COMP 202

Fall 2022

Case study 1

Friday, October 7, 2022

Question 1

Consider a function `is_palindrome` that takes a string as input and returns `True` if it is a palindrome, `False` otherwise. A palindrome is a word (or a sentence) that reads the same way both in the forward and reverse direction.

Examples:

```
>>> is_palindrome("racecar")
True
>>> is_palindrome("mom")
True
>>> is_palindrome("a nut for a jar of tuna")
True
>>> is_palindrome("apple")
False
```

The body of the function contains the following pieces of code, but they have been shuffled around. You must place the letters in the correct order to recompose the function. After each letter, write a number corresponding to the number of tabs (indentation level) that code should have. E.g., A0 means to keep the A code block at the margin (no extra indentation); A1 means to indent all the code in A by one tab, and so on. You cannot repeat letters in your solution. Also, you may not have to use all the letters in your solution.

A. `s = str(n)`

B. `while i < len(s):`

C. `return False`

D. `i = 0`

E. `if s[i] != s[i+1]:`

F. `i = i - 1`

G. `def is_palindrome(n):`

H. `if s[i] != s[-i-1]:`

I. `i += 1`

J. `return True`

K. `if i != i-1:`

Question 2

Write a function `count_common_letters` which takes two strings `s1` and `s2` as input and returns how many letters the two strings have in common. The function should be case-insensitive (e.g., a lowercase 'A' and uppercase 'A' should be considered the same). Examples:

```
>>> count_common_letters("banana", "cat")
1
>>> count_common_letters("Silent", "listen")
6
```

Note that it should always be the case that for any string `s` and `t`:

```
count_common_letters(s, t) == count_common_letters(t, s)
```

Question 3

The following code has several errors (stylistic, syntax, runtime and logical). Identify and fix each error.

```
def average(s):
    ''' (int) -> void
    Takes a string of digits as argument, and returns the average of the digits.
    If there are any non-integer elements in the list, return -1 instead.

    >>> average('48')
    6
    >>> average('abc33')
    -1
    '''
    avg == 0
    for i in range(len(numbers)):
        avg += i
    return avg / len(numbers)

if __main__ == '__name__':
    NuMs = input(Enter the digits)
    print(average(NuMs)
```