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.

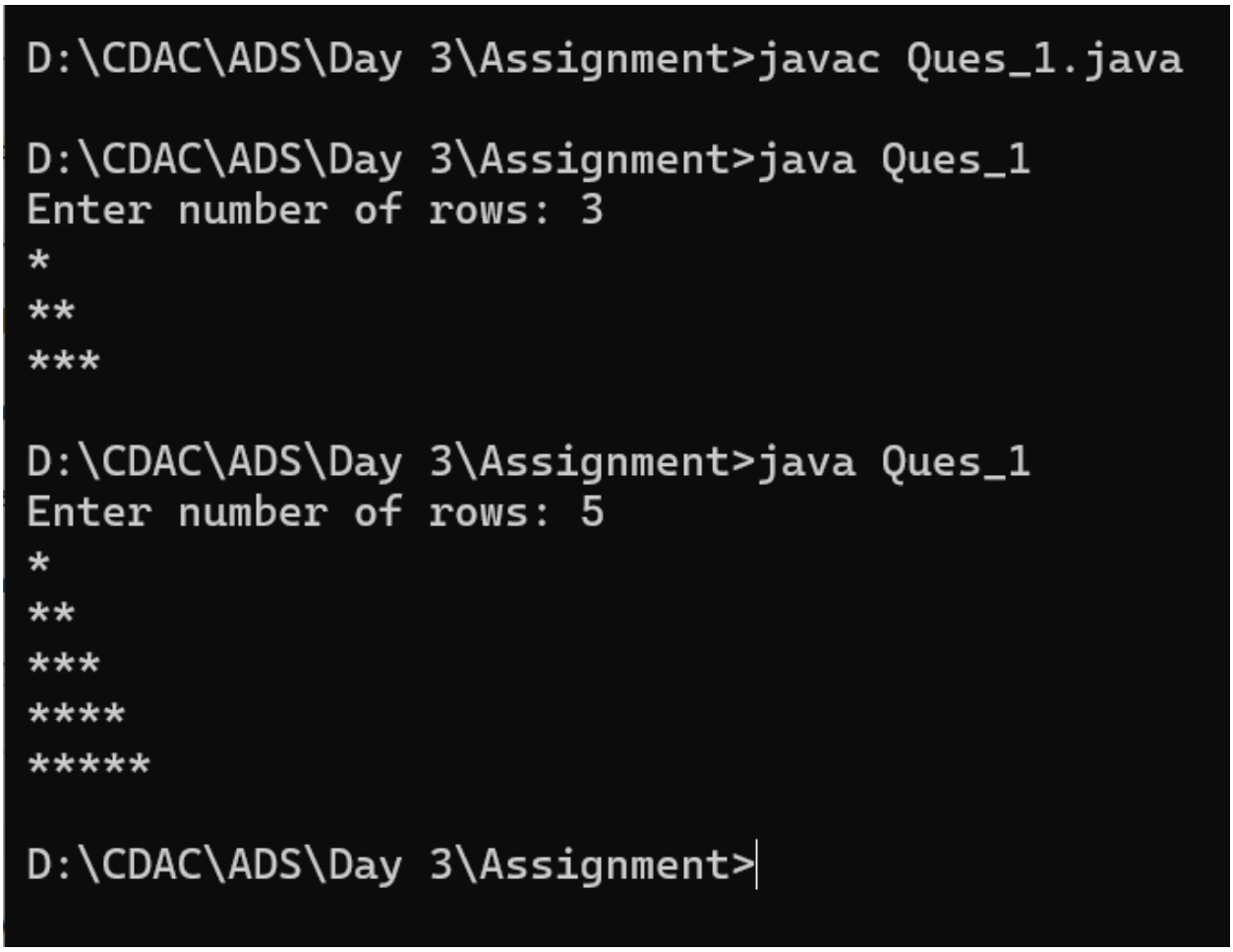
Test Cases:

Input: n = 3
Output:
*
**
***
Input: n = 5
Output:
*
**
***
****

```
import java.util.Scanner;

public class Ques_1 {
    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();
        }
    }
}
```

Explanation:

1. Take n value
2. It then prints i stars in each line, where i ranges from 1 to n.



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.*;
public class Ques_2 {
    public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter array size: ");
            int n = sc.nextInt();
```

```
            int[] arr = new int[n];
            System.out.print("\nEnter array element: ");
            for (int i = 0; i < n; i++)
            {
                  arr[i] = sc.nextInt();
            }
        int newLength = removeDuplicates(arr);
        System.out.println("New length: " + newLength);
    }

    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;
    }
}
```

Explanation:

1. compares each element with the previous one.
2. If an element is unique, it moves it to the next available position in the array.
3. The final value of index gives the count of unique elements

