

PSG COLLEGE OF TECHNOLOGY, COIMBATORE – 641 004
Department of Applied Mathematics and Computational Sciences

MSc Software Systems – Semester V
23XW57-JAVA PROGRAMMING LAB
PROBLEM SHEET 3 – Arrays and Strings
23pw30

1. Write a program that prompts the user for an integer, then asks the user to enter that many values.

Store these values in an array and print the array. Then reverse the array elements so that the first element becomes the last element, the second element becomes the second to last element,

and so on, with the old last element now first. Do not just reverse the order in which they are printed; actually change the way they are stored in the array. Do not create a second array; just rearrange the elements within the array you have.

Code:

```
import java.util.Scanner;
public class Array {
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter array size :");
        int user = scan.nextInt();
        int [] array = new int[user];
        System.out.print("Enter array elements :");
        for(int i=0;i<user;i++){
            array[i] = scan.nextInt();
        }
        System.out.print("Array elements :");
        for(int i=0;i<user;i++){
            System.out.print(array[i]+" ");
        }

        for(int i=0,j=user-1;i<user/2;i++,j--){
            int temp = array[i];
            array[i] = array[j];
            array[j] = temp;
        }
        System.out.print("\nReversed array elements :");
        for(int i=0;i<user;i++){
            System.out.print(array[i]+" ");
        }
    }
}
```

```
}  
}
```

OUTPUT:

```
Enter array size :5  
Enter array elements :1 2 3 4 5  
Array elements :1 2 3 4 5  
Reversed array elements :5 4 3 2 1
```

2. Given a positive integer n , generate an $n \times n$ matrix filled with elements from 1 to n^2 in spiral order and print the element.

CODE:

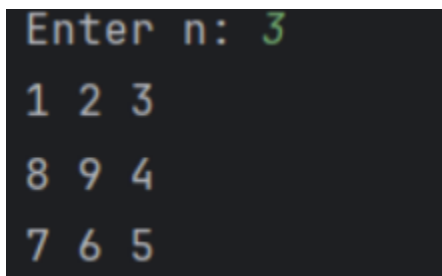
```
import java.util.Scanner;  
public class Spiral {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
        System.out.print("Enter n: ");  
        int n = scanner.nextInt();  
        int[][] matrix = generateSpiralMatrix(n);  
        for (int[] row : matrix) {  
            for (int val : row) {  
                System.out.print(val + " ");  
            }  
            System.out.println();  
        }  
    }  
    public static int[][] generateSpiralMatrix(int n) {  
        int[][] result = new int[n][n];  
        int value = 1;  
        int top = 0, bottom = n - 1;  
        int left = 0, right = n - 1;  
        while (value <= n * n) {  
            for (int i = left; i <= right; i++)  
                result[top][i] = value++;  
            top++;  
            for (int i = top; i <= bottom; i++)  
                result[i][right] = value++;
```

```

right--;
for (int i = right; i >= left; i--)
    result[bottom][i] = value++;
bottom--;
for (int i = bottom; i >= top; i--)
    result[i][left] = value++;
left++;
}
return result;
}
}

```

OUTPUT:



```

Enter n: 3
1 2 3
8 9 4
7 6 5

```

3) Write a menu driven program to do following operation on two dimensional array A of size m x n. You should use user-defined methods which accept 2-D array A, and its size m and n as arguments. The options are:

- ✓ To input elements into matrix of size m x n
- ✓ To display elements of matrix of size m x n
- ✓ Sum of all elements of matrix of size m x n
- ✓ To display row-wise sum of matrix of size m x n
- ✓ To display column-wise sum of matrix of size m x n
- ✓ To create transpose of matrix of size n x m

Code:

```

import java.util.Scanner;
public class twoD {
    public static void create2D(int arr[], int rows, int cols, Scanner
scan) {
        System.out.println("Enter 2D array:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                arr[i][j] = scan.nextInt();
            }
        }
    }
    public static void printArr(int arr[], int rows, int cols) {

