Name: JATIN SINGH
SECTION:631/A
UID:222BCS10887

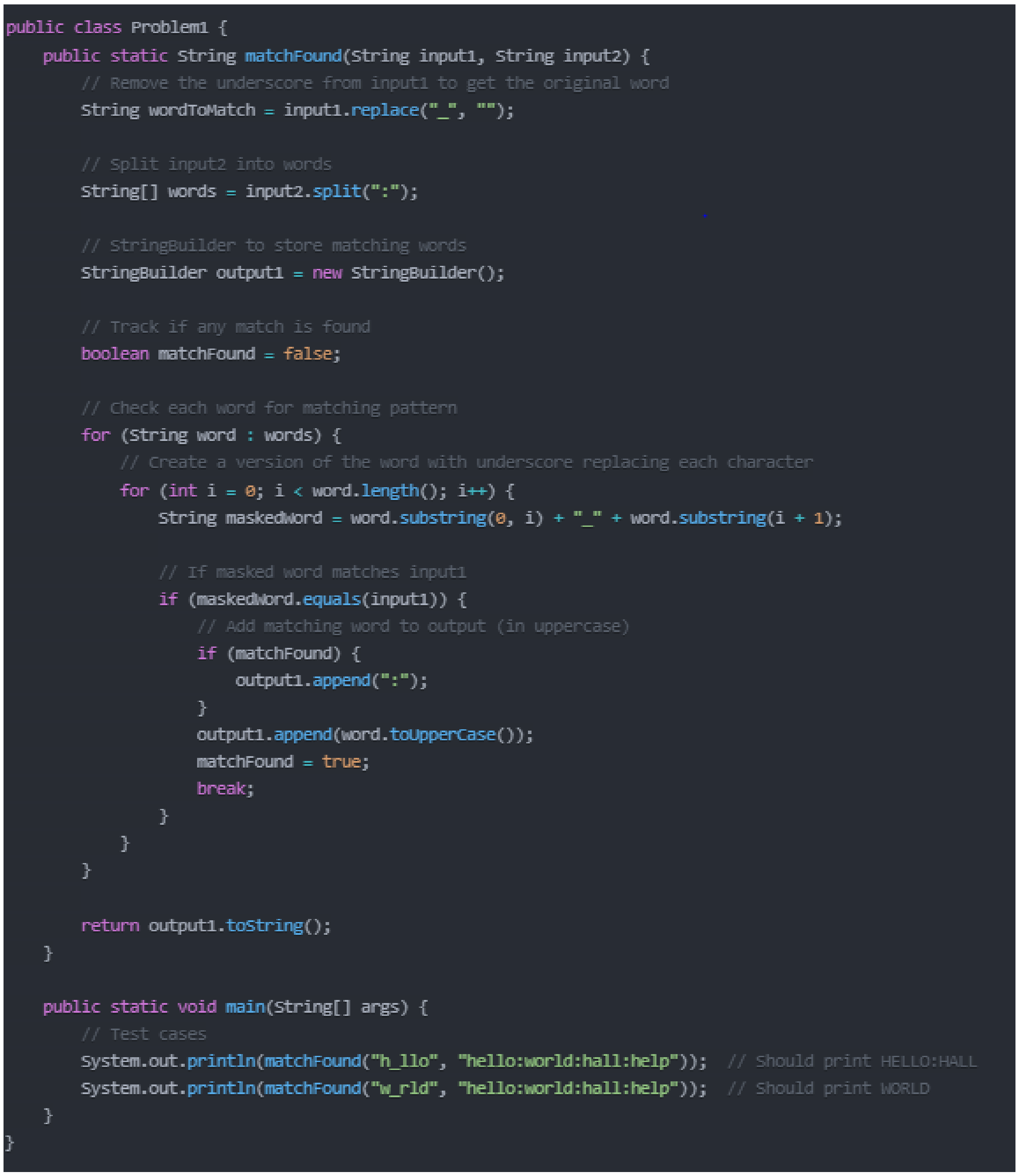
Subject: Project Based Learning in Java
Subject Code: 22CSH-359

Lab Based Complex Coding Problems

Problem 1.

