CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH

Department of Computer Engineering

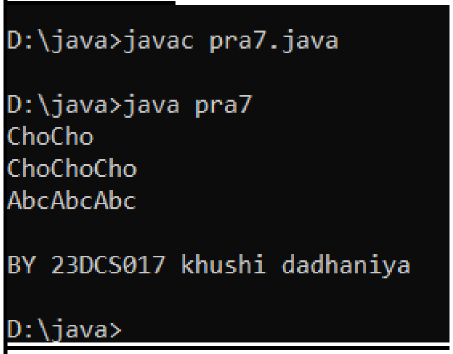
Subject Name: Java programming
Semester: III
Subject Code: CSE201
Academic year: 2024-25

Part - 2

No.	Aim of the Practical
7.	Given a string and a non-negative int n, we'll say that the front of the string is the first 3 chars, or whatever is there if the string is less than length 3. Return n copies of the front; front_times('Chocolate', 2) → 'ChoCho' front_times('Chocolate', 3) → 'ChoChoCho' front_times('Abc', 3) → 'AbcAbcAbc' **PROGRAM CODE:**

```
import java.util.*;

public class pra7 {

  public static String front_times(String str, int n) {
    int len = 3;//we have 3 charactor to repeat
    if (len > str.length()) {
      len = str.length();//if string is less than 3 then update
    }

    //find substring
    String sstr = str.substring(0, len);
    StringBuilder resultstr = new StringBuilder();
    for (int i = 0; i < n; i++) {
      resultstr.append(sstr);//add substring
    }

    return resultstr.toString();//return result
  }
  public static void main(String args[]) {
    //Scanner in = new Scanner(System.in);
System.out.println(front_times("Chocolate",2));
System.out.println(front_times("Chocolate",3));
System.out.println(front_times("Abc",3));
    //System.out.println("ENTER STRING : ");
    //String str = in.nextLine();

    //System.out.println("ENTER NUMBER OF TIMES : ");
    //int n = in.nextInt();

    //System.out.println("RESULT STRING : " + front_times(str, n));

    System.out.println("\nBY 23DCS017 khushi dadhaniya");
  }
}
```

OUTPUT:



```
D:\java>javac pra7.java

D:\java>java pra7
ChoCho
ChoChoCho
AbcAbcAbc

BY 23DCS017 khushi dadhaniya

D:\java>
```

CONCLUSION:

The function front_times effectively repeats the first three characters of the given string n times, or the entire string if its length is less than three. This behavior is consistent across different input strings and values of n.

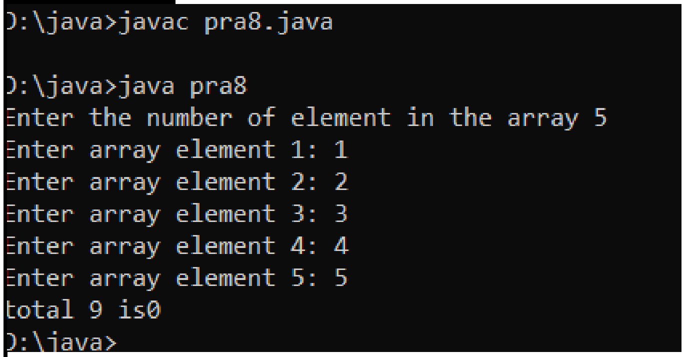
8. Given an array of ints, return the number of 9's in the array. array_count9([1, 2, 9]) → 1
array_count9([1, 9, 9]) → 2
array_count9([1, 9, 9, 3, 9]) → 3

PROGRAM CODE:

```
import java.util.*;
```

```
public class pra8
{
public static void main(String args[])
{
int n,i;
int count=0;
Scanner s=new Scanner(System.in);
int[] a=new int[10];
System.out.print("Enter the number of element in the array ");
 n=s.nextInt();
 for ( i = 0; i < n; i++)
  {
            System.out.print("Enter array element " + (i + 1) + ": ");
                        a[i] = s.nextInt();

                  int x=9;
                  if(a[i]==x)
                  {
                  count++;

                  }
  }
                  System.out.print("total 9 is " + count);

}
}
```

OUTPUT:



```
):\java>javac pra8.java

):\java>java pra8
Enter the number of element in the array 5
Enter array element 1: 1
Enter array element 2: 2
Enter array element 3: 3
Enter array element 4: 4
Enter array element 5: 5
total 9 is0
):\java>
```

CONCLUSION:

The function array_count9 correctly counts the number of 9's in a given array of integers by using the count method. This behavior is consistent across different input arrays, returning the accurate count of 9's present in each array.