```

```

System.out.println("\nOriginal Matrix:");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        System.out.print(arr[i][j] + " ");
    }
    System.out.println();
}

public static void Sum(int arr[], int rows, int cols) {
    int sum = 0;
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            sum += arr[i][j];
        }
    }
    System.out.println("\nTotal Sum of all elements: " + sum);
}

public static void Rowsum(int arr[], int rows, int cols) {

    System.out.println("\nRow-wise Sum:");
    for (int i = 0; i < rows; i++) {
        int rowsum = 0;
        for (int j = 0; j < cols; j++) {
            rowsum += arr[i][j];
        }
        System.out.println("Sum of row " + (i + 1) + " : " + rowsum);
    }
}

public static void Colsum(int arr[], int rows, int cols) {
    System.out.println("\nColumn-wise Sum:");
    for (int j = 0; j < cols; j++) {
        int colsum = 0;
        for (int i = 0; i < rows; i++) {
            colsum += arr[i][j];
        }
        System.out.println("Sum of column " + (j + 1) + " : " + colsum);
    }
}

public static void Trans(int arr[], int rows, int cols) {
    int[][] transpose = new int[cols][rows];
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            transpose[j][i] = arr[i][j];
        }
    }
}

```

```

    }
    System.out.println("\nTranspose of the Matrix:");
    for (int i = 0; i < cols; i++) {
        for (int j = 0; j < rows; j++) {
            System.out.print(transpose[i][j] + " ");
        }
        System.out.println();
    }
}

public static void main(String[] arg) {
    Scanner scan = new Scanner(System.in);
    System.out.println("Enter number of rows:");
    int rows = scan.nextInt();
    System.out.println("Enter number of columns:");
    int cols = scan.nextInt();
    int[][] arr = new int[rows][cols];
    int choice;
    do {
        System.out.println("\n--- Matrix Operations Menu ---");
        System.out.println("1. Input Matrix");
        System.out.println("2. Display Matrix");
        System.out.println("3. Sum of All Elements");

        System.out.println("4. Row-wise Sum");
        System.out.println("5. Column-wise Sum");
        System.out.println("6. Transpose");
        System.out.println("7. Exit");
        System.out.print("Enter your choice: ");
        choice = scan.nextInt();
        switch (choice) {
            case 1:
                create2D(arr, rows, cols, scan);

                break;
            case 2:
                printArr(arr, rows, cols);

                break;
            case 3:
                Sum(arr, rows, cols);

                break;
            case 4:
                Rowsum(arr, rows, cols);

```

```
break;
case 5:
    Colsum(arr, rows, cols);
break;
case 6:
    Trans(arr, rows, cols);

break;
case 7:
    System.out.println("Exiting Program.");
break;
default:
    System.out.println("Invalid Choice. Try again.");
}
} while (choice != 7);
}
```

OUTPUT:

--- Matrix Operations Menu ---

1. Input Matrix
2. Display Matrix
3. Sum of All Elements
4. Row-wise Sum
5. Column-wise Sum
6. Transpose
7. Exit

Enter your choice: 2

Original Matrix:

```
1 2 3
4 5 6
7 8 9
```

--- Matrix Operations Menu ---

1. Input Matrix
2. Display Matrix
3. Sum of All Elements
4. Row-wise Sum
5. Column-wise Sum
6. Transpose
7. Exit

4. You are given an array prices where prices[i] is the price of a given stock on the ith day. You want to maximize your profit by choosing a single day to buy one stock and choosing a different day in the future to sell that stock. Return the maximum profit you can achieve from this transaction. If you cannot achieve any profit, return 0.

