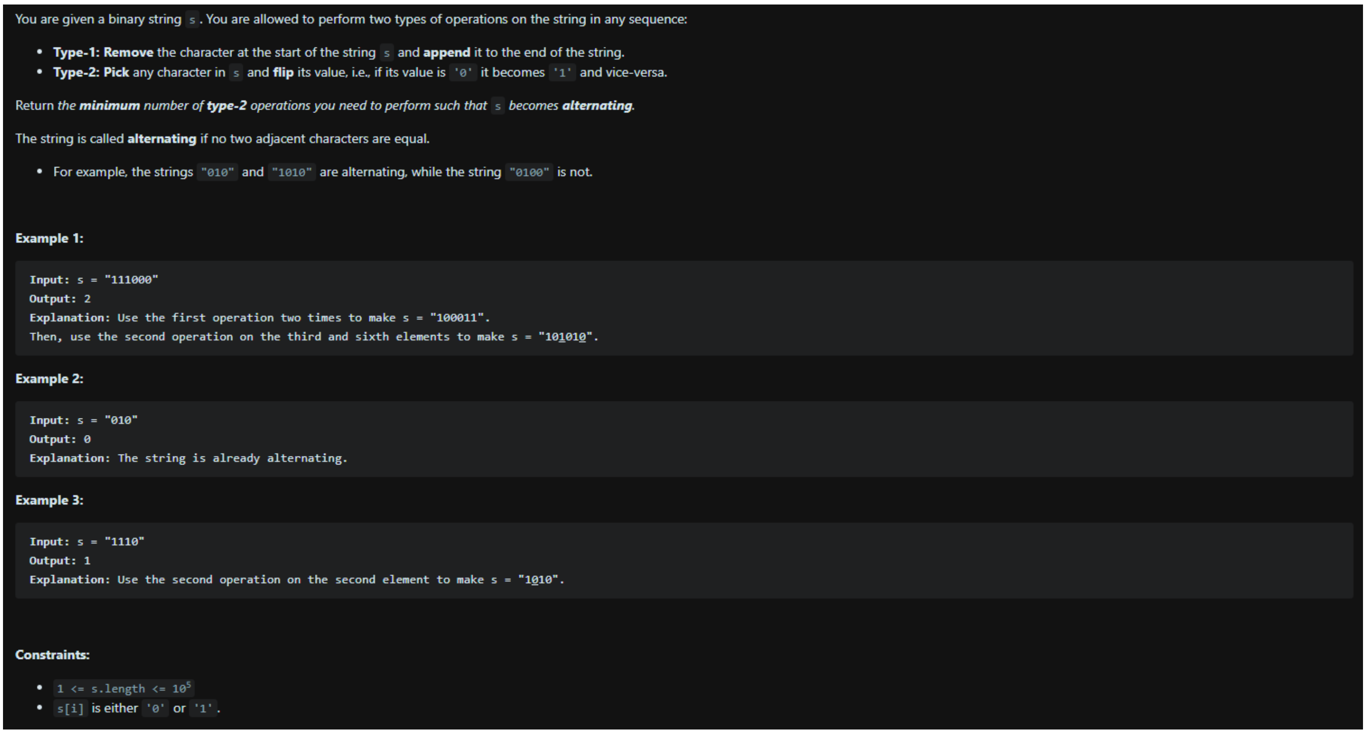

You are given a binary string `s`. You are allowed to perform two types of operations on the string in any sequence:

- **Type-1: Remove** the character at the start of the string `s` and **append** it to the end of the string.
- **Type-2: Pick** any character in `s` and **flip** its value, i.e., if its value is `'0'` it becomes `'1'` and vice-versa.

Return *the **minimum** number of **type-2** operations you need to perform such that* `s` *becomes **alternating**.*

The string is called **alternating** if no two adjacent characters are equal.

- For example, the strings `"010"` and `"1010"` are alternating, while the string `"0100"` is not.

Example 1:

```
Input: s = "111000"
Output: 2
Explanation: Use the first operation two times to make s = "100011".
Then, use the second operation on the third and sixth elements to make s = "101010".
```

Example 2:

```
Input: s = "010"
Output: 0
Explanation: The string is already alternating.
```

Example 3:

```
Input: s = "1110"
Output: 1
Explanation: Use the second operation on the second element to make s = "1010".
```

Constraints:

- $1 <= \texttt{s.length} <= 10^5$
- `s[i]` is either `'0'` or `'1'`.