**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:

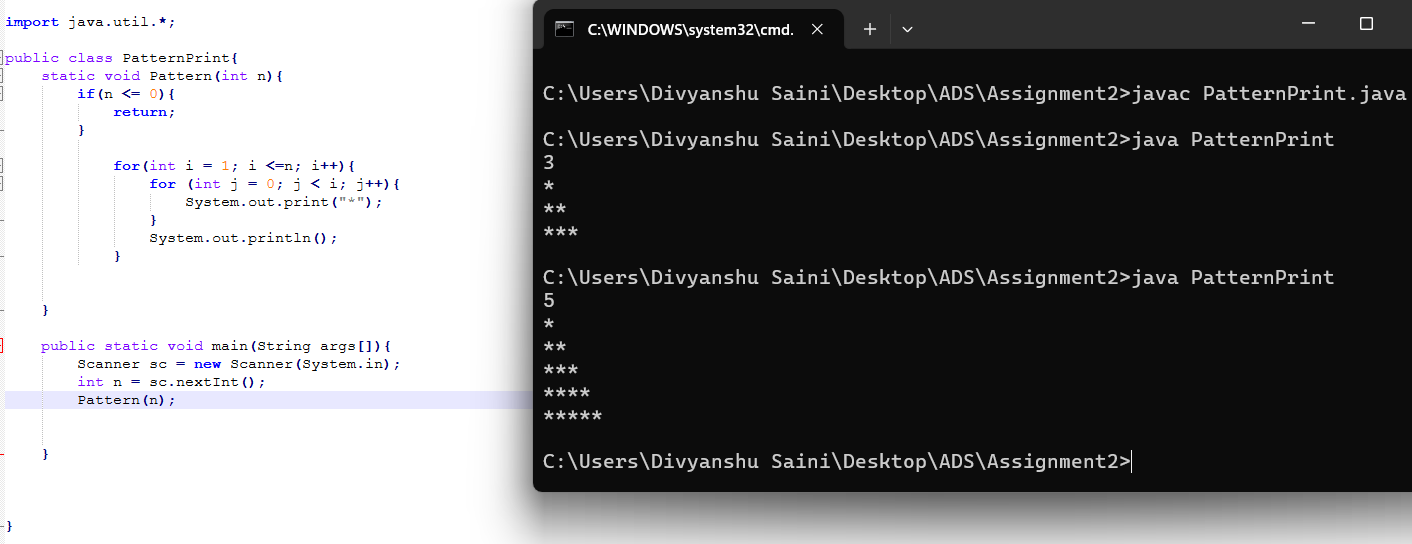
\*

\*\*

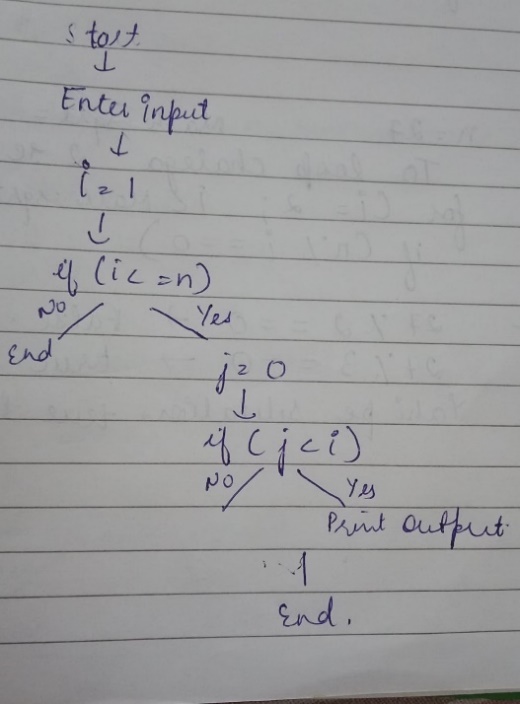
\*\*\*

\*\*\*\*

\*\*\*\*\*



**Flowchart-**

****

**Explanation :** Firstly, I’ve created a public class named Patterns. Inside the main method I’ve taken the input from the user using Scanner class. The outer for loop starts with i = 1 and runs until i is less than or equal to n. It controls the number of rows that will be printed. The value of i represents the current row number. Inside the outer loop, there is another for loop that initializes j to 0 and runs until j is less than i. This loop controls how many asterisks (\*) will be printed in the current row.

