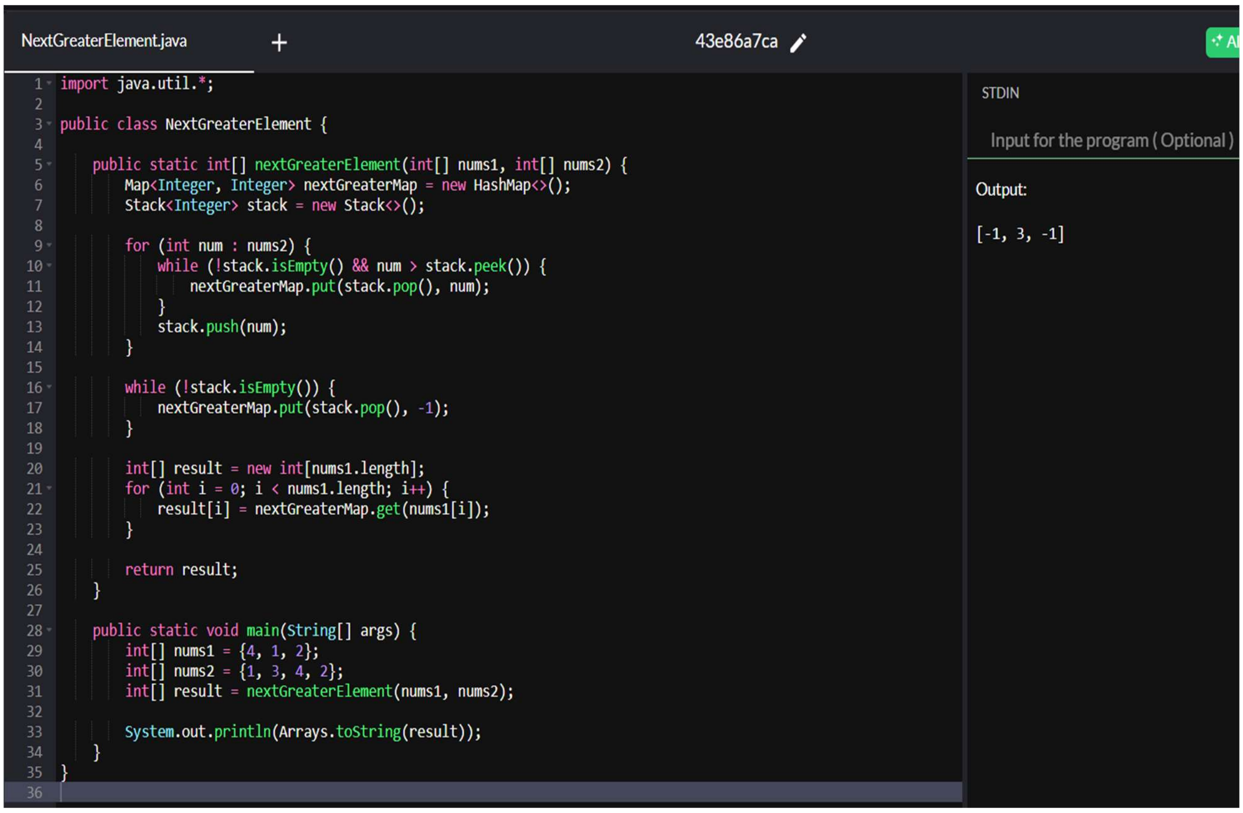
**Name: Rudra Kathait**
**UID: 22BCS14791**
**Subject: PBLJ**
**Section: IOT-631/A**

# Lab Based Average Coding Problems

## QUESTION 1.

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.



```java
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() && num > stack.peek()) {
                nextGreaterMap.put(stack.pop(), num);
            }
            stack.push(num);
        }

        while (!stack.isEmpty()) {
            nextGreaterMap.put(stack.pop(), -1);
        }

        int[] result = new int[nums1.length];
        for (int i = 0; i < nums1.length; i++) {
            result[i] = nextGreaterMap.get(nums1[i]);
        }

        return result;
    }

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

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

Output:

