

# Problem 1888: Minimum Number of Flips to Make the Binary String Alternating

## Problem Information

Difficulty: Medium

Acceptance Rate: 0.00%

Paid Only: No

## Problem Description

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 = "10

1

01

0

".

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 = "1

0

10".

Constraints:

$1 \leq s.length \leq 10$

5

$s[i]$

is either

'0'

or

'1'

## Code Snippets

### C++:

```
class Solution {  
public:  
    int minFlips(string s) {  
  
    }  
};
```

### Java:

```
class Solution {  
public int minFlips(String s) {  
  
}  
}
```

### Python3:

```
class Solution:  
    def minFlips(self, s: str) -> int:
```

### Python:

```
class Solution(object):  
    def minFlips(self, s):  
        """  
        :type s: str
```

```
:rtype: int  
"""
```

### JavaScript:

```
/**  
 * @param {string} s  
 * @return {number}  
 */  
var minFlips = function(s) {  
  
};
```

### TypeScript:

```
function minFlips(s: string): number {  
  
};
```

### C#:

```
public class Solution {  
    public int MinFlips(string s) {  
  
    }  
}
```

### C:

```
int minFlips(char* s) {  
  
}
```

### Go:

```
func minFlips(s string) int {  
  
}
```

### Kotlin:

```
class Solution {  
    fun minFlips(s: String): Int {  
        //  
        //  
        //  
    }  
}
```

### Swift:

```
class Solution {  
    func minFlips(_ s: String) -> Int {  
        //  
        //  
        //  
    }  
}
```

### Rust:

```
impl Solution {  
    pub fn min_flips(s: String) -> i32 {  
        //  
        //  
        //  
    }  
}
```

### Ruby:

```
# @param {String} s  
# @return {Integer}  
def min_flips(s)  
  
end
```

### PHP:

```
class Solution {  
  
    /**  
     * @param String $s  
     * @return Integer  
     */  
    function minFlips($s) {  
  
    }  
}
```

**Dart:**

```
class Solution {  
    int minFlips(String s) {  
  
    }  
}
```

**Scala:**

```
object Solution {  
    def minFlips(s: String): Int = {  
  
    }  
}
```

**Elixir:**

```
defmodule Solution do  
    @spec min_flips(s :: String.t) :: integer  
    def min_flips(s) do  
  
    end  
end
```

**Erlang:**

```
-spec min_flips(S :: unicode:unicode_binary()) -> integer().  
min_flips(S) ->  
.
```

**Racket:**

```
(define/contract (min-flips s)  
  (-> string? exact-integer?)  
)
```

## Solutions

**C++ Solution:**

```

/*
 * Problem: Minimum Number of Flips to Make the Binary String Alternating
 * Difficulty: Medium
 * Tags: array, string, dp
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

class Solution {
public:
    int minFlips(string s) {
}
};


```

### Java Solution:

```

/**
 * Problem: Minimum Number of Flips to Make the Binary String Alternating
 * Difficulty: Medium
 * Tags: array, string, dp
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

class Solution {
public int minFlips(String s) {
}

}


```

### Python3 Solution:

```

"""

Problem: Minimum Number of Flips to Make the Binary String Alternating
Difficulty: Medium
Tags: array, string, dp

```

```

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) or O(n * m) for DP table
"""

class Solution:

def minFlips(self, s: str) -> int:
# TODO: Implement optimized solution
pass

```

### Python Solution:

```

class Solution(object):
def minFlips(self, s):
"""
:type s: str
:rtype: int
"""

```

### JavaScript Solution:

```

/**
 * Problem: Minimum Number of Flips to Make the Binary String Alternating
 * Difficulty: Medium
 * Tags: array, string, dp
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

/**
 * @param {string} s
 * @return {number}
 */
var minFlips = function(s) {

};


```

### TypeScript Solution:

```

/**
 * Problem: Minimum Number of Flips to Make the Binary String Alternating
 * Difficulty: Medium
 * Tags: array, string, dp
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

function minFlips(s: string): number {
}

```

### C# Solution:

```

/*
 * Problem: Minimum Number of Flips to Make the Binary String Alternating
 * Difficulty: Medium
 * Tags: array, string, dp
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

public class Solution {
    public int MinFlips(string s) {
}
}

```

### C Solution:

```

/*
 * Problem: Minimum Number of Flips to Make the Binary String Alternating
 * Difficulty: Medium
 * Tags: array, string, dp
 *
 * Approach: Use two pointers or sliding window technique
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(n) or O(n * m) for DP table
 */

```

```
*/  
  
int minFlips(char* s) {  
  
}
```

### Go Solution:

```
// Problem: Minimum Number of Flips to Make the Binary String Alternating  
// Difficulty: Medium  
// Tags: array, string, dp  
//  
// Approach: Use two pointers or sliding window technique  
// Time Complexity: O(n) or O(n log n)  
// Space Complexity: O(n) or O(n * m) for DP table  
  
func minFlips(s string) int {  
  
}
```

### Kotlin Solution:

```
class Solution {  
    fun minFlips(s: String): Int {  
  
    }  
}
```

### Swift Solution:

```
class Solution {  
    func minFlips(_ s: String) -> Int {  
  
    }  
}
```

### Rust Solution:

```
// Problem: Minimum Number of Flips to Make the Binary String Alternating  
// Difficulty: Medium  
// Tags: array, string, dp
```

```
//  
// Approach: Use two pointers or sliding window technique  
// Time Complexity: O(n) or O(n log n)  
// Space Complexity: O(n) or O(n * m) for DP table  
  
impl Solution {  
    pub fn min_flips(s: String) -> i32 {  
  
    }  
}
```

### Ruby Solution:

```
# @param {String} s  
# @return {Integer}  
def min_flips(s)  
  
end
```

### PHP Solution:

```
class Solution {  
  
    /**  
     * @param String $s  
     * @return Integer  
     */  
    function minFlips($s) {  
  
    }  
}
```

### Dart Solution:

```
class Solution {  
    int minFlips(String s) {  
  
    }  
}
```

### Scala Solution:

```
object Solution {  
    def minFlips(s: String): Int = {  
        }  
        }  
    }
```

### Elixir Solution:

```
defmodule Solution do  
  @spec min_flips(s :: String.t) :: integer  
  def min_flips(s) do  
  
  end  
  end
```

### Erlang Solution:

```
-spec min_flips(S :: unicode:unicode_binary()) -> integer().  
min_flips(S) ->  
.
```

### Racket Solution:

```
(define/contract (min-flips s)  
  (-> string? exact-integer?)  
  )
```