**Flip-Flopping**

Mark enjoys seeing patterns in his everyday life. Today, he’s decided to look at the patterns in binary strings but has sadly not been able to find any patterns! Do you think you can find the ***minimum*** number of the ***second operations*** required to make the binary string alternate between 1’s and 0’s? The following operations are allowed:

* **Operation** #1: Remove the character at the start of the string and append it to the end of the string.
* **Operation** #2: Pick any character in the string and flip its value. (if it’s a 1, it becomes a 0 and vice versa).

**Input:** The first line of input contains **T**, the number of test cases. The next **T** lines contain a binary string.

**Output:** You will first output “CASE #(case number): “ followed by the minimum number times you have to do ***operation #2*** in order to flip-flop the binary string.

**Example Input:**

3

111000

010

1110

**Example Output:**

CASE #1: 2

CASE #2: 0

CASE #3: 1

**Explanation:** Test case #1: use the first operation two times to make the string = “100011”. Then, use the second operation on the third and sixth elements to make the string = “101010”.

Test case #2: It already alternates so there is nothing to be done.

Test case #3: Use the second operation to make the 2nd element into a 0 to make the string = “1010”.