**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.

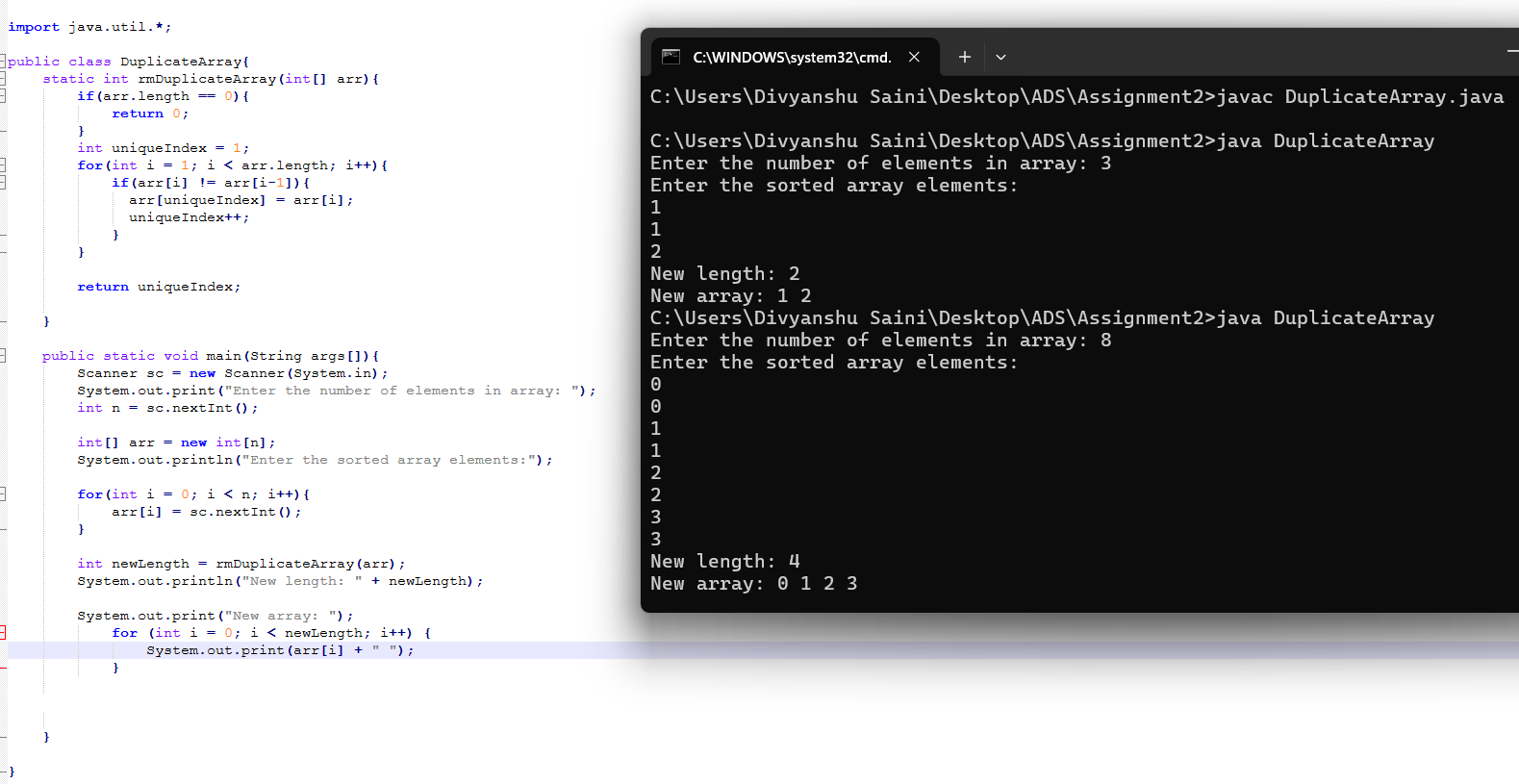
Test Cases:

Input: arr = [1, 1, 2]

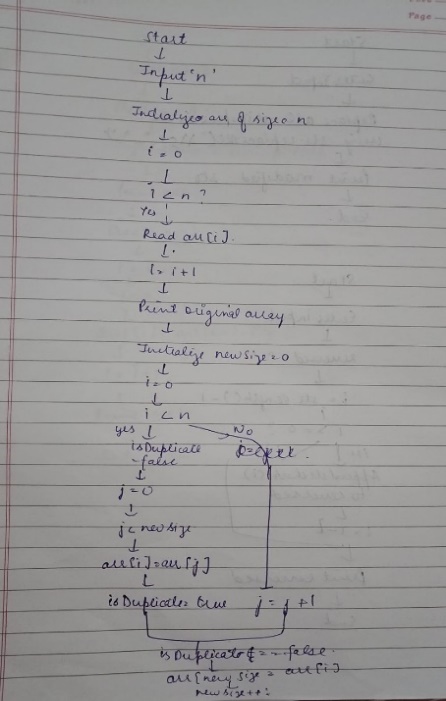
Output: 2

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

Output: 4



**Flowchart-**



**Explanation:** Firstly we have a Function rmDuplicateArray, Ittakes a sorted integer array as input.If the array is empty, it returns 0.Therewe initializes a uniqueIndex to track the position of unique elements.

Then we iterates through the array starting from the second element.

Whenever it finds an element that is not equal to the previous one, it assigns that element to the uniqueIndex and increments the uniqueIndex.

And the Main method contains test cases that demonstrate the functionality of the rmDuplicatemethod.

Prints the new length of the array after removing duplicates and shows the New array.

