**CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY**

**DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH**

Department of Computer Science & Engineering

**Subject Name: Java Programming**

**Semester: 3rd**

**Subject Code: CSE201**

**Academic year: 2024-2025**

**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.lang.\*;  import java.util.Scanner;  class chocho  {  static int front\_times(String s, int n)  {  String s1=s.substring(0,3);  int i;  for(i=0;i<n;i++)  {  System.out.print(s1);  }  return 0;  }  public static void main(String a[])  {  /\*Scanner s=new Scanner(System.in);  System.out.println("Enter your string: ");  String str=s.nextLine();  System.out.println("Enter the number of times you want to print the string: ");  int num=s.nextInt();\*/    String str="Chocolate";  front\_times(str,2);  System.out.println(" ");  front\_times(str,3);  System.out.println(" ");  front\_times("Abc",3);  System.out.println(" ");  }  }    **OUTPUT:**    **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.**  **SUPP.** | 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.lang.\*;  import java.util.Scanner;  class countarr  {  static void array\_count9(int []a)  {  int i;  int count=0;  for(i=0;i<a.length;i++)  {  if(a[i]==9)  {  count++;  }  }  System.out.println(count);  }    public static void main(String args[])  {  int []arr={1,2,9};  int []arr1={1,9,9};  int []arr2={1,9,9,3,9};  System.out.println("The number of 9(s) in array {1,2,9} is: ");  array\_count9(arr);  System.out.println("The number of 9(s) in array {1,9,9} is: ");  array\_count9(arr1);  System.out.println("The number of 9(s) in array {1,9,9,3,9} is: ");  array\_count9(arr2);  }  }  **OUTPUT:**    **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.  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 supp3  {  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:**    **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;  class char\_double  {  static void double\_char(String s)  {  int i;  for(i=0;i<s.length();i++)  {  char s1=s.charAt(i);  System.out.print(s1);  System.out.print(s1);  }  }    public static void main(String args[])  {  String str="The";  double\_char(str);  System.out.println("");  double\_char("AAbb");  System.out.println("");  double\_char("Hi-There");  }  }  **OUTPUT:**    **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.\*;  class str\_fn  {  public static String sortString(String s)  {  char[] charArray = s.toCharArray();  Arrays.sort(charArray);  return new String(charArray);  }  public static void main(String a[])  {  Scanner s=new Scanner(System.in);  System.out.println("Enter a string: ");  String s1=s.nextLine();    int len=s1.length();  System.out.println("The length of the given string is: "+len);    String up=s1.toUpperCase();  String low=s1.toLowerCase();  System.out.println("The string in upper case is: "+up);  System.out.println("The string in lower case is: "+low);    int i;  System.out.println("The reversed string is: ");  for(i=len-1;i>=0;i--)  {  System.out.print(s1.charAt(i));  }    System.out.println("The sorted string is: "+sortString(s1));        }  }  **OUTPUT:**    **CONCLUSION:**  The provided Java implementation effectively performs the specified functionalities on the string "JUGvi". Each functionality is implemented in a separate method to ensure modularity and reusability. |
| **11.**  **Supp:** | 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;  class charusat  {  public static void main(String a[])  {  String str="CHARUSAT UNIVERSITY";  System.out.println("Given string: "+str);    System.out.println("The length is: "+str.length());    System.out.println("The string in lower case is: "+str.toLowerCase());    System.out.println("The modified string is: "+str.replace('H','J'));  }  }  **OUTPUT:**    **CONCLUSION:**  The Java implementation effectively performs the specified functionalities on the string "CHARUSAT UNIVERSITY". It finds the length of the string, replaces 'H' with 'J', and converts all characters to lowercase.  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:**  import java.util.Scanner;  class duplicate  {  public static void main(String a[])  {  Scanner s=new Scanner(System.in);  System.out.println("Enter a string: ");  String s1=s.nextLine();  int i,j,count=0;  char str[]=s1.toCharArray();  System.out.println("The duplicate characters are: ");  for(i=0;i<str.length;i++)  {  count=1;  for(j = i+1;j <str.length;j++)  {  if(str[i] == str[j] && str[i] != ' ')  {  count++;  str[j] = '0';  }  }    if(count > 1 && str[i] != '0')  {  System.out.println(str[i]+" appears "+count+" times ");    }  }  }  }  **OUTPUT:**    **CONCLUSION:**  Hence, all the duplicate characters are displayed as the loop runs and the count is incremented and displayed accordingly. |