# Qingshan Loves Strings 2

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

Qingshan has a string $s$ which only contains `0` and `1`.

A string $a$ of length $k$ is good if and only if

- $a_i \neq a_{k-i+1}$ for all $i = 1, 2, \ldots, k$.

For Div. 2 contestants, note that this condition is different from the condition in problem B.

For example, `10`, `1010`, `111000` are good, while `11`, `101`, `001`, `001100` are not good.

Qingshan wants to make $s$ good. To do this, she can do the following operation **at most** 300 **times (possibly, zero):**

- **insert** `01` **to any position of** $s$ **(getting a new** $s$**).**

**Please tell Qingshan if it is possible to make $s$ good. If it is possible, print a sequence of operations that makes $s$ good.**

## Input

**The input consists of multiple test cases. The first line contains a single integer $t$ ($1 \leq t \leq 100$) — the number of test cases. The description of the test cases follows.**

**The first line of each test case contains a single integer $n$ ($1 \leq n \leq 100$) — the length of string $s$, respectively.**

**The second line of each test case contains a string $s$ with length $n$.**

**It is guaranteed that $s$ only consists of `0` and `1`.**

## Output

**For each test case, if it impossible to make $s$ good, output $-1$.**

**Otherwise, output $p$ ($0 \leq p \leq 300$) — the number of operations, in the first line.**

**Then, output $p$ integers in the second line. The $i$-th integer should be an index $x_i$ ($0 \leq x_i \leq n+2i-2$) — the position where you want to insert `01` in the current $s$. If $x_i = 0$, you insert `01` at the beginning of $s$. Otherwise, you insert `01` immediately after the $x_i$-th character of $s$.**

**We can show that under the constraints in this problem, if an answer exists, there is always an answer that requires at most 300 operations.**

## Example

| standard input | standard output |
|---|---|
| 6<br>2<br>01<br>3<br>000<br>4<br>1111<br>6<br>001110<br>10<br>0111001100<br>3<br>001 | 0<br><br>-1<br>-1<br>2<br>6 7<br>1<br>10<br>-1 |

## Note

**In the first test case, you can do zero operations and get $s = \texttt{01}$, which is good.**

**Another valid solution is to do one operation: (the inserted `01` is underlined)**

1. `0`<u>`01`</u>`1`

**and get $s = \texttt{0011}$, which is good.**

**In the second and the third test case, it is impossible to make $s$ good.**

**In the fourth test case, you can do two operations:**

1. `00111`<u>`01`</u>
2. `00111000`<u>`01`</u>`1`

**and get $s = \texttt{0011100011}$, which is good.**