**Time Complexity** – O(n^2)

**Space Complexity** – O(n)

3. Remove White Spaces from String

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

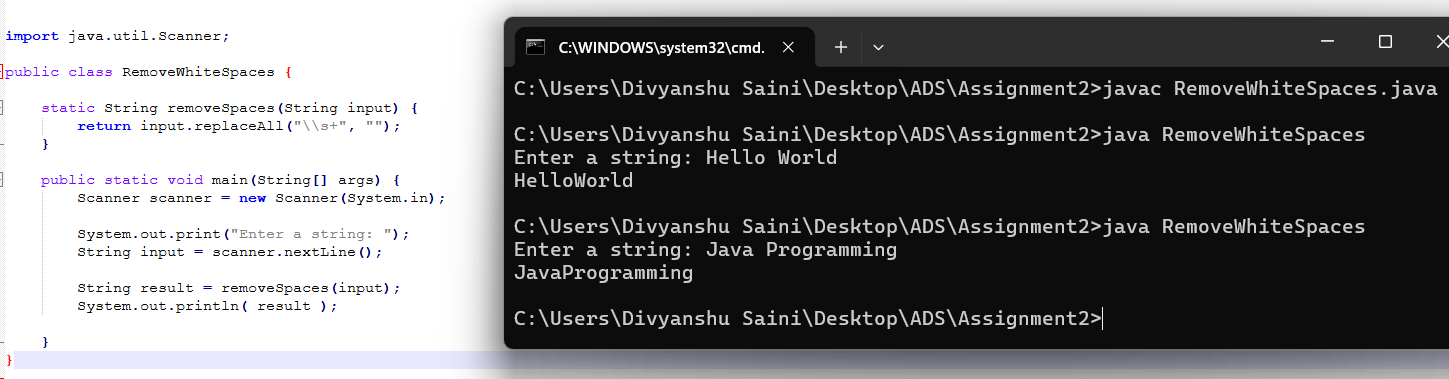
Test Cases:

Input: "Hello World"

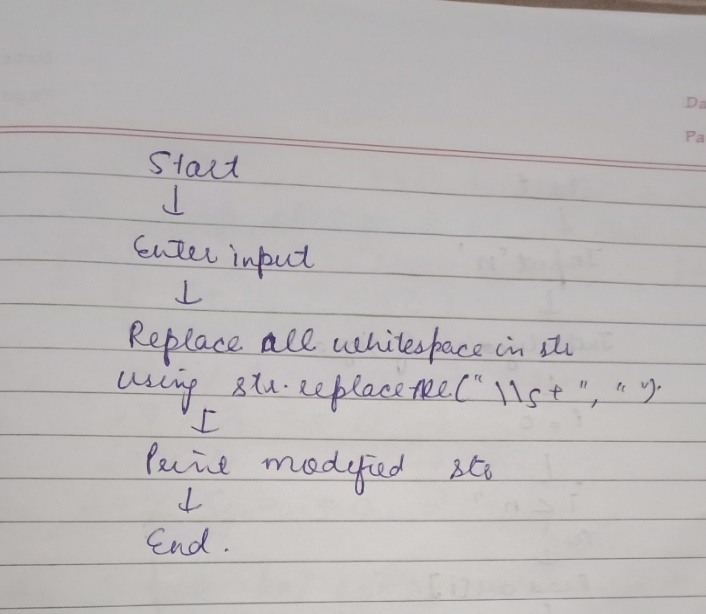
Output: "HelloWorld"

Input: " Java Programming "

Output: "JavaProgramming"



**Flowchart-**



**Explanation:** Firstly we have a Function removeSpaces it takes a string as input and uses replaceAll with the regex \\s+ to match all whitespace characters (spaces, tabs, etc.) and replace them with an empty string. The replaceAll method in Java can be used with the regex \\s+ to remove all whitespace characters from a string. The \\s matches any whitespace character (including spaces, tabs, and newlines), and the + quantifier means one or more occurrences.

And then Main method prompts the user to enter a string and reads the input. Then calls the removeSpaces method and prints the result.

**Time Complexity** – O(n)

**Space Complexity** – O(n)

4. Reverse a String

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

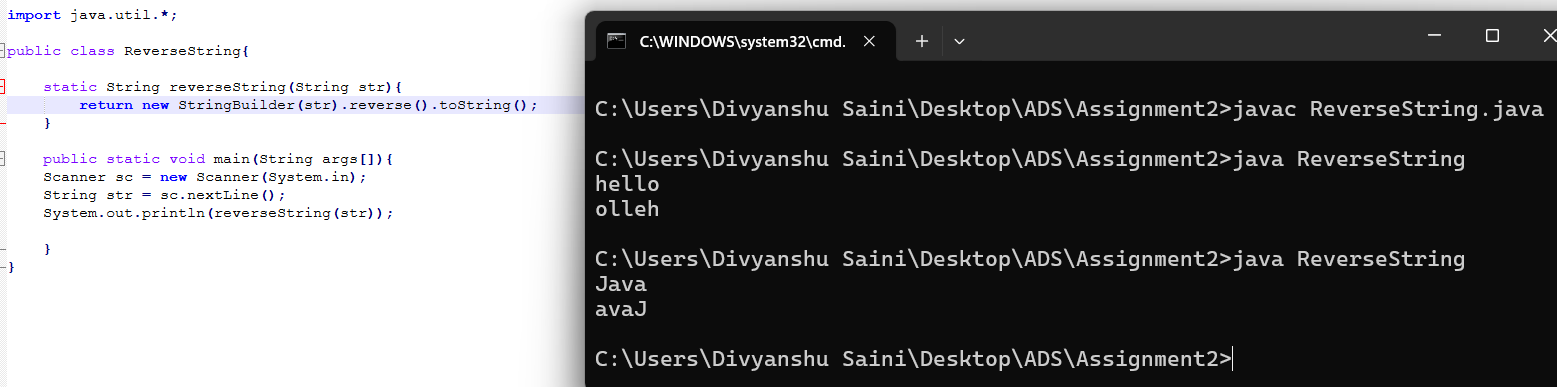
Test Cases:

Input: "hello"

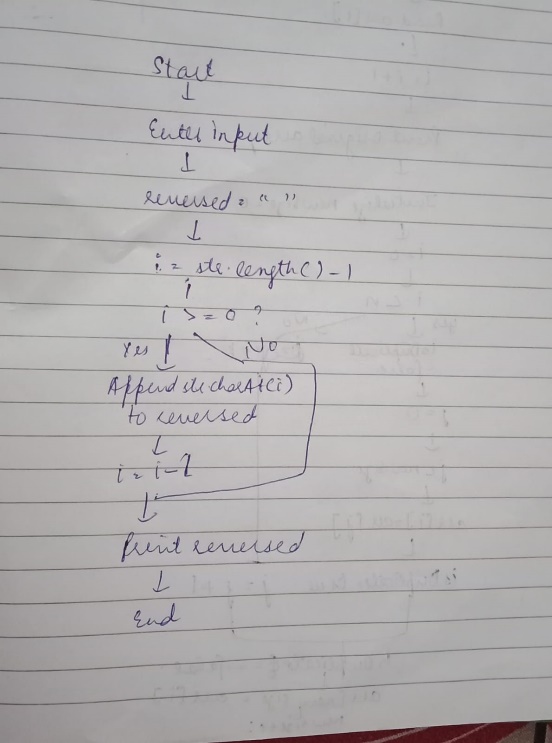
Output: "olleh"

Input: "Java"

Output: "avaJ"



**Flowchart-**

****

**Explanation:** Firstly we have a Function reverseString in which it takes a String as an argument,

and uses the StringBuilder class, which has a built-in reverse() method that reverses the characters in the string. StringBuilder provides a simple yet efficient way to reverse any given string, ensuring that users can see their input reversed instantly. Then toString() method converts the StringBuilder object back to a String.

In Main method we take the input then call the reverseString method which gives the reverse string in the output.

**Time Complexity** – O(n^2)

**Space Complexity** – O(n)

5. Reverse Array in Place

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

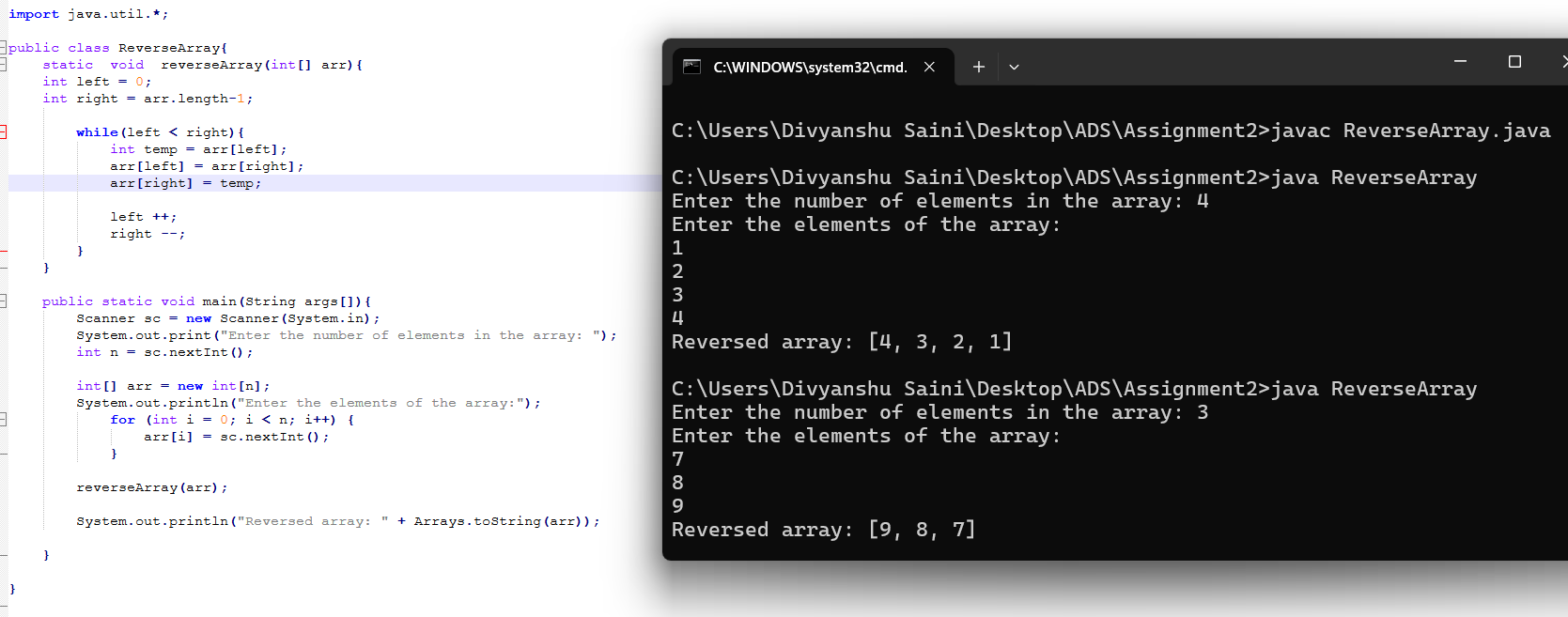
Test Cases:

