



**Devang Patel Institute of
Advance Technology and Research**
(A Constitute Institute of CHARUSAT)

Certificate

This is to certify that

Mr./Mrs. Ved N. Patel

of 3 CSE I *Class,*

ID. No. D24DCS148 *has satisfactorily completed*

his/ her term work in Java programming (201) *for*

the ending in Oct 2024/2025

Date : 17/10/24


Sign. of Faculty


Head of Department

CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY**DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH**

Department of Computer Science & Engineering

Subject Name : JAVA PROGRAMMING

Semester: 3

Subject Code: CSE201

Academic Year: 2024 - 2025

Part - 1

NO.	Aim of the practical
1	<p>Demonstration of installation steps of Java, Introduction to Object Oriented Concepts, comparison of Java with other object-oriented programming languages. Introduction to JDK, JRE, JVM, Javadoc, command line argument. Introduction to Eclipse or NetBeans IDE, or BlueJ and Console Programming.</p> <p>1. Installation Steps for Java</p> <p>1. Download JDK:</p> <ul style="list-style-type: none">• Go to the Oracle JDK download page.• Select the appropriate version for Windows and download the installer. <p>2. Install JDK:</p> <ul style="list-style-type: none">• Run the downloaded installer.• Follow the installation prompts and complete the installation. <p>3. Set Environment Variables:</p> <ul style="list-style-type: none">• Open the Start menu and search for "Environment Variables".• In System Properties, click on "Environment Variables".• Under System Variables, click "New" and add JAVA_HOME as the variable name and the JDK installation path as the variable value.• Find the Path variable in System Variables, click "Edit", and add %JAVA_HOME%\bin. <p>4. Verify Installation:</p> <ul style="list-style-type: none">• Open Command Prompt and type java -version to verify the installation.

2. Introduction to Object-Oriented Concepts

- **Object-Oriented Programming (OOP):** A programming paradigm based on the concept of objects, which can contain data and code that manipulates that data.
 - **Class:** A blueprint for creating objects (a particular data structure).
 - **Object:** An instance of a class.
 - **Inheritance:** A mechanism where a new class inherits the properties and behaviors of an existing class.
 - **Polymorphism:** The ability of different classes to be treated as instances of the same class through a common interface.
 - **Encapsulation:** Bundling of data and methods that operate on the data within one unit, e.g., a class.
 - **Abstraction:** The concept of hiding the complex implementation details and showing only the necessary features.

3. Comparison of Java with Other Object-Oriented Programming Languages

- **Java vs. C++:**
 - Java is platform-independent due to its JVM, while C++ is platform-dependent.
 - Java has automatic garbage collection, whereas C++ requires manual memory management.
 - Java does not support pointers, while C++ does.
- **Java vs. Python:**
 - Java is statically typed, meaning type checking is done at compile time. Python is dynamically typed.
 - Java is generally faster in execution due to its compiled nature, while Python is interpreted.
 - Java syntax is more verbose, while Python's syntax is simpler and more concise.

4. INTRODUCTION:

JDK (Java Development Kit)

- **Definition:** The JDK is a software development kit used to develop Java applications and applets. It includes tools for developing and testing programs written in the Java programming language and running on the Java platform.
- **Components:**

- **Compiler (javac):** Converts Java source code into bytecode.
- **Java Runtime Environment (JRE):** Allows running of Java applications.
- **Java Debugger (jdb):** Assists in debugging Java programs.
- **Other Tools:** Includes an archiver (jar), a documentation generator (javadoc), and various other development tools.

JRE (Java Runtime Environment)

- **Definition:** The JRE is a part of the JDK but can be downloaded separately. It provides the libraries, Java Virtual Machine (JVM), and other components necessary to run applications written in Java.
- **Components:**
 - **JVM:** Runs Java bytecode.
 - **Libraries:** Standard libraries required for Java programs to run.

JVM (Java Virtual Machine)

- **Definition:** The JVM is an abstract computing machine that enables a computer to run Java programs. It is platform-independent, meaning the same Java program can run on any device that has a JVM.
- **Functions:**
 - **Class Loader:** Loads class files.
 - **Bytecode Verifier:** Ensures the bytecode is valid and does not breach Java's security constraints.
 - **Interpreter/Just-In-Time Compiler:** Converts bytecode into machine code, executing it on the host machine.

