

# Problem 1781: Sum of Beauty of All Substrings

## Problem Information

Difficulty: **Medium**

Acceptance Rate: 0.00%

Paid Only: No

## Problem Description

The

beauty

of a string is the difference in frequencies between the most frequent and least frequent characters.

For example, the beauty of

"abaacc"

is

$3 - 1 = 2$

.

Given a string

s

, return

the sum of

beauty

of all of its substrings.

Example 1:

Input:

`s = "aabcb"`

Output:

5

Explanation:

The substrings with non-zero beauty are ["aab","aabc","aabcb","abcb","bcb"], each with beauty equal to 1.

Example 2:

Input:

`s = "aabcbaa"`

Output:

17

Constraints:

$1 \leq s.length \leq$

500

`s`

consists of only lowercase English letters.

## Code Snippets

### C++:

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

### Java:

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

### Python3:

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

### Python:

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

### JavaScript:

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

### TypeScript:

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

### C#:

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

### C:

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

### Go:

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

### Kotlin:

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

### Swift:

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

### Rust:

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

### Ruby:

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

### PHP:

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

### Dart:

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

### Scala:

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

### Elixir:

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

  end

end
```

### Erlang:

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

### Racket:

```
(define/contract (beauty-sum s)
  (-> string? exact-integer?)
)
```

## Solutions

### C++ Solution:

```
/*
 * Problem: Sum of Beauty of All Substrings
 * Difficulty: Medium
 * Tags: string, tree, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(h) for recursion stack where h is height
 */

class Solution {
public:
    int beautySum(string s) {

    }

};
```

## Java Solution:

```
/**
 * Problem: Sum of Beauty of All Substrings
 * Difficulty: Medium
 * Tags: string, tree, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(h) for recursion stack where h is height
 */

class Solution {
public int beautySum(String s) {

}

}
```

## Python3 Solution:

```
"""
Problem: Sum of Beauty of All Substrings
Difficulty: Medium
Tags: string, tree, hash

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(h) for recursion stack where h is height
"""

class Solution:
def beautySum(self, s: str) -> int:
# TODO: Implement optimized solution
pass
```

## Python Solution:

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

```
"""
```

### JavaScript Solution:

```
/**
 * Problem: Sum of Beauty of All Substrings
 * Difficulty: Medium
 * Tags: string, tree, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(h) for recursion stack where h is height
 */

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

};
```

### TypeScript Solution:

```
/**
 * Problem: Sum of Beauty of All Substrings
 * Difficulty: Medium
 * Tags: string, tree, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(h) for recursion stack where h is height
 */

function beautySum(s: string): number {

};
```

### C# Solution:

```

/*
 * Problem: Sum of Beauty of All Substrings
 * Difficulty: Medium
 * Tags: string, tree, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(h) for recursion stack where h is height
 */

public class Solution {
    public int BeautySum(string s) {

    }
}

```

### C Solution:

```

/*
 * Problem: Sum of Beauty of All Substrings
 * Difficulty: Medium
 * Tags: string, tree, hash
 *
 * Approach: String manipulation with hash map or two pointers
 * Time Complexity: O(n) or O(n log n)
 * Space Complexity: O(h) for recursion stack where h is height
 */

int beautySum(char* s) {

}

```

### Go Solution:

```

// Problem: Sum of Beauty of All Substrings
// Difficulty: Medium
// Tags: string, tree, hash
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

```

```

func beautySum(s string) int {

}

```

### Kotlin Solution:

```

class Solution {
    fun beautySum(s: String): Int {

    }
}

```

### Swift Solution:

```

class Solution {
    func beautySum(_ s: String) -> Int {

    }
}

```

### Rust Solution:

```

// Problem: Sum of Beauty of All Substrings
// Difficulty: Medium
// Tags: string, tree, hash
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

impl Solution {
    pub fn beauty_sum(s: String) -> i32 {

    }
}

```

### Ruby Solution:

```

# @param {String} s
# @return {Integer}
def beauty_sum(s)

```

```
end
```

### PHP Solution:

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

### Dart Solution:

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

### Scala Solution:

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

### Elixir Solution:

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

### Erlang Solution:

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

### Racket Solution:

```
(define/contract (beauty-sum s)  
  (-> string? exact-integer?)  
)
```