Assignment-2

1. Printing Patterns

Problem: Write a Java program to print patterns such as a right triangle of stars.

Test Cases:

Input: n = 3

Output:

*

**

***

Input: n = 5

Output:

*

**

***

****

*****

ANS:

```
import java.util.Scanner;

public class StarPattern {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input number of rows
        System.out.print("Enter the number of rows: ");
        int n = scanner.nextInt();

        // Declare 2D array
        char[][] stars = new char[n][n];
```

```
        // Fill the array to create the pattern
        for (int i = 0; i < n; i++) {
            for (int j = 0; j <= i; j++) {
                stars[i][j] = '*';
            }
        }

        // Print the pattern
        for (int i = 0; i < n; i++) {
            for (int j = 0; j <= i; j++) {
                System.out.print(stars[i][j] + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}
```

2. Remove Array Duplicates

Problem: Write a Java program to remove duplicates from a sorted array and return the new length of the array.

Test Cases:

Input: arr = [1, 1, 2]

Output: 2

Input: arr = [0, 0, 1, 1, 2, 2, 3, 3]

Output: 4

ANS:

```
import java.util.Scanner;

public class RemoveDuplicates {
```

```
public static int removeDuplicates(int[] arr) {
    if (arr.length == 0) return 0;

    // Pointer for the new length of the array
    int newLength = 1;

    // Traverse the array, compare current element with the previous one
    for (int i = 1; i < arr.length; i++) {
        if (arr[i] != arr[i - 1]) {
            arr[newLength] = arr[i];
            newLength++;
        }
    }
    return newLength;
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    // Input the array size
    System.out.print("Enter the size of the array: ");
    int size = scanner.nextInt();

    // Declare and input the array elements
    int[] arr = new int[size];
    System.out.println("Enter the sorted array elements:");
    for (int i = 0; i < size; i++) {
        arr[i] = scanner.nextInt();
    }

    // Call the method to remove duplicates and get the new length
    int newLength = removeDuplicates(arr);
```

```
        // Output the new array length and elements
        System.out.println("New length: " + newLength);
        System.out.print("Array after removing duplicates: ");
        for (int i = 0; i < newLength; i++) {
            System.out.print(arr[i] + " ");
        }

        scanner.close();
    }
}
```

3. Remove White Spaces from String

Problem: Write a Java program to remove all white spaces from a given string.

Test Cases:

Input: "Hello World"

Output: "HelloWorld"

Input: " Java   Programming "

ANS:

```
import java.util.Scanner;

public class RemoveWhiteSpaces {
    public static String removeWhiteSpaces(String str) {
        // Using the built-in replaceAll() method to remove all white spaces
        return str.replaceAll("\\s+", "");
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the string
        System.out.print("Enter the string: ");
        String input = scanner.nextLine();
```

```
        // Call the method to remove white spaces
        String result = removeWhiteSpaces(input);

        // Output the string after removing white spaces
        System.out.println("String after removing white spaces: " + result);

        scanner.close();
    }
}
```

4. Reverse a String

Problem: Write a Java program to reverse a given string.

Test Cases:

Input: "hello"

Output: "olleh"

Input: "Java"

Output: "avaJ"

```
import java.util.Scanner;

public class ReverseString {
    public static String reverse(String str) {
        String reversed = "";
        for (int i = str.length() - 1; i >= 0; i--) {
            reversed += str.charAt(i);  // Append characters in reverse order
        }
        return reversed;
    }

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

```
        // Input the string
        System.out.print("Enter the string: ");
        String input = scanner.nextLine();

        // Call the method to reverse the string
        String result = reverse(input);

        // Output the reversed string
        System.out.println("Reversed string: " + result);

        scanner.close();
    }
}
```

5. Reverse Array in Place

Problem: Write a Java program to reverse an array in place.

Test Cases:

Input: arr = [1, 2, 3, 4]

Output: [4, 3, 2, 1]

Input: arr = [7, 8, 9]

Output: [9, 8, 7]

ANS:

```
import java.util.Scanner;

public class ReverseArrayInPlace {
    public static void reverseArray(int[] arr) {
        int left = 0;
        int right = arr.length - 1;

        // Swap elements from both ends until middle is reached
        while (left < right) {
            int temp = arr[left];
            arr[left] = arr[right];
```

```
            arr[right] = temp;

            left++;
            right--;
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the array size
        System.out.print("Enter the size of the array: ");
```

6. Reverse Words in a String

Problem: Write a Java program to reverse the words in a given sentence.

Test Cases:

Input: "Hello World"

Output: "World Hello"

Input: "Java Programming"

Output: "Programming Java"

ANS:

```
import java.util.Scanner;

public class ReverseWordsInString {
    public static String reverseWords(String str) {
        // Split the sentence into words using space as a delimiter
        String[] words = str.split(" ");

        // Initialize an empty string to hold the result
        String reversedSentence = "";

        // Traverse the array of words in reverse order and append each word to the result
```

```
        for (int i = words.length - 1; i >= 0; i--) {
            reversedSentence += words[i];
            if (i != 0) {
                reversedSentence += " ";  // Add a space between words (except after the last word)
            }
        }

        return reversedSentence;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the sentence
        System.out.print("Enter the sentence: ");
        String input = scanner.nextLine
```

7. Reverse a Number

Problem: Write a Java program to reverse a given number.

Test Cases:

Input: 12345

Output: 54321

Input: -9876

Output: -6789

ANS:

```
import java.util.Scanner;

public class ReverseNumber {
    public static int reverse(int num) {
        boolean isNegative = num < 0;  // Check if the number is negative
        num = Math.abs(num);  // Make the number positive for easier processing
```

```
        int reversed = 0;
        while (num != 0) {
            int digit = num % 10;  // Get the last digit
            reversed = reversed * 10 + digit;  // Append the digit to the reversed number
            num /= 10;  // Remove the last digit from the original number
        }

        return isNegative ? -reversed : reversed;  // Restore the negative sign if needed
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the number
        System.out.print("Enter a number: ");
        int input = scanner.nextInt();

        // Call the method to reverse the number
        int result = reverse(input);

        // Output the reversed number
        System.out.println("Reversed number: " + result);

        scanner.close();
    }
}
```

8. Array Manipulation

Problem: Perform a series of operations to manipulate an array based on range update queries. Each query adds a value to a range of indices.

Test Cases:

Input: n = 5, queries = [[1, 2, 100], [2, 5, 100], [3, 4, 100]]

Output: 200

Input: n = 4, queries = [[1, 3, 50], [2, 4, 70]]

Output: 120

ANS:

```
import java.util.Scanner;


public class ArrayManipulation {
    public static long arrayManipulation(int n, int[][] queries) {
        long[] arr = new long[n + 1]; // Create an array with an extra space for easy manipulation

        // Apply the difference array technique for each query
        for (int[] query : queries) {
            int start = query[0];   // Start index of the range
            int end = query[1];     // End index of the range
            long value = query[2];  // Value to add

            arr[start] += value;    // Add value at the start index
            if (end + 1 <= n) {
                arr[end + 1] -= value; // Subtract value just after the end index
            }
        }

        long max = 0, current = 0;

        // Compute the final values in the array
        for (int i = 1; i <= n; i++) {
            current += arr[i]; // Calculate the current value
            if (current > max) {
                max = current; // Update max if current is greater
            }
        }
```

```
        return max; // Return the maximum value in the array
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the size of the array
        System.out.print("Enter the size of the array: ");
        int n = scanner.nextInt();

        // Input the number of queries
        System.out.print("Enter the number of queries: ");
        int q = scanner.nextInt();

        int[][] queries = new int[q][3]; // Create an array to hold the queries

        // Input each query
        for (int i = 0; i < q; i++) {
            System.out.print("Enter query " + (i + 1) + " (start end value): ");
            queries[i][0] = scanner.nextInt();
            queries[i][1] = scanner.nextInt();
            queries[i][2] = scanner.nextInt();
        }

        // Perform the array manipulation and output the result
        long result = arrayManipulation(n, queries);
        System.out.println("Maximum value after queries: " + result);

        scanner.close();
    }
}
```

9. String Palindrome

Problem: Write a Java program to check if a given string is a palindrome.

Test Cases:

Input: "madam"

Output: true

Input: "hello"

Output: false

Here’s a continuation of the list of assignment questions starting from question 21, with two test cases for each:

ANS:

```
import java.util.Scanner;

public class StringPalindrome {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the string to check
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();

        // Check if the string is a palindrome
        boolean isPalindrome = checkPalindrome(input);

        // Output the result
        if (isPalindrome) {
            System.out.println("Output: true");
        } else {
            System.out.println("Output: false");
        }

        scanner.close(); // Close the scanner
    }
```

```
    // Method to check if a string is a palindrome
    public static boolean checkPalindrome(String str) {
        int left = 0; // Start index
        int right = str.length() - 1; // End index

        while (left < right) {
            // Compare characters from both ends
            if (str.charAt(left) != str.charAt(right)) {
                return false; // Not a palindrome
            }
            left++; // Move towards the center
            right--; // Move towards the center
        }

        return true; // It's a palindrome
    }
}
```

10. Array Left Rotation

Problem: Write a Java program to rotate an array to the left by d positions.

Test Cases:

Input: arr = [1, 2, 3, 4, 5], d = 2

Output: [3, 4, 5, 1, 2]

Input: arr = [10, 20, 30, 40], d = 1

Output: [20, 30, 40, 10]

ANS:

```
import java.util.Scanner;
import java.util.Arrays;
```

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

        // Step 1: Get the size of the array
        System.out.print("Enter the size of the array: ");
        int n = scanner.nextInt();

        // Step 2: Create the array and get elements from the user
        int[] arr = new int[n];
        System.out.print("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt(); // Fill the array
        }

        // Step 3: Get the number of positions to rotate
        System.out.print("Enter the number of positions to rotate left: ");
        int d = scanner.nextInt();

        // Step 4: Call the rotation method and get the result
        int[] rotatedArray = leftRotate(arr, d);

        // Step 5: Print the result
        System.out.println("Rotated Array: " + Arrays.toString(rotatedArray));

        scanner.close(); // Close the scanner
    }

    // Method to rotate the array to the left by d positions
    public static int[] leftRotate(int[] arr, int d) {
        int n = arr.length;

        // Step 6: Adjust d in case it's greater than n
```

```
        d = d % n; // This ensures we don't rotate more than necessary

        // Step 7: Create a new array for the rotated values
        int[] rotated = new int[n];

        // Step 8: Copy elements from the original array to the new array
        for (int i = 0; i < n - d; i++) {
            rotated[i] = arr[i + d]; // Copying elements after d positions
        }

        // Step 9: Copy the first d elements to the end of the new array
        for (int i = 0; i < d; i++) {
            rotated[n - d + i] = arr[i]; // Moving first d elements to the end
        }

        // Step 10: Return the rotated array
        return rotated;
    }
}
```