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

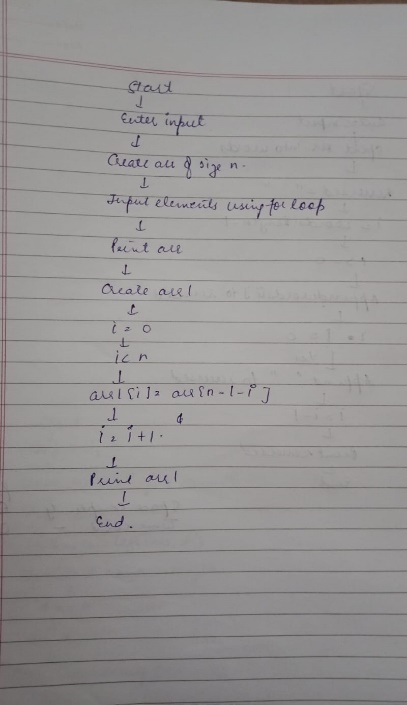
Output: [4, 3, 2, 1]

Input: arr = [7, 8, 9]

Output: [9, 8, 7]



**Flowchart-**



**Explanation:** Firstly we have a Function reverseArray which takes the array as input and swaps elements from the end to starting makes array reverse.

This method uses two pointers, left starting from the beginning and right from the end. While left is less than right, we swap the elements at these positions. After each swap, we move the left pointer to the right and the right pointer to the left, effectively moving towards the center of the array until the entire array is reversed.

**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.

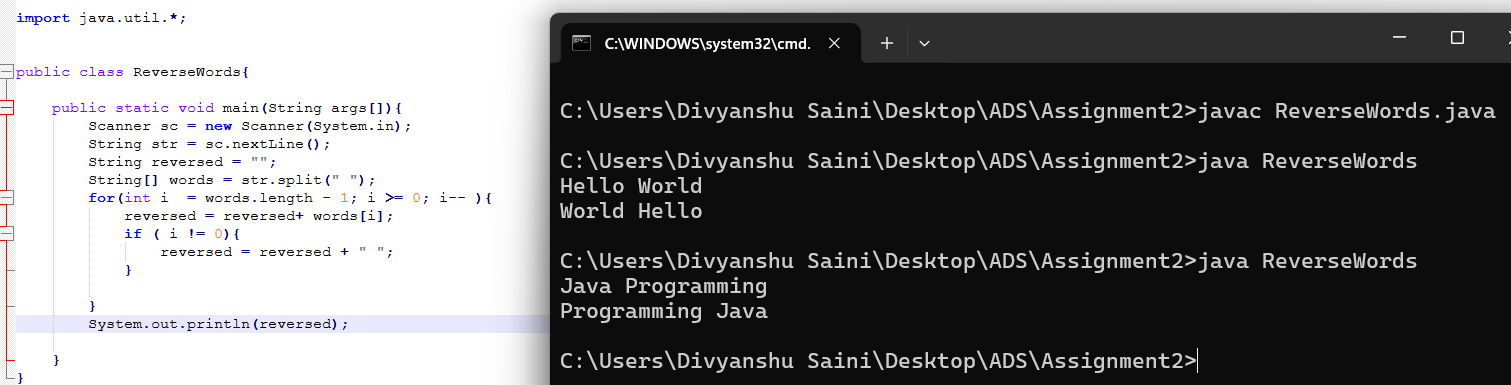
Test Cases:

Input: "Hello World"

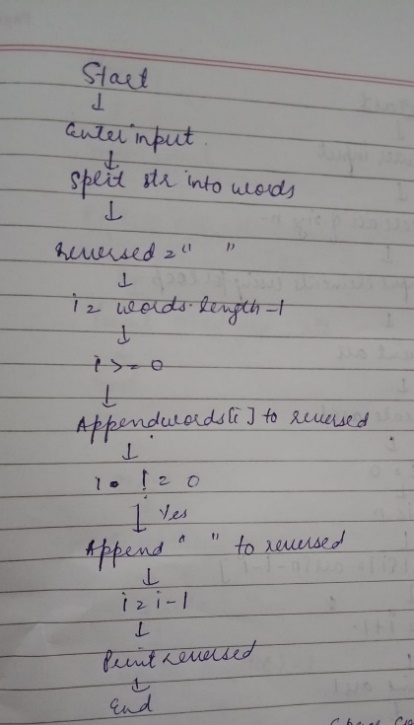
Output: "World Hello"

Input: "Java Programming"

Output: "Programming Java"



**Flowchart-**

****

**Explanation -** Firstly, I’ve created a public class named ReverseWords. Inside the main method I’ve taken a string as input with the help of Scanner class. An empty String variable named reversed is initialized. The split(" ") method splits the input string str into an array of substrings. The for loop iterates from the last element of words to the first element. Appended each word from the words array to the reversed string. Printed the reversed string.