Javadoc

- **Definition:** Javadoc is a tool provided by the JDK for generating API documentation in HTML format from Java source code comments.
- **Usage:**
 - **Tags:** Use special tags like @param, @return, and @throws to describe method parameters, return values, and exceptions.
 - **Command:** Run javadoc <source files> to generate documentation.

Command Line Arguments

- **Definition:** Arguments passed to a Java program at the time of execution from the command line.
- **Usage:**
 - **Accessing Arguments:** They are accessed in the main method's args array parameter.

Introduction to Eclipse, NetBeans IDE, BlueJ, and Console Programming

Eclipse IDE

- **Definition:** Eclipse is a popular open-source integrated development environment (IDE) for Java development.
- **Features:**
 - **Code Editor:** Provides syntax highlighting, code completion, and refactoring tools.
 - **Debugging Tools:** Integrated debugger for stepping through code and setting breakpoints.
 - **Plugins:** Supports various plugins to extend functionality.
- **Installation:**
 - Download from the Eclipse website.
 - Extract the downloaded file and run the eclipse executable.
- **Creating a Java Project:**
 - Open Eclipse.
 - Go to File -> New -> Java Project.
 - Enter the project name and click Finish.

NetBeans IDE

- **Definition:** NetBeans is an open-source IDE for Java development, known for its simplicity and ease of use.
- **Features:**
 - **Code Editor:** Offers syntax highlighting, code templates, and easy navigation.
 - **GUI Builder:** Allows for drag-and-drop design of user interfaces.
 - **Integrated Tools:** Includes support for version control systems, debugging, and profiling.

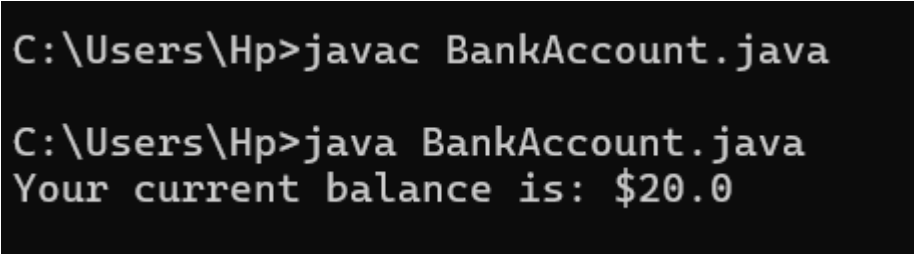
- **Installation:**
 - Download from the [NetBeans website](#).
 - Run the installer and follow the setup instructions.
- **Creating a Java Project:**
 - Open NetBeans.
 - Go to File -> New Project.
 - Select Java and Java Application.
 - Enter the project name and click Finish.

BlueJ

- **Definition:** BlueJ is a simple IDE for beginners, designed to teach object-oriented programming.
- **Features:**
 - **User-Friendly Interface:** Simplified interface aimed at new programmers.
 - **Interactive:** Allows direct interaction with objects.
 - **Visualization:** Provides class structure visualization.
- **Installation:**
 - Download from the BlueJ website.
 - Run the installer and follow the setup instructions.
- **Creating a Java Project:**
 - Open BlueJ.
 - Go to Project -> New Project.
 - Enter the project name and click Create.

Console Programming

- **Definition:** Console programming refers to writing programs that interact with the user through a command-line interface.

2	<p>Imagine you are developing a simple banking application where you need to display the current balance of a user account. For simplicity, let's say the current balance is \$20. Write a java program to store this balance in a variable and then display it to the user.</p> <p>PROGRAM CODE:</p> <pre>public class BankAccount { public static void main(String[] args) { // Store the current balance in a variable double currentBalance = 20.00; // Display the balance to the user System.out.println("Your current balance is: \$" + currentBalance); } }</pre> <p>OUTPUT:</p>  <p>CONCLUSION:</p> <p>In this program, we defined a BankAccount class with a main method. Inside the main method, we stored the user's current balance in a variable named currentBalance and then displayed it using the System.out.println method.</p>
3	<p>Write a program to take the user for a distance (in meters) and the time taken (as three numbers: hours, minutes, seconds), and display the speed, in meters per second, kilometers per hour and miles per hour (hint: 1 mile = 1609 meters).</p> <p>PROGRAM CODE:</p> <pre>import java.util.Scanner; public class SpeedCalculator { public static void main(String[] args) { Scanner scanner = new Scanner(System.in); System.out.print("Enter distance (in meters): "); double distance = scanner.nextDouble(); System.out.print("Enter time - hours: "); int hours = scanner.nextInt(); System.out.print("Enter time - minutes: "); int minutes = scanner.nextInt(); System.out.print("Enter time - seconds: "); int seconds = scanner.nextInt();</pre>

