Problem D. Xoractive

Time limit: 1 second
Memory limit: 256 megabytes

Aidos has come up with a puzzle and challenged Temirulan to solve it. He picked a sequence a of n non-negative integers numbered from 1 to n: $a_1, a_2, ..., a_n$.
Temirulan can ask two types of questions:

- *ask* — Reveal the number at position i of the given sequence.
- *get_pairwise_xor* — For the given sequence of distinct integer numbers: $i_1, i_2, ..., i_k$, get a set of pairwise values of *xor* for the elements of the sequence a at indexes $i_1, i_2, ..., i_k$, $\{a_{i_x} \oplus a_{i_y} \mid 1 \le x, y \le k\}$.

For example, let's assume that Aidos has picked the sequence $[1, 5, 6, 3]$. Then for the question $ask(2)$, Aidos will answer with the number 5 and for the question $get\_pairwise\_xor(\{3, 4\})$, Aidos will answer with the sequence $[0, 0, 5, 5]$, because

- $a_3 \oplus a_4 = 6 \oplus 3 = 5$
- $a_4 \oplus a_3 = 3 \oplus 6 = 5$
- $a_3 \oplus a_3 = 6 \oplus 6 = 0$
- $a_4 \oplus a_4 = 3 \oplus 3 = 0$.

Temirulan failed to cope with the puzzle and your task is to help him. Find the hidden sequence using the questions described above.

Interaction Protocol

YOUR SUBMISSIONS MUST NOT READ FROM THE STANDARD INPUT, PRINT TO THE STARDARD OUTPUT OR INTERACT WITH ANY OTHER FILE.
Your task is to implement the following function: `int[] guess(int n)`

- n: the length of the hidden sequence.
- The function is called exactly one time for each test.
- The function has to return the hidden sequence in the same order.

Your function can call the following functions:

1. `int ask(int i)`
   - i: index of the number in sequence, $1 \le i \le n$.
   - The function returns the i-th number of the hidden sequence.
2. `int[] get_pairwise_xor(int[] pos)`
   - *pos*: **non empty** list of indexes of the sequence.
   - All of the elements in *pos* must be **distinct** integers.
   - Let k be the number of elements in *pos*. Then $1 \le pos_i \le n$ for each $1 \le i \le k$.
   - The function returns sorted list of k^2 elements: a set of pairwise values *xor*, $\{a_{pos_x} \oplus a_{pos_y} \mid 1 \le x, y \le k\}$.

You can call both functions no more than 200 times in total for each test. If any of the above conditions are violated, your program will get **Wrong Answer** verdict. Otherwise, your program will get **Accepted** verdict and your score is calculated based on the total number of calls of the functions *ask* and *get_pairwise_xor* (Refer to the section "Scoring").

Scoring

- $2 \leq n \leq 100$
- $0 \leq a_i \leq 10^9$ for each $1 \leq i \leq n$.

In this task, the grader is NOT adaptive. It means that the sequence a is fixed at the beginning of the running of the grader and does not depend on calls from your program.

1. (6 points) $n \leq 4$
2. (94 points) No additionals constraints. For this subtask, your score is calculated in the following manner. Let q be the total number of calls of the functions *ask* and *get_pairwise_xor*.
   - If $q \leq 15$, your score is 94.
   - If $15 < q \leq 40$, your score is $84 - 2(q - 16)$.
   - If $40 < q \leq 50$, your score is 35.
   - Otherwise, your score is 0.

Note that your score for each subtask is the minimum score among all the results on tests of the corresponding subtask.

Note

The *xor* operation is the bitwise exclusive OR.
Let the hidden sequence a be $[1, 5, 6, 3]$. Grader calls the function. Example of the interaction is below.

Call	Result
$ask(2)$	5
$get\_pairwise\_xor(\{1, 2, 3\})$	$\{0, 0, 0, 3, 3, 4, 4, 7, 7\}$
$ask(3)$	6
$get\_pairwise\_xor(\{4, 2\})$	$\{0, 0, 6, 6\}$
$get\_pairwise\_xor(\{2\})$	$\{0\}$

The sample grader reads the input in the following format:

- Line 1: n
- Line 2: $a_1, a_2, ..., a_n$

YOU CAN DOWNLOAD `xoractive.zip` in the system that contains examples for languages Java, C++11, FPC, Python 2.
All the examples of calling the functions can be found above. For `Python 2`, you have to implement the function `def guess(n, interactor)`, where *interactor* is an instance of the class being tested. Functions *ask* and *get_pairwise_xor* are the methods of this class.
`xoractive.zip` contains examples of solutions for each language.
For the solutions in `Java` language, file and class name have to be named as `Xoractive.java` and `Xoractive` respectively.
For the solutions in `Pascal` language, file has to be named as `xoractive.pas`.

