Third Longest Stick

Input file: `sticks.in` **100 points**
Output file: `sticks.out` **Time limit: 1sec**
Source Code: `sticks.pas/.c/.cpp` **Memory limit: 64 MB**

We are given a set of wooden sticks, labeled from ***1*** to ***N***. We want to select a third longest stick. A stick ***c*** is a third longest one, if there are two other sticks, say ***a*** and ***b,*** such that ***a*** is shorter than ***b***, ***b*** is shorter than ***c***, and the length of any other stick that is shorter than ***c*** equals either to the length of ***a*** or the length of ***b***. We can not measure the length of a stick, but we can compare two sticks, say ***a*** and ***b,*** and observe whether ***a*** is shorter, is of the same length, or is longer than **b**.

Task

You are to write a program that selects a third longest stick.

Library

To perform queries, you are given a library **`look`** with three operations:

- **`GetN`**, to be called once at the beginning, without arguments; it returns ***N***, the number of sticks. **`GetN`** must be called before the first call to **`Compare`**.
- **`Compare`**, to be called with two stick labels as arguments. **`Compare(x,y)`** returns ***-1*** if stick ***x*** is shorter than stick ***y***, returns ***0*** if ***x*** and ***y*** are of equal length, and returns ***1*** if stick ***x*** is longer than stick ***y***.
- **`Answer`**, to be called once in the end, with one stick label as argument; it reports the label of a third longest stick and it properly terminates the execution of your program. If there are more than one third longest sticks, your program should report only one; it does not matter which one. If there is no solution then your program must pass the argument ***0*** to **`Answer`**.

Instruction for Pascal programmers: include the import statement

```
uses look;
```

in your source code.

Instructions for C/C++ programmers: use the instruction

```
#include "look.h"
```

in your source code.

Create a project file in the task directory, add the files `sticks.c` (`sticks.cpp`), `look.h` and `look.o` into this project, and then *compile* and/or *make* your program.

(Using Dev-C++ IDE, choose the Project/Project Options/Files menu, select the file `look.o` and unset "include in compilation" and set "include in linking").

Command line compilation: `gcc/g++ –o sticks sticks.c look.o`

Experimentation

You are provided with a toolset that contains the libraries for WinXP and Linux. You can download it from the competition server as a zip archive. Copy the appropriate library files into your task directory.

You can experiment with the library by creating a text file `sticks.in`. The file must contain two lines. The first line must contain a single integer, N ($1 \le N \le 1\,000\,000$), the number of sticks. The second line must contain N integers: the ith integer L_i ($1 \le L_i \le 1\,000\,000$) is the length of the stick with label i.

The library `look` produces a text file `sticks.out` that contains the solution your program submitted by a call to **`Answer`**.

Constraints

- Your program must not read or write any files.
- For the number of sticks N, we have $1 \le N \le 1\,000\,000$.
- FreePascal library file names: `look.ppu` and `look.o` for WinXP, and `look.o` for Linux.
- Pascal function declarations:

```
function GetN: longint;
function Compare(x, y: longint): integer;
procedure Answer(x: longint);
```

- C/C++ library file names: `look.h`, and `look.o`
- C/C++ function declarations:

```
long GetN(void);
int Compare(long x, long y);
void Answer(long x);
```

Example

sticks.in	sticks.out
10 1 9 3 9 8 4 9 7 4 21	6

Note that 9 is also a correct answer.