	<pre> int totalTimeInSeconds = hours * 3600 + minutes * 60 + seconds; double speedMetersPerSecond = distance / totalTimeInSeconds; double speedKilometersPerHour = (distance / 1000.0) / (totalTimeInSeconds / 3600.0); double speedMilesPerHour = (distance / 1609.0) / (totalTimeInSeconds / 3600.0); System.out.println("Speed in meters per second: " + speedMetersPerSecond); System.out.println("Speed in kilometers per hour: " + speedKilometersPerHour); System.out.println("Speed in miles per hour: " + speedMilesPerHour); scanner.close(); } } </pre> <p>OUTPUT:</p> <pre> C:\Users\Hp>javac SpeedCalculator.java C:\Users\Hp>java SpeedCalculator.java Enter distance (in meters): 50000 Enter time - hours: 2 Enter time - minutes: 50 Enter time - seconds: 32 Speed in meters per second: 4.886630179827991 Speed in kilometers per hour: 17.591868647380764 Speed in miles per hour: 10.933417431560452 </pre> <p>CONCLUSION: This Java program demonstrates how to take input from the user, perform basic calculations, and display the results. The program calculates the speed in three different units: meters per second, kilometers per hour, and miles per hour, based on the distance and time provided by the user.</p>
4	<p>Imagine you are developing a budget tracking application. You need to calculate the total expenses for the month. Users will input their daily expenses, and the program should compute the sum of these expenses. Write a Java program to calculate the sum of elements in an array representing daily expenses.</p> <p>PROGRAM CODE:</p> <pre> import java.util.Scanner; public class BudgetTracker { public static void main(String[] args) { Scanner scanner = new Scanner(System.in); </pre>


```
System.out.print("Enter the number of days in the month: ");
int numberOfDays = scanner.nextInt();

double[] dailyExpenses = new double[numberOfDays];

for (int i = 0; i < numberOfDays; i++) {
    System.out.print("Enter the expense for day " + (i + 1) + ": ");
    dailyExpenses[i] = scanner.nextDouble();
}

double totalExpenses = 0;
for (int i = 0; i < numberOfDays; i++) {
    totalExpenses += dailyExpenses[i];
}

System.out.println("Total expenses for the month: $" + totalExpenses);

scanner.close();
}
```

OUTPUT:

```
C:\Users\Hp>javac BudgetTracker.java

C:\Users\Hp>java BudgetTracker
Enter the number of days in the month: 30
Enter the expense for day 1: 222
Enter the expense for day 2: 1212
Enter the expense for day 3: 156
Enter the expense for day 4: 54646
Enter the expense for day 5: 121
Enter the expense for day 6: 211
Enter the expense for day 7: 2255
Enter the expense for day 8: 221
Enter the expense for day 9: 222
Enter the expense for day 10: 565
Enter the expense for day 11: 551
Enter the expense for day 12: 5889
Enter the expense for day 13: 44
Enter the expense for day 14: 669
Enter the expense for day 15: 264
Enter the expense for day 16: 2664
Enter the expense for day 17: 1644
Enter the expense for day 18: 664
Enter the expense for day 19: 969
Enter the expense for day 20: 33
Enter the expense for day 21: 1547
Enter the expense for day 22: 546
Enter the expense for day 23: 5666
Enter the expense for day 24: 4456
Enter the expense for day 25: 1623
Enter the expense for day 26: 2213
Enter the expense for day 27: 213
Enter the expense for day 28: 1233
Enter the expense for day 29: 665
Enter the expense for day 30: 656
Total expenses for the month: $92040.0
```

CONCLUSION:

This Java program demonstrates how to use arrays and loops to collect multiple inputs from the user, perform a summation operation, and then display the result. The program prompts the user to input daily expenses for a specified number of days, stores these values in an array, calculates the sum of the array elements to find the total monthly expenses, and displays the result.