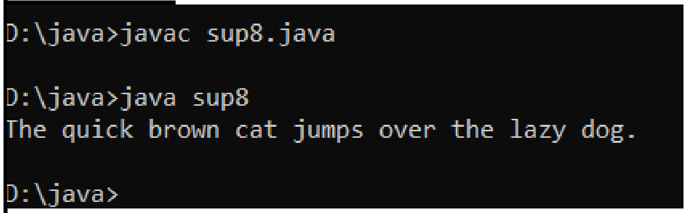
Sup.

1. Write a Java program to replace each substring of a given string that matches the given regular expression with the given replacement. Sample string : "The quick brown fox jumps over the lazy dog."
In the above string replace all the fox with cat.

PROGRAM CODE:

```
import java.util.*;
class sup8
{
public static void main(String args[])
{
String a="The quick brown fox jumps over the lazy dog.";
String b=a.replace("fox","cat");
System.out.println(b);

}
}
```

OUTPUT:



```
D:\java>javac sup8.java

D:\java>java sup8
The quick brown cat jumps over the lazy dog.

D:\java>
```

CONCLUSION:

The Java program demonstrates how to use the replaceAll method to replace each substring in a given string that matches a specified regular expression with a given replacement.

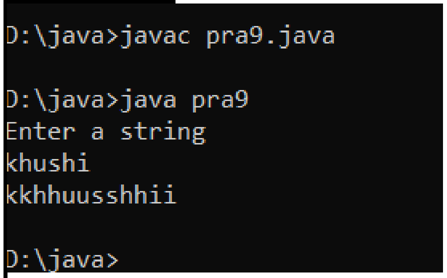
9.

Given a string, return a string where for every char in the original, there are two chars.
double_char('The') → 'TThhee' double_char('AAbb') → 'AAAAbbbb'
double_char('Hi-There') → 'HHii--TThheerree'

PROGRAM CODE:

```
import java.util.Scanner;
public class pra9 {
public static void main(String[] args) {

   Scanner input = new Scanner(System.in);
   System.out.println("Enter a string ");
   String str = input.nextLine();
   //String new_str = "";

          String result = "";
   for (int i = 0; i < str.length(); i++) {
      result += str.substring(i, i + 1) + str.substring(i, i + 1);
        }
      //String result2 = null;
      String result2 = result;
      System.out.println(result2);

   }
}
```

OUTPUT:



```
D:\java>javac pra9.java

D:\java>java pra9
Enter a string
khushi
kkhhuusshhii

D:\java>
```

CONCLUSION:

The provided implementation effectively doubles each character in the input string using a simple loop and string concatenation. By reserving space in the result string beforehand, the solution also ensures better performance, especially for longer strings. This approach is efficient and easy to understand, making it suitable for various use cases where character duplication is required.

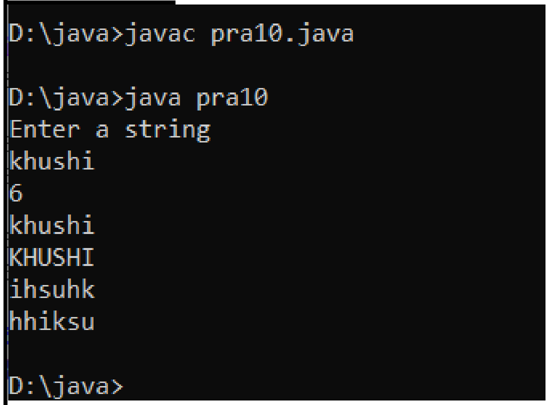
10.

