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.*;
class Pattern {
	public static void main(String args[]) {
		Scanner sc = new Scanner(System.in);
		System.out.print("n: ");
		int n = sc.nextInt();

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

	}
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS> javac Pattern.java

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java Pattern
n: 4
*
**
***
****

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java Pattern
n: 3
*
**
***

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>
```

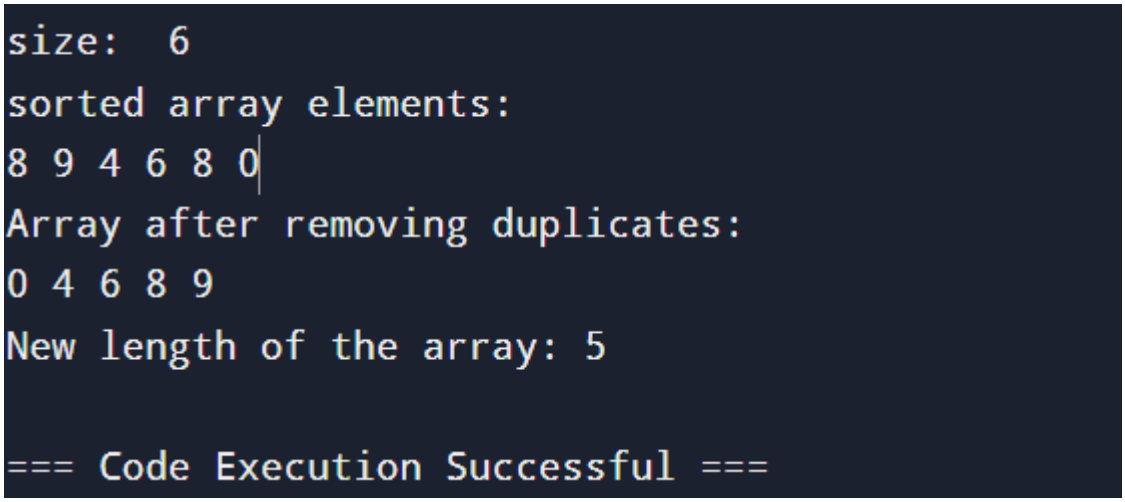
Explanation:
When we input the number of rows, n. It then uses nested loops to print a right-angled triangle made of asterisks (*). The outer loop iterates through each row, while the inner loop prints the appropriate number of asterisks for that row. Finally, the Scanner is closed to free up resources.

Time Complexity: O(n^2)
Space Complexity: O(1)

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.*;
 class Rotation {
    public static int removeDuplicates(int[] arr) {
        if (arr.length == 0) return 0;
        int index = 1;
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] != arr[i - 1]) {
                arr[index] = arr[i];
                index++;
            }
        }
        return index;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("size:  ");
        int size = sc.nextInt();
        int[] arr = new int[size];
        System.out.println("sorted array elements: ");
        for (int i = 0; i < size; i++) {
            arr[i] = sc.nextInt();
        }
        Arrays.sort(arr);
        int newLength = removeDuplicates(arr);
        System.out.println("Array after removing duplicates: ");
        for (int i = 0; i < newLength; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println("\nNew length of the array: " + newLength);
        sc.close();
    }
}
```



Explanation:

First input the array from the user , sorts the array, and then uses the removeDuplicates method to shift unique elements to the front of the array while returning the new length. Finally, it prints the modified array and its new length.

Time Complexity: O(nlogn)
Space Complexity: O(1)

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;
```

```
class RemoveSpaces {
	public static void main(String args[]) {
		Scanner sc = new Scanner(System.in);
		String str = sc.nextLine();
		String result = str.replaceAll("\\s", "");
		System.out.println(result);
		sc.close();
	}
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS> javac RemoveSpaces.java

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS> java RemoveSpaces
Hello world
Helloworld

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS> java RemoveSpaces
Welcome to the world of programming
Welcometotheworldofprogramming

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>
```

Explanation:
We take input from the user and removes all whitespace characters from the string using the replaceAll method with a regular expression. It then prints the modified string without spaces.

Time Complexity: O(n)
Space Complexity: O(n)

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 Reverse {
public static void main(String args[])  {
{
   Scanner sc = new Scanner(System.in);
         System.out.println();
         String str = sc.nextLine();
   int len = str.length();
   int i;
         Stack<Character> chr = new Stack<>();

  for(i=0;i<len;i++)
      stack.push(str.charAt(i));  // pushes an element into stack
```

```
  for(i=0;i<len;i++)
    System.out.println(stack.pop());

}
}
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>javac Reverse.java

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java Reverse

Hello
olleH
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java Reverse

Programming
gnimmargorP
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>
```

Explanation:
We take the input from the user in String format . It reads the input string, pushes each character onto the stack, and then pops the characters from the stack to print them in reverse order.

