島巡り (Island Hopping)

JOI 国には N 個の島があり，1 から N までの番号が付けられている．また，JOI 国には $N-1$ 本の航路があり，1 から $N-1$ までの番号が付けられている．航路 j $(1 \leqq j \leqq N-1)$ は島 A_j と島 B_j を双方向に結んでいる．どの島からどの島へも，いくつかの航路を乗り継ぐことで移動できる．

葵は JOI 国での旅行を計画している．しかし，葵は JOI 国の航路について知らない．そこで，JOI 国に住んでいるビ太郎と以下のようなやりとりを行うことにした．

1. 葵は 1 以上 N 以下の整数 v と 1 以上 $N-1$ 以下の整数 k をビ太郎に伝える．
2. ビ太郎は島 v 以外の $N-1$ 個の島のうち，島 v から k 番目に近い島の番号を葵に伝える．具体的には，整数 i $(1 \leqq i \leqq N,\ i \neq v)$ について $\mathrm{dist}(v, i)$ を島 v から島 i に移動する際に必要な航路の数の最小値としたとき，ビ太郎は $\mathrm{dist}(v, i) \times N + i$ が k 番目に小さくなるような整数 i を葵に伝える．

葵はビ太郎とのやりとりを通じて，どの島とどの島を結ぶ航路があるのかを知りたい．しかし，葵は忙しいので，ビ太郎とのやりとりを L 回までしかできない．

JOI 国の島の数とビ太郎とのやりとりをできる回数が与えられたとき，どの島とどの島を結ぶ航路があるのかを求める葵の戦略を実装した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 `island.cpp` という名前である．`island.cpp` は `#include` プリプロセッサ指令によって `island.h` を読み込み，以下の関数を実装していなければならない．

- `void solve(int N, int L)`
  この関数は各テストケースにおいて 1 回だけ呼び出される．
  - 引数 `N` は JOI 国の島の数 N である．
  - 引数 `L` は葵がビ太郎とのやりとりをできる回数 L である．

`island.cpp` 内では以下の関数を呼び出すことができる．

★ `int query(int v, int k)`
この関数を用いて，ビ太郎に質問を行う．
- ◇ 引数 `v` は，1 以上 N 以下でなければならない．これが満たされていない場合，**不正解 [1]** と判定される．
- ◇ 引数 `k` は，1 以上 $N-1$ 以下でなければならない．これが満たされていない場合，**不正解 [2]** と判定される．
- ◇ 戻り値は，島 `v` 以外の $N-1$ 個の島のうち，島 `v` から `k` 番目に近い島の番号である．より正確な定義については，問題文を参照せよ．
- ◇ 関数 `query` を L 回を超えて呼び出してはならない．L 回を超えて呼び出した場合，**不正解 [3]** と判定される．

★ `void answer(int x, int y)`
この関数を用いて，JOI 国の航路の情報を解答する．
- ◇ 引数 `x` と引数 `y` は，ある航路が結ぶ 2 つの島の番号である．
- ◇ 引数 `x` と引数 `y` は，1 以上 N 以下でなければならない．これが満たされていない場合，**不正解 [4]** と判定される．
- ◇ 島 `x` と島 `y` を結ぶ航路が存在しなければならない．すなわち，`x` $= A_j$，`y` $= B_j$ もしくは `x` $= B_j$，`y` $= A_j$ となる整数 j $(1 \leqq j \leqq N-1)$ が存在しなければならない．これが満たされていない場合，**不正解 [5]** と判定される．
- ◇ 同じ航路の情報を累計で 2 回以上解答してはならない．これが満たされていない場合，**不正解 [6]** と判定される．
- ◇ 関数 `answer` はちょうど $N-1$ 回呼び出される必要がある．関数 `solve` の実行の終了時に関数 `answer` の呼び出し回数が $N-1$ 回でなかった場合，**不正解 [7]** と判定される．

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ことは自由に行える．
- 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`, `island.cpp`, `island.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 island.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．

$N\ L$
$A_1\ B_1$
$A_2\ B_2$
$\vdots$
$A_{N-1}\ B_{N-1}$

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標準出力へ以下の情報を出力する（引用符は実際には出力されない）.

- 正解の場合，関数 `query` の呼び出し回数が “`Accepted: 2024`” のように出力される.
- 不正解の場合，不正解の種類が “`Wrong Answer [4]`” のように出力される.