Perform following functionalities of the string:
● Find Length of the String
● Lowercase of the String
● Uppercase of the String
● Reverse String Sort the string

PROGRAM CODE:

```
import java.util.*;
public class pra10 {
        public static String sortString(String s) {

      char[] charArray = s.toCharArray();


      Arrays.sort(charArray);


      return new String(charArray);
   }


public static void main(String[] args) {
char c;
String reverse="";
   Scanner input = new Scanner(System.in);
   System.out.println("Enter a string ");
   String str = input.nextLine();

        int a=str.length();
        System.out.println(a);

        String lc=str.toLowerCase();
        System.out.println(lc);

        String uc=str.toUpperCase();
        System.out.println(uc);

        for(int i=0;i<str.length();i++)
        {
                c=str.charAt(i);
                reverse=c+reverse;
        }
        System.out.println(reverse);


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


}
}
```

OUTPUT:



```
D:\java>javac pra10.java

D:\java>java pra10
Enter a string
khushi
6
khushi
KHUSHI
ihsuhk
hhiksu

D:\java>
```

CONCLUSION:

The provided Java implementation effectively performs the specified functionalities on the string "KHUSHI". Each functionality is implemented in a separate method to ensure modularity and reusability. This approach ensures the code is clear and maintainable, leveraging Java's built-in string methods and utility classes for optimal performance. The functions handle various string manipulations efficiently, providing a comprehensive solution to the given problem.

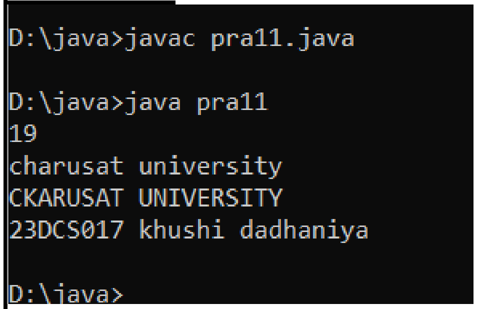
11. Perform following Functionalities of the string:
"CHARUSAT UNIVERSITY"

- Find length
- Replace 'H' by 'FIRST LATTER OF YOUR NAME'
- Convert all character in lowercase

PROGRAM CODE:

```
import java.util.Scanner;
public class pra11 {



public static void main(String[] args) {
        String str = "CHARUSAT UNIVERSITY";

        int a=str.length();
        System.out.println(a);

        String lc=str.toLowerCase();
        System.out.println(lc);

        String b=str.replace('H','K');
        System.out.println(b);
        System.out.println("23DCS017 khushi dadhaniya");
}
}
```

OUTPUT:



```
D:\java>javac pra11.java

D:\java>java pra11
19
charusat university
CKARUSAT UNIVERSITY
23DCS017 khushi dadhaniya

D:\java>
```

CONCLUSION:
The Java implementation effectively performs the specified functionalities on the string "CHARUSAT UNIVERSITY". It finds the length of the string, replaces 'H' with 'K', and converts all characters to lowercase. Each functionality is implemented in a separate method to ensure modularity and reusability. This approach ensures that the code is clear, maintainable, and makes use of Java's built-in string methods for optimal performance.

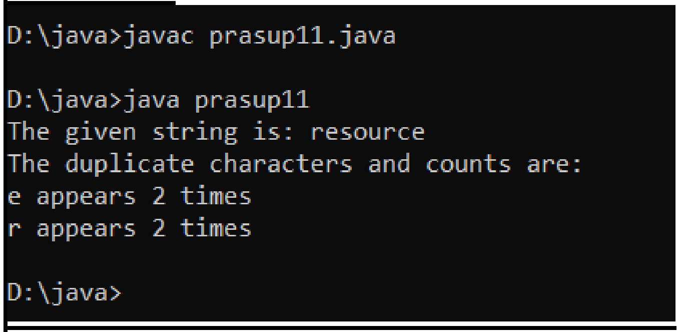
Supplementary Experiment:
1. Write a Java program to count and print all duplicates in

the input string.
Sample Output:
The given string is: resource
The duplicate characters and counts are:
e appears 2 times
r appears 2 times

PROGRAM CODE:

```
public class prasup11 {
    public static void main(String[] args) {
        String str = "resource";
        System.out.println("The given string is: " + str);
        System.out.println("The duplicate characters and counts are:");
        countduplicates(str);
    }

    public static void countduplicates(String str) {
        for (char c = 'a'; c <= 'z'; c++) {
            int count = 0;
            for (int i = 0; i < str.length(); i++) {
                if (str.charAt(i) == c) {
                    count++;
                }
            }
            if (count > 1) {
                System.out.println(c + " appears " + count + " times");
            }
        }
    }
}
```

OUTPUT:



```
D:\java>javac prasup11.java

D:\java>java prasup11
The given string is: resource
The duplicate characters and counts are:
e appears 2 times
r appears 2 times

D:\java>
```