```
[-1, 3, -1]
```

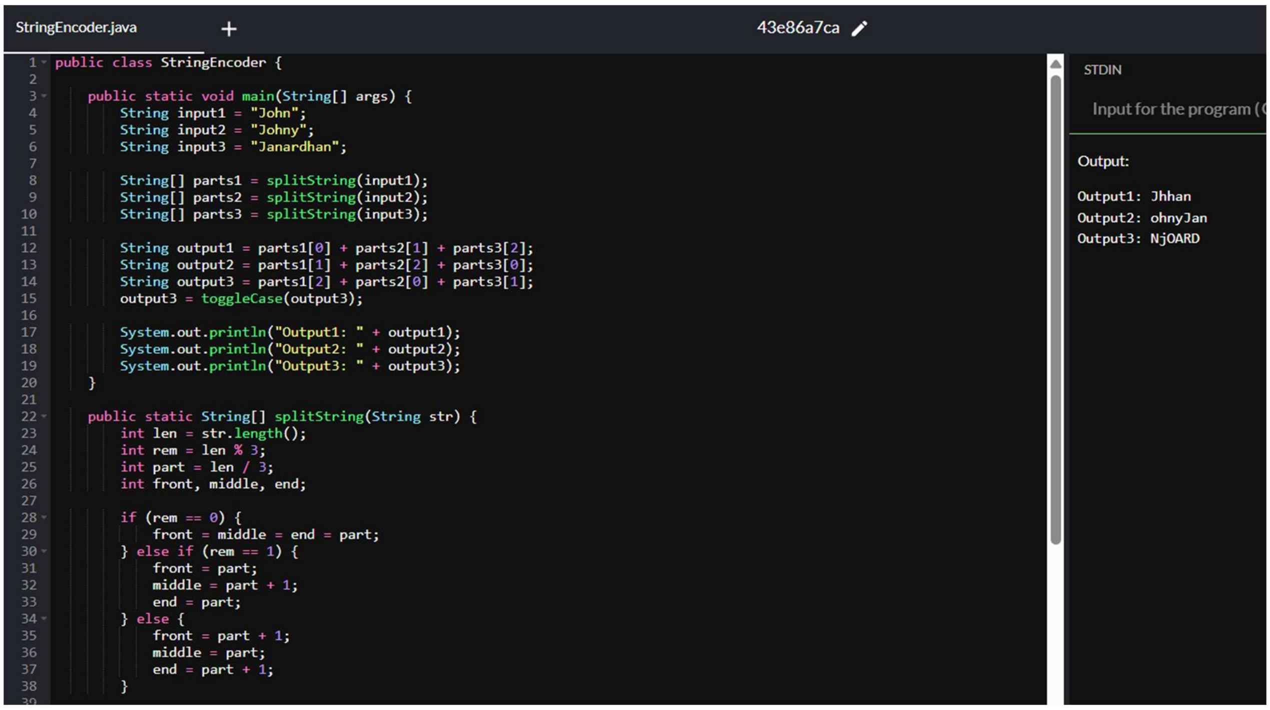
## QUESTION 2.

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 string. Further processing is required only for the third output string as per below rule-
“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”.

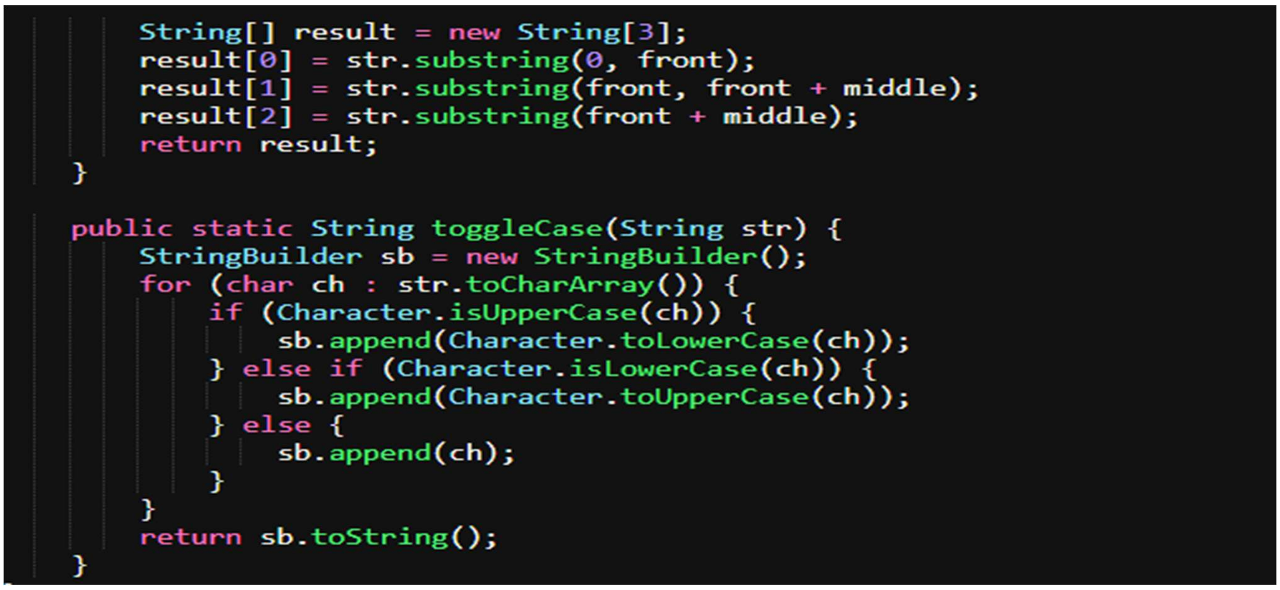
Final Result – The three output strings after applying the above three steps i.e. for the above example .
Output1 =”Jnhan”
Output2=”ohnyJan’
Output3 = “NJOARD”
Help Anand to write a program that would do the above.



