Minerals

Professor JOI's laboratory is researching N kinds of minerals. There are 2 slices of each kind of mineral. There are $2N$ slices in total, numbered from 1 to $2N$.

One day, as assistant Bitaro has dropped the box containing the $2N$ slices, he has no idea which slice and which slice are of the same kind of mineral.

The laboratory owns a device which can count the number of kinds of minerals when some slices are inserted in it, by measuring the wavelength each mineral absorbs. Bitaro is going to determine the N pairs of the same kind of mineral from the $2N$ slices. At first, there are no slices inserted in the device. Bitaro can perform the following operations:

- Inserting one slice into the device, Bitaro knows how many kinds of minerals are in the device.
- Extracting one slice from the device, Bitaro knows how many kinds of minerals are in the device.

So that Professor JOI will not find Bitaro bringing about troubles, Bitaro can perform these operations at most 1 000 000 times in total.

Write a program which, given the number of kinds of minerals, using the device, determines all the pairs of the same kind of mineral.

Implementation Details

You need to submit one file.

The name of the file is `minerals.cpp`. It should implement the following function. The program should include `minerals.h`.

- `void Solve(int N)`
  This function is called exactly once for each test case.
  - The parameter `N` represents N, the number of kinds of minerals.

  Your program can call the following functions.
  ★ `int Query(int x)`
  For the index of the slice you specify, this function extracts the slice from the device if the slice is already in the device, or inserts the slice into the device otherwise. Then it returns the number of kinds of minerals in the device.
    ◇ You specify the index x of the slice via parameter `x`. This must satisfy $1 \leq x \leq 2N$. Otherwise, your program is considered as **Wrong Answer [1]**.

  ◇ The function `Query` must not be called more than 1 000 000 times. Otherwise, your program is considered as **Wrong Answer [2]**.

★ `void Answer(int a, int b)`

Using this function, you answer the pairs of the same mineral.

  ◇ The parameters `a` and `b` represent that the a-th slice and the b-th slice are the same kind of mineral. These must satisfy $1 \leq a \leq 2N$ and $1 \leq b \leq 2N$. Otherwise, your program is considered as **Wrong Answer [3]**. If the same value appears more than once in a or b in total, your program is considered as **Wrong Answer [4]**. If slices of different kinds of mineral are specified, your program is considered as **Wrong Answer [5]**.

The function `Answer` must be called exactly N times. If the number of calls to the function `Answer` is not N at the end of the execution of the function `Solve`, your program is considered as **Wrong Answer [6]**.

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`, `minerals.cpp`, and `minerals.h` in the same directory, and run the following command to compile your programs.

```
g++ -std=gnu++14 -O2 -o grader grader.cpp minerals.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
X_1 Y_1
$\vdots$
X_N Y_N

X_i and Y_i ($1 \leq i \leq N$) denote that the X_i-th slice and the Y_i-th slice are the same kind of mineral.

Output of the Sample Grader

When the program terminates successfully, the sample grader writes the following information to the standard output (quotes for clarity).

- If your program is considered correct, it writes the number of calls to the function `Query` as "`Accepted: 100`".
- If your program is considered as Wrong Answer, it writes its type as "`Wrong Answer [1]`".

If your program is considered as several types of Wrong Answer, the sample grader reports only one of them.

Constraints

Refer to the "Input for the Sample Grader" section for the definition of X_i and Y_i.

- $1 \leq N \leq 43\,000$.
- $1 \leq X_i \leq 2N$ ($1 \leq i \leq N$).
- $1 \leq Y_i \leq 2N$ ($1 \leq i \leq N$).
- $X_i \neq X_j$ ($1 \leq i < j \leq N$).
- $Y_i \neq Y_j$ ($1 \leq i < j \leq N$).
- $X_i \neq Y_j$ ($1 \leq i \leq N, 1 \leq j \leq N$).

Subtasks

1. (6 points) $N \leq 100$.
2. (25 points) $N \leq 15\,000$, $1 \leq X_i \leq N$, $N + 1 \leq Y_i \leq 2N$ $(1 \leq i \leq N)$.
3. (9 points) $N \leq 15\,000$.
4. (30 points) $N \leq 38\,000$.
5. (5 points) $N \leq 39\,000$.
6. (5 points) $N \leq 40\,000$.
7. (5 points) $N \leq 41\,000$.
8. (5 points) $N \leq 42\,000$.
9. (10 points) No additional constraints.

Sample Communication

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

Sample Input 1	Sample Calls		
	Call	Call	Return
4	`Solve(4)`		
1 5		`Query(1)`	1
2 6		`Query(2)`	2
3 4		`Query(5)`	2
7 8		`Query(2)`	1
		`Answer(3, 4)`	(none)
		`Answer(5, 1)`	(none)
		`Answer(8, 7)`	(none)
		`Answer(2, 6)`	(none)