CS23333-Object Oriented Programming Using Java-2023

Quiz navigation

1 2 3

Show one page at a time

Finish review

Status	Finished
Started	Tuesday, 19 November 2024, 2:43 PM
Completed	Tuesday, 19 November 2024, 2:55 PM
Duration	11 mins 59 secs

Question **1**

Correct

Marked out of 5.00

Flag question

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 %)

```java
import java.util.Scanner;

public class ReverseWordsWithCaseControl {

    // Function to reverse words and control case
    public static String reverseWordsWithCaseControl(String sentence, int caseOption) {
        // Split the sentence into words using space as the delimiter
        String[] words = sentence.split(" ");
        StringBuilder result = new StringBuilder();

        // Iterate through each word
        for (String word : words) {
            // Reverse each word
            String reversedWord = new StringBuilder(word).reverse().toString();

            if (caseOption == 1) {
                // Case reversal with position maintenance (ignoring non-alphabetic characters)
                StringBuilder caseReversedWord = new StringBuilder();

                for (int i = 0; i < word.length(); i++) {
                    char originalChar = word.charAt(i);
                    char reversedChar = reversedWord.charAt(i);

                    // If the character is alphabetic, apply case transformation
                    if (Character.isAlphabetic(originalChar)) {
                        if (Character.isUpperCase(originalChar)) {
                            caseReversedWord.append(Character.toUpperCase(reversedChar));
                        } else {
                            caseReversedWord.append(Character.toLowerCase(reversedChar));
                        }
                    } else {
                        // If it's a non-alphabetic character, keep it as is
                        caseReversedWord.append(reversedChar);
                      }
                }

                result.append(caseReversedWord).append(" ");
            } else {
                // Normal case, just append the reversed word
                result.append(reversedWord).append(" ");
            }
        }

        // Remove the extra space at the end
        return result.toString().trim();
    }

    public static void main(String[] args) {
        // Create a scanner to read input from user
        Scanner sc = new Scanner(System.in);

        // Read input sentence
```

	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	

Passed all tests!

Question **2**

Correct

Marked out of 5.00

Flag question

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 %)

```
public class CommonCharsAsciiSum{
    public static int calculatesingledigitsum(char[] input1,char[] input2){
        StringBuilder commonChars=new StringBuilder();
        for(char c1:input1){
            for(char c2:input2){
                if(c1==c2){
                    commonChars.append(c1);
                    break;
                }
            }
        }
        int asciisum=0;
        for(int i=0;i<commonChars.length();i++){
            asciisum+=commonChars.charAt(i);
        }
        while(asciisum>=10){
            int tempsum=0;
            while(asciisum>0){
                tempsum+=asciisum%10;
                asciisum=asciisum/10;
            }
            asciisum=tempsum;
        }
        return asciisum;
    }
    public static void main(String[] args){
        char[] input1={'a','b','c'};
        char[] input2={'b','c'};
        System.out.println(calculatesingledigitsum(input1,input2));
    }
}
```

	Input	Expected	Got	
	a b c b c	8	8	

Passed all tests!

Question **3**

Correct

Marked out of 5.00

Flag question

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: 00001000000000000000000100000000000100000000010000000000001

The decoded string (original word) will be: WIPRO

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

For example:

Input	Result
010010001	ZYX
00001000000000000000000100000000000100000000010000000000001	WIPRO

Answer: (penalty regime: 0 %)

```java
import java.util.Scanner;

public class DecodeString {

    // Function to decode the string
    public static String decodeString(String input) {
        // Split the input string by '1', as '1' is the delimiter
        String[] parts = input.split("1");
        StringBuilder result = new StringBuilder();

        // Iterate over each part
        for (String part : parts) {
            // Skip empty parts (which could appear due to multiple consecutive '1's)
            if (!part.isEmpty()) {
                int length = part.length();

                // Calculate the letter corresponding to the length of 0's
                // 'Z' corresponds to 1 zero, 'Y' to 2 zeros, ..., 'A' corresponds to 26 zeros
                char letter = (char) ('Z' - (length - 1));
                result.append(letter);
            }
        }

        return result.toString();
    }

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


        String input = sc.nextLine();
        // Decode the string and output the result
        String decodedString = decodeString(input);
        System.out.println(decodedString);

        sc.close();
    }
}


```

	Input	Expected	Got	
	010010001	ZYX	ZYX	
	00001000000000000000000100000000000100000000010000000000001	WIPRO	WIPRO	

Passed all tests!

Finish review

◄ Lab-12-MCQ

Jump to...

Identify possible words ►