Consider a function **public String matchFound(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.

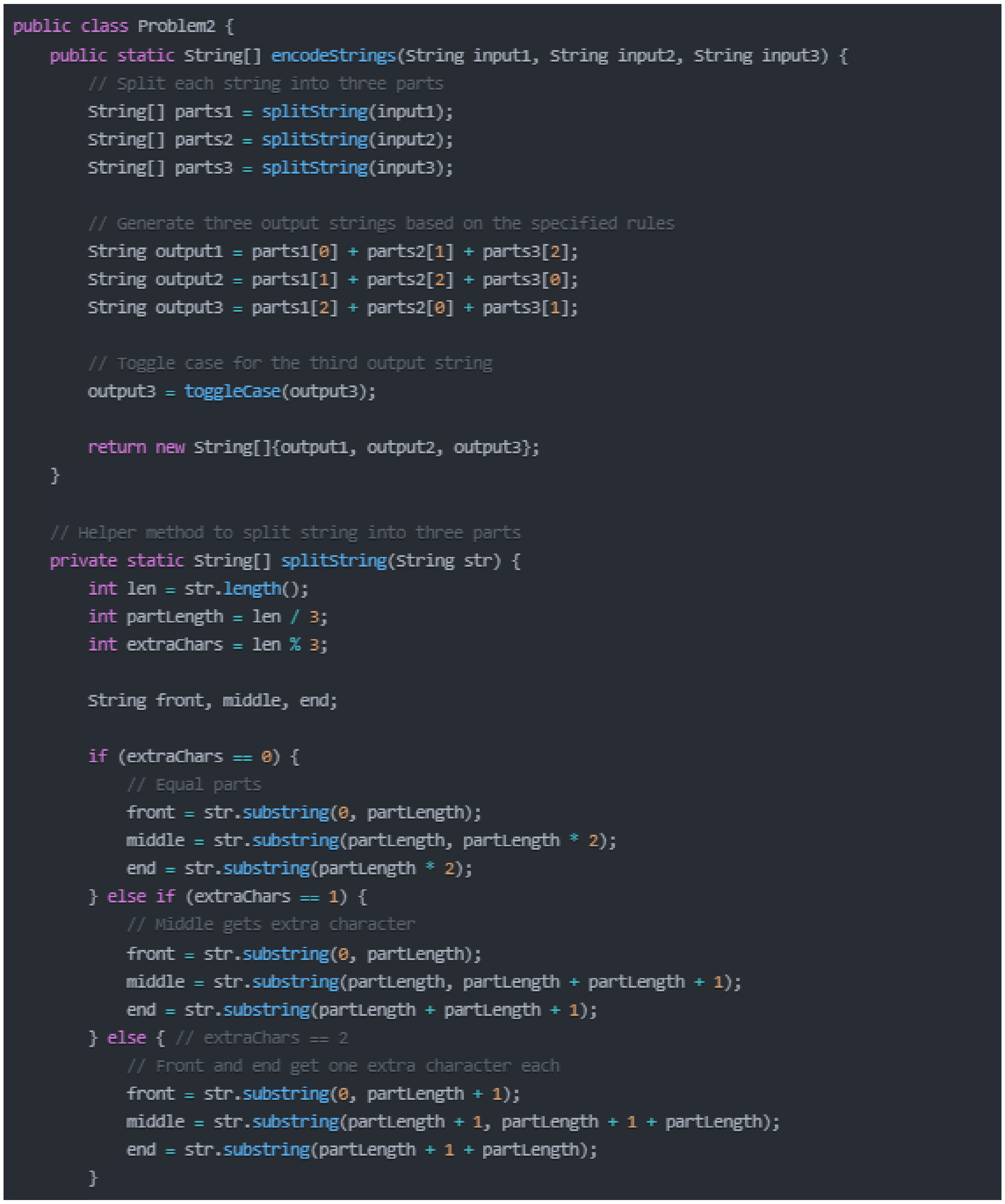


```java
public class Problem1 {
    public static String matchFound(String input1, String input2) {
        // Remove the underscore from input1 to get the original word
        String wordToMatch = input1.replace("_", "");

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

        // StringBuilder to store matching words
        StringBuilder output1 = new StringBuilder();

        // Track if any match is found
        boolean matchFound = false;

        // Check each word for matching pattern
        for (String word : words) {
            // Create a version of the word with underscore replacing each character
            for (int i = 0; i < word.length(); i++) {
                String maskedWord = word.substring(0, i) + "_" + word.substring(i + 1);

                // If masked word matches input1
                if (maskedWord.equals(input1)) {
                    // Add matching word to output (in uppercase)
                    if (matchFound) {
                        output1.append(":");
                    }
                    output1.append(word.toUpperCase());
                    matchFound = true;
                    break;
                }
            }
        }

        return output1.toString();
    }

    public static void main(String[] args) {
        // Test cases
        System.out.println(matchFound("h_llo", "hello:world:hall:help"));  // Should print HELLO:HALL
        System.out.println(matchFound("w_rld", "hello:world:hall:help"));  // Should print WORLD
    }
}
```