```java
public class StringEncoder {

    public static void main(String[] args) {
        String input1 = "John";
        String input2 = "Johny";
        String input3 = "Janardhan";

        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);
    }

    public static String[] splitString(String str) {
        int len = str.length();
        int rem = len % 3;
        int part = len / 3;
        int front, middle, end;

        if (rem == 0) {
            front = middle = end = part;
        } else if (rem == 1) {
            front = part;
            middle = part + 1;
            end = part;
        } else {
            front = part + 1;
            middle = part;
            end = part + 1;
        }
```

Output:

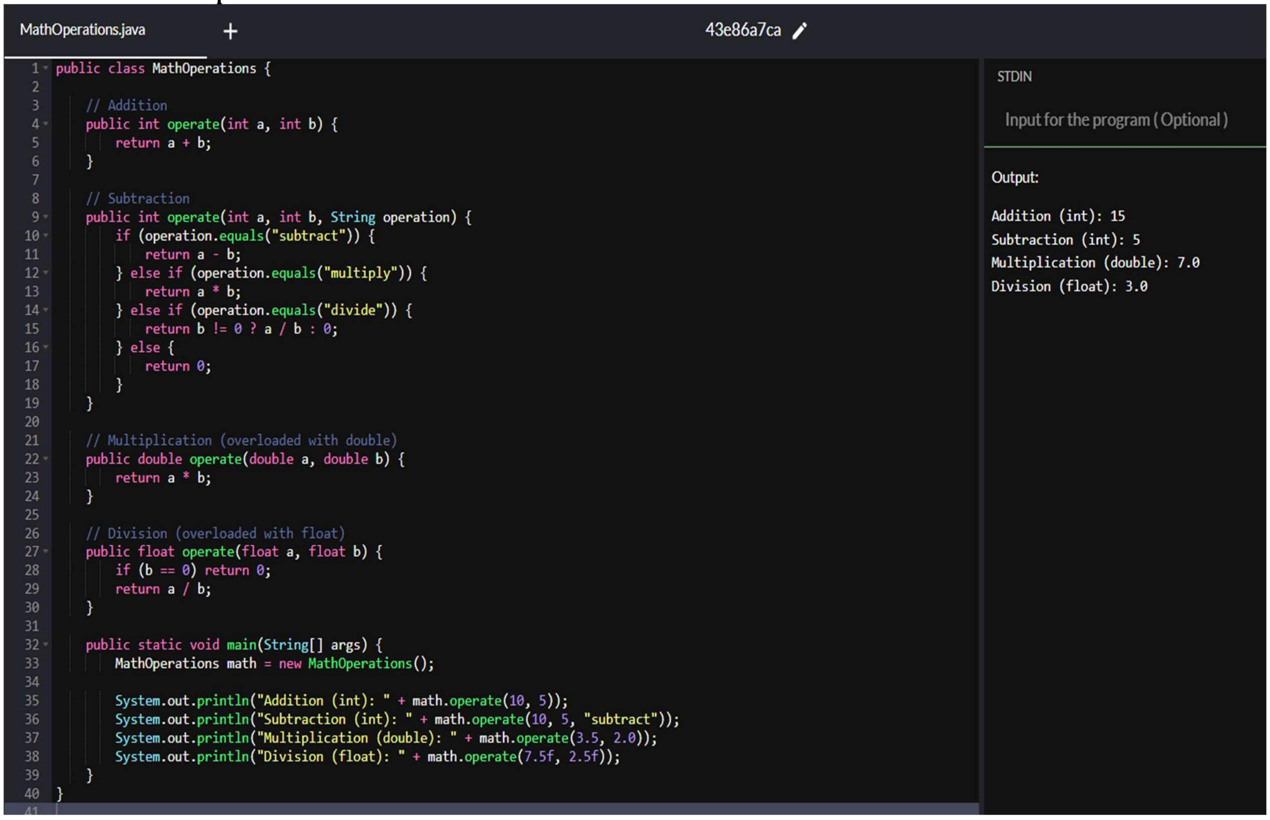
```
Output1: Jhhan
Output2: ohnyJan
Output3: NjOARD
```



