

DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Lab Assignment

# Student Name: Puneet Yadav UID: 22BCS17246

**Branch: BE-CSE Section/Group:636 B**

**Semester: 6th Date of Performance: 04/04/25 Subject Name: PBLJ with Lab Subject Code: 22CSH-359**

**Problem 1:-** Consider a function public String match Found(String input 1, String input 2), where:

* input1 will contain only a single word with only 1 character replaces by an underscore ‘\_’
* input2 will contain a series of words separated by colons and no space character in between
* input2 will not contain any other special character other than underscore and alphabetic characters. The methods should return output in a String type variable “output1” which contains all the words from input2 separated by colon which matches with input 1. All words in output1 should be in uppercase. **Code:-**

public class MatchFinder {

    public static String matchFound(String input1, String input2) {

        StringBuilder output1 = new StringBuilder();

        String[] words = input2.split(":");

        String regex = input1.replace("\_", ".");

        for (String word : words) {

            if (word.matches(regex)) {

                if (output1.length() > 0) {

                    output1.append(":");

                }

                output1.append(word.toUpperCase());

            }

        }

        return output1.toString();

    }

    public static void main(String[] args) {

        String input1 = "h\_llo";

        String input2 = "hello:hallo:hullo:hillo:hiello";

        System.out.println(matchFound(input1, input2));

        System.out.println("Anita 22BCS10868");

    }

}

# Output:-

# 

**Problem 2:-** Encoding Three Strings: Anand was assigned the task of coming up with an encoding mechanism for any given three strings.He has come up with the following plan.

Step ONE :- Given any three strings,break each string into 3 parts each. For example- if the three strings are below:

Input 1: “John”

Input 2: “Johny”

Input 3 : “Janardhan”

“John” should be split into”J”,”oh”,”n,” as the FRONT ,MIDDLE and END part repectivly.

“Johny” should be spilt into “jo”,” h”, “ny” as the FRONT ,MIDDLE and END respectively.

“Janardhan” should be split into “Jan” ,” ard” ,”han” as the FRONT ,MIDDLE and END part respectively. i.e. If the no. of characters in the string are in multiples of 3 ,then each split –part will contain equal no of characters , as seen in the example of “Janadhan”.

If the no. of characters in the string are NOT in multiples of 3 ,and if there is one character more than multiple of 3, then the middle part will get the extra character ,as seen in the example of “john”.

If the no. of characters in the string are Not in multiples of 3 and if there are two characters more than multiple of 3, then the FRONT and END parts will get one extra character each, as seen in the example of “Johny”.

Step TWO : Concatenate (join) the FRONT ,MIDDLE and END parts of the string as per the below specified concatenation – rule to from three Output strings.

Output 1: FRONT part of input 1 + MIDDLE part of input 2 +END part of input 3 Output2:- MIDDLE part of input1+ END part of input2 + FRONT part of input3 Output3: END part of the input1+FRONT part of input2 +MIDDLE part of input3 For example , for the above example input strings:

Output1 = “J” +”h”+han”=”jhhan” Output2 =”oh”+”ny”+”Jan”=”ohnyjan” Output3=”n”+Jo”+ard”+=“njoard” Step THREE :-

Process the resulting output strings based on the output-processing rule .After the above two steps, we will now have three output strings. Further processing is required only for the third output string as per below rule-



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

“Toggle the case of each character in the string “ ,i.e. in the third output string, all lower-case characters should be made upper-case and vice versa.

For example , for the above example strings ,output3 is “nJoard”, so after applying the toggle rule.

Output3 should become “NjOARD”.

Final Result – The three output strings after applying the above three steps i.e. for the above example . Output1 =”Jnhan”

Output2=”ohnyJan’ Output3 = “NJOARD”

Help Anand to write a program that would do the above.

**Code:-**

import java.util.Scanner;

public class StringEncoder {

    public static String[] splitString(String input) {

        int len = input.length();

        int partSize = len / 3;

        int remainder = len % 3;

        int frontSize = partSize + (remainder == 2 ? 1 : 0);

        int middleSize = partSize + (remainder == 1 ? 1 : 0);

        int endSize = partSize;

        String front = input.substring(0, frontSize);

        String middle = input.substring(frontSize, frontSize + middleSize);

        String end = input.substring(frontSize + middleSize);

        return new String[]{front, middle, end};

    }

    public static String toggleCase(String input) {

        StringBuilder toggled = new StringBuilder();

        for (char c : input.toCharArray()) {

            if (Character.isUpperCase(c)) {

                toggled.append(Character.toLowerCase(c));

            } else {

                toggled.append(Character.toUpperCase(c));

            }

        }

return toggled.toString();

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter first string: ");

        String input1 = scanner.nextLine();

        System.out.print("Enter second string: ");

        String input2 = scanner.nextLine();

        System.out.print("Enter third string: ");

        String input3 = scanner.nextLine();

        scanner.close();

        String[] parts1 = splitString(input1);

        String[] parts2 = splitString(input2);

        String[] parts3 = splitString(input3);

        String output1 = parts1[0] + parts2[1] + parts3[2];

        String output2 = parts1[1] + parts2[2] + parts3[0];

        String output3 = parts1[2] + parts2[0] + parts3[1];

        output3 = toggleCase(output3);

        System.out.println("Output1: " + output1);

        System.out.println("Output2: " + output2);

        System.out.println("Output3: " + output3);

    }

}

# Output:-

# 



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

**Problem 3:-** Given a String (In Uppercase alphabets or Lowercase alphabets), new alphabets is

to be appended with following rule:

If the alphabet is present in the input string, use the numeric value of that alphabet.

