOI 国には $N-1$ 本の道路があり，0 から $N-2$ までの番号が付けられている．道路 j $(0 \leqq j \leqq N-2)$ は街 U_j と街 V_j を双方向に結んでいる．どの街からどの街へも何本かの道路を通ることによって移動することができる．

IOI 国のそれぞれの街には飲食店が 1 店舗あり，街 i $(0 \leqq i \leqq N-1)$ にある飲食店の種類は整数 F_i によって表される．具体的には，

- $F_i = 0$ のとき，街 i にある飲食店はジュース専門店である．
- $F_i = 1$ のとき，街 i にある飲食店はオムレツ専門店である．
- $F_i = 2$ のとき，街 i にある飲食店はアイスクリーム専門店である．

IOI 国のツアーガイドである理恵さんは，**JOI ツアー**という観光ツアーを企画している．JOI ツアーとは，以下のような 3 種類の飲食店を巡るツアーである．

1. ジュース専門店のある街 i_0 $(0 \leqq i_0 \leqq N-1)$ を選び，街 i_0 から JOI ツアーを開始する．
2. 街 i_0 にあるジュース専門店を訪れる．
3. オムレツ専門店のある街 i_1 $(0 \leqq i_1 \leqq N-1)$ を選び，バスで街 i_0 から街 i_1 へ最短経路で移動する．
4. 街 i_1 にあるオムレツ専門店を訪れる．
5. アイスクリーム専門店のある街 i_2 $(0 \leqq i_2 \leqq N-1)$ を選び，バスで街 i_1 から街 i_2 へ最短経路で移動する．
6. 街 i_2 にあるアイスクリーム専門店を訪れる．
7. 街 i_2 で JOI ツアーを終了する．

理恵さんは，参加者を飽きさせないために，ツアー中同じ道路を複数回通らないように 3 つの街 (i_0, i_1, i_2) を選ぶことにした．そのような JOI ツアーを**良い JOI ツアー**と呼ぶことにする．あなたの仕事は，理恵さんが最善のツアープランを見つけられるよう，良い JOI ツアーの個数を理恵さんに伝えることである．すなわち，以下の条件を満たす 3 つの街 (i_0, i_1, i_2) の選び方の個数を求めなければならない．

- 街 i_0 にある飲食店はジュース専門店である．
- 街 i_1 にある飲食店はオムレツ専門店である．
- 街 i_2 にある飲食店はアイスクリーム専門店である．
- 街 i_0 から街 i_1 へ最短経路で移動し，街 i_1 から街 i_2 へ最短経路で移動したとき，同じ道路を複数回通らない．

IOI 国では，飲食店の種類の変更が Q 回予定されている．$k+1$ 回目 $(0 \leqq k \leqq Q-1)$ の変更の際には，$0 \leqq X_k \leqq N-1$ を満たす整数 X_k と $0 \leqq Y_k \leqq 2$ を満たす整数 Y_k が与えられる．これは，街 X_k にある飲食

店が整数 Y_k に対応した専門店に変更されることを意味する．すなわち，$Y_k = 0$ ならばジュース専門店に，$Y_k = 1$ ならばオムレツ専門店に，$Y_k = 2$ ならばアイスクリーム専門店に変更される．あなたは，飲食店の変更の情報が与えられたとき，すぐにその変更を反映し，良い JOI ツアーの個数を理恵さんに伝えなければならない．IOI 国の道路と飲食店の情報が与えられたとき，飲食店の変更がなされるたび，それを反映して良い JOI ツアーの個数を返すプログラムを作成せよ．

実装の詳細

あなたは 1 つのファイルを提出しなければならない．

あなたの提出するファイルは `joitour.cpp` という名前である．そのプログラムは `#include` プリプロセッサ指令によって `joitour.h` を読み込むこと．