Problem E. Bigger segments

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	256 megabytes

Our small boy Askhat noticed an interesting phenomenon — trying to cover an array with "jumps" of bigger and bigger sums may not be as simple as it seems. Of course, now you need to find a way to do it.
You are given a sequence of positive integer numbers of length N.
Divide the given sequence into the maximal number of segments so that:

1. Every element of the sequence belongs to exactly one segment.
2. Sum of the numbers in every segment, except for the first one, is not less than in the previous.

Input

The first line of the input contains the integer N ($1 \le N \le 5 \cdot 10^5$).
The next line contains N positive integers $a_i (1 \le a_i \le 10^9)$, separated by spaces.

Output

Output a single integer — the maximal number of segments the given sequence can be divided into.

Scoring

This task contains five subtasks, with additional constraints:

1. $1 \le N \le 20$, $a_i \le 10^6$. Scored 13 points.
2. $1 \le N \le 500$. Scored 14 points.
3. $1 \le N \le 3000$. Scored 10 points.
4. $1 \le N \le 10^5$. Scored 36 points.
5. Original constraints. Scored 27 points.

Examples

standard input	standard output
4 2 3 1 7	3
5 6 2 3 9 13	3
3 3 1 2	2

Problem F. "The Lyuboyn" code

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The "Lyuboyn" team — Small boy Askhat, middle boy Sanzhar, and big boy Nurbakyt — have decided to broaden their knowledge and study a bit of electronics. They came up with a weird electromechanical switch that modifies the analogue signal it receives in a special way. Happy with their invention, they proudly named it "The Lyuboyn switch".

To be precise, a signal may be represented as a sequence of N bits, and "The Lyuboyn switch" is such that the output signal it produces differs from the input in exactly K bits and no two input signals produce the same output signal. To make their invention even more gorgeous, the boys added the final feature: if the binary parameter T is set to 1, the linked sequence of outputs will be looped, i.e. if you start with a signal, replace it with its output from the switch and repeat the procedure enough times, at some point you will come back to the signal you originally started with. This, however, will not hold true if the parameter T is set to 0.

In this task, you are required to repeat the team's achivement and generate "The Lyuboyn code" — the mapping of output signals to given input signals the switch produces. To make things easier, you only need to output the linked sequence of outputs as described above, starting with a signal S.

Formally, you need to find a sequence f of length 2^N consisting of binary numbers of length N (including leading zeroes), such that:

1. $f_0 = S$
2. For every i and j $(i \neq j)$, $f_i \neq f_j$
3. For any i $(0 \leq i < 2^N - 1)$, f_i differs from f_{i+1} in exactly K digits in binary representation. Also, if the parameter T equals to 1, then the sequence must be looped, i.e. f_{2^N-1} should also differ from f_0 in exactly K digits in binary representation.

Input

The first line of the input contains three integer numbers N, K, and T $(2 \leq N \leq 18,\ 1 \leq K < N,\ 0 \leq T \leq 1)$.

The second line contains the binary representation of the starting number S.

Output

If no such sequence exists, print a single line containing -1.

Otherwise, the first line of the output should contain the number of values in the linked sequence — 2^N. The lines numbered from 2 to $2^N + 2$ should contain a single binary number each — the value of f_{i-2}.

If there are multiple valid solutions, you may output any of them.

Scoring

This task contains eight sub-tasks:

1. Sample test. Scored 0 points.
2. $N = 4,\ K = 3,\ T = 1,\ S = 0$. Scored 5 points.
3. $2 \le N \le 18,\ K$ is even, $T \le 1,\ S < 2^N$. Scored 3 points.
4. $2 \le N \le 18,\ K = 1,\ T = 1,\ S = 0$. Scored 11 points.
5. $2 \le N \le 18,\ K = 3,\ T = 0,\ S = 0$. Scored 15 points.
6. $2 \le N \le 18,\ K \cdot 2 < N,\ T = 0,\ S < 2^N$. Scored 18 points.
7. $2 \le N \le 18,\ K < N,\ T = 0,\ S < 2^N$. Scored 31 points.
8. $2 \le N \le 18,\ K < N,\ T = 1,\ S < 2^N$. Original constraints. Scored 17 points.

Example

standard input	standard output
2 1 1 10	4 10 11 01 00