Input: prices = [7,1,5,3,6,4]

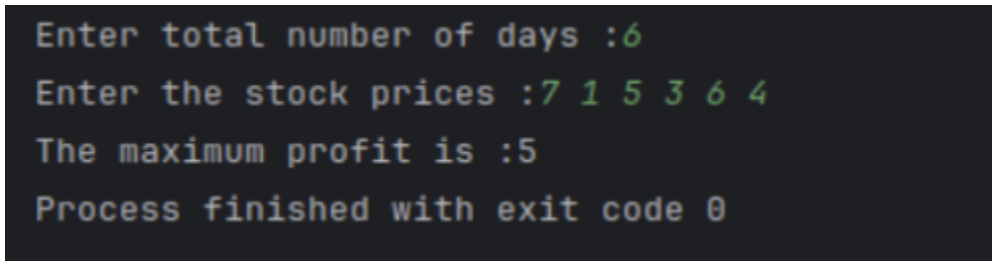
Output: 5

Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5. Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.

```
import java.util.Scanner;
public class Maxprofit {
    public static void main(String [] args) {
        Scanner scan = new Scanner(System.in);

        System.out.print("Enter total number of days :");
        int tot = scan.nextInt();
        int [] prices = new int[tot];
        System.out.print("Enter the stock prices :");
        for(int i=0;i<tot;i++){
            prices[i] = scan.nextInt();
        }
        int min =Integer.MAX_VALUE;
        int profit =0;
        for(int i=0;i<tot;i++){
            if(profit<prices[i]-min){
                profit = prices[i] - min;
            }
            if(min>prices[i]){
                min = prices[i];
            }
        }
        System.out.print("The maximum profit is :"+profit);
    }
}
```

OUTPUT:



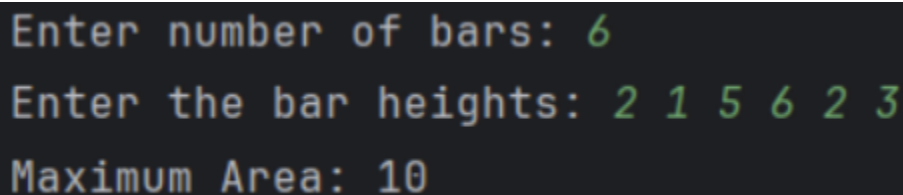
```
Enter total number of days :6
Enter the stock prices :7 1 5 3 6 4
The maximum profit is :5
Process finished with exit code 0
```

5) Given an array of integers heights representing the histogram's bar height where the width of each bar is 1, Write a program to find the area of the largest rectangle in Histogram.

Code:

```
import java.util.Scanner;
import java.util.Stack;
public class LargestRectangleHistogram {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of bars: ");
        int n = scanner.nextInt();
        int[] heights = new int[n];
        System.out.print("Enter the bar heights: ");
        for (int i = 0; i < n; i++) {
            heights[i] = scanner.nextInt();
        }
        int maxArea = largestRectangleArea(heights);
        System.out.println("Maximum Area: " + maxArea);
    }
    public static int largestRectangleArea(int[] heights) {
        Stack<Integer> stack = new Stack<>();
        int maxArea = 0;
        int n = heights.length;
        for (int i = 0; i <= n; i++) {
            int currHeight = (i == n) ? 0 : heights[i];
            while (!stack.isEmpty() && currHeight < heights[stack.peek()]) {
                int height = heights[stack.pop()];
                int width = (stack.isEmpty()) ? i : i - stack.peek() - 1;
                int area = height * width;
                maxArea = Math.max(maxArea, area);
            }
            stack.push(i);
        }
        return maxArea;
    }
}
```

A screenshot of a terminal window with a dark background. It shows the output of the Java program. The first line is "Enter number of bars: 6", the second line is "Enter the bar heights: 2 1 5 6 2 3", and the third line is "Maximum Area: 10".

```
Enter number of bars: 6
Enter the bar heights: 2 1 5 6 2 3
Maximum Area: 10
```

6) You are given a 0-indexed integer array `nums` of even length consisting of an equal number of positive and negative integers. You should rearrange the elements of `nums` such that the modified array follows the given conditions:

Every consecutive pair of integers have opposite signs.

For all integers with the same sign, the order in which they were present in `nums` is preserved. The rearranged array begins with a positive integer.

Return the modified array after rearranging the elements to satisfy the aforementioned conditions.