```
D:\CDAC\ADS\Day 3\Assignment>javac Ques_2.java

D:\CDAC\ADS\Day 3\Assignment>java Ques_2
Enter array size: 3

Enter array element: 1 1 2
New length: 2

D:\CDAC\ADS\Day 3\Assignment>java Ques_2
Enter array size: 8

Enter array element: 0 0 1 1 2 2 3 3
New length: 4

D:\CDAC\ADS\Day 3\Assignment>
```

Time Complexity: O(n)
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;

public class Ques_3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String s = sc.nextLine();

        String result = "";

        for (int i = 0; i < s.length(); i++) {
            char c = s.charAt(i);

            if (c != ' ') {
                result += c;
            }
        }

        System.out.println("String without spaces: " + result);
    }
}
```

Explanation:

1. The program takes an input string from the user.
2. It initializes an empty string result to store the final string without spaces.
3. It loops through each character of the input string. If the character is not a space, it adds (concatenates) the character to result.
4. Finally, it prints the modified string without spaces.

```
D:\CDAC\ADS\Day 3\Assignment>javac Ques_3.java

D:\CDAC\ADS\Day 3\Assignment>java Ques_3
Enter a string: Hello World
String without spaces: HelloWorld

D:\CDAC\ADS\Day 3\Assignment>java Ques_3
Enter a string: Java Programming
String without spaces: JavaProgramming

D:\CDAC\ADS\Day 3\Assignment>
```

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 Ques_4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String s = sc.nextLine();

        String r = "";
        for (int i = s.length() - 1; i >= 0; i--) {
            r += s.charAt(i);
        }

        System.out.println("Reversed string: " + r);
    }
}
```

```
D:\CDAC\ADS\Day 3\Assignment>javac Ques_4.java

D:\CDAC\ADS\Day 3\Assignment>java Ques_4
Enter a string: hello
Reversed string: olleh

D:\CDAC\ADS\Day 3\Assignment>java Ques_4
Enter a string: java
Reversed string: avaj

D:\CDAC\ADS\Day 3\Assignment>
```

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]

```
import java.util.*;
class Ques_5{
	static void reverse(int[] arr,int n)
	{
		for(int i=0;i<n/2;i++)
		{
			int temp = arr[i];
			arr[i]= arr[n-i-1];
			arr[n-i-1]=temp;
		}
	}
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter Size: ");
		int n = sc.nextInt();
		int[] arr = new int[n];
```

```
            System.out.print("\nEnter array element: ");
            for(int i=0;i<n;i++)
            {
                  arr[i]= sc.nextInt();
            }
            reverse(arr,n);
            System.out.print("\nReversed Array: ");
            for(int i=0;i<n;i++)
            {
                  System.out.print(arr[i]+" ");
            }
            System.out.println();
      }
}
```

