# 696. Count Binary Substrings

445 83

## Approach #1: Group By Character [Accepted]

**Intuition**

We can convert the string `s` into an array `groups` that represents the length of same-character contiguous blocks within the string. For example, if `s = "110001111000000"`, then `groups = [2, 3, 4, 6]`.

For every binary string of the form `'0' * k + '1' * k` or `'1' * k + '0' * k`, the middle of this string must occur between two groups.

Let's try to count the number of valid binary strings between `groups[i]` and `groups[i+1]`. If we have `groups[i] = 2, groups[i+1] = 3`, then it represents either `"00111"` or `"11000"`. We clearly can make `min(groups[i], groups[i+1])` valid binary strings within this string. Because the binary digits to the left or right of this string must change at the boundary, our answer can never be larger.

**Algorithm**

Let's create `groups` as defined above. The first element of `s` belongs in it's own group. From then on, each element either doesn't match the previous element, so that it starts a new group of size 1; or it does match, so that the size of the most recent group increases by 1.

Afterwards, we will take the sum of `min(groups[i-1], groups[i])`.

**Python**

```
class Solution(object):
    def countBinarySubstrings(self, s):
        groups = [1]
        for i in xrange(1, len(s)):
            if s[i-1] != s[i]:
                groups.append(1)
            else:
                groups[-1] += 1

        ans = 0
        for i in xrange(1, len(groups)):
            ans += min(groups[i-1], groups[i])
        return ans
```

*Alternate Implentation*

```
class Solution(object):
    def countBinarySubstrings(self, s):
        groups = [len(list(v)) for _, v in itertools.groupby(s)]
        return sum(min(a, b) for a, b in zip(groups, groups[1:]))
```

**Java**

```
class Solution {
    public int countBinarySubstrings(String s) {
        int[] groups = new int[s.length()];
        int t = 0;
        groups[0] = 1;
        for (int i = 1; i < s.length(); i++) {
            if (s.charAt(i-1) != s.charAt(i)) {
                groups[++t] = 1;
            } else {
                groups[t]++;
            }
        }

        int ans = 0;
        for (int i = 1; i <= t; i++) {
            ans += Math.min(groups[i-1], groups[i]);
        }
        return ans;
    }
}
```

**Complexity Analysis**

- Time Complexity: $O(N)$, where $N$ is the length of `s`. Every loop is through $O(N)$ items with $O(1)$ work inside the for-block.
- Space Complexity: $O(N)$, the space used by `groups`.

## Approach #2: Linear Scan [Accepted]

**Intuition and Algorithm**

We can amend our *Approach #1* to calculate the answer on the fly. Instead of storing `groups`, we will remember only `prev = groups[-2]` and `cur = groups[-1]`. Then, the answer is the sum of `min(prev, cur)` over each different final `(prev, cur)` we see.

**Python**

```
class Solution(object):
    def countBinarySubstrings(self, s):
        ans, prev, cur = 0, 0, 1
        for i in xrange(1, len(s)):
            if s[i-1] != s[i]:
                ans += min(prev, cur)
                prev, cur = cur, 1
            else:
                cur += 1

        return ans + min(prev, cur)
```

**Java**

```
class Solution {
    public int countBinarySubstrings(String s) {
        int ans = 0, prev = 0, cur = 1;
        for (int i = 1; i < s.length(); i++) {
            if (s.charAt(i-1) != s.charAt(i)) {
                ans += Math.min(prev, cur);
                prev = cur;
                cur = 1;
            } else {
                cur++;
            }
        }
        return ans + Math.min(prev, cur);
    }
}
```

**Complexity Analysis**

- Time Complexity: $O(N)$, where $N$ is the length of `s`. Every loop is through $O(N)$ items with $O(1)$ work inside the for-block.
- Space Complexity: $O(1)$, the space used by `prev`, `cur`, and `ans`.

Analysis written by: @awice (https://leetcode.com/awice).

## Comments: 9

Sort By

Type comment here... (Markdown is supported)

Preview Post

makikvues (/makikvues) ★ 3 June 30, 2018 8:41 AM

(/makikvues)

```
public int CountBinarySubstrings(string s)
        {
            int count = 0;
            int consecutiveCount = 1;
```

Read More

2 Share Reply

SHOW 1 REPLY

dbalagula (/dbalagula) ★ 1 March 23, 2018 11:53 PM

(/dbalagula)

var i = 0;
var sum = 0;
while (i < s.length){
var type = (s[i] == 0 ? 0 : 1);
if (type == 0){

Read More

1 Share Reply

sxj1232456 (/sxj1232456) ★ 11 March 15, 2018 3:33 AM

(/sxj1232456)

can't understand, it's too hard for a rookie

1 Share Reply

SHOW 2 REPLIES