```java
        String[] result = new String[3];
        result[0] = str.substring(0, front);
        result[1] = str.substring(front, front + middle);
        result[2] = str.substring(front + middle);
        return result;
    }

    public static String toggleCase(String str) {
        StringBuilder sb = new StringBuilder();
        for (char ch : str.toCharArray()) {
            if (Character.isUpperCase(ch)) {
                sb.append(Character.toLowerCase(ch));
            } else if (Character.isLowerCase(ch)) {
                sb.append(Character.toUpperCase(ch));
            } else {
                sb.append(ch);
            }
        }
        return sb.toString();
    }
```

**QUESTION 3:**

Implement a Java program that uses method overloading to perform different mathematical operations.



```java
public class MathOperations {

    // Addition
    public int operate(int a, int b) {
        return a + b;
    }

    // Subtraction
    public int operate(int a, int b, String operation) {
        if (operation.equals("subtract")) {
            return a - b;
        } else if (operation.equals("multiply")) {
            return a * b;
        } else if (operation.equals("divide")) {
            return b != 0 ? a / b : 0;
        } else {
            return 0;
        }
    }

    // Multiplication (overloaded with double)
    public double operate(double a, double b) {
        return a * b;
    }

    // Division (overloaded with float)
    public float operate(float a, float b) {
        if (b == 0) return 0;
        return a / b;
    }

    public static void main(String[] args) {
        MathOperations math = new MathOperations();

        System.out.println("Addition (int): " + math.operate(10, 5));
        System.out.println("Subtraction (int): " + math.operate(10, 5, "subtract"));
        System.out.println("Multiplication (double): " + math.operate(3.5, 2.0));
        System.out.println("Division (float): " + math.operate(7.5f, 2.5f));
    }
}
```

Output:

```
Addition (int): 15
Subtraction (int): 5
Multiplication (double): 7.0
Division (float): 3.0
```

**QUESTION 4.**

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

(i) If alphabet is present in input string, use 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, new string will be a0y.

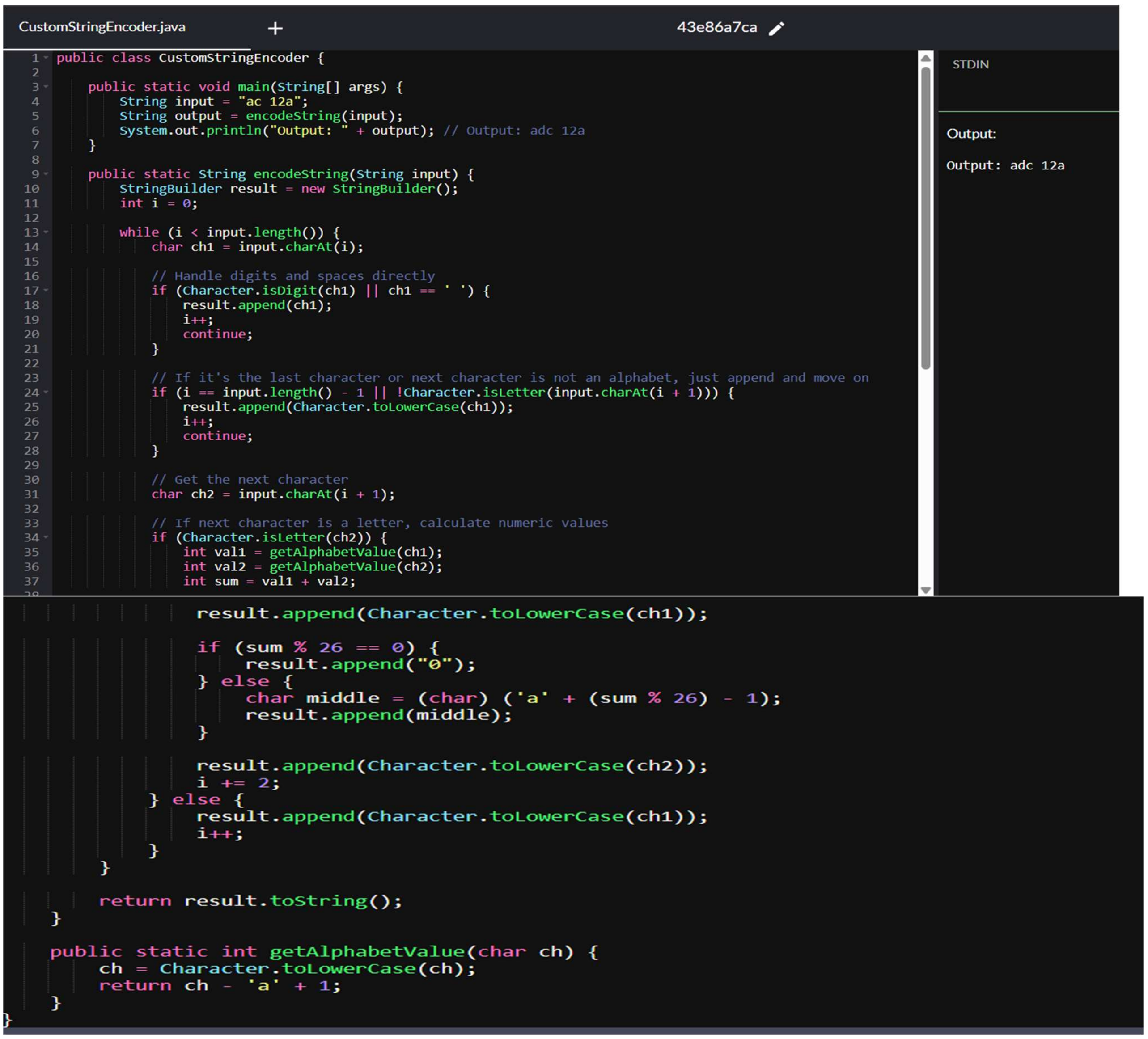
(b) Otherwise (sum of numeric value of 2 alphabets) %26 numeric value alphabet is to appended. E.g. ac is string. Numeric value of a is 1, c is 3, sum is 4. Remainder with 26 is 4. Alphabet to be appended is d. output will be adc.

(ii) If digit is present, it will be same in output string. E.g. string is 12, output string is 12.

(iii) If only single alphabet is present, it will be same in output string. E.g. input string is 1a, output will be 1a.

(iv) If space is present, it will be same in output string. E.g. string is ac 12a, output will be adc 12a.

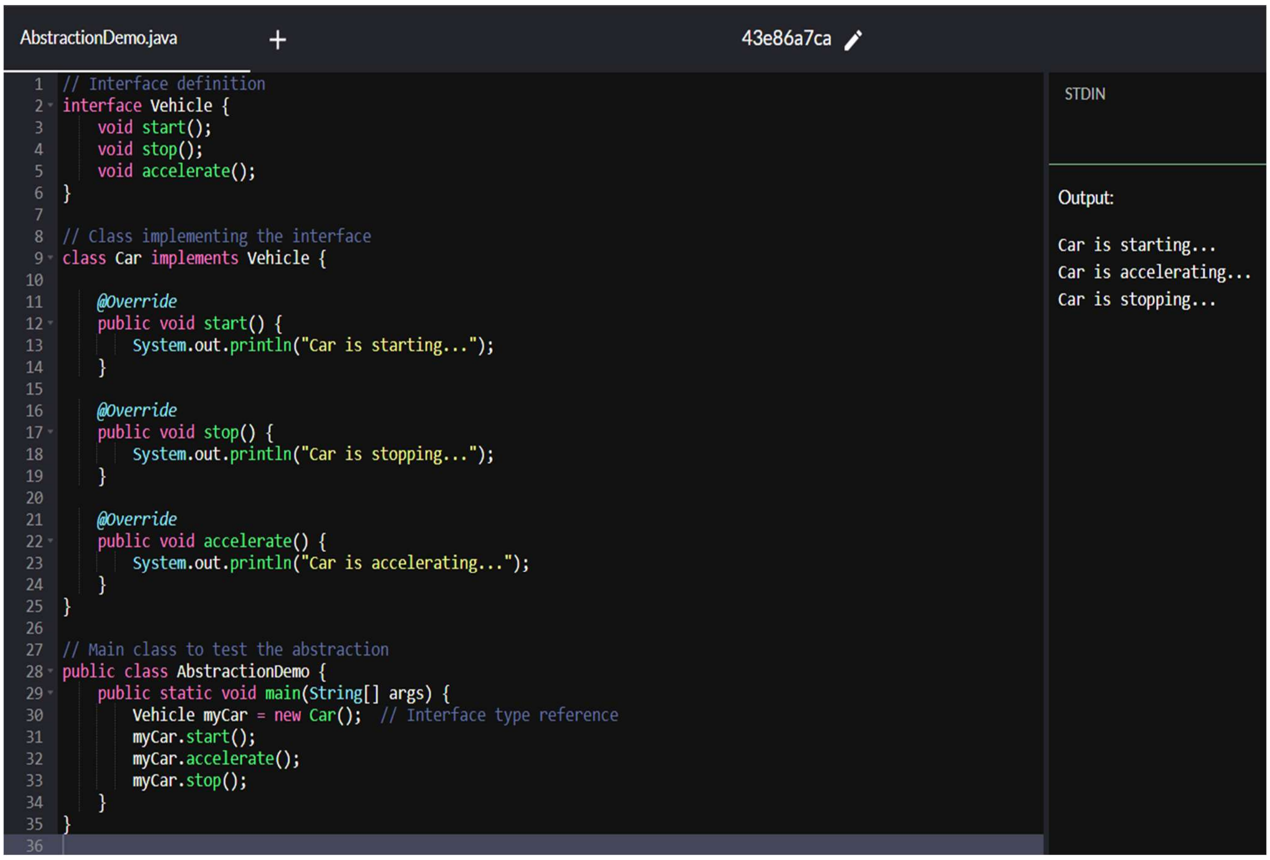
Constraint: Whether string alphabets are In Uppercase or Lowercase, appended alphabets must be in lower case. Output string must also be in lowercase.