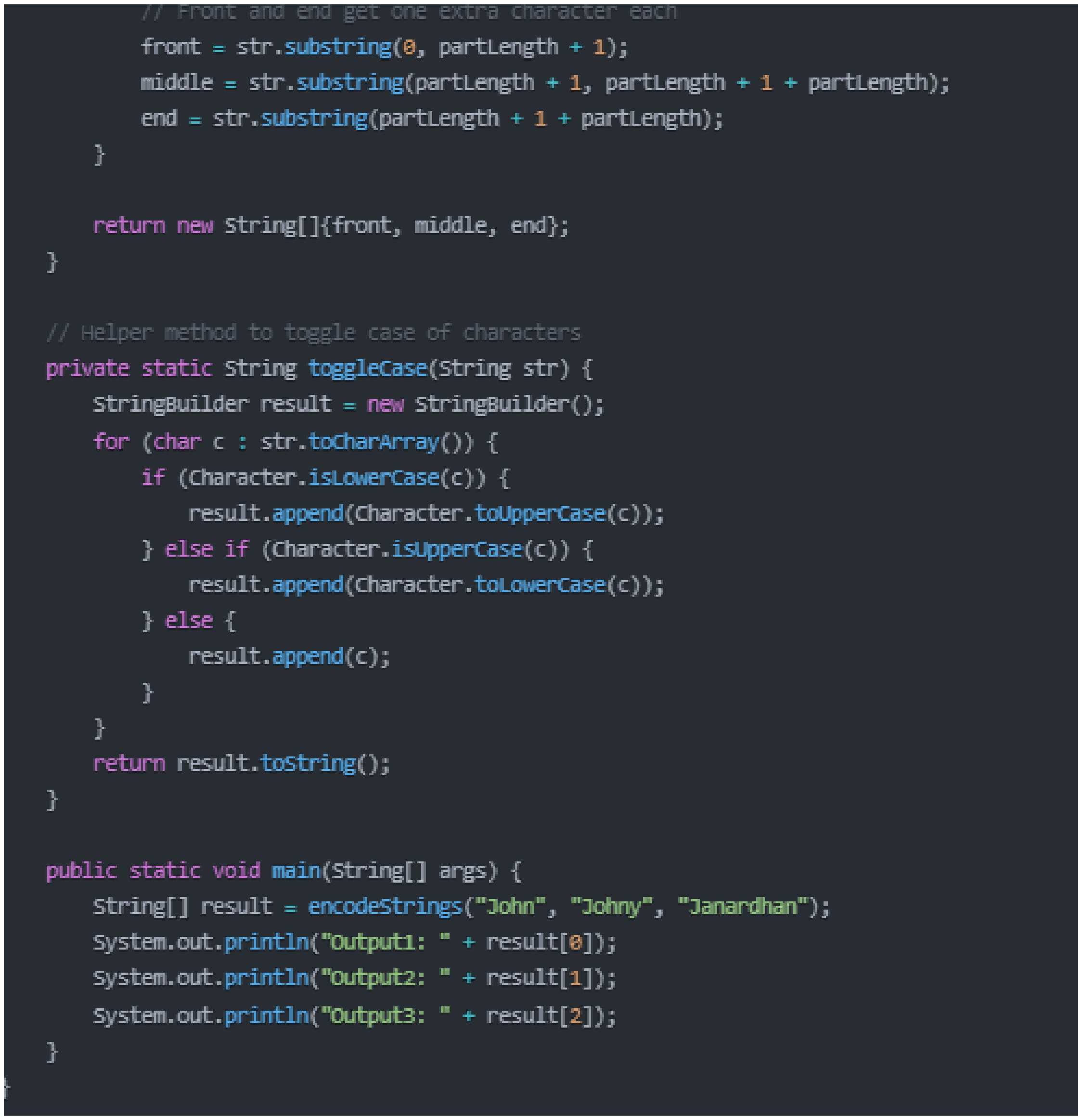
-

Name: JATIN SINGH **Subject: Project Based Learning in Java**
SECTION:631/A **Subject Code: 22CSH-359**
UID:222BCS10887

- **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.



```java
public class Problem2 {
    public static String[] encodeStrings(String input1, String input2, String input3) {
        // Split each string into three parts
        String[] parts1 = splitString(input1);
        String[] parts2 = splitString(input2);
        String[] parts3 = splitString(input3);

        // Generate three output strings based on the specified rules
        String output1 = parts1[0] + parts2[1] + parts3[2];
        String output2 = parts1[1] + parts2[2] + parts3[0];
        String output3 = parts1[2] + parts2[0] + parts3[1];

        // Toggle case for the third output string
        output3 = toggleCase(output3);

        return new String[]{output1, output2, output3};
    }

    // Helper method to split string into three parts
    private static String[] splitString(String str) {
        int len = str.length();
        int partLength = len / 3;
        int extraChars = len % 3;

        String front, middle, end;

        if (extraChars == 0) {
            // Equal parts
            front = str.substring(0, partLength);
            middle = str.substring(partLength, partLength * 2);
            end = str.substring(partLength * 2);
        } else if (extraChars == 1) {
            // Middle gets extra character
            front = str.substring(0, partLength);