```
D:\CDAC\ADS\Day 3\Assignment>javac Ques_5.java

D:\CDAC\ADS\Day 3\Assignment>java Ques_5
Enter Size: 4

Enter array element: 1 2 3 4

Reversed Array: 4 3 2 1

D:\CDAC\ADS\Day 3\Assignment>java Ques_5
Enter Size: 3

Enter array element: 7 8 9

Reversed Array: 9 8 7

D:\CDAC\ADS\Day 3\Assignment>
```

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

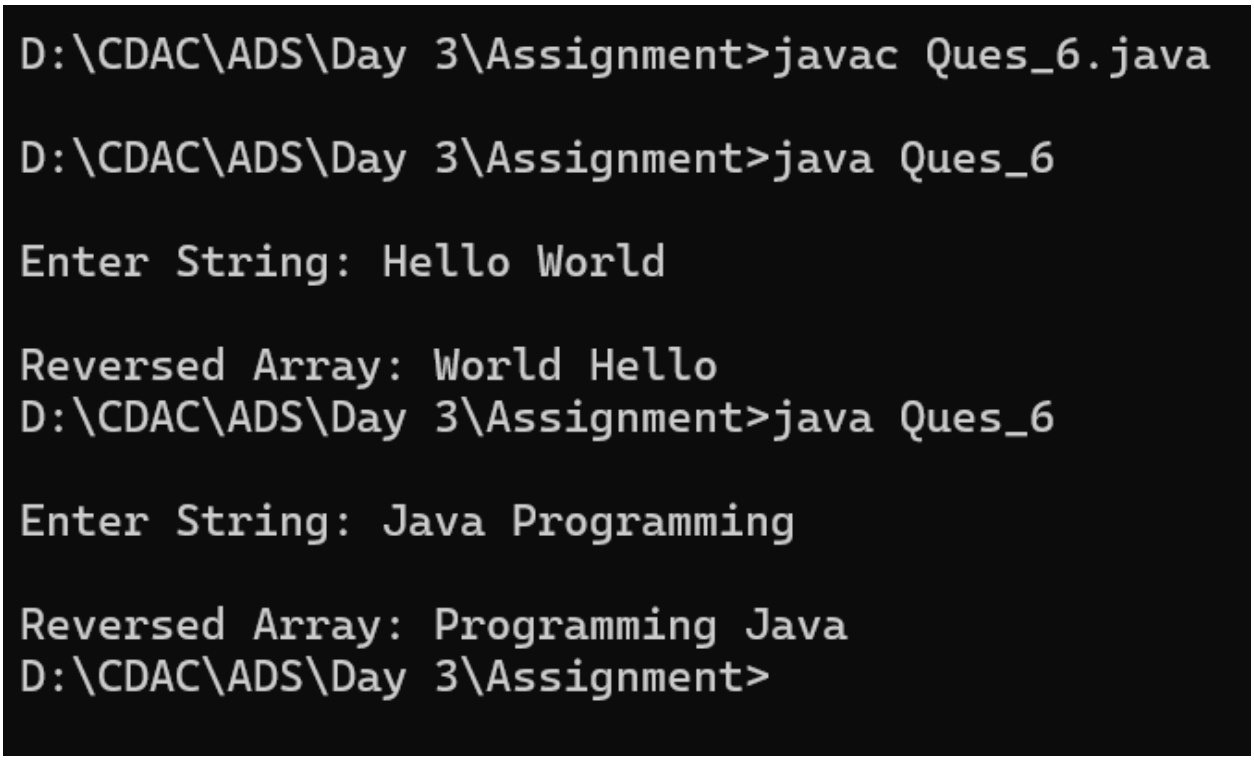
---

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.*;
class Ques_6{
	static void reverse(String[] arr)
	{
		for(int i=0;i<arr.length/2;i++)
		{
			String temp = arr[i];
			arr[i]= arr[arr.length-i-1];
			arr[arr.length-i-1]=temp;
		}
	}
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		System.out.print("\nEnter String: ");
		String sent = sc.nextLine();
		String[] arr = sent.split(" ");

		reverse(arr);
		String reverse = String.join(" ",arr);
		System.out.print("\nReversed Array: "+reverse);

	}
}
```



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

---

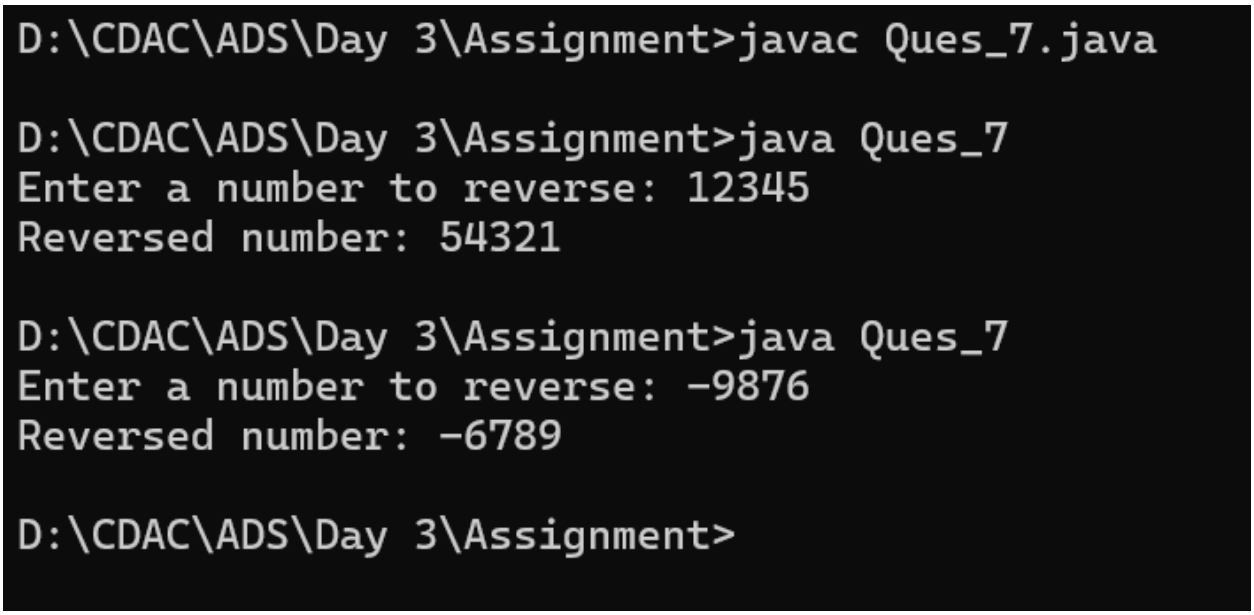
7. Reverse a Number
Problem: Write a Java program to reverse a given number.