```java
public class CustomStringEncoder {

    public static void main(String[] args) {
        String input = "ac 12a";
        String output = encodeString(input);
        System.out.println("Output: " + output); // Output: adc 12a
    }

    public static String encodeString(String input) {
        StringBuilder result = new StringBuilder();
        int i = 0;

        while (i < input.length()) {
            char ch1 = input.charAt(i);

            // Handle digits and spaces directly
            if (Character.isDigit(ch1) || ch1 == ' ') {
                result.append(ch1);
                i++;
                continue;
            }

            // If it's the last character or next character is not an alphabet, just append and move on
            if (i == input.length() - 1 || !Character.isLetter(input.charAt(i + 1))) {
                result.append(Character.toLowerCase(ch1));
                i++;
                continue;
            }

            // Get the next character
            char ch2 = input.charAt(i + 1);

            // If next character is a letter, calculate numeric values
            if (Character.isLetter(ch2)) {
                int val1 = getAlphabetValue(ch1);
                int val2 = getAlphabetValue(ch2);
                int sum = val1 + val2;
                result.append(Character.toLowerCase(ch1));

                if (sum % 26 == 0) {
                    result.append("0");
                } else {
                    char middle = (char) ('a' + (sum % 26) - 1);
                    result.append(middle);
                }

                result.append(Character.toLowerCase(ch2));
                i += 2;
            } else {
                result.append(Character.toLowerCase(ch1));
                i++;
            }
        }

        return result.toString();
    }

    public static int getAlphabetValue(char ch) {
        ch = Character.toLowerCase(ch);
        return ch - 'a' + 1;
    }
}
```

```
Output:

Output: adc 12a
```

**QUESTION 5.** Define an interface in Java and create a class that implements it, demonstrating the concept of abstraction.



```java
// Interface definition
interface Vehicle {
    void start();
    void stop();
    void accelerate();
}

// Class implementing the interface
class Car implements Vehicle {

    @Override
    public void start() {
        System.out.println("Car is starting...");
    }

    @Override
    public void stop() {
        System.out.println("Car is stopping...");
    }

    @Override
    public void accelerate() {
        System.out.println("Car is accelerating...");
    }
}

// Main class to test the abstraction
public class AbstractionDemo {
    public static void main(String[] args) {
        Vehicle myCar = new Car();  // Interface type reference
        myCar.start();
        myCar.accelerate();
        myCar.stop();
    }
}
```

Output:

```
Car is starting...
Car is accelerating...
Car is stopping...
```