Supplementary Experiment:

You are creating a library management system. The library has two separate lists of books for fiction and non-fiction. The system should merge these lists into a single list for Inventory purposes. Write a Java program to merge two arrays.

PROGRAM CODE:

```
public class LibraryManagementSystem {
    public static void main(String[] args) {
        String[] fictionBooks = {
            "To Kill a Mockingbird",
            "1984",
            "The Great Gatsby",
            "The Catcher in the Rye"
        };

        String[] nonFictionBooks = {
            "Sapiens: A Brief History of Humankind",
            "Educated",
            "The Immortal Life of Henrietta Lacks",
            "Unbroken"
        };

        String[] inventory = mergeArrays(fictionBooks, nonFictionBooks);

        System.out.println("Merged Inventory List:");
        for (String book : inventory) {
            System.out.println(book);
        }

        public static String[] mergeArrays(String[] array1, String[] array2) {
            String[] mergedArray = new String[array1.length + array2.length];
            int index = 0;

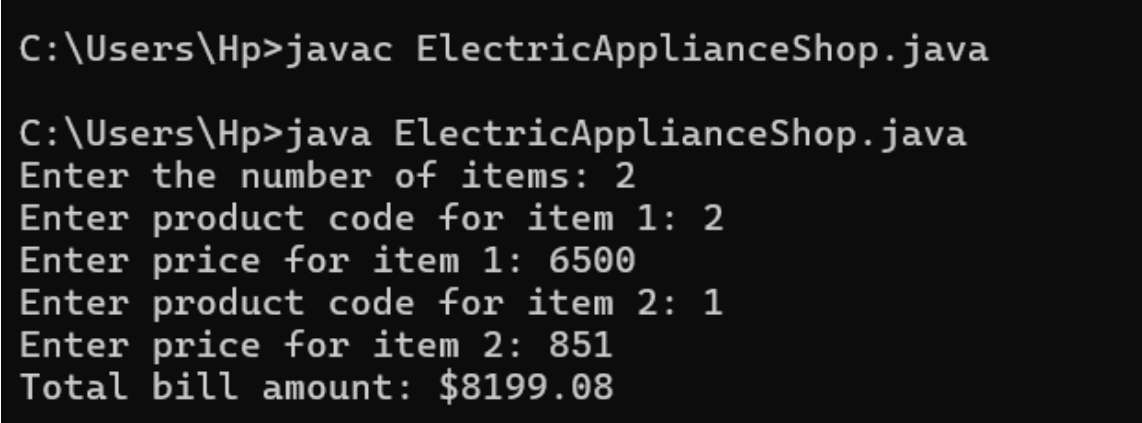
            for (String element : array1) {
                mergedArray[index++] = element;
            }

            for (String element : array2) {
                mergedArray[index++] = element;
            }

            return mergedArray;
        }
    }
}
```

OUTPUT:

	<pre> C:\Users\Hp>javac LibraryManagementSystem.java C:\Users\Hp>java LibraryManagementSystem Merged Inventory List: To Kill a Mockingbird 1984 The Great Gatsby The Catcher in the Rye Sapiens: A Brief History of Humankind Educated The Immortal Life of Henrietta Lacks Unbroken </pre>
5	<p>An electric appliance shop assigns code 1 to motor,2 to fan,3 to tube and 4 for wires. All other items have code 5 or more. While selling the goods, a sales tax of 8% to motor,12% to fan,5% to tube light,7.5% to wires and 3% for all other items is charged. A list containing the product code and price in two different arrays. Write a java program using switch statement to prepare the bill.</p> <p>PROGRAM CODE:</p> <pre> import java.util.Scanner; public class ElectricApplianceShop { public static void main(String[] args) { Scanner scanner = new Scanner(System.in); System.out.print("Enter the number of items: "); int numberOfItems = scanner.nextInt(); int[] productCodes = new int[numberOfItems]; double[] prices = new double[numberOfItems]; for (int i = 0; i < numberOfItems; i++) { System.out.print("Enter product code for item " + (i + 1) + ": "); productCodes[i] = scanner.nextInt(); System.out.print("Enter price for item " + (i + 1) + ": "); prices[i] = scanner.nextDouble(); } double totalBill = 0.0; for (int i = 0; i < numberOfItems; i++) { double taxRate = 0.0; switch (productCodes[i]) { case 1: </pre>

