

Minimums on Arcs

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 1024 megabytes

This problem can only be solved in the C++ programming language.

On a circle, there are n distinct numbers from 1 to n , with the number p_i written at the i -th vertex when traversing counterclockwise. For each pair of values x and y , there are two ways to get from x to y — going clockwise or counterclockwise. In this problem, n is always odd, so these two paths will have different lengths.

Let $f(x, y)$ be the minimum value we encounter on the way if we travel from number x to number y via the shortest path.

For example, if $p = [1, 5, 3, 2, 7, 4, 6]$, then the values 5 and 7 are connected by two paths, and the values of the vertices on these paths are, respectively, $[5, 3, 2, 7]$ and $[5, 1, 6, 4, 7]$. The shortest path is the first one mentioned, so $f(5, 7) = \min\{5, 3, 2, 7\} = 2$.

We say that two permutations p and q are *equivalent* if the value of the function f for these permutations is the same for all x and y . You are given a permutation p , and you need to guess any permutation q that is equivalent to it by making queries to the function $f(x, y)$.

Solution Format

This is an unusual problem. It has a testing format with a grader, where you need to implement only the function `guess` with the solution. This function will be called by the testing program of the jury (the grader), and the returned value of the function will be accepted as the solution to the problem.

In particular, this means that the code you submit **must not input or output any data**. Your code **must not** contain a function `main`. If necessary, you can implement any number of helper functions, structures, classes, and global variables, but all the code of your solution must be in one file.

You must implement the following function:

```
std::vector<int> guess(int n);
```

The function `guess` takes as input the number n — the length of the permutation.

In the implementation of the function `guess`, you can use the function `min_value`, which is implemented by the grader. This function takes two values x and y as input and returns $f(x, y)$ as the result. If an incorrect query is made or the number of queries is exhausted, the program will automatically terminate.

To access the function `min_value` in your solution, the first line of your code must include the header file with the following line:

```
#include "checkpoint.h"
```

The definition of the function `min_value` in the header file is as follows:

```
int min_value(int x, int y);
```

All parameters (values x , y , as well as the permutation q you return) are given in **1-indexing**.

Your function `guess` must determine the unknown permutation p , up to the specified equivalence, using calls to the function `min_value`. That is, if the jury has chosen the permutation p , and the function `guess` returns any permutation q that is equivalent to it, then that answer is considered correct.

During one run of the grader, **the jury may make several calls to the function `guess`**, in which case the function `min_value` will return the values $f(i, j)$ defined for the currently chosen permutation p within the specific call to the function `guess`.

The grader is not adaptive, meaning that the permutations p are fixed in advance and do not depend

on the implementation of the function `guess`.

Testing

You are provided with a solution template `checkpoint.cpp`, as well as a header file `checkpoint.h`, containing the definitions of the functions `min_value` and `guess`.

For convenience in testing, you are provided with a grader — the file `grader.cpp`. This file implements input reading from the standard input stream, calls the function `guess`, and outputs the returned value of the function `guess` to the standard output stream. In the testing system, these grader files may differ.

To compile your code `checkpoint.cpp` in C++, use the command

```
g++ -std=c++20 grader.cpp checkpoint.cpp -o grader
```

After executing this command, an executable file named `grader` or `grader.exe` will be created, depending on your operating system. You can run this file to input tests in the specified format.

If you encounter difficulties compiling via commands, for local testing, you can copy the implementation of the function `guess` into the file `grader.cpp` and run the file `grader.cpp`. However, before submitting your solution to the testing system, you need to leave only the implementation of the function `guess`, remembering to include the header file at the beginning of the code.

Input

The grader reads the test in the following format:

The first line contains the number t ($1 \leq t \leq 30\,000$) — the number of input data sets.

The first line of each input data set contains an **odd** number n ($1 \leq n \leq 30\,000$) — the length of the permutation.

The second line contains n distinct numbers p_1, \dots, p_n ($1 \leq p_i \leq n$) — the chosen permutation.

Output

The grader outputs the results of the function `guess` — the guessed permutation for each input data set.

In the file `grader.cpp`, there is a variable `verbose`, which is initially set to 0. By increasing its value, the grader will provide more detailed information about your solution and its queries.

Example

standard input	standard output
2	queries count => 3
3	queries count => 10
1 2 3	
5	
1 4 2 3 5	

Scoring

Tests for this problem consist of four groups.

Points for each of the first three groups are awarded only if all tests in the group are passed and all tests in some of the previous groups. The score for the last group equals the minimum of the scores obtained for each test in the fourth group.

For each test, let N be the sum of n across all input data sets, and Q be the limit on the total number of calls to the function `min_value` across all input data sets.

Group	Points	Constraints		Required	Comment
		N	Q		
0	0	–	$Q = 4950$	–	Samples
1	10	$N \leq 100$	$Q = 4950$		$p_{2i-1} = \frac{n-1}{2} + i$ for $1 \leq i \leq \frac{n+1}{2}$
2	20	$N \leq 100$	$Q = 4950$	0, 1	
3	20	$N \leq 1\,000$	$Q = 100\,000$	0, 1, 2	
4	60	$N \leq 30\,000$	$Q = 1\,000\,000$	–	

The last group is scored according to the formula. If you made x queries in the test, your score will be:

$$\text{score} = \min\left(60, \left\lfloor 95 \cdot \left(1 - \frac{x}{10^6}\right) \right\rfloor\right)$$

Your score for the last group equals the minimum of the scores obtained on the tests in the fourth group.