実行するプログラムが複数の不正解の条件を満たした場合，表示される不正解の種類はそれらのうち 1 つのみである.

採点に関する注意

すべてのテストケースについて，実際の採点プログラムは適応的 (adaptive) **ではない**．これは，採点プログラムは初めから固定された答えを持つということを意味する.

制約

すべての入力データは以下の条件を満たす．

- $3 \leqq N \leqq 300$.
- $1 \leqq A_j \leqq N$ $(1 \leqq j \leqq N-1)$.
- $1 \leqq B_j \leqq N$ $(1 \leqq j \leqq N-1)$.
- $A_j \neq B_j$ $(1 \leqq j \leqq N-1)$.
- どの島からどの島へも，いくつかの航路を乗り継ぐことで移動できる．
- 入力される値はすべて整数である．

小課題

1. (2 点) $N = 3$，$L = 9$.
2. (4 点) $L = N^2$，どの島についてもその島から 1 本の航路で移動できる島は 2 つ以下である．
3. (7 点) $L = 2N$，どの島についてもその島から 1 本の航路で移動できる島は 2 つ以下である．
4. (9 点) $L = N^2$，島 1 から 1 本の航路で移動できる島は 3 つである．島 1 以外の他のどの島についてもその島から 1 本の航路で移動できる島は 2 つ以下である．
5. (13 点) $L = 3N$，島 1 から 1 本の航路で移動できる島は 3 つである．島 1 以外の他のどの島についてもその島から 1 本の航路で移動できる島は 2 つ以下である．
6. (15 点) $L = N^2$.
7. (22 点) $L = 3N$.
8. (28 点) $L = 2N$.

やりとりの例

採点プログラムのサンプルが読み込む入力の例と，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	関数の呼び出しの例		
	呼び出し	呼び出し	戻り値
`4 16`	`solve(4, 16)`		
`1 2`		`query(2, 1)`	`1`
`2 4`		`query(3, 1)`	`4`
`4 3`		`answer(2, 4)`	
		`query(2, 2)`	`4`
		`answer(2, 1)`	
		`query(3, 2)`	`2`
		`query(2, 1)`	`1`
		`answer(3, 4)`	

島 2 から島 1，島 3，島 4 へ移動するのに必要な航路の数の最小値はそれぞれ $1, 2, 1$ である．例えば，島 2 から島 3 へは航路 2，航路 3 をこの順に乗り継ぐことで移動できる．

島 2 以外の島 i を $\mathrm{dist}(2, i) \times N + i$ が小さい順に並び替えると島 1，島 4，島 3 となる．したがって，関数 `query(2, 1)` の戻り値は 1 となる．また，関数 `query(2, 2)` の戻り値は 4 となる．

入力例 1 は小課題 $2, 6$ の制約を満たす．コンテストサイトからダウンロードできるファイルのうち，`sample-01-in.txt` は入力例 1 に対応する．

入力例 2	関数の呼び出しの例		
	呼び出し	呼び出し	戻り値
5 25	solve(5, 25)		
5 2		query(1, 3)	5
3 1		query(1, 4)	2
1 4		answer(3, 1)	
1 5		query(2, 4)	4
		query(3, 1)	1
		query(3, 2)	4
		answer(1, 5)	
		answer(4, 1)	
		answer(2, 5)	

島 1 から島 2，島 3，島 4，島 5 へ移動するのに必要な航路の数の最小値はそれぞれ 2, 1, 1, 1 である．例えば，島 1 から島 2 へは航路 4，航路 1 をこの順に乗り継ぐことで移動できる．

島 1 以外の島 i を $\mathrm{dist}(1, i) \times N + i$ が小さい順に並び替えると島 3，島 4，島 5，島 2 となる．したがって，関数 query(1, 3) の戻り値は 5 となる．また，関数 query(1, 4) の戻り値は 2 となる．

入力例 2 は小課題 4, 6 の制約を満たす．コンテストサイトからダウンロードできるファイルのうち，sample-02-in.txt は入力例 2 に対応する．

コンテストサイトからダウンロードできるファイルに含まれる sample-01-in.txt, sample-02-in.txt は，採点プログラムのサンプルの入力として用いることができる．