	<pre> taxRate = 0.08; // 8% for motor break; case 2: taxRate = 0.12; // 12% for fan break; case 3: taxRate = 0.05; // 5% for tube light break; case 4: taxRate = 0.075; // 7.5% for wires break; default: taxRate = 0.03; // 3% for all other items break; } double taxAmount = prices[i] * taxRate; double itemTotal = prices[i] + taxAmount; totalBill += itemTotal; } System.out.printf("Total bill amount: \$%.2f\n", totalBill); scanner.close(); } } </pre> <p>OUTPUT:</p>  <pre> C:\Users\Hp>javac ElectricApplianceShop.java C:\Users\Hp>java ElectricApplianceShop.java Enter the number of items: 2 Enter product code for item 1: 2 Enter price for item 1: 6500 Enter product code for item 2: 1 Enter price for item 2: 851 Total bill amount: \$8199.08 </pre> <p>CONCLUSION: This Java program demonstrates how to use arrays and a switch statement to calculate the total bill for items purchased in an electric appliance shop. The program applies different tax rates based on the product code and sums up the total amount including tax for each item</p>
6	Create a Java program that prompts the user to enter the number of days (n) for which they want to generate their exercise routine. The program should then calculate and

display the first n terms of the Fibonacci series, representing the exercise duration for each day.

PROGRAM CODE:

```
import java.util.Scanner;

public class ExerciseRoutine {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of days for exercise routine: ");
        int n = scanner.nextInt();

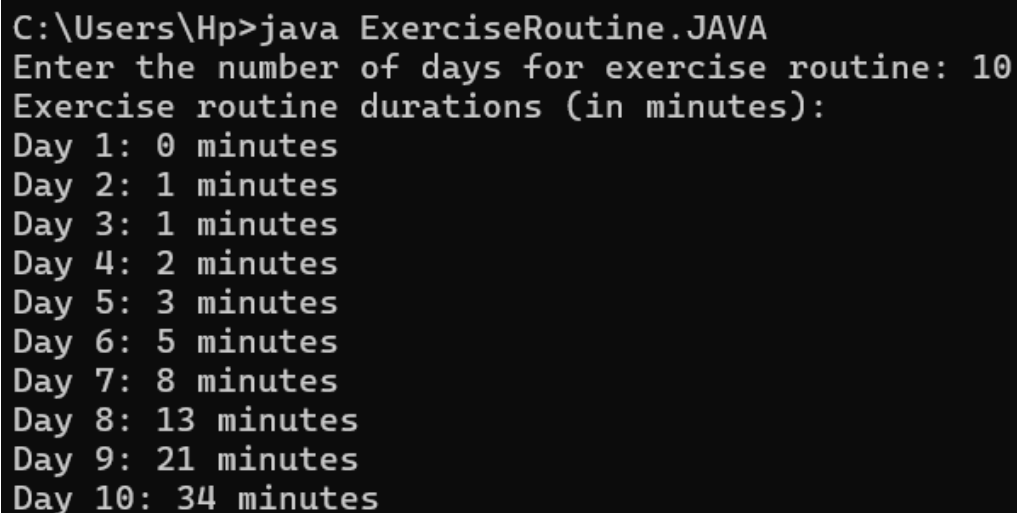
        int a = 0, b = 1;

        System.out.println("Exercise routine durations (in minutes):");

        for (int i = 1; i <= n; i++) {
            System.out.println("Day " + i + ": " + a + " minutes");
            int next = a + b;
            a = b;
            b = next;
        }

        scanner.close();
    }
}
```

OUTPUT:



```
C:\Users\Hp>java ExerciseRoutine.JAVA
Enter the number of days for exercise routine: 10
Exercise routine durations (in minutes):
Day 1: 0 minutes
Day 2: 1 minutes
Day 3: 1 minutes
Day 4: 2 minutes
Day 5: 3 minutes
Day 6: 5 minutes
Day 7: 8 minutes
Day 8: 13 minutes
Day 9: 21 minutes
Day 10: 34 minutes
```

CONCLUSION:

This Java program demonstrates how to use the Fibonacci series to generate an exercise routine. The program prompts the user for the number of days and calculates the Fibonacci series up to that number of terms, displaying the exercise duration for each day

Supplementary Experiment: Imagine you are developing a classroom management system. You need to keep track of the grades of students in a class. After collecting the grades, you want to display each student's grade along with a message indicating if they have passed or failed. Let's assume the passing grade is 50.

