**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:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

Ans = public class Solution1 {

public static void main(String[] args) {

int n =5;

for (int i =0; i<n ; i++) {

for (int j = 0; j<=i; j++) {

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

}

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

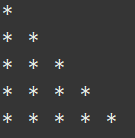
}

}

}

**Explanation:**

1. Start by declaring a class named “RightTriangleStarPattern”.
2. Inside the class, declare a “main” method.
3. Initialize a variable “rows” to store the number of rows to be printed in the pattern. In this example, we will use the value 5.
4. Use a “for” loop to iterate over each row of the pattern. Set the initial value of the loop variable “i” to 1 and continue looping while “i” is less than or equal to “rows”.
5. Increment “i” by 1 after each iteration.
6. Inside the outer loop, use another “for” loop to print the asterisks on each line. Set the initial value of the loop variable “j” to 1 and continue looping while “j” is less than or equal to “i”. Increment “j” by 1 after each iteration.
7. Inside the inner loop, print an asterisk followed by a space using the “System.out.print” statement.
8. After the inner loop completes, print a newline character using the “System.out.println” statement to move to the next line.
9. Run the program to see the output.



Complexity :

Time : O(n2)  
**Space :** 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

Ans= public class Solution1 {

static int removeDuplicates(int[] arr) {

int n = arr.length;

if (n <= 1)

return n;

int idx = 1; // Start from the second element

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

if (arr[i] != arr[i - 1]) {

arr[idx++] = arr[i];

}

}

return idx;

}

public static void main(String[] args) {

int[] arr = {1, 2, 2, 3, 4, 4, 4, 5, 5};

int newSize = *removeDuplicates*(arr);

for (int i = 0; i < newSize; i++) {

System.***out***.print(arr[i] + " ");

}

}

}

Explanation :

* Start with idx = 1 (idx is going to hold the index of the next distinct item. Since there is nothing before the first item, we consider it as the first distinct item and begin idx with 1.
* Loop through the array for i = 0 to arr.size() – 1
  + At each index i, if arr[i] is different from arr[i-1], assign arr[idx] = arr[i] and increment idx.
* After the loop, arr contains the unique elements in the first idx positions.



**Complexity = O(n) time and O(1) Space**

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"

Ans = public class Solution1 {

public static void white(String s) {

String str = s.replace(" ", "");

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

}

public static void main(String[] args) {

Solution1.*white*(" Java Programming ");

}

}

Explanation: Use the String inbuilt method .replace() to remove the whitespace.



4. Reverse a String

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

Test Cases:

Input: "hello"

Output: "olleh"

Input: "Java"

Output: "avaJ"

Ans = public class Solution1 {

public static void main(String[] args) {

String str = "Hello";

String rev = "";

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

rev = str.charAt(i) + rev;

}

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

}

}

Explanation :

1. **Initialization:**
   * originalStr is initialized to "Hello".
   * reversedStr is initialized to an empty string "".
2. **For Loop:**
   * The loop runs from i = 0 to i = originalStr.length() - 1, which means it iterates over each character of originalStr from the first to the last character.
3. **Reversing Process:**
   * In each iteration, the current character (originalStr.charAt(i)) is taken and **prepended** to the reversedStr.
   * This is the key point: by prepending the character, the string is reversed.
   * Iteration details:
     + **i = 0:** reversedStr = "H" + "" = "H"
     + **i = 1:** reversedStr = "e" + "H" = "eH"
     + **i = 2:** reversedStr = "l" + "eH" = "leH"
     + **i = 3:** reversedStr = "l" + "leH" = "lleH"
     + **i = 4:** reversedStr = "o" + "lleH" = "olleH"

output : 

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 = public class Solution1 {

public static void main(String[] args) {

int arr [] = {1,2,3,4,5,6,7,8};

int arr1 [] = new int[arr.length];

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

arr1[i]=arr[arr.length-i-1];

for (int i = 0; i < arr1.length; i++)

System.***out***.print(arr1[i]);

}

}

O/P:



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 = class Reverse {

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

String str = sc.nextLine();

String rev = "";

*//splitting the sentence in words*

String s[] = str.split(" ");

*//Adding the words in rev*

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

rev = s[j]+" "+rev;

}

System.out.println("Reversed sentence: " + rev);

}

}

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 = **class** ReverseNumber

{

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

{

**int** number = 987654, reverse = 0;

**while**(number != 0)

{

**int** remainder = number % 10;

reverse = reverse \* 10 + remainder;

number = number/10;

}

System.out.println("The reverse of the given number is: " + reverse);

}

}

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** **Main** {

public static void main(String[] args) {

String str = "madam"; // String **for** testing

StringBuilder str1 = new StringBuilder(str);

str1.reverse();

**if** (str.equals(str1.toString())) {

System.out.println("Palindrome String");

} **else** {

System.out.println("Not a palindrome String");

}

}

}

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 = class RotateLeft {

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

        //Initialize array

**int** [] arr = **new** **int** [] {1, 2, 3, 4, 5};

        //n determine the number of times an array should be rotated

**int** n = 3;

        //Displays original array

        System.out.println("Original array: ");

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

            System.out.print(arr[i] + " ");

        }

        //Rotate the given array by n times toward left

**for**(**int** i = 0; i < n; i++){

**int** j, first;

            //Stores the first element of the array

            first = arr[0];

**for**(j = 0; j < arr.length-1; j++){

                //Shift element of array by one

                arr[j] = arr[j+1];

            }

            //First element of array will be added to the end

            arr[j] = first;

        }

        System.out.println();

        //Displays resulting array after rotation

        System.out.println("Array after left rotation: ");

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

            System.out.print(arr[i] + " ");

        }

    }

}