**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;**

**class Pattern{**

**public static void main(String []args)**

**{**

**Scanner sc = new Scanner(System.in);**

**System.out.println("enter a number");**

**int n = sc.nextInt();**

**for(int i =1; i<=n ; i++)**

**{**

**for(int j=1; j<=i ; j++)**

**{**

**System.out.print("\*");**

**}**

**System.out.println();**

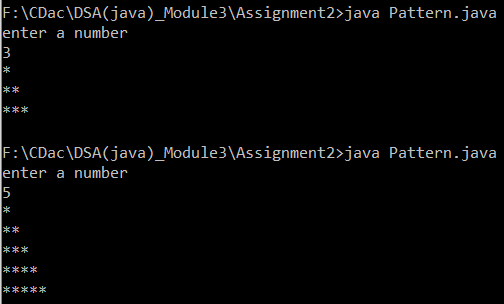
**}**

**}**

**}**

Time complexity : O(n²)

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.Arrays;**

**import java.util.Scanner;**

**public class RemoveDuplicate{**

**public static void main(String[] args) {**

**Scanner sc = new Scanner(System.in);**

**System.out.println("enter size of array");**

**int size = sc.nextInt();**

**int[] arr1 = new int[size];**

**int count = 0;**

**System.out.println("enter elemnts");**

**for(int i=0; i<size;i++){**

**arr1[i] = sc.nextInt();**

**}**

**System.out.println("array with duplicates "+Arrays.toString(arr1));**

**int newlength = removeDuplicates(arr1);**

**System.out.println("New length after removing duplicates from arr1: " + newlength);**

**System.out.println("Array after removing duplicates: " + Arrays.toString(Arrays.copyOf(arr1,newlength)));**

**sc.close();**

**}**

**// Method to remove duplicates from a sorted array**

**public static int removeDuplicates(int[] arr) {**

**if (arr.length == 0) return 0;**

**int uniqueIndex = 0; // Index for the next unique element**

**// Iterate through the array**

**for (int i = 1; i < arr.length; i++) {**

**// If the current element is different from the last unique element**

**if (arr[i] != arr[uniqueIndex]) {**

**uniqueIndex++; // Move to the next position**

**arr[uniqueIndex] = arr[i]; // Update the unique element**

**}**

**}**

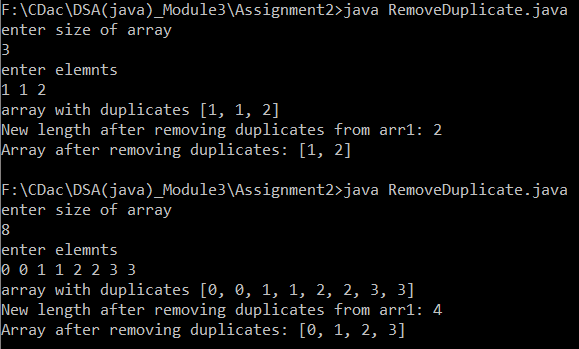
**return uniqueIndex +1;**

**}**

**}**

Time complexity : O(n)

Space complexity : O(n)



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 WhiteSpaceRomove{**

**public static void main(String[] args)**

**{**

**Scanner sc = new Scanner(System.in);**

**String s = sc.nextLine();**

**/\***

**String[] strs = s.split(" ");**

**String ans = "";**

**//String str = s.replaceAll(" ","");**

**// System.out.println("answer: "+str);**

**System.out.println("answer: "+ans);**

**} \*/**

**StringBuilder result = new StringBuilder();**

**for(int i=0; i<s.length();i++){**

**if(s.charAt(i) != ' '){**

**result.append(s.charAt(i));**

**}**

**}**

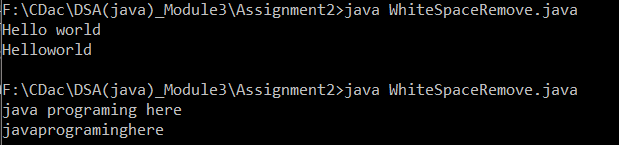
**System.out.println(result);**

**}**

**}**

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;**

**class ReverseString{**

**public static void main(String []args)**

**{**

**Scanner sc = new Scanner(System.in);**

**String str = sc.nextLine();**

**String revstr = "";**

**for(int i=str.length()-1 ; i>=0; i--){**

**revstr += str.charAt(i);**

**}**

**System.out.println(revstr);**

**}**

**}**

Time complexity : O(n^2)

Space complexity : O(n)

**import java.util.Scanner;**

**class ReverseString{**

**public static void main(String []args)**

**{**

**Scanner sc = new Scanner(System.in);**

**String str = sc.nextLine();**

**StringBuilder revstr = new StringBuilder();**

**for(int i=str.length()-1 ; i>=0; i--){**

**revstr.append(str.charAt(i));**

**}**

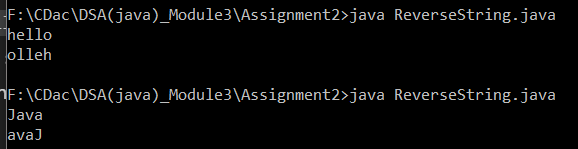
**System.out.println(revstr.toString());**