Code:

```
import java.util.*;

public class RearrangeBySign {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter space-separated integers (equal number of positives and negatives): ");
        String[] input = scanner.nextLine().split(" ");
        int[] nums = new int[input.length];
        for (int i = 0; i < input.length; i++) {
            nums[i] = Integer.parseInt(input[i]);
        }
        int[] result = rearrangeArray(nums);
        System.out.println("Rearranged array: " + Arrays.toString(result));
    }

    public static int[] rearrangeArray(int[] nums) {
        int n = nums.length;
        int[] result = new int[n];
        List<Integer> pos = new ArrayList<>();
        List<Integer> neg = new ArrayList<>();
        for (int num : nums) {
            if (num >= 0) pos.add(num);
            else neg.add(num);
        }
        for (int i = 0; i < n / 2; i++) {
            result[2 * i] = pos.get(i);
            result[2 * i + 1] = neg.get(i);
        }
        return result;
    }
}
```

OUTPUT:

```
Enter space-separated integers (equal number of positives and negatives): 1 1 -1 -2 6 -9 6 4 -8
Rearranged array: [1, -1, 1, -2, 6, -9, 6, -8, 0]
```

7) Zeller's congruence is an algorithm developed by Christian Zeller to calculate the day of the week. The formula is,

where,

- h is the day of the week (0: Saturday, 1: Sunday, 2: Monday, 3: Tuesday, 4: Wednesday, 5: Thursday, 6: Friday).
- q is the day of the month.
- m is the month (3: March, 4: April, ..., 12: December). January and February are counted as months 13 and 14 of the previous year.
- j is the century (i.e., year / 100).

- k is the year of the century (i.e., year % 100).

Note that the division in the formula performs an integer division. Write a program that prompts the user to enter a year, month, and day of the month, and displays the name of the day of the week. Here are some sample runs:

Code:

```
import java.util.Scanner;
public class ZellerCongruence {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter year (e.g., 2024): ");
        int year = scan.nextInt();
        System.out.print("Enter month (1-12): ");
        int month = scan.nextInt();
        System.out.print("Enter day of the month (1-31): ");
        int day = scan.nextInt();
        if (month == 1 || month == 2) {
            month += 12;
            year -= 1;
        }
        int q = day;
        int m = month;
        int k = year % 100;
        int j = year / 100;
        int h = (q + (13 * (m + 1)) / 5 + k + (k / 4) + (j / 4) + 5 * j) %
        7;
        String[] days = {"Saturday", "Sunday", "Monday", "Tuesday",
            "Wednesday", "Thursday", "Friday"};
        System.out.println("The day of the week is: " + days[h]);
    }
}
```

OUTPUT:

```
Enter year (e.g., 2024): 2015
Enter month (1-12): 1
Enter day of the month (1-31): 25
The day of the week is: Sunday
```

8. Create a jagged array of integers. This array should consist of two 2-D arrays. First 2-D array should contain 3 rows having length of 4, 3 and 2, respectively. Second 2-D array should contain 2 rows with length 3 and 4, respectively. Initialize array with suitable elements and display them.

Code:

```
public class JaggedArray{
    public static void main(String[] args) {
        int[][] firstArray = {
            {1, 2, 3, 4},
            {5, 6, 7},
            {8, 9}
        };
        int[][] secondArray = {
            {10, 11, 12},
            {13, 14, 15, 16}
        };
        System.out.println("First 2-D Array:");
        for (int i = 0; i < firstArray.length; i++) {
            for (int j = 0; j < firstArray[i].length; j++) {
                System.out.print(firstArray[i][j] + " ");
            }
            System.out.println();
        }
        System.out.println("\nSecond 2-D Array:");
        for (int i = 0; i < secondArray.length; i++) {
            for (int j = 0; j < secondArray[i].length; j++) {
                System.out.print(secondArray[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

OUTPUT:

```
First 2-D Array:
1 2 3 4
5 6 7
8 9

Second 2-D Array:|
10 11 12
13 14 15 16
```

9. Each week, the Pickering Trucking Company randomly selects one of its 30 employees to take

a drug test. Write an application that determines which employee will be selected each week for the next 52 weeks. Use the `Math.random()` function to generate an employee number between 1 and 30; you use a statement similar to:

```
testedEmployee = 1 + (int) (Math.random() * 30);
```

After each selection, display the number of the employee to test. Display four employee numbers on each line. It is important to note that if testing is random, some employees will be tested multiple times, and others might never be tested. Run the application several times until you are confident that the selection is random.

Code :

```
import java.util.Scanner;
import java.lang.Math;
public class Test {
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        System.out.println("Employees tested");
        for(int i=1;i<=52;i++){
            int testedEmployee = 1 + (int)(Math.random()*30);
            System.out.print(testedEmployee+ " ");
            if(i%4==0){
                System.out.println("");
            }
        }
    }
}
```

Output:

Employees tested

5 13 8 20

10 11 10 10

5 17 27 12

13 22 7 3

17 14 11 3

12 12 6 7

8 19 6 1

10. Write a program that inputs a line of text, tokenizes the line with String method split and outputs the tokens in reverse order. Use space characters as delimiters.

```
import java.util.Scanner;
import java.lang.String;
public class ReverseString {
    public static void main(String[] args){
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter a line of strings :");
        String line = scan.nextLine();
        String[] tokens = line.split(" ");
        for(int i = tokens.length-1;i>=0;i--){
            System.out.print(tokens[i]+" ");
        }
    }
}
```

OUTPUT:

```
Enter a line of strings :Enter a line of strings
strings of line a Enter
```

11)

11. Write an application that prompts the user for a password that contains at least two uppercase letters, at least three lowercase letters, and at least one digit. Continuously re-prompt the user until a valid password is entered. Display a message indicating whether the password is valid; if not, display the reason the password is not valid.

```
import java.util.Scanner;
public class Password {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
```

```

boolean valid = false;
while (!valid) {
    System.out.print("Enter a password: ");
    String password = scanner.nextLine();
    int upperCount = 0;
    int lowerCount = 0;
    int digitCount = 0;
    for (int i = 0; i < password.length(); i++) {
        char ch = password.charAt(i);
        if (Character.isUpperCase(ch)) {
            upperCount++;
        } else if (Character.isLowerCase(ch)) {
            lowerCount++;
        } else if (Character.isDigit(ch)) {
            digitCount++;
        }
    }
    if (upperCount >= 2 && lowerCount >= 3 && digitCount >= 1) {
        valid = true;
        System.out.println("Password is valid!");
    } else {
        System.out.println("Password is not valid. Please fix the
        following:");
        if (upperCount < 2) {
            System.out.println("- Must contain at least 2 uppercase
            letters.");
        }
        if (lowerCount < 3) {
            System.out.println("- Must contain at least 3 lowercase
            letters.");
        }
        if (digitCount < 1) {
            System.out.println("- Must contain at least 1 digit.");
        }
    }
}
}

```

OUTPUT:

```
Enter a password: shyam
Password is not valid. Please fix the following:
Must contain at least 2 uppercase letters.
- Must contain at least 1 digit.
Enter a password: shyam@123
Password is not valid. Please fix the following:
Must contain at least 2 uppercase letters.
Enter a password: Shyam@123
Password is not valid. Please fix the following:
Must contain at least 2 uppercase letters.
Enter a password: SHyam@123
Password is valid!
```

12)

12. An anagram is a word or a phrase made by transposing the letters of another word or phrase;

for example, "parliament" is an anagram of "partial men," and "software" is an anagram of "swear oft." Write a program that figures out whether one string is an anagram of another string. The program should ignore white space and punctuation.

