Island Hopping

There are N islands in JOI Kingdom, numbered from 1 to N. There are $N-1$ sea routes in JOI Kingdom, numbered from 1 to $N-1$. The sea route j ($1 \leq j \leq N-1$) connects island A_j and island B_j bi-directionally. It is possible to travel from any island to any other island by using some sea routes.

Aoi is planning a trip in JOI Kingdom. However, she doesn't know about sea routes in JOI Kingdom. She is going to ask questions with Bitaro, a citizen in JOI Kingdom, in the following way:

1. Aoi tells Bitaro an integer v, where $1 \leq v \leq N$, and an integer k, where $1 \leq k \leq N-1$.
2. Bitaro tells Aoi the number on the island which is the k-th closest from island v among the $N-1$ islands except for island v. To be specific, he tells her an integer i that $\mathrm{dist}(v, i) \times N + i$ ($1 \leq i \leq N$, $i \neq v$) is the k-th smallest, where $\mathrm{dist}(v, i)$ is the minimum number of sea routes to move from island v to island i.

Aoi wants to know all sea routes in JOI Kingdom by asking questions to Bitaro. Since Aoi does not want to spend too much time, she can ask at most L questions to Bitaro.

Write a program which, given the number of islands in JOI Kingdom and the limit on number of questions to Bitaro, implements Aoi's strategy to find all sea routes.

Implementation Details

You need to submit one file.

The file is `island.cpp`. It should implement the following function. The program should include `island.h` using the preprocessing directive `#include`.

- `void solve(int N, int L)`

  This function is called only once for each test case.
  - The parameter `N` is the number of islands N.
  - The parameter `L` is the limit on number of questions L.

In `island.cpp`, you can call the following function.

★ `int query(int v, int k)`

Using this function, Aoi asks question to Bitaro.
- ◇ The parameter `v` must be between 1 and N. If not, your program is judged as **Wrong Answer [1]**.
- ◇ The parameter `k` must be between 1 and $N-1$. If not, your program is judged as **Wrong Answer [2]**.
- ◇ The return value is the number on the island which is the k-th closest from island v among the $N-1$ islands except for island v. See the problem statement for more detailed definition.
- ◇ You must not call function `query` more than L times. If it is called more than L times, your program is judged as **Wrong Answer [3]**.

★ `void answer(int x, int y)`

Your program answers a sea route in JOI Kindom by calling this function.
- ◇ The parameters `x` and `y` are the numbers of two islands connected by a sea route.
- ◇ The parameters `x` and `y` must be between 1 and N. If not, your program is judged as **Wrong Answer [4]**.
- ◇ There must exist a sea route which connects islands `x` and `y`. In other words, there must exist an integer j $(1 \leq j \leq N-1)$ that $\texttt{x} = A_j$ and $\texttt{y} = B_j$, or $\texttt{x} = B_j$ and $\texttt{y} = A_j$. If not, your program is judged as **Wrong Answer [5]**.
- ◇ The program must not answer the information of the same sea route twice or more. If this condition is not met, your program is judged as **Wrong Answer [6]**.
- ◇ Function `answer` must be called exactly $N-1$ times. If the number of calls of function `answer` is not $N-1$ when the execution of function `solve` ends, your program is judged as **Wrong Answer [7]**.

Important Notices

- Your program can implement other functions for internal use, or use global variables.
- Your program must not use the standard input and the standard output. Your program must not communicate with other files by any methods. However, your program may output debugging information to the standard error.

Compilation and Test Run

You can download an archive file from the contest webpage which contains the sample grader to test your program. The archive file also contains a sample source file of your program.

The sample grader is the file `grader.cpp`. In order to test your program, put `grader.cpp`, `island.cpp`, `island.h` in the same directory, and run the following command to compile your programs. Instead, you may run `compile.sh` contained in the archive file.

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

When the compilation succeeds, the executable file `grader` is generated.

Note that the actual grader is different from the sample grader. The sample grader will be executed as a single process, which will read input data from the standard input and write the results to the standard output.

