Chameleon's Love

In JOI Zoo, there are $2N$ chameleons, numbered from 1 to $2N$. Among them, N chameleons have sex X. The remaining N chameleons have sex Y.

Each chameleon has its **original color**. The following are known about the original colors.

- The original colors of the N chameleons of sex X are distinct.
- For each chameleon of sex X, there exists a unique chameleon of sex Y with the same original color.

Now, JOI Zoo is in the season of new loves. Each chameleon **loves** another chameleon. The following are known about the chameleons' loves.

- Each chameleon loves exactly one chameleon whose sex is different from itself.
- A chameleon and the chameleon which it loves have different original colors.
- There do not exist two chameleons which love the same chameleon.

You can gather some of the chameleons and organize a meeting. For each chameleon s attending the meeting, let t be the chameleon which s loves. The **skin color** of s is decided as follows.

- If t attends the meeting, the skin color of s is the original color of t.
- If t does not attend the meeting, the skin color of s is the original color of s.

The skin color of a chameleon may change for different meetings. For each meeting you organize, you can count the number of kinds of skin colors of the chameleons attending the meeting.

By organizing meetings at most 20 000 times, you want to determine all pairs of chameleons with the same original color.

Write a program which, given the number of chameleons, determines all pairs of chameleons with the same original color by organizing meetings at most 20 000 times.

Implementation Details

You need to submit one file.

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

- `void Solve(int N)`
  This function is called exactly once for each test case.

- The parameter `N` is N, the number of chameleons whose sexes are X.

Your program can call the following function.

★ `int Query(const std::vector<int> &p)`

You organize a meeting of chameleons by calling this function.

- The parameter `p` is the list of chameleons attending the meeting.
- The return value is the number of kinds of skin colors of the chameleons attending the meeting.
- Each element of the parameter `p` should be an integer greater between 1 and $2N$, inclusive. If this condition is not satisfied, your program is judged as **Wrong Answer [1]**.
- The elements of the parameter `p` should be distinct. If this condition is not satisfied, your program is judged as **Wrong Answer [2]**.
- Your program should not call the function `Query` more than 20 000 times. If this condition is not satisfied, your program is judged as **Wrong Answer [3]**.

★ `void Answer(int a, int b)`

By calling this function, you answer a pair of chameleons with the same original color.

- The parameters `a` and `b` mean the chameleon a and the chameleon b have the same original color.
- It must hold that $1 \leq a \leq 2N$ and $1 \leq b \leq 2N$. If these conditions are not satisfied, your program is judged as **Wrong Answer [4]**.
- Your program should not call this function with same values of a or b more than once in total. If this condition is not satisfied, your program is judged as **Wrong Answer [5]**.
- If the original colors of the chameleon a and the chameleon b are different, your program is judged as **Wrong Answer [6]**.
- Your program should call the function `Answer` exactly N times. When the function `Solve` terminates, if the number of calls to the function `Answer` is different from N, 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`, `chameleon.cpp`, `chameleon.h` in the same directory, and run the following command to compile your programs.

```
g++ -std=gnu++14 -O2 -o grader grader.cpp chameleon.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
$Y_1 \cdots Y_{2N}$
$C_1 \cdots C_{2N}$
$L_1 \cdots L_{2N}$

Y_i ($1 \leq i \leq 2N$) is the sex of the chameleon i, which is 0 or 1. It is 0 if the sex is X, and 1 if the sex is Y.

C_i ($1 \leq i \leq 2N$) is the original color of the chameleon i, which is an integer greater than or equal to 1 and less than or equal to N.

L_i ($1 \leq i \leq 2N$) is the number of the chameleon which the chameleon i loves.

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

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

Constraints

All input data satisfy the following conditions. For the meaning of Y, C, and L, see Input for the Sample Grader.

- $2 \leq N \leq 500$.
- $0 \leq Y_i \leq 1$ $(1 \leq i \leq 2N)$.
- $1 \leq C_i \leq N$ $(1 \leq i \leq 2N)$.
- For each j $(1 \leq j \leq N)$, there exists a unique i $(1 \leq i \leq 2N)$ satisfying $Y_i = 0$ and $C_i = j$.
- For each j $(1 \leq j \leq N)$, there exists a unique i $(1 \leq i \leq 2N)$ satisfying $Y_i = 1$ and $C_i = j$.
- $1 \leq L_i \leq 2N$ $(1 \leq i \leq 2N)$.
- $Y_i \neq Y_{L_i}$ $(1 \leq i \leq 2N)$.
- $C_i \neq C_{L_i}$ $(1 \leq i \leq 2N)$.
- $L_k \neq L_l$ $(1 \leq k < l \leq 2N)$.

Subtasks

1. (4 points) $L_{L_i} = i$ $(1 \leq i \leq 2N)$.
2. (20 points) $N \leq 7$.
3. (20 points) $N \leq 50$.
4. (20 points) $Y_i = 0$ $(1 \leq i \leq N)$.
5. (36 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 0 1 0 0 1 1 0		`Query([])`	`0`
4 4 1 2 1 2 3 3		`Query([6, 2])`	`2`
4 3 8 7 6 5 2 1		`Query([8, 1, 6])`	`2`
		`Query([7, 1, 3, 5, 6, 8])`	`4`
		`Query([8, 6, 4, 1, 5])`	`3`
		`Answer(6, 4)`	
		`Answer(7, 8)`	
		`Answer(2, 1)`	
		`Answer(3, 5)`	

Among the files you can download from the contest site, `sample-02.txt` satisfies the constraints for Subtask 1, and `sample-03.txt` satisfies the constraints for Subtask 4.