PROGRAM CODE:

```
import java.util.Scanner;

public class ClassroomManagement {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of students: ");
        int numberOfStudents = scanner.nextInt();

        String[] studentNames = new String[numberOfStudents];
        int[] grades = new int[numberOfStudents];

        \ for (int i = 0; i < numberOfStudents; i++) {
            System.out.print("Enter name of student " + (i + 1) + ": ");
            studentNames[i] = scanner.next();
            System.out.print("Enter grade for " + studentNames[i] + ": ");
            grades[i] = scanner.nextInt();
        }

        System.out.println("\nStudent Grades:");
        for (int i = 0; i < numberOfStudents; i++) {
            String status = grades[i] >= 50 ? "Passed" : "Failed";
            System.out.println(studentNames[i] + ": " + grades[i] + " - " + status);
        }

        scanner.close();
    }
}
```

OUTPUT:

```
C:\Users\Hp>java ClassroomManagement.java
Enter the number of students: 2
Enter name of student 1: Akai
Enter grade for Akai: 78
Enter name of student 2: Khloe
Enter grade for Khloe: 99

Student Grades:
Akai: 78 - Passed
Khloe: 99 - Passed
```

Part - 2

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:

```
package str;
public class p1 {
    public static String frontTimes(String str, int n) {
        String front = str.length() < 3 ? str : str.substring(0, 3);

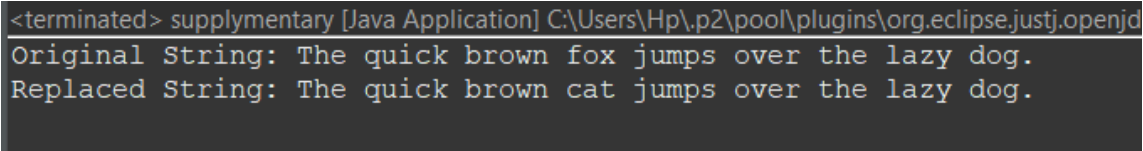
        String result = "";
        for (int i = 0; i < n; i++) {
            result += front;
        }

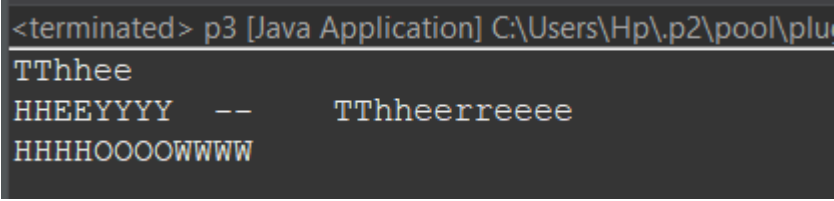
        return result;
    }

    public static void main(String[] args) {
        System.out.println(frontTimes("Chocolate", 2));
        System.out.println(frontTimes("Chocolate", 4));
        System.out.println(frontTimes("Abc", 3));
    }
}
```

OUTPUT:

	<pre><terminated> p1 [Java Application] C:\Users\ ChoCho ChoChoChoCho AbcAbcAbc</pre> <p>CONCLUSION: In this practical, we used Java's string manipulation capabilities to extract a substring and repeat it. By employing the <code>substring()</code> method, we efficiently obtained the first three characters (or the entire string if shorter). Using string concatenation, we repeated this front portion <i>n</i> times, illustrating how Java handles string operations seamlessly.</p>
08	<p>Given an array of ints, return the number of 9's in the : array_count9([1, 2, 9]) → 1 array_count9([1, 9, 9]) → 2 array_count9([1, 9, 9, 3, 9]) → 3</p> <p>PROGRAM CODE:</p> <pre>package str; public class p2 { public static int arrc(int[] arr) { int count = 0 ; for(int i : arr) { if (i == 9) { count++; } } return count; } public static void main(String[] args) { System.out.println("count of 9 in : {1,2,3,4,5,6,7,8,9,9,9} is : " + arrc(new int[] {1,2,3,4,5,6,7,8,9,9,9})); System.out.println("count of 9 in :{90,99,87,2,9} is : "+ arrc(new int[] {90,99,87,2,9})); } }</pre> <p>OUTPUT:</p> <pre><terminated> p2 [Java Application] C:\Users\Hp\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot count of 9 in : {1,2,3,4,5,6,7,8,9,9,9} is : 3 count of 9 in :{90,99,87,2,9} is : 1</pre> <p>CONCLUSION:</p>

