Kevin and Binary String (Hard Version)

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

This is the hard version of the problem. The difference between the versions is that in this version, string t consists of '0', '1' and '?'. You can hack only if you solved all versions of this problem.

Kevin has a binary string s of length n. Kevin can perform the following operation:

- Choose two adjacent blocks of s and swap them.

A block is a maximal substring* of identical characters. Formally, denote $s[l, r]$ as the substring $s_l s_{l+1} \ldots s_r$. A block is $s[l, r]$ satisfying:

- $l = 1$ or $s_l \neq s_{l-1}$.
- $s_l = s_{l+1} = \ldots = s_r$.
- $r = n$ or $s_r \neq s_{r+1}$.

Adjacent blocks are two blocks $s[l_1, r_1]$ and $s[l_2, r_2]$ satisfying $r_1 + 1 = l_2$.

For example, if $s =$ `000`**`11`**`00`**`111`**, Kevin can choose the two blocks $s[4, 5]$ and $s[6, 7]$ and swap them, transforming s into `000`**`00`**`11`**`111`**.

Given a string t of length n consisting of '0', '1' and '?', Kevin wants to determine the minimum number of operations required to perform such that for any index i ($1 \leq i \leq n$), if $t_i \neq$ '?' then $s_i = t_i$. If it is impossible, output -1.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains a string s consisting of '0' and '1'.

The second line of each test case contains a string t consisting of **'0', '1' and '?'**.

It is guaranteed that the lengths of s and t are the same.

It is guaranteed that the sum of the length of s over all test cases will not exceed $4 \cdot 10^5$.

Output

For each test case, output one integer — the minimum number of operations required. If it is impossible, output -1.

*A string a is a substring of a string b if a can be obtained from b by the deletion of several (possibly, zero or all) characters from the beginning and several (possibly, zero or all) characters from the end.

Examples

standard input	standard output
6 0001100111 0000011111 010101 111000 0101 0110 0101 1010 011001 001110 0 1	1 3 1 -1 -1 -1
6 010101 ?0?0?? 0101 ?0?0 11100101 ???????? 11100101 ???11?1? 1000100011 ?11?000?0? 10101 ?1011	2 -1 0 2 2 -1

Note

In the first test case of the first example, the possible way is shown in the statement.

In the second test case of the first example, one possible way could be:

- Swap blocks $[2, 2], [3, 3]$, s will become `001101`.
- Swap blocks $[3, 4], [5, 5]$, s will become `000111`.
- Swap blocks $[1, 3], [4, 6]$, s will become `111000`.

In the first test case of the second example, one possible way could be:

- Swap blocks $[1, 1], [2, 2]$, s will become `100101`.
- Swap blocks $[4, 4], [5, 5]$, s will become `100011`.