            middle = str.substring(partLength, partLength + partLength + 1);
            end = str.substring(partLength + partLength + 1);
        } else { // extraChars == 2
            // Front and end get one extra character each
            front = str.substring(0, partLength + 1);
            middle = str.substring(partLength + 1, partLength + 1 + partLength);
            end = str.substring(partLength + 1 + partLength);
        }
```



```
            // Front and end get one extra character each
            front = str.substring(0, partLength + 1);
            middle = str.substring(partLength + 1, partLength + 1 + partLength);
            end = str.substring(partLength + 1 + partLength);
        }

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

    // Helper method to toggle case of characters
    private static String toggleCase(String str) {
        StringBuilder result = new StringBuilder();
        for (char c : str.toCharArray()) {
            if (Character.isLowerCase(c)) {
                result.append(Character.toUpperCase(c));
            } else if (Character.isUpperCase(c)) {
                result.append(Character.toLowerCase(c));
            } else {
                result.append(c);
            }
        }
        return result.toString();
    }

    public static void main(String[] args) {
        String[] result = encodeStrings("John", "Johny", "Janardhan");
        System.out.println("Output1: " + result[0]);
        System.out.println("Output2: " + result[1]);
        System.out.println("Output3: " + result[2]);
    }
}
```

Problem 3:

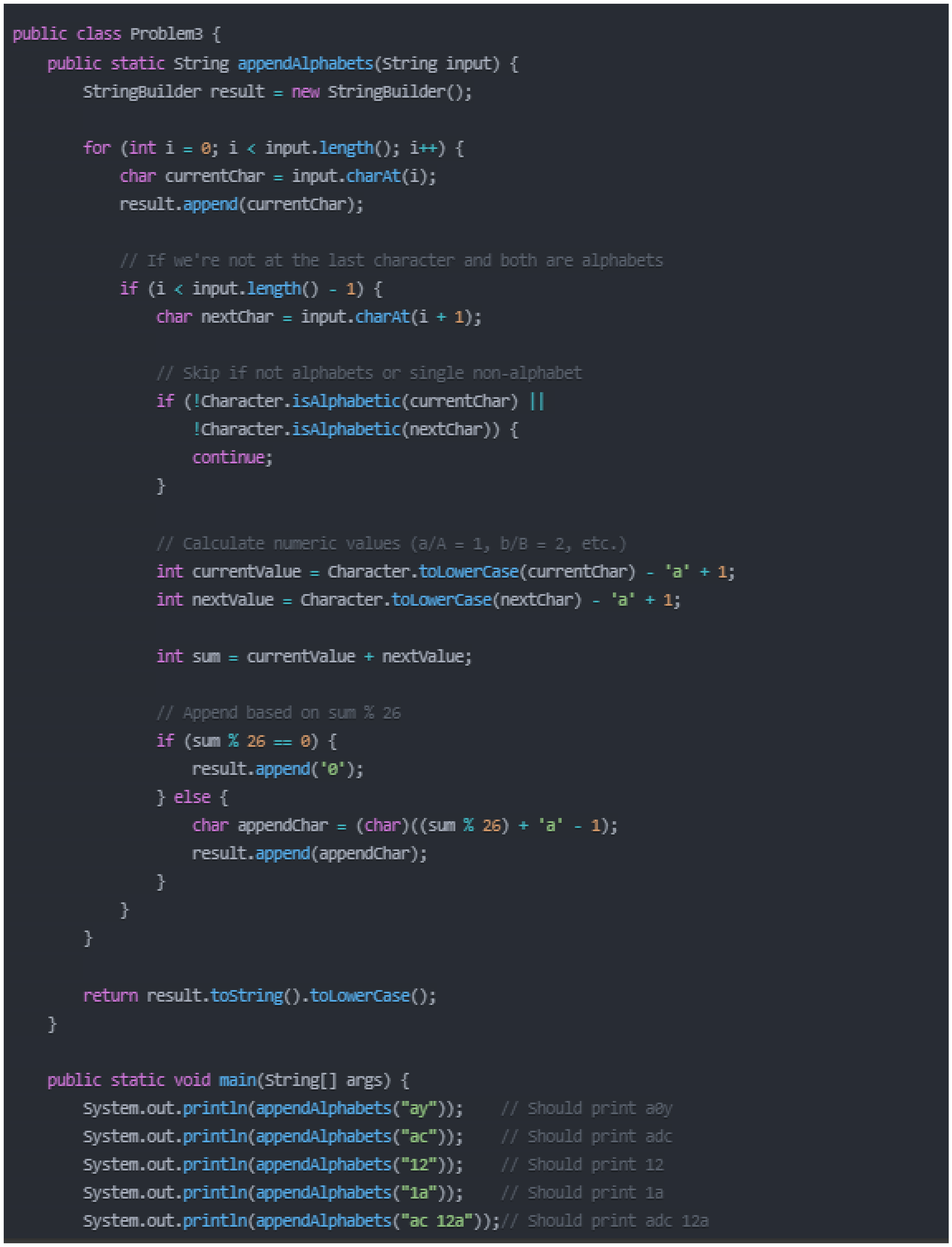
Given a String (In Uppercase alphabets or Lowercase alphabets), new alphabets is to be appended with following rule:

(i) If the alphabet is present in the input string, use the numeric value of that alphabet. E.g. a or A numeric value is 1 and so on. New alphabet to be appended between 2 alphabets:

(a) If (sum of numeric value of 2 alphabets) %26 is 0, then append 0. 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.