	<p>This exercise demonstrated Java's powerful array handling by iterating over elements to count occurrences of a specific integer, such as 9. Using a simple <code>for</code> loop and conditional statements, we counted the target number efficiently. This task highlights Java's straightforward approach to array manipulation and reinforces the use of loops and conditionals for basic data processing.</p> <p>Supplementary Experiment:</p> <p>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."</p> <p>In the above string replace all the fox with cat.</p> <p>PROGRAM CODE:</p> <pre>package str; public class supplementary { public static void main(String[] args) { String originalString = "The quick brown fox jumps over the lazy dog."; String regex = "fox"; String replacement = "cat"; String replacedString = originalString.replaceAll(regex, replacement); System.out.println("Original String: " + originalString); System.out.println("Replaced String: " + replacedString); } }</pre> <p>OUTPUT:</p> 
09	<p>Given a string, return a string where for every char in the original, there are two chars.</p> <pre>double_char('The') → 'TThhee' double_char('AAAb') → 'AAAAbbbb' double_char('Hi-There') → 'HHii--TThheerree'</pre> <p>PROGRAM CODE:</p> <pre>package str;</pre>

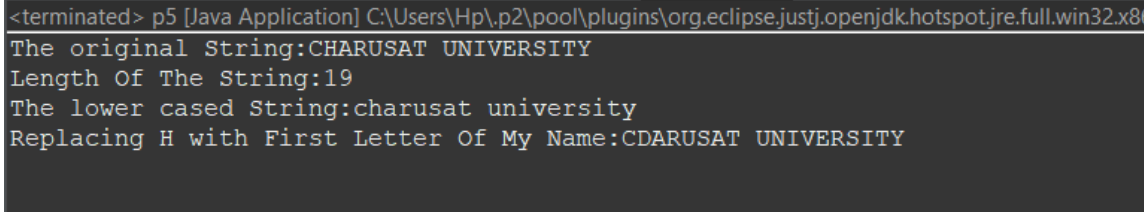
	<pre> public class p3 { public static String doublechar(String str) { String res = ""; for (int i =0;i<str.length();i++) { res += str.charAt(i) +"" + str.charAt(i); } return res; } public static void main(String[] args) { System.out.println(doublechar("The")); System.out.println(doublechar("HEYY - Theree")); System.out.println(doublechar("HHOOWW")); } } </pre> <p>OUTPUT:</p>  <p>CONCLUSION:</p> <p>The `doublechar` method effectively demonstrates string manipulation by duplicating each character in the input string using iteration and concatenation. While functional, the program highlights the importance of considering efficiency in string operations, potentially using `StringBuilder` for better performance in larger inputs. This exercise reinforces fundamental Java concepts such as string handling and loop constructs.</p>
10	<p>Perform following functionalities of the string:</p> <ul style="list-style-type: none"> ● Find Length of the String ● Lowercase of the String ● Uppercase of the String ● Reverse String ● Sort the String <p>PROGRAM CODE:</p> <pre> package str; import java.util.*; public class p4 { </pre>

```
public static String sortstr(String input_str) {  
    char[] temp = input_str.toCharArray();  
    Arrays.sort(temp);  
  
    return new String(temp);  
  
}  
  
public static void main(String[] args) {  
    String s = new String("Java is OOP Language");  
    String a = s.toLowerCase();  
    System.out.println("Original String: "+s);  
    System.out.println("Lower case String:"+a);  
    System.out.println("Uppercase String: "+s.toUpperCase());  
    StringBuffer sb = new StringBuffer(s);  
    sb.reverse();  
    String rev = sb.toString();  
    System.out.println("Reversed String:"+rev);  
  
    String ip = "FEELSFOR";  
    String op = sortstr(ip);  
  
    System.out.println("Input String:"+ip);  
    System.out.println("Output String:"+op);  
  
    }}  
  
OUTPUT:
```

