Status	Finished
Started	Friday, 8 November 2024, 5:58 PM
Completed	Friday, 8 November 2024, 6:36 PM
Duration	37 mins 41 secs

Question **1**

Incorrect

Marked out of 5.00

Write a function that takes an input String (sentence) and generates a new String (modified sentence) by reversing the words in the original String, maintaining the words position.

In addition, the function should be able to control the reversing of the case (upper or lowercase) based on a case_option parameter, as follows:

If case_option = 0, normal reversal of words i.e., if the original sentence is “Wipro TechNologies BangaLore”, the new reversed sentence should be “orpiW seigoloNhceT eroLagnaB”.

If case_option = 1, reversal of words with retaining position’s case i.e., if the original sentence is “Wipro TechNologies BangaLore”, the new reversed sentence should be “Orpiw SeigOlonhcet EroLaGnab”.

Note that positions 1, 7, 11, 20 and 25 in the original string are uppercase W, T, N, B and L.

Similarly, positions 1, 7, 11, 20 and 25 in the new string are uppercase O, S, O, E and G.

NOTE:
1. Only space character should be treated as the word separator i.e., “Hello World” should be treated as two separate words, “Hello” and “World”. However, “Hello,World”, “Hello;World”, “Hello-World” or “Hello/World” should be considered as a single word.

2. Non-alphabetic characters in the String should not be subjected to case changes. For example, if case option = 1 and the original sentence is “Wipro TechNologies, Bangalore” the new reversed sentence should be “Orpiw ,seiGolonhceT Erolagnab”. Note that comma has been treated as part of the word “Technologies,” and when comma had to take the position of uppercase T it remained as a comma and uppercase T took the position of comma. However, the words “Wipro and Bangalore” have changed to “Orpiw” and “Erolagnab”.

3. Kindly ensure that no extra (additional) space characters are embedded within the resultant reversed String.

Examples:

S. No.	input1	input2	output
1	Wipro Technologies Bangalore	0	orpiW seigolonhceT erolagnaB
2	Wipro Technologies, Bangalore	0	orpiW ,seigolonhceT erolagnaB
3	Wipro Technologies Bangalore	1	Orpiw Seigolonhcet Erolagnab
4	Wipro Technologies, Bangalore	1	Orpiw ,seigolonhceT Erolagnab

For example:

Input	Result
`Wipro Technologies Bangalore` `0`	`orpiW seigolonhceT erolagnaB`
`Wipro Technologies, Bangalore` `0`	`orpiW ,seigolonhceT erolagnaB`
`Wipro Technologies Bangalore` `1`	`Orpiw Seigolonhcet Erolagnab`
`Wipro Technologies, Bangalore` `1`	`Orpiw ,seigolonhceT Erolagnab`

Answer: (penalty regime: 0 %)

```
import java.util.*;

public class ReverseWordsWithCase {

    public static void main(String[] args) {
        // Scanner to accept input
        Scanner sc = new Scanner(System.in);

        // Input sentence and case_option
        String sentence = sc.nextLine();
        int caseOption = sc.nextInt();

        // Call function to reverse words and handle case
        String result = reverseWords(sentence, caseOption);

        // Output the result
```

```
        System.out.println(result);
    }

    // Function to reverse words based on case_option
    public static String reverseWords(String sentence, int caseOption) {
        // Split the sentence into words based on spaces
        String[] words = sentence.split(" ");
        StringBuilder result = new StringBuilder();

        // Process each word
        for (String word : words) {
            String reversedWord = reverseWord(word, caseOption);
            result.append(reversedWord).append(" ");
        }

        // Remove trailing space
        return result.toString().trim();
    }

    // Function to reverse a single word and apply case_option
    public static String reverseWord(String word, int caseOption) {
        StringBuilder reversedWord = new StringBuilder(word);

        // Reverse the word
        reversedWord.reverse();

        // Apply the case option
        if (caseOption == 0) {
            return reverseCase(reversedWord.toString());  // Reverse case for case_option 0
        } else if (caseOption == 1) {
            return capitalizeFirstLetterAndLastLetter(reversedWord.toString());  // Handle case_option 1
        }

        return reversedWord.toString(); // Default case, should not happen
    }

```

	Input	Expected	Got	
✓	Wipro Technologies Bangalore 0	orpiW seigolonhceT erolagnaB	orpiW seigolonhceT erolagnaB	✓
✓	Wipro Technologies, Bangalore 0	orpiW ,seigolonhceT erolagnaB	orpiW ,seigolonhceT erolagnaB	✓
✓	Wipro Technologies Bangalore 1	Orpiw Seigolonhcet Erolagnab	Orpiw Seigolonhcet Erolagnab	✓
✗	Wipro Technologies, Bangalore 1	Orpiw ,seigolonhceT Erolagnab	Orpiw ,seigolonhcet Erolagnab	✗

Your code must pass all tests to earn any marks. Try again.