Name: JATIN SINGH
SECTION:631/A
UID:222BCS10887

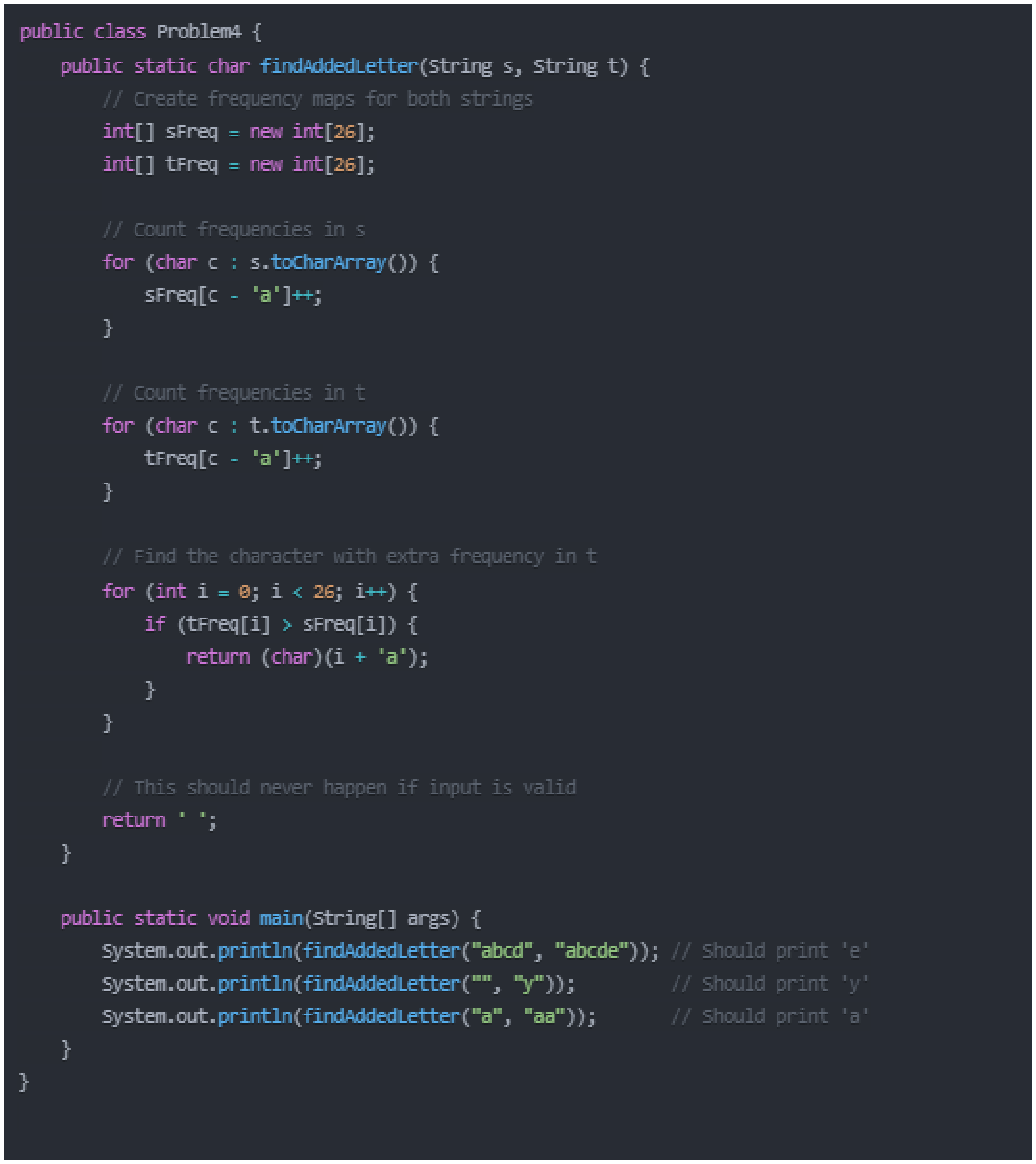
Subject: Project Based Learning in Java
Subject Code: 22CSH-359



```java
public class Problem3 {
    public static String appendAlphabets(String input) {
        StringBuilder result = new StringBuilder();

        for (int i = 0; i < input.length(); i++) {
            char currentChar = input.charAt(i);
            result.append(currentChar);

            // If we're not at the last character and both are alphabets
            if (i < input.length() - 1) {
                char nextChar = input.charAt(i + 1);

                // Skip if not alphabets or single non-alphabet
                if (!Character.isAlphabetic(currentChar) ||
                    !Character.isAlphabetic(nextChar)) {
                    continue;
                }

                // Calculate numeric values (a/A = 1, b/B = 2, etc.)
                int currentValue = Character.toLowerCase(currentChar) - 'a' + 1;
                int nextValue = Character.toLowerCase(nextChar) - 'a' + 1;

                int sum = currentValue + nextValue;

                // Append based on sum % 26
                if (sum % 26 == 0) {
                    result.append('0');
                } else {
                    char appendChar = (char)((sum % 26) + 'a' - 1);
                    result.append(appendChar);
                }
            }
        }

        return result.toString().toLowerCase();
    }

    public static void main(String[] args) {
        System.out.println(appendAlphabets("ay"));    // Should print a0y
        System.out.println(appendAlphabets("ac"));    // Should print adc
        System.out.println(appendAlphabets("12"));    // Should print 12
        System.out.println(appendAlphabets("1a"));    // Should print 1a
        System.out.println(appendAlphabets("ac 12a"));// Should print adc 12a
```

