C-DAC Mumbai

Date 26/09/2024

Subject: Algorithm and Data Structure
Assignment 1

Solve the assignment with following thing to be added in each question.

-Program
-Flow chart
-Explanation
-Output
-Time and Space complexity

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:
```
*
**
***
****
```

```
import java.util.Scanner;

public class StarPattern {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of rows: ");
        int n = sc.nextInt();

        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```
==

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

```
import java.util.Arrays;

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

        int uniqueCount = 1;
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] != arr[i - 1]) {
                arr[uniqueCount] = arr[i];
                uniqueCount++;
            }
        }
        return uniqueCount;
    }

    public static void main(String[] args) {
        int[] arr = {1, 1, 2};
        int newLength = removeDuplicates(arr);

        System.out.println("New array length: " + newLength);
        System.out.println("Array after removing duplicates: " + Arrays.toString(Arrays.copyOf(arr, 
newLength)));
    }
}
```
===
=========

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 "
Output: "JavaProgramming"

```
import java.util.Scanner;

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

        // Remove all white spaces using replaceAll
        String result = input.replaceAll("\\s", "");

        System.out.println("String after removing white spaces: " + result);
    }
}
```
===

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

        // Reverse the string using StringBuilder
        String reversed = new StringBuilder(input).reverse().toString();

        System.out.println("Reversed string: " + reversed);
    }
}
```
==
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]

```
import java.util.Arrays;

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

        // Reverse the array in place
        int start = 0;
        int end = arr.length - 1;

        while (start < end) {
            // Swap arr[start] and arr[end]
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;

            start++;
            end--;
        }

        // Output the reversed array
        System.out.println("Reversed array: " + Arrays.toString(arr));
    }
}
```
==

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"

7. Reverse a Number
Problem: Write a Java program to reverse a given number.

Test Cases:

Input: 12345
Output: 54321
Input: -9876
Output: -6789

```
import java.util.Scanner;

public class ReverseNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int input = sc.nextInt();

        // Variable to hold the reversed number
        int reversed = 0;
        int original = input; // Store the original number to check for negativity

        // Handle negative numbers
        input = Math.abs(input);

        // Reverse the number
        while (input != 0) {
            int digit = input % 10; // Get the last digit
            reversed = reversed * 10 + digit; // Build the reversed number
            input /= 10; // Remove the last digit from the input
        }

        // Restore the negative sign if the original number was negative
        if (original < 0) {
            reversed = -reversed;
        }

        System.out.println("Reversed number: " + reversed);
    }
}
```

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

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:

```
import java.util.Scanner;

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

        // Remove spaces and convert to lowercase for case-insensitive comparison
        String cleanedInput = input.replaceAll("\\s+", "").toLowerCase();

        // Check if the cleaned string is a palindrome
        boolean isPalindrome = isPalindrome(cleanedInput);

        System.out.println("Is the string a palindrome? " + isPalindrome);
    }

    // Helper method to check palindrome
    private static boolean isPalindrome(String str) {
        int start = 0;
        int end = str.length() - 1;

        while (start < end) {
            if (str.charAt(start) != str.charAt(end)) {
                return false; // Not a palindrome
            }
            start++;
            end--;
        }
        return true; // Is 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]

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

public class ArrayLeftRotation {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.print("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        System.out.print("Enter the number of positions to rotate: ");
        int d = sc.nextInt();

        // Perform left rotation
        rotateLeft(arr, d);

        System.out.println("Array after left rotation: " + Arrays.toString(arr));
    }

    // Method to rotate the array to the left by d positions
    private static void rotateLeft(int[] arr, int d) {
        int n = arr.length;
        // Ensure d is within the bounds of the array length
        d = d % n;

        // Reverse the first part
        reverse(arr, 0, d - 1);
        // Reverse the second part
        reverse(arr, d, n - 1);
        // Reverse the entire array
        reverse(arr, 0, n - 1);
    }

    // Helper method to reverse a portion of the array
    private static void reverse(int[] arr, int start, int end) {
        while (start < end) {
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;
            start++;
            end--;
        }
    }
}
```