1. E.g. a or A numeric value is 1 and so on. New alphabet to be appended between 2 alphabets:
2. If (sum of numeric value of 2 alphabets) %26 is 0, then append 0.
3. E.g. string is ay. Numeric value of a is 1, y is 25. Sum is 26. Remainder is 0, the new string will be a0y.
4. Otherwise (sum of numeric value of 2 alphabets) %26 numeric value alphabet is to be appended. E.g. ac is string. Numeric value of a is 1, c is 3, sum is 4. Remainder with 26 is 4. Alphabet to be appended is d. output will be adc.
5. If a digit is present, it will be the same in the output string. E.g. string is 12, output string is 12.
6. If only a single alphabet is present, it will be the same in the output string. E.g. input string is 1a, output will be 1a.
7. If space is present, it will be the same in the output string. E.g. string is ac 12a, output will be adc 12a.

Constraint: Whether string alphabets are In Uppercase or Lowercase, appended alphabets must be in lower case. Output string must also be in lowercase.

# Code:-

import java.util.Scanner;

public class AlphabetAppender {

    public static String appendAlphabets(String input) {

        StringBuilder output = new StringBuilder();

        for (int i = 0; i < input.length(); i++) {

            char current = input.charAt(i);

            output.append(Character.toLowerCase(current));

            if (i < input.length() - 1) {

                char next = input.charAt(i + 1);

                if (Character.isLetter(current) && Character.isLetter(next)) {

                    int sum = (Character.toLowerCase(current) - 'a' + 1) + (Character.toLowerCase(next) - 'a' + 1);

                    char newChar = (sum % 26 == 0) ? '0' : (char) ('a' + (sum % 26) - 1);

                    output.append(newChar);

                }

            }

        }

        return output.toString();

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter input string: ");

        String input = scanner.nextLine();

        scanner.close();

        System.out.println("Output string: " + appendAlphabets(input));

    }

}

# Output:-

# 

**Problem 4:-** String t is generated by random shuffling string s and then add one more letter at a random position.

Return the letter that was added to t. Hint:-

Input: s = "abcd", t = "abcde"

Output: "e"

# Code:-

import java.util.Scanner;

public class FindAddedLetter {

    public static char findTheDifference(String s, String t) {

        int sumS = 0, sumT = 0;

        for (char c : s.toCharArray()) {

            sumS += c;

        }

        for (char c : t.toCharArray()) {

            sumT += c;

        }

        return (char) (sumT - sumS);

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter string s: ");

        String s = scanner.nextLine();

        System.out.print("Enter string t: ");

        String t = scanner.nextLine();

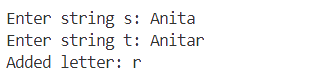
        scanner.close();

        System.out.println("Added letter: " + findTheDifference(s, t));

    }

}

# Output:-



**Problem 5:-** The next greater element of some element x in an array is the first greater element that is to the right of x in the same array.

You are given two distinct 0-indexed integer arrays nums1 and nums2, where nums1 is a subset of nums2.

For each 0 <= i < nums1.length, find the index j such that nums1[i] == nums2[j] and determine the next greater element of nums2[j] in nums2. If there is no next greater element, then the answer for this query is -1.

Return an array ans of length nums1.length such that ans[i] is the next greater element as described above. Hint:

Input: nums1 = [4,1,2], nums2 = [1,3,4,2] Output: [-1,3,-1]

Explanation: The next greater element for each value of nums1 is as follows:

* 4 is underlined in nums2 = [1,3,4,2]. There is no next greater element, so the answer is -1.
* 1 is underlined in nums2 = [1,3,4,2]. The next greater element is 3.
* 2 is underlined in nums2 = [1,3,4,2]. There is no next greater element, so the answer is -1.



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

# Code:-

import java.util.\*;

public class NextGreaterElement {

    public static int[] nextGreaterElement(int[] nums1, int[] nums2) {

        Map<Integer, Integer> nextGreaterMap = new HashMap<>();

        Stack<Integer> stack = new Stack<>();

        for (int num : nums2) {

            while (!stack.isEmpty() && stack.peek() < num) {

                nextGreaterMap.put(stack.pop(), num);

            }

            stack.push(num);

        }

        while (!stack.isEmpty()) {

            nextGreaterMap.put(stack.pop(), -1);

        }

        int[] ans = new int[nums1.length];

        for (int i = 0; i < nums1.length; i++) {

            ans[i] = nextGreaterMap.get(nums1[i]);

        }

        return ans;

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter nums1 size: ");

        int n1 = scanner.nextInt();

        int[] nums1 = new int[n1];

        System.out.println("Enter elements of nums1: ");

        for (int i = 0; i < n1; i++) {

            nums1[i] = scanner.nextInt();

        }

        System.out.print("Enter nums2 size: ");

        int n2 = scanner.nextInt();

        int[] nums2 = new int[n2];

        System.out.println("Enter elements of nums2: ");

        for (int i = 0; i < n2; i++) {

            nums2[i] = scanner.nextInt();

        }

        scanner.close();

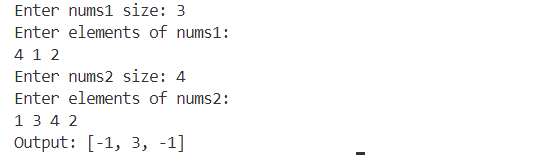
        int[] result = nextGreaterElement(nums1, nums2);

        System.out.println("Output: " + Arrays.toString(result));

    }

}

**Output:-**

****