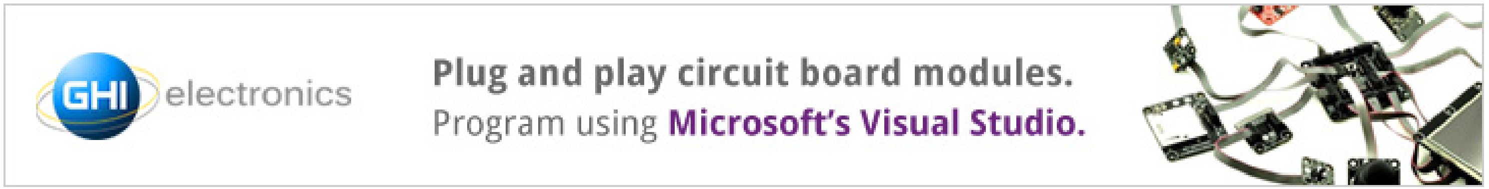
Longest substring in alphabetical order [closed]


GHI electronics
Plug and play circuit board modules.
Program using Microsoft's Visual Studio.

Write a program that prints the longest substring of s in which the letters occur in alphabetical order. For example, if s = 'azcbobobegghakl', then your program should print

Longest substring in alphabetical order is: beggh

In the case of ties, print the first substring. For example, if s = 'abcbcd', then your program should print

Longest substring in alphabetical order is: abc

count subclass slice itertools

asked Oct 25 '13 at 5:13
user2918562
4 2

closed as off-topic by CSᵠ, Erstwhilelll, EdChum, Hamad, Soner Gönül Oct 22 '14 at 7:26

This question appears to be off-topic. The users who voted to close gave this specific reason:

- "Questions seeking debugging help ("**why isn't this code working?**") must include the desired behavior, a *specific problem or error* and *the shortest code necessary* to reproduce it **in the question itself**. Questions without **a clear problem statement** are not useful to other readers. See: How to create a Minimal, Complete, and Verifiable example." – CSᵠ, Erstwhilelll, EdChum, Hamad, Soner Gönül

If this question can be reworded to fit the rules in the help center, please edit the question.

from itertools import count maxsubstr = s[0:0] # empty slice (to accept subclasses of str) for start in range(len(s)): # O(n) for end in count(start + len(maxsubstr) + 1): # O(m) substr = s[start:end] # O(m) if len(set(substr)) != (end - start): # found duplicates or EOS break – user2918562 Oct 25 '13 at 5:13

3 Answers

Here you go edx student i've been helped to finish the code :

```
from itertools import count

def long_sub(input_string):
    maxsubstr = input_string[0:0] # empty slice (to accept subclasses of str)
    for start in range(len(input_string)): # O(n)
        for end in count(start + len(maxsubstr) + 1): # O(m)
            substr = input_string[start:end] # O(m)
            if len(substr) != (end - start): # found duplicates or EOS
                break
            if sorted(substr) == list(substr):
                maxsubstr = substr
    return maxsubstr

sub = (long_sub(s))
print "Longest substring in alphabetical order is: %s" %sub
```

answered Oct 26 '13 at 2:47

spacegame
41 1 6

These are all assuming you have a string (s) and are needing to find the longest substring in alphabetical order.

Option A

```
test = s[0]      # seed with first letter in string s
best = ''        # empty var for keeping track of longest sequence

for n in range(1, len(s)):    # have s[0] so compare to s[1]
    if len(test) > len(best):
        best = test
    if s[n] >= s[n-1]:
        test = test + s[n]    # add s[1] to s[0] if greater or equal
    else:                     # if not, do one of these options
        test = s[n]

print "Longest substring in alphabetical order is:", best
```

Option B

```
maxSub, currentSub, previousChar = '', '', ''
for char in s:
    if char >= previousChar:
        currentSub = currentSub + char
        if len(currentSub) > len(maxSub):
            maxSub = currentSub
    else: currentSub = char
    previousChar = char
print maxSub
```

Option C

```
matches = []
current = [s[0]]
for index, character in enumerate(s[1:]):
    if character >= s[index]: current.append(character)
    else:
        matches.append(current)
        current = [character]
print "".join(max(matches, key=len))
```

Option D

```
def longest_ascending(s):
    matches = []
    current = [s[0]]
    for index, character in enumerate(s[1:]):
        if character >= s[index]:
            current.append(character)
        else:
            matches.append(current)
            current = [character]
    matches.append(current)
    return "".join(max(matches, key=len))
print(longest_ascending(s))
```

answered Nov 11 '13 at 8:47

Ernesto
114 8

The following code solves the problem using the `reduce` method:

```
solution = ''

def check(substr, char):
    global solution
    last_char = substr[-1]
    substr = (substr + char) if char >= last_char else char
    if len(substr) > len(solution):
        solution = substr
    return substr

def get_largest(s):
    global solution
    solution = ''
    reduce(check, list(s))
    return solution
```

answered Oct 26 '13 at 18:32

dparpyani
831 1 3 14