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.



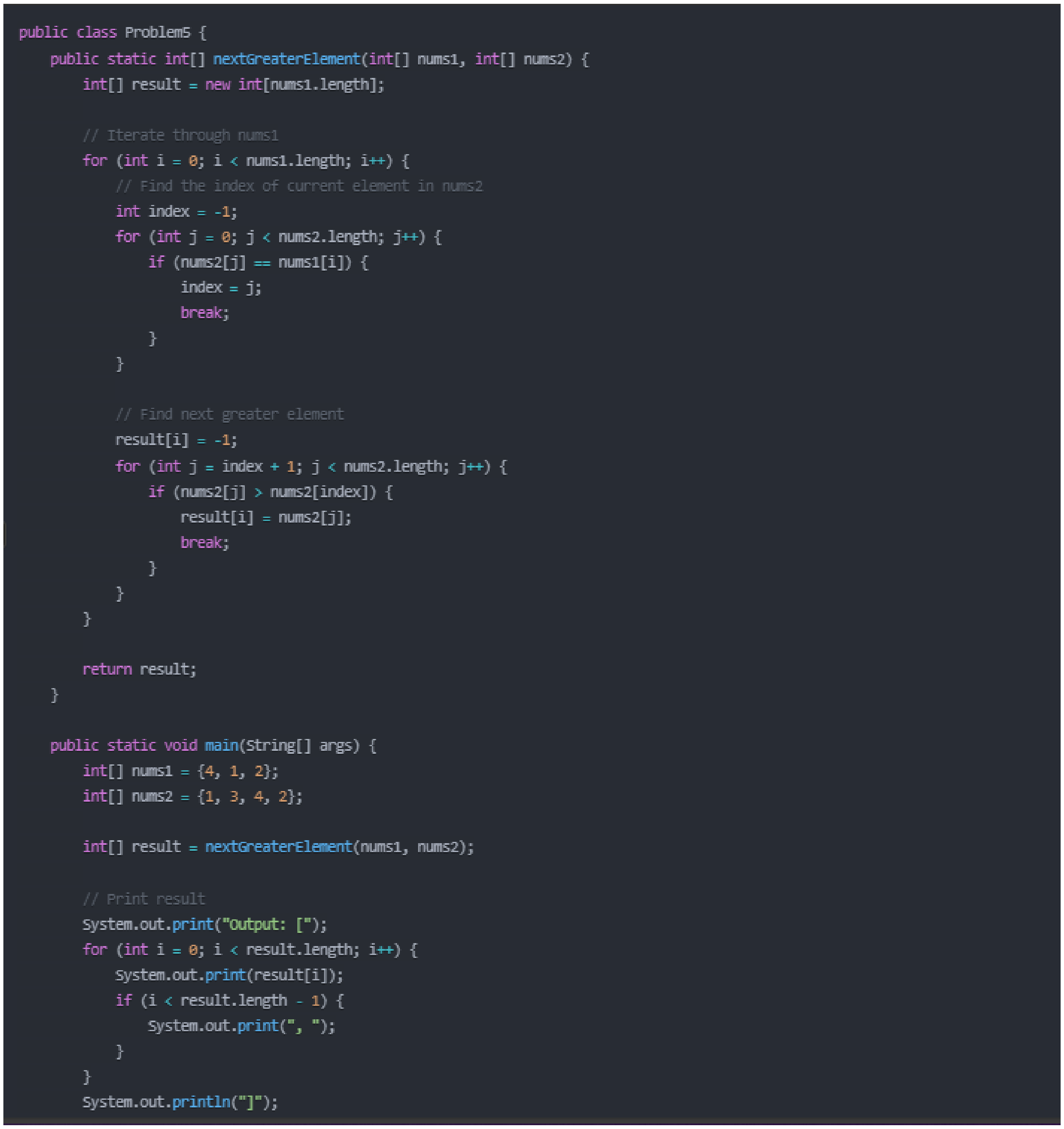
```java
public class Problem4 {
    public static char findAddedLetter(String s, String t) {
        // Create frequency maps for both strings
        int[] sFreq = new int[26];
        int[] tFreq = new int[26];

        // Count frequencies in s
        for (char c : s.toCharArray()) {
            sFreq[c - 'a']++;
        }

        // Count frequencies in t
        for (char c : t.toCharArray()) {
            tFreq[c - 'a']++;
        }

        // Find the character with extra frequency in t
        for (int i = 0; i < 26; i++) {
            if (tFreq[i] > sFreq[i]) {
                return (char)(i + 'a');
            }
        }

        // This should never happen if input is valid
        return ' ';
    }

    public static void main(String[] args) {
        System.out.println(findAddedLetter("abcd", "abcde")); // Should print 'e'
        System.out.println(findAddedLetter("", "y"));         // Should print 'y'
        System.out.println(findAddedLetter("a", "aa"));       // Should print 'a'
    }
}
```

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.

Name: JATIN SINGH
SECTION:631/A
UID:222BCS10887

Subject: Project Based Learning in Java
Subject Code: 22CSH-359



```java
public class Problem5 {
    public static int[] nextGreaterElement(int[] nums1, int[] nums2) {
        int[] result = new int[nums1.length];

        // Iterate through nums1
        for (int i = 0; i < nums1.length; i++) {
            // Find the index of current element in nums2
            int index = -1;
            for (int j = 0; j < nums2.length; j++) {
                if (nums2[j] == nums1[i]) {
                    index = j;
                    break;
                }
            }

            // Find next greater element
            result[i] = -1;
            for (int j = index + 1; j < nums2.length; j++) {
                if (nums2[j] > nums2[index]) {
                    result[i] = nums2[j];
                    break;
                }
            }
        }

        return result;
    }

    public static void main(String[] args) {
        int[] nums1 = {4, 1, 2};
        int[] nums2 = {1, 3, 4, 2};

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

        // Print result
        System.out.print("Output: [");
        for (int i = 0; i < result.length; i++) {
            System.out.print(result[i]);
            if (i < result.length - 1) {
                System.out.print(", ");
            }
        }
        System.out.println("]");
```