```
import java.util.Scanner;
public class Anagram {
    public static void main(String [] args){
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter the first string");
        String s1 = scan.nextLine();
        System.out.print("Enter the second string");
        String s2 = scan.nextLine();
        int [] alpha = new int[26];
        for(int i=0;i<s1.length();i++){
            if(s1.charAt(i)<'a' || s1.charAt(i)>'z'){
                continue;
            }
            int index = s1.charAt(i) - 'a';
            alpha[index] +=1;
        }
        for(int i=0;i<s2.length();i++){
            if(s2.charAt(i)<'a' || s2.charAt(i)>'z'){
```



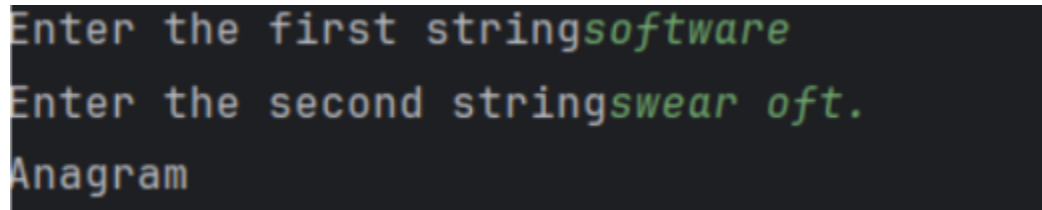
```

continue;
}
int index = s2.charAt(i) - 'a';
alpha[index] -= 1;
}
for(int i:alpha){
if(i!=0){

System.out.println("Not an anagram");
return;
}
}
System.out.println("Anagram");
}
}

```

OUTPUT:



```

Enter the first stringsoftware
Enter the second stringswear oft.
Anagram

```

13)

13. Write program in Java for String handling which perform followings

- i) Checks the capacity of StringBuffer objects
- ii) Reverse the contents of a string given on console and convert the resultant string in upper case
- iii) Read a string from console and append it to the resultant string of ii.

```

import java.util.Scanner;
public class Buff {
public static void main(String[] args){
Scanner scan = new Scanner(System.in);
System.out.print("Enter a string :");
String str = scan.nextLine();
StringBuffer strb = new StringBuffer(str);
System.out.println("Capacity of the buffer :"+strb.capacity());
strb.reverse();
String reverseUpper = strb.toString().toUpperCase();
System.out.println("Reverse of the string :"+reverseUpper);
System.out.println("Enter a string to append");
String str1 = scan.nextLine();

```

```
String append = reverseUpper + str1;
System.out.println("Final concatenated string :"+append);
}
}
```

OUTPUT:

```
Enter a string :java problem sheet 3
Capacity of the buffer :36
Reverse of the string :3 TEEHS MELBORP AVAJ
Enter a string to append
completed
Final concatenated string :3 TEEHS MELBORP AVAJcompleted
```

14)

14. Write an application that accepts a word from a user and converts it to Pig Latin. If a word starts with a consonant, the Pig Latin version removes all consonants from the beginning of the word and places them at the end, followed by ay. For example, cricket becomes icketcray. If a word starts with a vowel, the Pig Latin version is the original word with ay added to the end.

For example, apple becomes appeley. If y is the first letter in a word, it is treated as a consonant;

otherwise, it is treated as a vowel. For example, young becomes oungeyay, but system becomes ystemsay. For this program, assume that the user will enter only a single word consisting of all lowercase letters.

```
import java.util.Scanner;
public class PigLatinConverter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a word: ");
        String word = scanner.nextLine();
        String pigLatin = convertToPigLatin(word);
        System.out.println("Pig Latin: " + pigLatin);
    }
    public static String convertToPigLatin(String word) {
        String vowels = "aeiou";
        if (vowels.indexOf(word.charAt(0)) != -1) {
            return word + "ay";
        }
        for (int i = 0; i < word.length(); i++) {
            char c = word.charAt(i);
            if (vowels.indexOf(c) != -1 || (c == 'y' && i != 0)) {
                return word.substring(i) + word.substring(0, i) + "ay";
            }
        }
    }
}
```

```

}
return word + "ay";
}
}

```

OUTPUT:

Enter a word: cricket

Pig Latin: icketcray

15. Write a program inputs a line of text and uses String method to determine the total number of occurrences of each letter of the alphabet in the text. Uppercase and lowercase letters should be counted together. Store the totals for each letter in an array, and print the values in tabular format after the totals have been determined.

