Q1:

PROGRAM:

```
import java.util.Scanner;

public class StringLengthWithoutLength {

    public static int findLength(String str) {
        int count = 0;
        try {
            while (true) {
                str.charAt(count);
                count++;
            }
        } catch (IndexOutOfBoundsException e) {

        }
        return count;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = sc.next();

        int customLength = findLength(input);
        int builtInLength = input.length();

        System.out.println("Length (custom method): " + customLength);
        System.out.println("Length (built-in method): " + builtInLength);
    }
}
```

OUTPUT:

```
Enter a string: hello
Length (custom method): 5
Length (built-in method): 5
```

Q2:

PROGRAM:

```
import java.util.Scanner;

public class Split {

    public static int findLength(String str) {
        int count = 0;
        try {
            while (true) {
                str.charAt(count);
                count++;
            }
        } catch (IndexOutOfBoundsException e) {

        }
        return count;
    }


    public static String[] customSplit(String str) {
        int len = findLength(str);


        int wordCount = 0;
        boolean inWord = false;
        int[] spaceIndexes = new int[len + 1]; // Store indexes of spaces
        int spaceIdx = 0;

        for (int i = 0; i < len; i++) {
            if (str.charAt(i) == ' ') {
                spaceIndexes[spaceIdx++] = i;
                inWord = false;
            } else if (!inWord) {
                wordCount++;
                inWord = true;
            }
        }
        spaceIndexes[spaceIdx++] = len; // Add end index
```

```
    String[] words = new String[wordCount];
    int start = 0, wordIdx = 0;
    for (int i = 0; i < spaceIdx; i++) {
        int end = spaceIndexes[i];
        if (start < end) {
            StringBuilder sb = new StringBuilder();
            for (int j = start; j < end; j++) {
                sb.append(str.charAt(j));
            }
            words[wordIdx++] = sb.toString();
        }
        start = end + 1;
    }
    return words;
}


public static boolean compareArrays(String[] arr1, String[] arr2) {
    if (arr1.length != arr2.length) return false;
    for (int i = 0; i < arr1.length; i++) {
        if (!arr1[i].equals(arr2[i])) return false;
    }
    return true;
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter text: ");
    String input = sc.nextLine();

    String[] customWords = customSplit(input);
    String[] builtInWords = input.split(" ");

    System.out.println("Custom split result:");
    for (String word : customWords) {
        System.out.println(word);
    }

    System.out.println("Built-in split result:");
    for (String word : builtInWords) {
        System.out.println(word);
    }

    boolean areEqual = compareArrays(customWords, builtInWords);
```

```
        System.out.println("Are both arrays equal? " + areEqual);
    }
}
```

OUTPUT : Enter text: hello
Custom split result:
hello
Built-in split result:
hello
Are both arrays equal? true

Q3 :

```
import java.util.Scanner;

public class Splitlength {


    public static int findLength(String str) {
        int count = 0;
        try {
            while (true) {
                str.charAt(count);
                count++;
            }
        } catch (IndexOutOfBoundsException e) {
            // End of string
        }
        return count;
    }

    // Method to split text into words using charAt(), without split()
    public static String[] customSplit(String str) {
        int len = findLength(str);

        int wordCount = 0;
        boolean inWord = false;
        int[] spaceIndexes = new int[len + 1];
        int spaceIdx = 0;

        for (int i = 0; i < len; i++) {
            if (str.charAt(i) == ' ') {
```

```
            spaceIndexes[spaceIdx++] = i;
            inWord = false;
        } else if (!inWord) {
            wordCount++;
            inWord = true;
        }
    }
    spaceIndexes[spaceIdx++] = len;

    String[] words = new String[wordCount];
    int start = 0, wordIdx = 0;
    for (int i = 0; i < spaceIdx; i++) {
        int end = spaceIndexes[i];
        if (start < end) {
            StringBuilder sb = new StringBuilder();
            for (int j = start; j < end; j++) {
                sb.append(str.charAt(j));
            }
            words[wordIdx++] = sb.toString();
        }
        start = end + 1;
    }
    return words;
}

// Method to create a 2D array of word and its length as String
public static String[][] wordsWithLengths(String[] words) {
    String[][] result = new String[words.length][2];
    for (int i = 0; i < words.length; i++) {
        result[i][0] = words[i];
        result[i][1] = String.valueOf(findLength(words[i]));
    }
    return result;
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter text: ");
    String input = sc.nextLine();

    String[] words = customSplit(input);
    String[][] wordLengthArr = wordsWithLengths(words);

    System.out.println("Word\tLength");
```

```
        for (int i = 0; i < wordLengthArr.length; i++) {
            String word = wordLengthArr[i][0];
            int length = Integer.parseInt(wordLengthArr[i][1]);
            System.out.println(word + "\t" + length);
}
    sc.close();
    }
}
```