Input for the Sample Grader

The sample grader reads the following data from the standard input.

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

Output for the Sample Grader

The sample grader outputs the following information to the standard output (quotes for clarity).

- If your program is judged as correct, it reports the number of calls to `query` as "`Accepted: 2024`".

- If your program is judged as any type of Wrong Answer, the sample grader writes its type as "`Wrong Answer [4]`".

If your program satisfies the conditions of several types of Wrong Answer, the sample grader reports only one of them.

Notices for Grading

The actual grader is **not** adaptive for all test cases. It means that the grader has a pre-determined answer for each test case.

Constraints

All the input data satisfy the following conditions.

- $3 \leq N \leq 300$.
- $1 \leq A_j \leq N$ $(1 \leq j \leq N-1)$.
- $1 \leq B_j \leq N$ $(1 \leq j \leq N-1)$.
- $A_j \neq B_j$ $(1 \leq j \leq N-1)$.
- It is possible to travel from any island to any other island by using some sea routes.
- Given values are all integers.

Subtasks

1. (2 points) $N = 3$, $L = 9$.
2. (4 points) $L = N^2$. Each island has at most 2 connecting sea routes.
3. (7 points) $L = 2N$. Each island has at most 2 connecting sea routes.
4. (9 points) $L = N^2$. Island 1 has 3 connecting sea routes, and each of the other island has at most 2 connecting sea routes.
5. (13 points) $L = 3N$. Island 1 has 3 connecting sea routes, and each of the other island has at most 2 connecting sea routes.
6. (15 points) $L = N^2$.
7. (22 points) $L = 3N$.
8. (28 points) $L = 2N$.

Sample Communication

Here is a sample input for the sample grader and corresponding function calls.

Sample Input 1	Sample Function Calls		
	Calls	Calls	Return Values

`4 16`
`1 2`
`2 4`
`4 3`
	`solve(4, 16)`		

		`query(2, 1)`	`1`
		`query(3, 1)`	`4`
		`answer(2, 4)`	
		`query(2, 2)`	`4`
		`answer(2, 1)`	
		`query(3, 2)`	`2`
		`query(2, 1)`	`1`
		`answer(3, 4)`	

The minimum number of sea routes to move from island 2 to islands $1, 3, 4$, are $1, 2, 1$, respectively. For example, to move from island 2 to island 3, we can use sea route 2 and then sea route 3.

Sorting the islands i ($\neq 2$) in increasing order of $\mathrm{dist}(2, i) \times N + i$, the result is island 1, island 4, island 3, in this order. Therefore, the return value of `query(2, 1)` is 1, and the return value of `query(2, 2)` is 4.

Sample Input 1 satisfies the constraints of Subtasks $2, 6$. Among the files which can be obtained from the contest webpage, `sample-01-in.txt` corresponds to Sample Input 1.

Sample Input 2	Sample Function Calls		
	Calls	Calls	Return Values

`5 25`
`5 2`
`3 1`
`1 4`
`1 5`
	`solve(5, 25)`		

		`query(1, 3)`	`5`
		`query(1, 4)`	`2`
		`answer(3, 1)`	
		`query(2, 4)`	`4`
		`query(3, 1)`	`1`
		`query(3, 2)`	`4`
		`answer(1, 5)`	
		`answer(4, 1)`	
		`answer(2, 5)`	

The minimum number of sea routes to move from island 1 to islands 2, 3, 4, 5, are 2, 1, 1, 1, respectively. For example, to move from island 1 and island 2, we can use sea route 4 and then sea route 1.

Sorting the islands i ($\neq 1$) in increasing order of $\mathrm{dist}(1, i) \times N + i$, the result is island 3, island 4, island 5, island 2, in this order. Therefore, the return value of `query(1, 3)` is 5, and the return value of `query(1, 4)` is 2.

Sample Input 2 satisfies the constraints of Subtasks 4, 6. Among the files which can be obtained from the contest webpage, `sample-02-in.txt` corresponds to Sample Input 2.

Both `sample-01-in.txt` and `sample-02-in.txt` can be used as inputs for the sample grader.