**Time Complexity** – O(n^2)

**Space Complexity** – O(n)

7. Reverse a Number

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

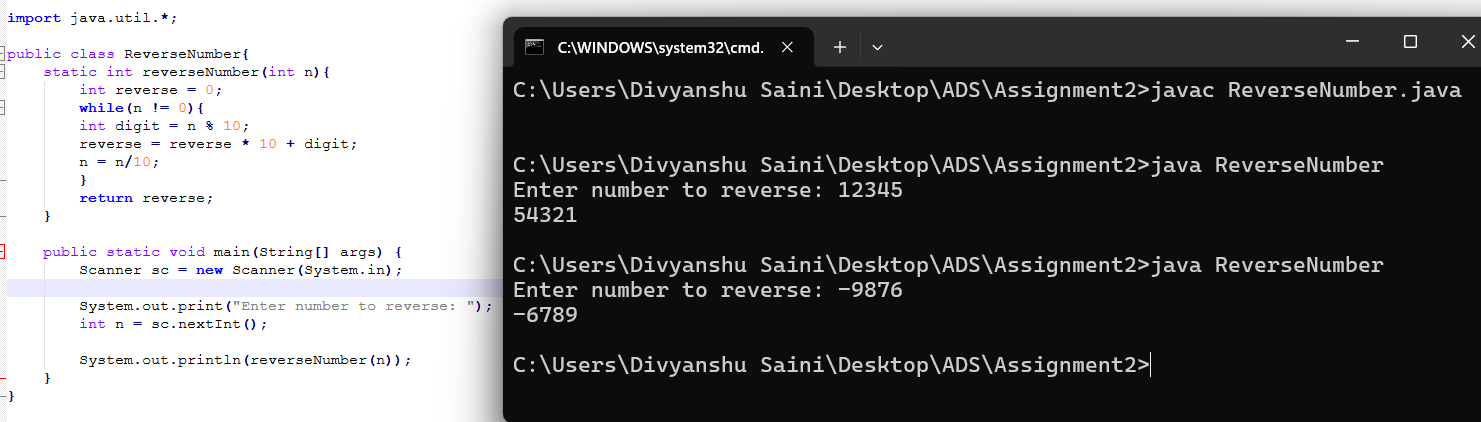
Test Cases:

Input: 12345

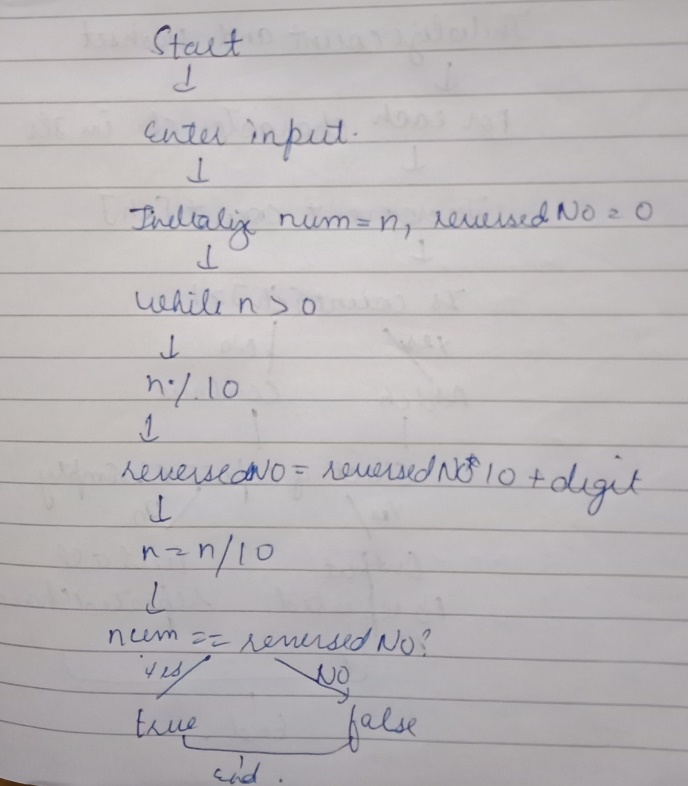
Output: 54321

Input: -9876

Output: -6789



**Flowchart:**



**Explanation:** Firstly we have a Function reverseNumber which takes the input n which is the number given by the user. Then while loop checks the condition and perform the steps to reverse the number and return that number.

In Main method it takes the input then call the reverseNumber method and print the output.

