

Name: Talha khan (2303-009-KHI-DEG)

Character with longest consecutive repetition

DESCRIPTION:

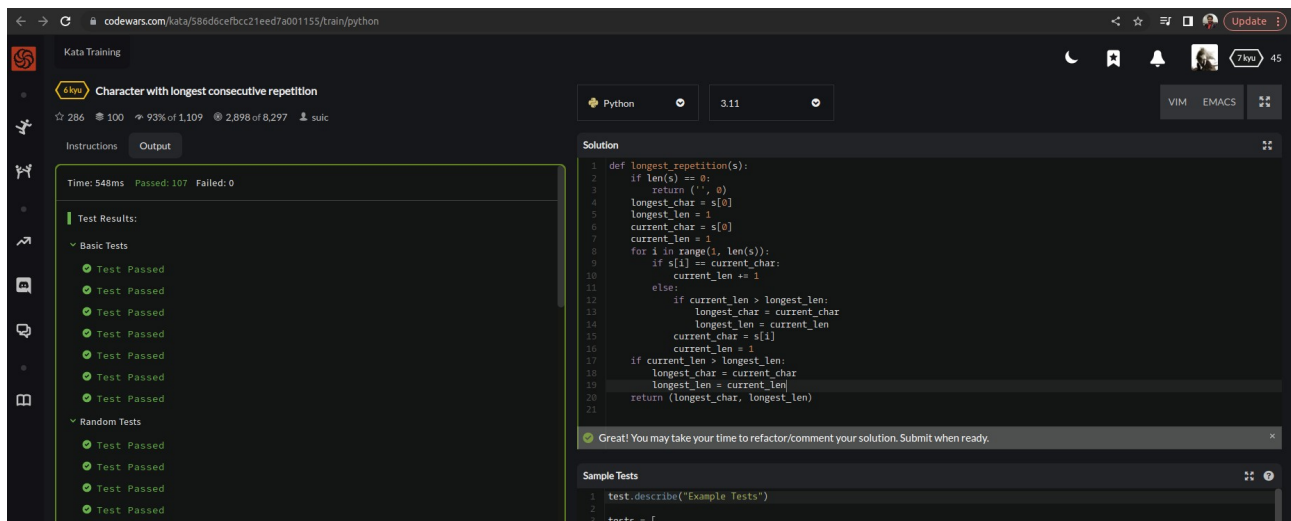
For a given string `S` find the character `C` (or `C`) with longest consecutive repetition and return: `(c, l)`

where `l` (or `L`) is the length of the repetition. If there are two or more characters with the same `l` return the first in order of appearance.

For empty string return:

`(' ', 0)`

SOLUTION:



The screenshot shows a web-based coding environment. On the left, the problem title 'Character with longest consecutive repetition' is displayed with a 6kyu difficulty rating. Below the title, test results are shown: 'Time: 548ms', 'Passed: 107', and 'Failed: 0'. A list of tests follows, with 'Basic Tests' and 'Random Tests' sections, all marked as 'Test Passed'. On the right, the 'Solution' tab is active, displaying a Python function `longest_repetition(s)`. The function logic is as follows: it returns `(' ', 0)` for an empty string. Otherwise, it initializes `longest_char` and `longest_len` to the first character and its length (1), and sets `current_char` and `current_len` to the same values. It then iterates through the string from index 1 to the end. For each character `s[i]`, if it matches `current_char`, `current_len` is incremented. If it doesn't match, `current_len` is reset to 1. After each iteration, the function checks if `current_len` is greater than `longest_len`. If so, it updates `longest_char` to `current_char` and `longest_len` to `current_len`. Finally, it returns the tuple `(longest_char, longest_len)`. A message at the bottom of the code editor says: 'Great! You may take your time to refactor/comment your solution. Submit when ready.'

```
1 def longest_repetition(s):
2     if len(s) == 0:
3         return (' ', 0)
4     longest_char = s[0]
5     longest_len = 1
6     current_char = s[0]
7     current_len = 1
8     for i in range(1, len(s)):
9         if s[i] == current_char:
10            current_len += 1
11        else:
12            if current_len > longest_len:
13                longest_char = current_char
14                longest_len = current_len
15            current_char = s[i]
16            current_len = 1
17        if current_len > longest_len:
18            longest_char = current_char
19            longest_len = current_len
20    return (longest_char, longest_len)
```

EXPLANATION:

This function takes a string `S` as input and returns a tuple containing the character with longest consecutive repetition and its length. If the string is empty, it returns `(' ', 0)`. We start by checking if the string is empty and returning the appropriate value. Otherwise, we initialize `longest_char` and `longest_len` to the first character in the string and its length, and `current_char` and `current_len` to the same values. We then iterate through the rest of the string, updating `current_len` when we encounter the same character as before and comparing it to `longest_len` when we encounter a new character. If `current_len` is greater than `longest_len`, we update `longest_char` and `longest_len` accordingly. Finally, we check if `current_len` is greater than `longest_len` one last time to handle the end of the string. We return the tuple `(longest_char, longest_len)` as the result.