```

import java.util.Scanner;
public class Occurence {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scan.nextLine();
        int[] counter = new int[26];
        for (int i = 0; i < str.length(); i++) {

            char letter = str.charAt(i);
            if (letter >= 'a' && letter <= 'z') {
                counter[letter - 'a']++;
            } else if (letter >= 'A' && letter <= 'Z') {
                counter[letter - 'A']++;
            }
        }
        for (int i = 0; i < 26; i++) {
            System.out.print((char)('a' + i) + ":" + counter[i] + " ");
            if ((i + 1) % 4 == 0) {
                System.out.println();
            }
        }
    }
}

```

OUTPUT:

```

Enter a string: hello
a:0 b:0 c:0 d:0
e:1 f:0 g:0 h:1
i:0 j:0 k:0 l:2
m:0 n:0 o:1 p:0
q:0 r:0 s:0 t:0
u:0 v:0 w:0 x:0
y:0 z:0

```

16) Biologists use a sequence of the letters A, C, T, and G to model a genome. A gene is a substring of a genome that starts after a triplet ATG and ends before a triplet TAG, TAA, or TGA. Furthermore, the length of a gene string is a multiple of 3, and the gene does not contain any of the triplets ATG, TAG, TAA, or TGA. Write a program that prompts the user to enter a genome and displays all genes in the genome. If no gene is found in the input sequence, display

“no gene is found”. Here are the sample runs:

Code:

```

import java.util.Scanner;
public class GenomeParser {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter genome sequence: ");
        String genome = scanner.nextLine();
        boolean found = false;
        for (int i = 0; i < genome.length() - 2; i++) {
            if (genome.substring(i, i + 3).equals("ATG")) {

                int start = i + 3;
                for (int j = start + 3; j <= genome.length() - 3; j += 3) {
                    String codon = genome.substring(j, j + 3);

                    if (codon.equals("TAG") || codon.equals("TAA") ||

                        codon.equals("TGA")) {
                        String gene = genome.substring(start, j);

                        if (gene.length() % 3 == 0 &&

```

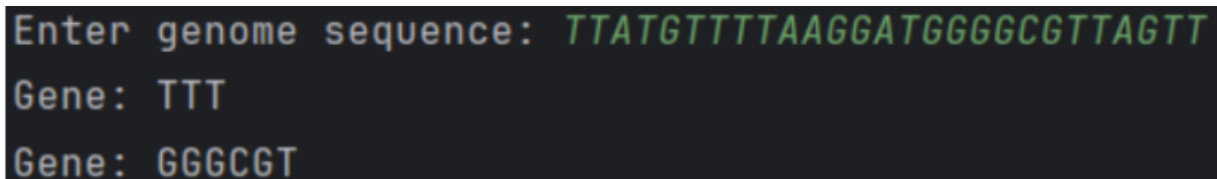
```

!containsStopOrStart(gene)) {
System.out.println("Gene: " + gene);
found = true;
}
break;

}
}
}
}
if (!found) System.out.println("no gene is found");
}
public static boolean containsStopOrStart(String gene) {
for (int i = 0; i <= gene.length() - 3; i += 3) {
String triplet = gene.substring(i, i + 3);
if (triplet.equals("ATG") || triplet.equals("TAG") ||
triplet.equals("TAA") || triplet.equals("TGA")) {
return true;
}
}
return false;
}
}
}

```

Output:



```

Enter genome sequence: TTATGTTTTAAGGATGGGGCGTTAGTT
Gene: TTT
Gene: GGGCGT

```

17. Write a program to find the number of valid words in a given sentence. A sentence consists of

lowercase letters ('a' to 'z'), digits ('0' to '9'), hyphens ('-'), punctuation marks ('!', '.', and ','), and spaces (' ') only. Each sentence can be broken down into one or more tokens separated by one or more spaces ' '.

A token is a valid word if all three of the following are true:

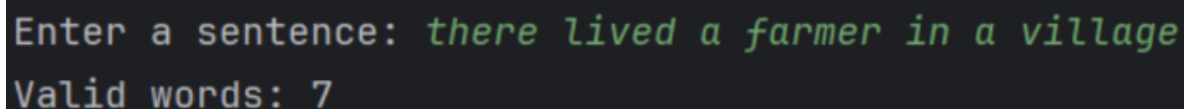
- It only contains lowercase letters, hyphens, and/or punctuation (no digits).
- There is at most one hyphen '-'. If present, it must be surrounded by lowercase characters ("a-b" is valid, but "-ab" and "ab-" are not valid).
- There is at most one punctuation mark. If present, it must be at the end of the token ("ab,", "cd!", and "." are valid, but "a!b" and "c.," are not valid).

Examples of valid words include "a-b.", "afad", "ba-c", "a!", and "!".

Given a string sentence, return the number of valid words in sentence.