Time Complexity: O(n)
Space Complexity: O(n)

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.*;

 public class ReverseArray {

        public static void rev(int[] arr)
        {
                Stack<Integer> stack = new Stack<>();

                for(int element : arr)
                stack.push(element);

                for(int i =0; i < arr.length; i++) {
```

```
                arr[i] = stack.pop();
            }
    }

        public static void main(String args[]) {
                Scanner sc = new Scanner(System.in);
            System.out.print("arr=  ");
            int n = sc.nextInt();
                int[] arr = new int[n];

                System.out.println();
                for(int i = 0; i < n; i++) {
                        arr[i]= sc.nextInt();
                }

                rev(arr);

                System.out.println();
                for(int num : arr) {
                        System.out.print(num+ " ");
                }
        }
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>javac ReverseArray.java

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java ReverseArray
arr=  5

1 2 3 4 5

5 4 3 2 1
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java ReverseArray
arr=  10

1 3 5 7 9 11 13 15 17 19

19 17 15 13 11 9 7 5 3 1
```

Explanation:

By using stack we can reverse the array . The rev method pushes each element of the array onto a stack and then pops them back into the array in reverse order. The main method reads the size of the array and its elements from the user, calls the rev method to reverse the array, and then prints the reversed array.

Time Complexity: O(n)
Space Complexity: O(n)

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"

```
import java.util.*;

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


      System.out.println( );
      String ip = sc.nextLine();

      String rev = rever(ip);
      System.out.println(rev);

      sc.close();
   }

   public static String rever(String sen) {

      String[] Array = sen.split(" ");

      List<String> words = new ArrayList<>();
      Collections.addAll(words, Array);

      Collections.reverse(words);

      return String.join(" ", words);
   }
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>javac ReverseWords.ja

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java ReverseWords

Hello World
World Hello

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java ReverseWords

Java Programming
Programming Java
```

Explanation:
In this program we are reversing the order of Words . The main method reads a line of text from the user, calls the rever method to reverse the order of the words, and then prints the result. The rever method splits the input string into words, stores them in a list, reverses the list, and then joins the words back into a single string.

Time Complexity: O(n)
Space Complexity: O(n)

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.println( );
      int n = sc.nextInt();
      System.out.println(reverseNumber(n));
      sc.close();
   }

   public static int reverseNumber(int num) {
      int reversed = 0;
      while (num != 0) {
         reversed = reversed * 10 + num % 10;
         num /= 10;
      }
      return reversed;
   }
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>javac ReverseNumber.java

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java ReverseNumber

123456
654321

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java ReverseNumber

-6899
-9986
```

Explanation:

In this program we are reversing the interger from the input of user . The main method reads an integer, calls the reverseNumber method to perform the reversal, and then prints the reversed number. The reverseNumber method uses a loop to extract the last digit of the number and build the reversed integer.

Time Complexity: O(d) (where d is the number of digits in the input)
Space Complexity: O(1)

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

```
import java.util.Scanner;
 class Palindrome1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println(" ");
        String ip = sc.nextLine();
        System.out.println(isPalindrome(ip));
        sc.close();
    }

    public static boolean isPalindrome(String str) {
        StringBuilder revStr = new StringBuilder(str);
        revStr.reverse();
        return str.equals(revStr.toString());
    }
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>javac Palindrome1.java

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java Palindrome1

madam
true

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java Palindrome1

Hello
false

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java Palindrome1

asha
false
```

Explanation:
In this program we checks weather a string is a palindrome, meaning it reads the same forwards and backwards. The main method reads a string input from the user and calls the isPalindrome method to determine if it's a palindrome, then prints the result. The isPalindrome method uses a StringBuilder to reverse the input string and compares it with the original.

Time Complexity: O(n)
Space Complexity: O(n)

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;

public class LeftRotation {
    public static void main(String[] args) {
        int[] arr1 = {1, 2, 3, 4, 5};
        int d1 = 2;
        rotateLeft(arr1, d1);
        System.out.println("Rotated Array: " + Arrays.toString(arr1)); // Output: [3, 4, 5, 1, 2]

        int[] arr2 = {10, 20, 30, 40};
        int d2 = 1;
        rotateLeft(arr2, d2);
        System.out.println("Rotated Array: " + Arrays.toString(arr2)); // Output: [20, 30, 40, 10]
    }
```

```
    public static void rotateLeft(int[] arr, int d) {
        int n = arr.length;
        d = d % n; // In case d > n
        reverseArray(arr, 0, d - 1);   // Reverse the first part
        reverseArray(arr, d, n - 1);   // Reverse the second part
        reverseArray(arr, 0, n - 1);   // Reverse the entire array
    }

    public static void reverseArray(int[] arr, int start, int end) {
        while (start < end) {
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;
            start++;
            end--;
        }
    }
}
```

```
C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>java LeftRotation
Rotated Array1: [3, 4, 5, 1, 2]

Rotated Array2: [20, 30, 40, 10]

C:\Users\Shreya\OneDrive\Desktop\CDAC\ADS>
```

Explanation:

This Java program rotates an array to the left by a specified number of positions. The rotateLeft method takes an array and the number of positions to rotate (d) and uses a three-step reversal process to achieve the rotation. It first reverses the first part of the array, then the second part, and finally the entire array. The reverseArray helper method performs the in-place reversal.

Time Complexity: O(n)
Space Complexity: O(1)