**}**

**}**

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.Scanner;**

**import java.util.Arrays;**

**class ReverseArray{**

**public static void main(String []args){**

**int arr[] = {1,2,3,4};**

**System.out.println(Arrays.toString(arr));**

**int start =0;**

**int end = arr.length-1;**

**while(start < end){**

**int temp = arr[start];**

**arr[start] = arr[end];**

**arr[end] = temp;**

**start++;**

**end--;**

**}**

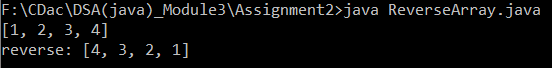
**System.out.println("reverse: "+Arrays.toString(arr));**

**}**

**}**

Time complexity : O(n)

Space complexity : O(1)



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.Scanner;**

**import java.io.\*;**

**public class ReverseWord**

**{**

**// Method to reverse the order of words in a string**

**static String reverse(String s){**

**String[] word = s.split(" ");// Split the string into words**

**String revstr =""; // Variable to store the reversed words**

**for(int i = word.length-1 ; i>=0 ; i--)**

**{**

**revstr += word[i]; // Append each word to the result**

**if(i != 0){**

**revstr += " ";// Add space between words, but not after the last word**

**}**

**}**

**return revstr;**

**}**

**public static void main(String srgd[]){**

**Scanner sc = new Scanner(System.in);**

**String str = sc.nextLine();**

**System.out.println("reverse string: "+ reverse(str));**

**}**

**}**

Time complexity : O(n^2) 🡪 O(m^2)

Space complexity : O(n)

**import java.util.Scanner; public class ReverseWord {**

**// Method to reverse the order of words in a string**

**static String reverse(String s) {**

**String[] word = s.split(" "); // Split the string into words**

**StringBuilder revstr = new StringBuilder(); // Use StringBuilder for efficiency**

**for (int i = word.length - 1; i >= 0; i--) {**

**revstr.append(word[i]); // Append each word to the result**

**if (i != 0) { revstr.append(" "); // Add space between words, but not after the last word**

**}**

**}**

**return revstr.toString();**

**}**

**public static void main(String[] args) {**

**Scanner sc = new Scanner(System.in);**

**String str = sc.nextLine();**

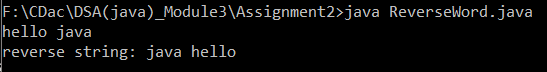
**System.out.println("Reversed string: " + reverse(str));**

**}**

**}**

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;

class ReverseNumber{

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

System.out.println("Enter a number");

int n = sc.nextInt();//123

int temp=0;

int sign = (n<0) ? -1 :1;// Determine the sign of the number

n = Math.abs(n); // Work with the absolute value of the number

while(n>0){

temp = temp\*10 + n%10;//3 -->

n = n/10;//12

}

temp \*= sign;

if(sign == -1)

System.out.print("reverse number: "+ temp);

else

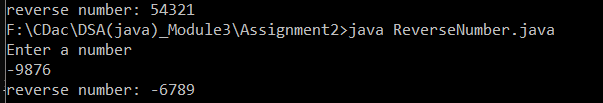
System.out.print("reverse number: "+ temp);

}

}

Time complexity : O(n) -> O(d)//d 🡪 number of digit

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

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

**import java.util.Scanner;**

**class StringPellindrom{**

**static boolean pellindrome(String s)**

**{**

**//String orgi = s;**

**String temp = "";**

**for(int i=s.length()-1; i>=0; i--)**

**{**

**temp += s.charAt(i);**

**// s.charAt[i] = s.charAt[s.length];**

**}**

**System.out.println("reverse String: "+temp);**

**if(s.equals(temp))**

**return true;**

**else**

**return false;**

**}**

**public static void main(String[] args){**

**Scanner sc = new Scanner(System.in);**

**System.out.println("enter a string");**

**String str = sc.nextLine();**

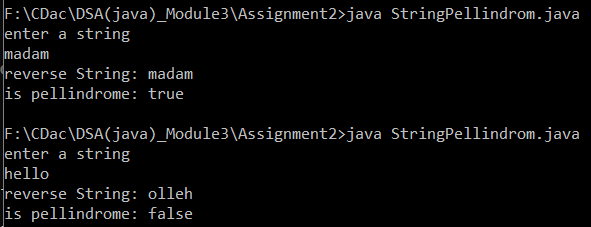
**System.out.println("is pellindrome: " + pellindrome(str));**

**}**

**}**

Time complexity : O(n)

Space complexity : O(1)



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.Scanner;**

**import java.util.Arrays;**

**class ArrayLeftRotation{**

**public static void main(String []args){**

**Scanner sc = new Scanner(System.in);**

**System.out.println("enter a size of array!!!");**

**int size = sc.nextInt();**

**int arr[] = new int[size];**

**System.out.println("enter position to left rotate");**

**int d = sc.nextInt();**

**int temp[] = new int[size];**

**System.out.println("enter array elements");**

**for(int i=0; i<size ; i++){**

**arr[i] = sc.nextInt();**

**}**

**System.out.println("original : "+Arrays.toString(arr));**

**for(int i=0; i<size-d ; i++){**

**temp[i] = arr[d+i];**

**}**

**for(int i=0; i<d ; i++){**

**temp[size - d+i] =arr[i];**

**}**

**System.out.println("after rotate : "+Arrays.toString(temp));**

**}**

**}**

Time complexity : O(n)

Space complexity : O(n)