```
import java.util.Scanner;
public class ValidWordCounter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a sentence: ");
        String sentence = scanner.nextLine();
        String[] tokens = sentence.trim().split("\\s+");
        int count = 0;
        for (String token : tokens) {
            if (isValid(token)) count++;
        }
        System.out.println("Valid word count: " + count);
    }
    public static boolean isValid(String word) {
        int hyphen = 0, punctuation = 0;
        int len = word.length();
        for (int i = 0; i < len; i++) {
            char c = word.charAt(i);
            if (Character.isDigit(c)) return false;
            if (c == '-') {
                hyphen++;
                if (hyphen > 1 || i == 0 || i == len - 1) return false;
                if (!Character.isLowerCase(word.charAt(i - 1)) ||
                    !Character.isLowerCase(word.charAt(i + 1)))
                    return false;
            } else if (c == '!' || c == '.' || c == ',') {
                punctuation++;
                if (punctuation > 1 || i != len - 1) return false;
            } else if (!Character.isLowerCase(c)) {
                return false;
            }
        }
        return true;
    }
}
```

Output:

A screenshot of a terminal window with a dark background. It shows the program's output for the input sentence "there lived a farmer in a village". The first line is "Enter a sentence: there lived a farmer in a village" and the second line is "Valid words: 7".

```
Enter a sentence: there lived a farmer in a village
Valid words: 7
```

18. The international standard letter/number mapping found on the telephone is:

Write a method that returns a number, given an uppercase letter, as follows: `public static int getNumber(char uppercaseLetter)`

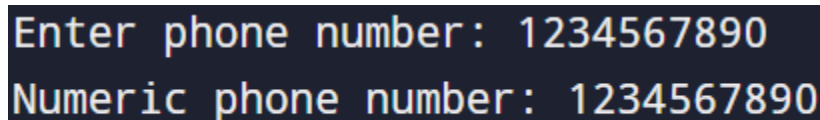
Write a test program that prompts the user to enter a phone number as a string. The input number may contain letters. The program translates a letter (upper- or lowercase) to a digit and leaves all other characters intact. Here is a sample run of the program:

```
import java.util.Scanner;

public class PhoneNumberTranslator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter phone number: ");
        String input = scanner.nextLine();
        StringBuilder result = new StringBuilder();
        for (char c : input.toCharArray()) {
            if (Character.isLetter(c)) {
                result.append(getNumber(Character.toUpperCase(c)));
            } else {
                result.append(c);
            }
        }
        System.out.println("Numeric phone number: " + result);
    }

    public static int getNumber(char letter) {
        if ("ABC".indexOf(letter) >= 0) return 2;
        if ("DEF".indexOf(letter) >= 0) return 3;
        if ("GHI".indexOf(letter) >= 0) return 4;
        if ("JKL".indexOf(letter) >= 0) return 5;
        if ("MNO".indexOf(letter) >= 0) return 6;
        if ("PQRS".indexOf(letter) >= 0) return 7;
        if ("TUV".indexOf(letter) >= 0) return 8;
        if ("WXYZ".indexOf(letter) >= 0) return 9;
        return 0;
    }
}
```

Output:

A screenshot of a terminal window with a dark background. It shows two lines of text: "Enter phone number: 1234567890" and "Numeric phone number: 1234567890". The text is in a light-colored, monospaced font.

```
Enter phone number: 1234567890
Numeric phone number: 1234567890
```