Test Cases:

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

```
import java.util.*;
class Ques_7 {
      static int reverse(int number,int reverse)
      {
            while (number != 0) {
             int lastDigit = number % 10;

             reverse = reverse * 10 + lastDigit;

             number = number / 10;
        }
            return reverse;
      }
```

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

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter a number to reverse: ");
        int number = sc.nextInt();

        int reverse = 0;
        int originalNumber = number;
        System.out.println("Reversed number: " +reverse(number,reverse) );
    }
}
```



Time Complexity: 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

```
import java.util.Scanner;

public class Ques_8 {
```

```
    public static int ArrayManu(int n, int[][] q) {
        int[] arr = new int[n + 1];

        for (int i = 0; i < q.length; i++) {
            int start = q[i][0] - 1;
            int end = q[i][1];
            int value = q[i][2];

            arr[start] += value;
            if (end < n) {
                arr[end] -= value;
            }
        }
        int max = 0;
        int currentSum = 0;

        for (int i = 0; i < n; i++) {
            currentSum += arr[i];
            if (currentSum > max) {
                max = currentSum;
            }
        }

        return max;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter  size : ");
        int n = sc.nextInt();
        System.out.print("Enter the number of q: ");
        int que = sc.nextInt();
        int[][] q = new int[que][3];

        System.out.println("Enter the q in the format (start end value)");
        for (int i = 0; i < que; i++) {
            q[i][0] = sc.nextInt();
            q[i][1] = sc.nextInt();
            q[i][2] = sc.nextInt();
        }
        int result = ArrayManu(n, q);
        System.out.println("Maximum value after all operations: " +
result);
    }
}
```

```
D:\CDAC\ADS\Day 3\Assignment>javac Ques_8.java

D:\CDAC\ADS\Day 3\Assignment>java Ques_8
Enter  size : 5
Enter the number of q: 3
Enter the q in the format (start end value)
1 2 100
2 5 100
3 4 100
Maximum value after all operations: 200

D:\CDAC\ADS\Day 3\Assignment>java Ques_8
Enter  size : 4
Enter the number of q: 2
Enter the q in the format (start end value)
1 3 50
2 4 70
Maximum value after all operations: 120

D:\CDAC\ADS\Day 3\Assignment>
```

Time Complexity: O(n+m) // n-size of array m-number of queries
Space Complexity: O(n)

---

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.*;

class Ques_9 {
	public static boolean palinCheck(String s) {
```

```
        char[] c = s.toCharArray();
        String s1 = "";
        for(int i=c.length-1 ; i>=0 ; i--) {
            s1 = s1+c[i];
        }
        if(s1.equals(s)) {
            return true;
        }
        else {
            return false;
        }
    }
    public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter String: ");
            String s1 = sc.nextLine();
        System.out.println(palinCheck(s1));
    }

}
```

```
D:\CDAC\ADS\Day 3\Assignment>javac Ques_9.java

D:\CDAC\ADS\Day 3\Assignment>java Ques_9
Enter String: madam
true

D:\CDAC\ADS\Day 3\Assignment>java Ques_9
Enter String: hello
false

D:\CDAC\ADS\Day 3\Assignment>
```

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

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

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.*;
class Ques_10 {
      public static void arrayRotate(int[] arr,int d)
      {
       int n = arr.length;
         d = d % n;
       for(int i=0 ; i<d ; i++)
         {
           int temp = arr[0];

           for(int j=0 ; j<n-1 ; j++)
               {
                arr[j] = arr[j+1];
           }
           arr[n-1] = temp;
       }

    }
    public static void main(String[] args)
      {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter size: ");
            int size = sc.nextInt();
        int[] arr = new int[size];
            System.out.print("Enter array: ");
            for(int i=0 ; i<size ;i++)
            {
                  arr[i]=sc.nextInt();
            }
            System.out.print("Enter d: ");
        int d = sc.nextInt() ;
        arrayRotate(arr,d);
            System.out.print("Rotated array: ");
        for (int i = 0; i < size; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

```
D:\CDAC\ADS\Day 3\Assignment>javac Ques_10.java

D:\CDAC\ADS\Day 3\Assignment>java Ques_10
Enter size: 5
Enter array: 1 2 3 4 5
Enter d: 2
Rotated array: 3 4 5 1 2
D:\CDAC\ADS\Day 3\Assignment>java Ques_10
Enter size: 4
Enter array: 10 20 30 40
Enter d: 1
Rotated array: 20 30 40 10
D:\CDAC\ADS\Day 3\Assignment>
```

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