Given a number s in their binary representation. Return the number of steps to reduce it to 1 under the following rules:

* If the current number is even, you have to divide it by 2.
* If the current number is odd, you have to add 1 to it.

It's guaranteed that you can always reach to one for all testcases.

**Example 1:**

**Input:** s = "1101"

**Output:** 6

**Explanation:** "1101" corressponds to number 13 in their decimal representation.

Step 1) 13 is odd, add 1 and obtain 14.

Step 2) 14 is even, divide by 2 and obtain 7.

Step 3) 7 is odd, add 1 and obtain 8.

Step 4) 8 is even, divide by 2 and obtain 4.

Step 5) 4 is even, divide by 2 and obtain 2.

Step 6) 2 is even, divide by 2 and obtain 1.

**Example 2:**

**Input:** s = "10"

**Output:** 1

**Explanation:** "10" corressponds to number 2 in their decimal representation.

Step 1) 2 is even, divide by 2 and obtain 1.

**Example 3:**

**Input:** s = "1"

**Output:** 0

**Constraints:**

* 1 <= s.length <= 500
* s consists of characters '0' or '1'
* s[0] == '1'