Show differences

Question **2**

Correct

Marked out of 5.00

You are provided with a string which has a sequence of 1's and 0's.

This sequence is the encoded version of a English word. You are supposed write a program to decode the provided string and find the original word.

Each alphabet is represented by a sequence of 0s.

This is as mentioned below:

Z : 0

Y : 00

X : 000

W : 0000

V : 00000

U : 000000

T : 0000000

and so on upto A having 26 0's (00000000000000000000000000).

The sequence of 0's in the encoded form are separated by a single 1 which helps to distinguish between 2 letters.

Example 1:

input1: 010010001

The decoded string (original word) will be: ZYX

Example 2:

input1: 00001000000000000000001000000000001000000000100000000000001

The decoded string (original word) will be: WIPRO

Note: The decoded string must always be in UPPER case.

For example:

Input	Result
010010001	ZYX
00001000000000000000001000000000001000000000100000000000001	WIPRO

Answer: (penalty regime: 0 %)

```
import java.util.Scanner;

public class Decoder {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Taking input string from user
        String encoded = scanner.nextLine();

        // Decoding the string
        String decoded = decode(encoded);

        // Printing the decoded word
        System.out.println( decoded);

        scanner.close();
    }

    private static String decode(String encoded) {
        // Split the string by '1', which separates the sequences of 0's
        String[] zeroSequences = encoded.split("1");

        StringBuilder decoded = new StringBuilder();

        // Iterate through each zero sequence
```

```
        for (String seq : zeroSequences) {
            if (!seq.isEmpty()) {
                // Calculate the length of the zero sequence
                int length = seq.length();

                // Map the length of zeros to the corresponding alphabet letter
                char letter = (char) ('Z' - (length - 1)); // 'Z' corresponds to 1 zero
                decoded.append(letter);
            }
        }

        return decoded.toString();
    }
}
```

	Input	Expected	Got	
✓	010010001	ZYX	ZYX	✓
✓	00001000000000000000001000000000001000000000100000000000001	WIPRO	WIPRO	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 5.00

Given two char arrays input1[] and input2[] containing only lower case alphabets, extracts the alphabets which are present in both arrays (common alphabets).

Get the ASCII values of all the extracted alphabets.

Calculate sum of those ASCII values. Lets call it sum1 and calculate single digit sum of sum1, i.e., keep adding the digits of sum1 until you arrive at a single digit.

Return that single digit as output.

Note:

1. Array size ranges from 1 to 10.
2. All the array elements are lower case alphabets.
3. Atleast one common alphabet will be found in the arrays.

Example 1:

input1: {'a', 'b', 'c'}

input2: {'b', 'c'}

output: 8

Explanation:

'b' and 'c' are present in both the arrays.

ASCII value of 'b' is 98 and 'c' is 99.

98 + 99 = 197

1 + 9 + 7 = 17

1 + 7 = 8

For example:

Input	Result
a b c b c	8

Answer: (penalty regime: 0 %)

```
import java.util.HashSet;

public class CommonCharactersSum {

    public static void main(String[] args) {
        // Example input
        char[] input1 = {'a', 'b', 'c'};
        char[] input2 = {'b', 'c'};

        // Function call to process and get the final result
        int result = getSingleDigitSumOfCommonASCIIValues(input1, input2);

        // Output the result
        System.out.println(result);
    }

    public static int getSingleDigitSumOfCommonASCIIValues(char[] input1, char[] input2) {
        // Step 1: Find common characters using a set
        HashSet<Character> set1 = new HashSet<>();
        for (char c : input1) {
            set1.add(c);
        }

        HashSet<Character> commonChars = new HashSet<>();
        for (char c : input2) {
            if (set1.contains(c)) {
                commonChars.add(c);
            }
        }
```

```
30
31          // Step 2: Calculate the sum of ASCII values of common characters
32          int sum1 = 0;
33          for (char c : commonChars) {
34              sum1 += (int) c; // Get ASCII value of the character
35          }
36
37          // Step 3: Calculate the single digit sum (digital root)
38          return digitalRoot(sum1);
39      }
40
41      // Method to calculate the digital root (single-digit sum)
42      public static int digitalRoot(int n) {
43          if (n == 0) {
44              return 0;
45          }
46          return 1 + (n - 1) % 9;
47      }
48  }
49
```

	Input	Expected	Got	
✓	a b c b c	8	8	✓

Passed all tests! ✓

◄ Lab-12-MCQ

Jump to...

Identify possible words ►