OUTPUT: Enter text: hello
Word Length
hello 5

Q4:

```
import java.util.Scanner;

public class Splitwords {

    // Method to find length without using length()
    public static int findLength(String str) {
        int count = 0;
        try {
            while (true) {
                str.charAt(count);
                count++;
            }
        } catch (IndexOutOfBoundsException e) {
            // End of string
        }
        return count;
    }

    // Method to split text into words using charAt(), without split()
    public static String[] customSplit(String str) {
        int len = findLength(str);

        int wordCount = 0;
        boolean inWord = false;
        int[] spaceIndexes = new int[len + 1];
```

```
    int spaceIdx = 0;

    for (int i = 0; i < len; i++) {
        if (str.charAt(i) == ' ') {
            spaceIndexes[spaceIdx++] = i;
            inWord = false;
        } else if (!inWord) {
            wordCount++;
            inWord = true;
        }
    }
    spaceIndexes[spaceIdx++] = len;

    String[] words = new String[wordCount];
    int start = 0, wordIdx = 0;
    for (int i = 0; i < spaceIdx; i++) {
        int end = spaceIndexes[i];
        if (start < end) {
            StringBuilder sb = new StringBuilder();
            for (int j = start; j < end; j++) {
                sb.append(str.charAt(j));
            }
            words[wordIdx++] = sb.toString();
        }
        start = end + 1;
    }
    return words;
}

// Method to create a 2D array of word and its length as String
public static String[][] wordsWithLengths(String[] words) {
    String[][] result = new String[words.length][2];
    for (int i = 0; i < words.length; i++) {
        result[i][0] = words[i];
        result[i][1] = String.valueOf(findLength(words[i]));
    }
    return result;
}

// Method to find shortest and longest word, returns their indexes in a 1D int array
public static int[] findShortestLongest(String[][] wordLengthArr) {
    int minIdx = 0, maxIdx = 0;
    int minLen = Integer.parseInt(wordLengthArr[0][1]);
    int maxLen = minLen;
```

```
        for (int i = 1; i < wordLengthArr.length; i++) {
            int len = Integer.parseInt(wordLengthArr[i][1]);
            if (len < minLen) {
                minLen = len;
                minIdx = i;
            }
            if (len > maxLen) {
                maxLen = len;
                maxIdx = i;
            }
        }
        return new int[]{minIdx, maxIdx};
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter text: ");
        String input = sc.nextLine();

        String[] words = customSplit(input);
        String[][] wordLengthArr = wordsWithLengths(words);

        System.out.println("Word\tLength");
        for (int i = 0; i < wordLengthArr.length; i++) {
            String word = wordLengthArr[i][0];
            int length = Integer.parseInt(wordLengthArr[i][1]);
            System.out.println(word + "\t" + length);
        }

        int[] minMaxIdx = findShortestLongest(wordLengthArr);
        System.out.println("Shortest word: " + wordLengthArr[minMaxIdx[0]][0] + " (Length: " +
wordLengthArr[minMaxIdx[0]][1] + ")");
        System.out.println("Longest word: " + wordLengthArr[minMaxIdx[1]][0] + " (Length: " +
wordLengthArr[minMaxIdx[1]][1] +
}
        sc.close();
    }
}
```

Q5:

```
import java.util.Scanner;

public class Vowels {

    // Method to check if a character is a vowel, consonant, or not a letter
    public static String checkCharType(char ch) {
        // Convert to lowercase using ASCII if uppercase
        if (ch >= 'A' && ch <= 'Z') {
            ch = (char)(ch + 32);
        }
        if (ch >= 'a' && ch <= 'z') {
            if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
                return "Vowel";
            } else {
                return "Consonant";
            }
        }
        return "Not a Letter";
    }

    // Method to count vowels and consonants in a string
    public static int[] countVowelsConsonants(String str) {
        int vowels = 0, consonants = 0;
        for (int i = 0; ; i++) {
            try {
                char ch = str.charAt(i);
                String type = checkCharType(ch);
                if (type.equals("Vowel")) vowels++;
                else if (type.equals("Consonant")) consonants++;
            } catch (IndexOutOfBoundsException e) {
                break;
            }
        }
        return new int[]{vowels, consonants};
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = sc.nextLine();

        int[] counts = countVowelsConsonants(input);
```

```
        System.out.println("Vowels: " + counts[0]);
        System.out.println("Consonants: " + counts[1]);
    }
}
```

OUTPUT:
Enter a string: hello
Vowels: 2
Consonants: 3

Q6:
PROGRAM :

```
import java.util.Scanner;

public class VOWELSCONSONANT {

    // Method to check if the character is a vowel, consonant, or not a letter
    public static String checkCharType(char ch) {
        // Convert to lowercase using ASCII if uppercase
        if (ch >= 'A' && ch <= 'Z') {
            ch = (char)(ch + 32);
        }
        if (ch >= 'a' && ch <= 'z') {
            if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
                return "Vowel";
            } else {
                return "Consonant";
            }
        }
        return "Not a Letter";
    }

    // Method to find vowels and consonants in a string and return a 2D array
    public static String[][] charTypes(String str) {
        int len = 0;
        try {
            while (true) {
                str.charAt(len);
                len++;
            }
        } catch (IndexOutOfBoundsException e) {
```

```
            // End of string
        }
        String[][] result = new String[len][2];
        for (int i = 0; i < len; i++) {
            char ch = str.charAt(i);
            result[i][0] = String.valueOf(ch);
            result[i][1] = checkCharType(ch);
        }
        return result;
    }

    // Method to display the 2D array in tabular format
    public static void displayCharTypes(String[][] arr) {
        System.out.println("Character\tType");
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i][0] + "\t\t" + arr[i][1]);
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = sc.nextLine();

        String[][] charTypeArr = charTypes(input);
        displayCharTypes(charTypeArr);
    }
}
```

Output:

```
Enter a string: HELLO
Vowels: 2
Consonants: 3
```