**Time Complexity –**  O (logn)

**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"

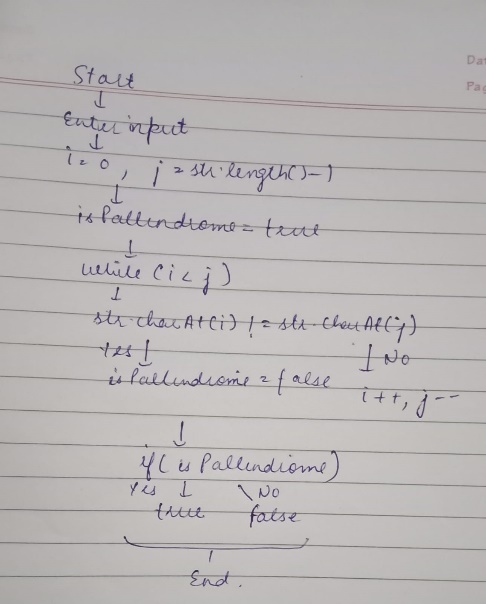
Output: true

Input: "hello"

Output: false



**Flowchart-**

****

**Explanation -** Firstly, I’ve created a public class named StringPallindrome. Inside it I’ve created a method named as Pallindrome of Boolean return type. i is initialized to 0 to point to the start of the string and j is initialized to str.length() - 1 to point to the end of the string. A boolean variable isPallindrome is initialized to true. A while loop that continues as long as i is less than j. If the characters at I and j are not equal, isPallindrome is set to false, and the loop is exited using break. This indicates that the string is not a palindrome. Moves i one step forward (i++) and j one step backward (j--) to compare the next pair of characters. If isPallindrome remains true , the method returns true.Otherwise, it returns false.

**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.

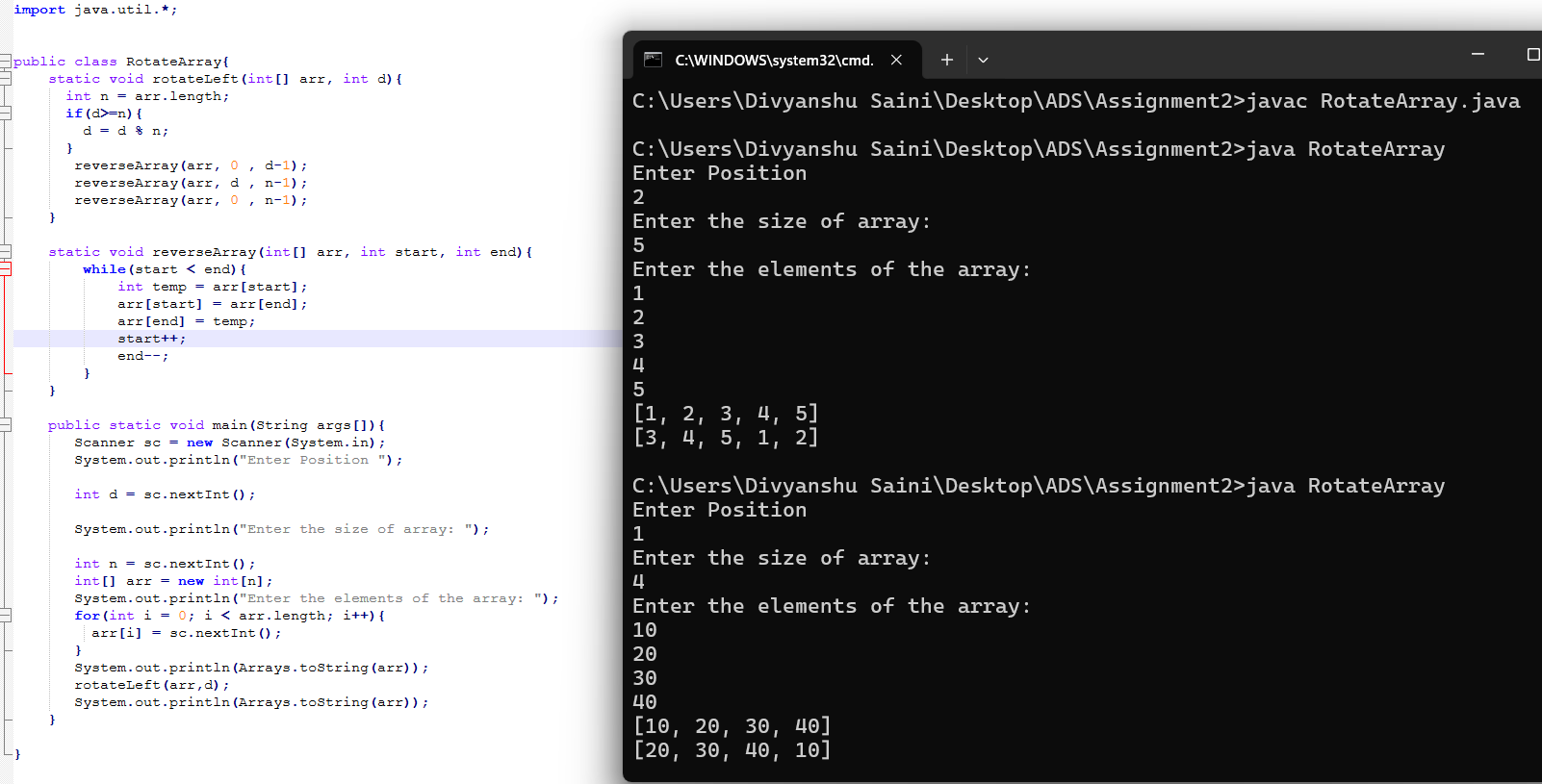
Test Cases:

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

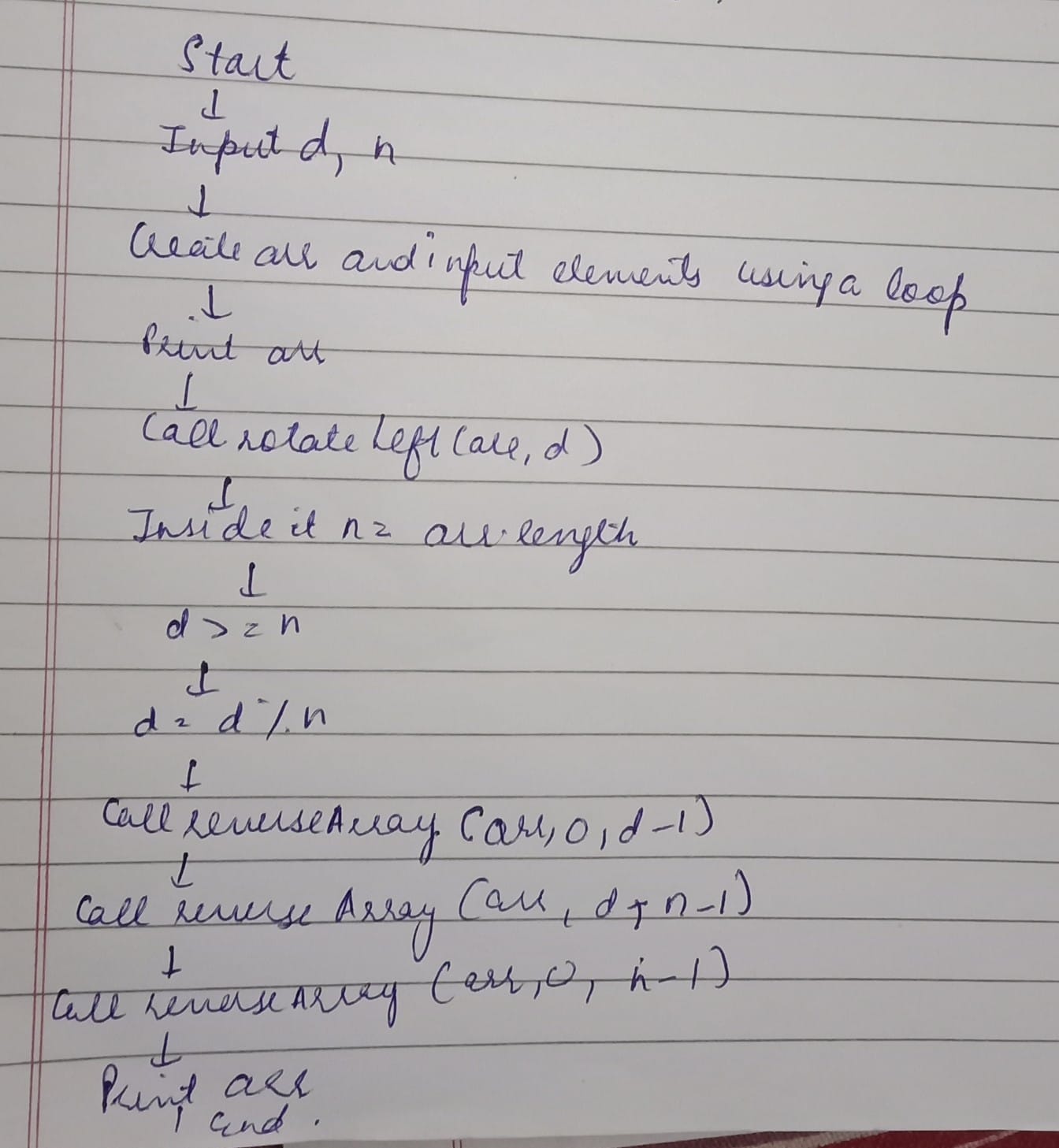
Output: [3, 4, 5, 1, 2]

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

Output: [20, 30, 40, 10]



**Flowchart –**

****

**Explanation -** Firstly, I’ve created a public class named RotateArray. Inside it I’ve created a method named as rotateLeft of void return type. In the rotateLeft method, the program first determines the length n of the array and handles cases where d is greater than or equal to n by computing d % n. The reverseArray method is used to reverse elements. The array is first reversed from index 0 to d-1, then from d to n-1, and finally, the entire array from 0 to n-1 is reversed, resulting in the desired left-rotated array. The main method handles user input, where d and n are read, followed by n array elements, and then calls the rotateLeft method to perform the rotation. The program prints the original array and the array after left rotation.

**Time Complexity** – O(n)

**Space Complexity** – O(1)