Black Panthers

Type	Time limit	Memory limit
Batch (interactive)	5 seconds (see notes)	128 MB

Statement

As a member of the most infamous organisation of thieves, the *Pink Panthers*[1], you have managed to infiltrate a rival thief organisation, the *Black Panthers*. Intelligence has told you the following information about the organisational structure of the *Black Panthers*:

- There are N members in the *Black Panthers*, with ID numbers from 0 through $N-1$.
- Each member except **the Boss** has 1 supervisor, such that the hierarchy forms a rooted binary tree structure if you consider the members as nodes, and the parent of each node as its supervisor, if any.
- That is, each member supervises at most 2 members, and the tree is rooted at the Boss's node.

You wish to determine the supervisor of each member on the organisation, as well as who the Boss is. Your contact in a intelligence unit can answer your questions of the form "For two members a, b what is the ID of the lowest common ancestor (LCA) of a and b on the hierarchy?" However, your intelligence contact does not want to be suspected for espionage, so you may only ask at most 20000 questions. Please note: for some subtasks, the grader on the contest system may take a significant amount of runtime. You are guaranteed that the grader will take at most 2 seconds of the total 5 seconds.

Implementation

You **must not** implement a main function. Instead, you should include the line in your code `#include "panther.h"` and implement the function:

```
vector<int> supervisors(int N, int S);
```

where:

- `N` is the number of members.
- `S` is the subtask number (see below).
- `supervisors` should return a vector A of N values where $A[i]$ is the member who is the supervisor of member i. If member i is the Boss, then $A[i] = -1$.

The grader will have implemented the function which may be called:

```
int query(int a, int b);
```

where:

[1] See Selection Exam 2017, *White Collar*

- `a, b` are the member IDs you wish to query.
- `query` will return the member ID of the lowest common ancestor.

When judging, `supervisors` will be called exactly once at the beginning.

Experimentation

The source files `panther.h` and `grader.cpp` are provided for you to experiment on your machine. Please note that the grader used for judging may have different behaviour to the provided grader.

Compile your solution with:

```
g++ -std=c++11 -O2 -Wall -static panther.h sol.cpp grader.cpp -o panther
```

Sample Grader

The sample grader reads the input in the following form:
Line 1: N S
Line 2: P_0 P_1 ... P_{N-1}

Here P_i is the supervisor of member i, and if i is the Boss then $P_i = -1$. S is the subtask number. The sample grader will then execute the aforementioned interaction procedure and print 2 lines:
Line 1: The number of queries used.
Line 2: The values of the vector returned by `supervisors` $A[0] \dots A[N-1]$.

Subtasks and Constraints

For all subtasks:

- $1 \leq N \leq 10^3$
- $1 \leq S \leq 4$ (see below)
- The organisational structure forms a binary tree

Subtasks

Number	Points	Max N	Other Constraints
1	29	200	None
2	14	1000	$P_i \neq P_j$ for $i \neq j$
3	17	1000	$P_i < i$ for all i and $P_0 = -1$
4	40	1000	None

In addition, the grader for subtask 3 will be **guaranteed to have a predetermined answer** before the function `supervisors` is called. In other words, it will not be adaptive.

Scoring

- For each test case if your function returns a incorrect hierarchy or uses more than 20000 queries it will be judged as wrong answer and receive 0 for the corresponding subtask.
- For subtasks 1,2 and 3, if your function returned the correct hierarchy in at most 20000 queries for each testcase, you will gain all the points for that subtask.
- Subtask 4 is scored via sliding scale. Let Q be the maximum number of queries used in a testcase in subtask 4. If $16000 < Q \leq 20000$ then your score is 8. Otherwise your score for subtask 4 is linearly interpolated, with $Q = 10000$ scoring 40 points and $Q = 16000$ scoring 16. Formally, your score $S(Q)$ is such that

$$S(Q) = \begin{cases} 40 & Q \leq 10000 \\ 40 - \frac{24(Q-10000)}{6000} & 10000 < Q \leq 16000 \\ 8 & 16000 < Q \leq 20000 \\ 0 & Q > 20000 \end{cases}$$

Sample Grader Input and Sample Session

The sample grader reads from standard input:

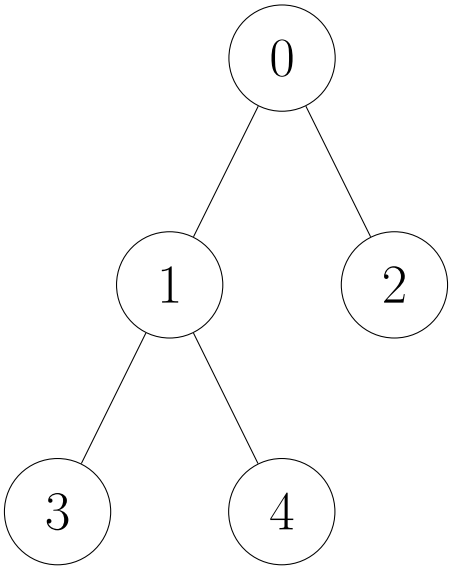
```
5 1
-1 0 0 1 1
```

The sample grader writes to standard output:

```
3
-1 0 0 1 1
```

Grader	Program	Description
supervisors(5, 1)		The grader calls the function.
	query(0, 1)	Member 0 is supervisor of member 1.
returns 0		
	query(3, 4)	Member 1 supervises members 3 and 4
returns 1		
	query(4 2)	Member 0 supervises member 2. Member 0 also supervises member 1, who is the supervisor of member 4.
returns 0		
Ends interaction	Returns [-1,0,0,1,1]	

Here is what the sample input looks like:



Notes

You may find on CMS a zipped package containing:

- A stub implementation of your function.
- Files for the sample grader and `panther.h` as well as compile command.
- 3 more sample inputs, adhering to subtasks 2,3,4 respectively.

You may assume the grader is adaptive for subtasks 1, 2 and 4, and the grader will take no more than 2 seconds of runtime.