Codility_

CodeCheck Report: trainingK6UUH7-HRR

Test Name:

Check out Codility training tasks

Summary Timeline

Tasks summary

Task	Time spent	Score
LongestPassword Python	6 min	100%

Total score

Tasks Details

1. LongestPassword

Easy

Given a string containing words, find the longest word that satisfies specific conditions.

Task Score	Correctness	Performance
100%	100%	Not assessed

Task description

You would like to set a password for a bank account. However, there are three restrictions on the format of the password:

- it has to contain only alphanumerical characters (`a-z`, `A-Z`, `0-9`);
- there should be an even number of letters;
- there should be an odd number of digits.

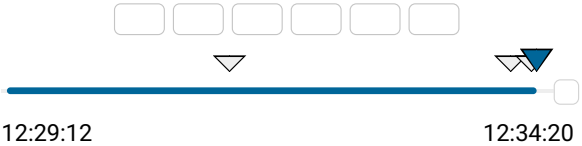
You are given a string S consisting of N characters. String S can be divided into *words* by splitting it at, and removing, the spaces. The goal is to choose the longest word that is a valid password. You can assume that if there are K spaces in string S then there are exactly K + 1 words.

For example, given "`test 5 a0A pass007 ?xy1`", there are five words and three of them are valid passwords: "`5`", "`a0A`" and "`pass007`". Thus the longest password is

Solution

Programming language used:	Python
Total time used:	6 minutes
Effective time used:	6 minutes
Notes:	*not defined yet*

Task timeline



"pass007" and its length is 7. Note that neither "test" nor "?xy1" is a valid password, because "?" is not an alphanumerical character and "test" contains an even number of digits (zero).

Write a function:

```
def solution(S)
```

that, given a non-empty string S consisting of N characters, returns the length of the longest word from the string that is a valid password. If there is no such word, your function should return −1.

For example, given S = "test 5 a0A pass007 ?xy1", your function should return 7, as explained above.

Assume that:

- N is an integer within the range [1..200];
- string S consists only of printable ASCII characters and spaces.

In your solution, focus on **correctness**. The performance of your solution will not be the focus of the assessment.

Copyright 2009–2023 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Code: 12:34:19 UTC, py, final, score: **100** show code in pop-up

```
1  # you can write to stdout for debugging pu
2  # print("this is a debug message")
3
4  def solution(S):
5      # Implement your solution here
6      def is_valid_password(word):
7          if not word.isalnum():
8              return False
9
10         letter_count = digit_count = 0
11         for char in word:
12             if char.isdigit():
13                 digit_count += 1
14             elif char.isalpha():
15                 letter_count += 1
16
17         # Check if there are an even numbe
18         return letter_count % 2 == 0 and d
19
20     words = S.split()
21     longest_valid_password = -1
22
23     for word in words:
24         if is_valid_password(word):
25             longest_valid_password = max(l
26
27     return longest_valid_password
```

Analysis summary

The solution obtained perfect score.

Analysis

expand all	Example tests	
▶	example example test	✔ OK
expand all	**Correctness tests**	
▶	simple short and simple tests	✔ OK
▶	one_character one character words	✔ OK
▶	one_word tests that contains one word only	✔ OK
▶	even_letters all words have even number of letters	✔ OK
▶	odd_digits all words have odd number of digits	✔ OK
▶	odd_length it's sufficient to test validity of characters and if length of word is odd	✔ OK
▶	all_alphanumerical all words contain only alphanumerical characters	✔ OK

▶	extra_characters valid passwords joined with some invalid characters	✔ OK
▶	large_random random tests	✔ OK
▶	maximum biggest possible tests with mixed types of words	✔ OK