`joitour.cpp` は以下の関数を実装していなければならない．

- `void init(int N, std::vector<int> F, std::vector<int> U, std::vector<int> V, int Q)`
  - この関数の呼び出しによって，IOI 国の道路と飲食店の情報が与えられる．
  - この関数は最初に 1 回だけ実行される．
  - 引数 `N` は街の数 N を表す．
  - 引数 `F` は長さ N の配列であり，`F[i]` $(0 \leqq i \leqq N-1)$ は街 i にある飲食店の種類 F_i を表す．
  - 引数 `U`, `V` は長さ $N-1$ の配列であり，`U[j]`, `V[j]` $(0 \leqq j \leqq N-2)$ は道路 j が結ぶ 2 つの街 U_j, V_j を表す．
  - 引数 `Q` は飲食店の種類が変更される回数 Q を表す．
- `void change(int X, int Y)`
  - この関数の呼び出しによって，飲食店の種類が変更されたという情報が与えられる．
  - この関数は Q 回実行される．
  - この関数の $k+1$ 回目 $(0 \leqq k \leqq Q-1)$ の呼び出しにおいて，引数 `X` は変更された飲食店のある街の番号 X_k を表す．
  - この関数の $k+1$ 回目 $(0 \leqq k \leqq Q-1)$ の呼び出しにおいて，引数 `Y` は変更後の飲食店の種類に対応する整数 Y_k を表す．変更後の飲食店の種類は，変更前の飲食店の種類と異なることが保証される．
- `long long num_tours()`
  - この関数は以下のタイミングに各 1 回，合計で $Q+1$ 回実行される．
    - 関数 `init` の実行の直後
    - 関数 `change` の各実行の直後
  - この関数は，現時点での良い JOI ツアーの個数を返さなければならない．

重要な注意

- 内部での使用のために他の関数を実装したり，グローバル変数を宣言するのは自由である．
- あなたの提出したプログラムは，標準入力・標準出力，あるいは他のファイルといかなる方法でもやりとりしてはならない．ただし，標準エラー出力にデバッグ情報等を出力することは許される．

コンパイル・実行の方法

作成したプログラムをテストするための採点プログラムのサンプルが，コンテストサイトからダウンロードできるアーカイブの中に含まれている．このアーカイブには，提出しなければならないファイルのサンプルも含まれている．

採点プログラムのサンプルは 1 つのファイルからなる．そのファイルは `grader.cpp` である．作成したプログラムをテストするには，これらのファイル `grader.cpp`, `joitour.cpp`, `joitour.h` を同じディレクトリに置き，次のようにコマンドを実行する．なお，アーカイブの中に含まれている `compile.sh` というファイルを代わりに実行してもよい．

```
g++ -std=gnu++20 -O2 -o grader grader.cpp joitour.cpp
```

コンパイルが成功すれば，`grader` という実行ファイルが生成される．

実際の採点プログラムは，採点プログラムのサンプルとは異なることに注意せよ．採点プログラムのサンプルは単一のプロセスとして起動する．このプログラムは，標準入力から入力を読み込み，標準出力に結果を出力する．

採点プログラムのサンプルの入力

採点プログラムのサンプルは標準入力から以下の形式で入力を読み込む．

N
$F_0\ F_1 \cdots F_{N-1}$
$U_0\ V_0$
$U_1\ V_1$
$\vdots$
$U_{N-2}\ V_{N-2}$
Q
$X_0\ Y_0$

X_1 Y_1
⋮
X_{Q-1} Y_{Q-1}

採点プログラムのサンプルの出力

採点プログラムのサンプルは，関数 `num_tours` を実行するたび，その戻り値を標準出力に 1 行で出力する．

制約

- $3 \leqq N \leqq 200\,000$.
- $0 \leqq F_i \leqq 2$ $(0 \leqq i \leqq N-1)$.
- $0 \leqq U_j < V_j \leqq N-1$ $(0 \leqq j \leqq N-2)$.
- どの 2 つの街の間も，いくつかの道路を経由して移動することができる．
- $0 \leqq Q \leqq 50\,000$.
- $0 \leqq X_k \leqq N-1$ $(0 \leqq k \leqq Q-1)$.
- $0 \leqq Y_k \leqq 2$ $(0 \leqq k \leqq Q-1)$.
- 関数 `change` の各呼び出しにおいて，変更後の飲食店の種類は変更前の飲食店の種類と異なる．
- 入力される値はすべて整数である．

