Problem 2311: Longest Binary Subsequence Less Than or Equal to K

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

You are given a binary string

s

and a positive integer

k

.

Return

the length of the

longest

subsequence of

s

that makes up a

binary

number less than or equal to

k

.

Note:

The subsequence can contain

leading zeroes

.

The empty string is considered to be equal to

0

.

A

subsequence

is a string that can be derived from another string by deleting some or no characters without changing the order of the remaining characters.

Example 1:

Input:

s = "1001010", k = 5

Output:

5

Explanation:

The longest subsequence of s that makes up a binary number less than or equal to 5 is "00010", as this number is equal to 2 in decimal. Note that "00100" and "00101" are also

possible, which are equal to 4 and 5 in decimal, respectively. The length of this subsequence is 5, so 5 is returned.

Example 2:

Input:

s = "00101001", k = 1

Output:

6

Explanation:

"000001" is the longest subsequence of s that makes up a binary number less than or equal to 1, as this number is equal to 1 in decimal. The length of this subsequence is 6, so 6 is returned.

Constraints:

1 <= s.length <= 1000

s[i]

is either

'0'

or

'1'

.

1 <= k <= 10

Code Snippets

C++:

```
class Solution {
public:
int longestSubsequence(string s, int k) {

}
};
```

Java:

```
class Solution {
public int longestSubsequence(String s, int k) {

}
}
```

Python3:

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

Python:

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

JavaScript:

```
/**
* @param {string} s
* @param {number} k
* @return {number}
*/
var longestSubsequence = function(s, k) {
```

```
};
```

TypeScript:

```
function longestSubsequence(s: string, k: number): number {

};
```

C#:

```
public class Solution {
public int LongestSubsequence(string s, int k) {

}
}
```

C:

```
int longestSubsequence(char* s, int k) {

}
```

Go:

```
func longestSubsequence(s string, k int) int {

}
```

Kotlin:

```
class Solution {
fun longestSubsequence(s: String, k: Int): Int {

}
}
```

Swift:

```
class Solution {
func longestSubsequence(_ s: String, _ k: Int) -> Int {
```

```
}
}
```

Rust:

```
impl Solution {
pub fn longest_subsequence(s: String, k: i32) -> i32 {

}
}
```

Ruby:

```
# @param {String} s
# @param {Integer} k
# @return {Integer}
def longest_subsequence(s, k)

end
```

PHP:

```
class Solution {

/**
* @param String $s
* @param Integer $k
* @return Integer
*/
function longestSubsequence($s, $k) {

}
}
```

Dart:

```
class Solution {
int longestSubsequence(String s, int k) {

}
}
```

Scala:

```
object Solution {
def longestSubsequence(s: String, k: Int): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec longest_subsequence(s :: String.t, k :: integer) :: integer
def longest_subsequence(s, k) do

end
end
```

Erlang:

```
-spec longest_subsequence(S :: unicode:unicode_binary(), K :: integer()) ->
integer().
longest_subsequence(S, K) ->
.
```

Racket:

```
(define/contract (longest-subsequence s k)
(-> string? exact-integer? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Longest Binary Subsequence Less Than or Equal to K
* Difficulty: Medium
* Tags: string, dp, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
```

```
*/

class Solution {
public:
int longestSubsequence(string s, int k) {

}
};
```

Java Solution:

```
/**
* Problem: Longest Binary Subsequence Less Than or Equal to K
* Difficulty: Medium
* Tags: string, dp, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

class Solution {
public int longestSubsequence(String s, int k) {

}
}
```

Python3 Solution:

```
"""
Problem: Longest Binary Subsequence Less Than or Equal to K
Difficulty: Medium
Tags: string, dp, greedy

Approach: String manipulation with hash map or two pointers
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) or O(n * m) for DP table
"""

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

```
# TODO: Implement optimized solution
pass
```

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Longest Binary Subsequence Less Than or Equal to K
* Difficulty: Medium
* Tags: string, dp, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

/**
* @param {string} s
* @param {number} k
* @return {number}
*/
var longestSubsequence = function(s, k) {

};
```

TypeScript Solution:

```
/**
* Problem: Longest Binary Subsequence Less Than or Equal to K
* Difficulty: Medium
* Tags: string, dp, greedy
*
```

```
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

function longestSubsequence(s: string, k: number): number {

};
```

C# Solution:

```
/*
* Problem: Longest Binary Subsequence Less Than or Equal to K
* Difficulty: Medium
* Tags: string, dp, greedy
*
* Approach: String manipulation with hash map or two pointers
* 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 LongestSubsequence(string s, int k) {

}
}
```

C Solution:

```
/*
* Problem: Longest Binary Subsequence Less Than or Equal to K
* Difficulty: Medium
* Tags: string, dp, greedy
*
* Approach: String manipulation with hash map or two pointers
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

int longestSubsequence(char* s, int k) {
```

```
}
```

Go Solution:

```
// Problem: Longest Binary Subsequence Less Than or Equal to K
// Difficulty: Medium
// Tags: string, dp, greedy
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

func longestSubsequence(s string, k int) int {

}
```

Kotlin Solution:

```
class Solution {
fun longestSubsequence(s: String, k: Int): Int {

}
}
```

Swift Solution:

```
class Solution {
func longestSubsequence(_ s: String, _ k: Int) -> Int {

}
}
```

Rust Solution:

```
// Problem: Longest Binary Subsequence Less Than or Equal to K
// Difficulty: Medium
// Tags: string, dp, greedy
//
// Approach: String manipulation with hash map or two pointers
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table
```

```
impl Solution {
pub fn longest_subsequence(s: String, k: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {String} s
# @param {Integer} k
# @return {Integer}
def longest_subsequence(s, k)

end
```

PHP Solution:

```
class Solution {

/**
* @param String $s
* @param Integer $k
* @return Integer
*/
function longestSubsequence($s, $k) {

}
}
```

Dart Solution:

```
class Solution {
int longestSubsequence(String s, int k) {

}
}
```

Scala Solution:

```
object Solution {
def longestSubsequence(s: String, k: Int): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec longest_subsequence(s :: String.t, k :: integer) :: integer
def longest_subsequence(s, k) do

end
end
```

Erlang Solution:

```
-spec longest_subsequence(S :: unicode:unicode_binary(), K :: integer()) ->
integer().
longest_subsequence(S, K) ->
.
```

Racket Solution:

```
(define/contract (longest-subsequence s k)
(-> string? exact-integer? exact-integer?)
)
```