Problem BinSearch

Input file stdin
Output file stdout

```
bool binary_search(int n, int p[], int target){
    int left = 1, right = n;
    while(left < right){
        int mid = (left + right) / 2;
        if(p[mid] == target)
            return true;
        else if(p[mid] < target)
            left = mid + 1;
        else
            right = mid - 1;
    }
    if(p[left] == target) return true;
    else return false;
}
```

It is well known that if `p` happens to be sorted, then this code returns `true` if and only if `target` appears within `p`. On the other hand, this may not be the case if `p` is not sorted.

You are given a positive integer n and a sequence $b_1, \ldots, b_n \in \{\texttt{true}, \texttt{false}\}$. It is guaranteed that $n = 2^k - 1$ for some positive integer k. You must generate a permutation p of $\{1, \ldots, n\}$ that follows certain conditions. Let $S(p)$ be the number of indices $i \in \{1, \ldots, n\}$ for which `binary_search(n, p, i)` does **not** return b_i. You must set p so that $S(p)$ is small (as detailed in the "Restrictions" section).

(Note: a permutation of $\{1, \ldots, n\}$ is a sequence of n integers that contains each integer from 1 to n *exactly* once.)

Input data

The input contains multiple test cases. The first line of input contains T, the number of test cases. The test cases follow.

The first line of a test case contains the integer n. The second line of a test case contains a string of length n containing only characters '0' and '1'. These characters are not separated by spaces. If the i^{th} character is '1', then $b_i = \texttt{true}$, and if it is '0', then $b_i = \texttt{false}$.

Output data

The output data consists of the answers for each of the T test cases. The answer for a particular test case consists of the permutation p generated for that test case.

Restrictions

- Let $\sum n$ be the sum of all values of n in a single input.
- $1 \leq \sum n \leq 100\,000$.
- $1 \leq T \leq 7\,000$.
- $n = 2^k - 1$ for some $k \in \mathbb{N}$, $k > 0$.
- If $S(p) \leq 1$ for all test cases within a subtask, then you are given 100% of the points for that subtask.
- Otherwise, if $0 \leq S(p) \leq \lceil \log_2 n \rceil$ (i.e. $1 \leq 2^{S(p)} \leq n + 1$) for all test cases within a subtask, then you are given 50% of the points for that subtask.

#	Points	Restrictions
1	3	$b_i =$ `true`.
2	4	$b_i =$ `false`.
3	16	$1 \leq n \leq 7$.
4	25	$1 \leq n \leq 15$.
5	22	$n = 2^{16} - 1$ and each b_i is selected uniformly and independently at random from $\{$`true`, `false`$\}$.
6	30	No additional constraints.

Examples

Input file	Output file

4
3
111
7
1111111
3
000
7
000000000
	1 2 3
	1 2 3 4 5 6 7
	3 2 1
	7 6 5 4 3 2 1

2 3 010 7 0010110	3 2 1 7 3 1 5 2 4 6

Explanations

Example 1. In the first two test cases of the first example, we have $S(p) = 0$.

In the third test case, we have $S(p) = 1$. This is because `binary_search(n, p, 2)` returns `true`, although $b_2 =$ `false`.

In the forth test case, we have $S(p) = 1$. This is because `binary_search(n, p, 4)` returns `true`, although $b_4 =$ `false`.

Example 2. We have $S(p) = 0$ for both test cases.