小課題

1. (6 点) $N \leqq 400$，$Q \leqq 100$.
2. (8 点) $N \leqq 4\,000$，$Q \leqq 1\,000$.
3. (6 点) $Q = 0$.
4. (16 点) $U_j = j$，$V_j = j+1$ $(0 \leqq j \leqq N-2)$.
5. (16 点) $U_j = \left\lfloor \frac{j}{2} \right\rfloor$，$V_j = j+1$ $(0 \leqq j \leqq N-2)$.（$\left\lfloor \frac{j}{2} \right\rfloor$ は $\frac{j}{2}$ を超えない最大の整数を表す．）
6. (34 点) $N \leqq 100\,000$，$Q \leqq 25\,000$.
7. (14 点) 追加の制約はない．

やりとりの例

採点プログラムのサンプルの入出力の例と，それに対応する関数の呼び出しの例を以下に示す．

入力例 1	呼び出し	戻り値	出力例 1
3	`init(3, [0, 1, 2], [0, 1], [1, 2], 0)`		1
0 1 2	`num_tours()`	1	
0 1			
1 2			
0			

良い JOI ツアーは $(i_0, i_1, i_2) = (0, 1, 2)$ の 1 通りである．これが良い JOI ツアーであることは以下のように確かめられる．

- $F_0 = 0$ であるから，街 0 にある飲食店はジュース専門店である．
- $F_1 = 1$ であるから，街 1 にある飲食店はオムレツ専門店である．
- $F_2 = 2$ であるから，街 2 にある飲食店はアイスクリーム専門店である．
- 街 0 から街 1 へ最短経路で移動し，街 1 から街 2 へ最短経路で移動したとき，同じ道路を複数回通らない．

したがって，1 回目の `num_tours` の呼び出しは 1 を返さなければならない．
この入力例は小課題 $1, 2, 3, 4, 6, 7$ の制約を満たす．

入力例 2	呼び出し	戻り値	出力例 2
3	`init(3, [0, 1, 2], [0, 1], [1, 2], 2)`		1
0 1 2	`num_tours()`	1	0
0 1	`change(2, 0)`		1
1 2	`num_tours()`	0	
2	`change(0, 2)`		
2 0	`num_tours()`	1	
0 2			

はじめの時点では，良い JOI ツアーは $(i_0, i_1, i_2) = (0, 1, 2)$ の 1 通りである．したがって，1 回目の `num_tours` の呼び出しは 1 を返さなければならない．

1 回目の変更では，街 2 にある飲食店がアイスクリーム専門店からジュース専門店に変更される．このとき，IOI 国からアイスクリーム専門店がなくなるため，良い JOI ツアーは存在しない．したがって，2 回目

の num_tours の呼び出しは 0 を返さなければならない．

2 回目の変更では，街 0 にある飲食店がジュース専門店からアイスクリーム専門店に変更される．このとき，良い JOI ツアーは $(i_0, i_1, i_2) = (2, 1, 0)$ の 1 通りである．したがって，3 回目の num_tours の呼び出しは 1 を返さなければならない．

この入力例は小課題 $1, 2, 4, 6, 7$ の制約を満たす．

入力例 3	出力例 3
7 1 0 2 2 0 1 0 0 1 0 2 1 3 1 4 2 5 2 6 7 0 0 1 1 2 0 3 0 4 2 5 2 6 2	3 0 4 4 0 4 5 5

この入力例は小課題 $1, 2, 5, 6, 7$ の制約を満たす．