length_of_last_word

May 14, 2025

1 Length of Last Word

1.1 Problem Definition

Given a string **s** consisting of words and spaces, return the length of the last word in the string.

A word is a maximal substring consisting of non-space characters only.

1.1.1 Example Cases

Example 1:

Input: s = "Hello World"
Output: 5
Explanation: The last word is "World" with length 5.

Example 2:

Input: s = " fly me to the moon "
Output: 4
Explanation: The last word is "moon" with length 4.

Example 3:

Input: s = "luffy is still joyboy"
Output: 6
Explanation: The last word is "joyboy" with length 6.

1.1.2 Constraints

Constraints:

- 1 <= s.length <= 10^4
- s consists of only English letters and spaces ' '.
- There will be at least one word in s.

1.2 Example test cases

[7]:
```python
def test_cases():
    assert lengthOfLastWord("Hello World") == 5
    assert lengthOfLastWord(" fly me to the moon ") == 4
    assert lengthOfLastWord("luffy is still joyboy") == 6
```

```
    print("All test cases passed!")
```

1.3 Solutions

$O(n)$

One way to solve the problem is to reverse the string (linear time) and then start iterating over the text. The length of the first word is the solution. Word can be found by looking for empty characters (" "). If there is one and after it comes a string, then word starts. The word ends when we encounter another empty character.

[9]:
```
def lengthOfLastWord(s):
    length = 0
    encountered_string = False
    for letter in s[::-1]:
        if letter != " " and not encountered_string:
            length += 1
            encountered_string = True
        elif letter != " " and encountered_string:
            length += 1
        elif letter == " " and encountered_string:
            return length
```

Another solution without reversing the string.

[20]:
```
def lengthOfLastWord(s):
    length = 0
    encountered_string = False
    lengths = []
    for letter in s:
        if letter != " " and not encountered_string:
            length += 1
            encountered_string = True
        elif letter != " " and encountered_string:
            length += 1
        elif letter == " " and encountered_string:
            lengths.append(length)
            encountered_string = False
            length = 0

    if encountered_string:
        lengths.append(length)

    return lengths[-1]
```

[21]:
```
test_cases()
```

```
All test cases passed!
```

Also, this can be solved using built-in functions, which probably is not the target, but still it's the simplest one:

[22]:
```
def lengthOfLastWord(s):
    return len(s.strip().split()[-1])
```

[23]:
```
test_cases()
```

```
All test cases passed!
```

A very interesting solution taken from the solutions. Reference: https://leetcode.com/problems/length-of-last-word/solutions/5774504/video-2-solutions-bonus/

Stripping the end while there is an empty character and then looking for all non-empty characters until we reach the next empty character.

[24]:
```
def lengthOfLastWord(s):
    end = len(s) - 1

    while s[end] == " ":
        end -= 1

    start = end
    while start >= 0 and s[start] != " ":
        start -= 1

    return end - start
```

[25]:
```
test_cases()
```

```
All test cases passed!
```