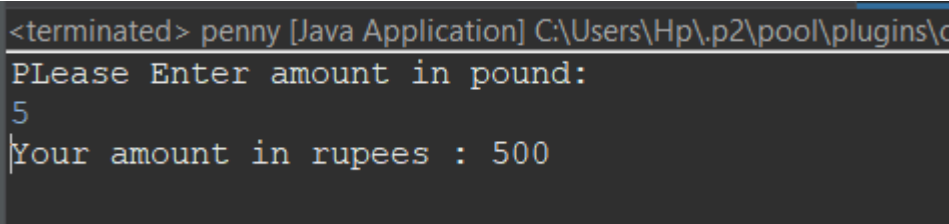
```
Original String: Java is OOP Language  
Lower case String:java is oop language  
Uppercase String: JAVA IS OOP LANGUAGE  
Reversed String:egaugnaL POO si avaJ  
Input String:FEELSFOR  
Output String:EEFFLORS
```

CONCLUSION:

Java's built-in methods for strings, such as `length()`, `toLowerCase()`, `toUpperCase()`, and custom methods for reversing, offer a comprehensive toolkit for basic string manipulations. This practical reinforces the understanding of these fundamental

	operations, showcasing Java's capability to handle string transformations efficiently and reliably.
11	<p>Perform following Functionalities of the string: "CHARUSAT UNIVERSITY"</p> <ul style="list-style-type: none"> ● Find length ● Replace 'H' by 'FIRST LATTER OF YOUR NAME' ● Convert all character in lowercase <p>PROGRAM CODE :</p> <pre>package str; public class p5 { public static void main(String[] args) { String s = new String("CHARUSAT UNIVERSITY"); System.out.println("The original String:"+s); System.out.println("Length Of The String:"+s.length()); String a = s.toLowerCase(); System.out.println("The lower cased String:"+a); String s2 = s.replace('H', 'D'); System.out.println("Replacing H with First Letter Of My Name:"+s2); } }</pre> <p>OUTPUT:</p>  <pre><terminated> p5 [Java Application] C:\Users\Hp\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64.jdk\bin\java.exe The original String:CHARUSAT UNIVERSITY Length Of The String:19 The lower cased String:charusat university Replacing H with First Letter Of My Name:CDARUSAT UNIVERSITY</pre> <p>CONCLUSION:</p> <p>This practical involved Java's string manipulation methods to perform tasks like finding the length, character replacement, and case conversion. By replacing 'H' with the first letter of a name and transforming the string to lowercase, we reinforced Java's capabilities in transforming strings using its standard library methods, highlighting its applicability in text processing.</p>

Part - 3

12	<p>Imagine you are developing a currency conversion tool for a travel agency. This tool should be able to convert an amount in Pounds to Rupees. For simplicity, we assume the conversion rate is fixed: 1 Pound = 100 Rupees. The tool should be able to take input both from command-line arguments and interactively from the user.</p> <p>PROGRAM CODE:</p> <pre>package part3; import java.util.*; public class penny { public static void main(String[] args) { Scanner sc = new Scanner(System.in); System.out.println("PLease Enter amount in pound: "); int a = sc.nextInt(); System.out.println("Your amount in rupees : "+ a*100); } }</pre> <p>OUTPUT:</p>  <p>CONCLUSION:</p> <p>In this practical, we used Java's string manipulation capabilities to extract a substring and repeat it. By employing the <code>substring()</code> method, we efficiently obtained the first three characters (or the entire string if shorter). Using string concatenation, we repeated this front portion <code>n</code> times, illustrating how Java handles string operations seamlessly.</p>
13	<p>Create a class called Employee that includes three pieces of information as instance variables—a first name (type String), a last name (type String) and a monthly salary (double). Your class should have a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the monthly salary is not positive, set it to 0.0. Write a test application named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10% raise and display each Employee's yearly salary again.</p> <p>PROGRAM CODE:</p> <p>- EMPLOY FILE :</p> <pre>package part3; public class Employ { private String fname;</pre>

```
private String lname;  
private Double salary;  
  
public Employ(String fname , String lname , Double salary) {  
    this.fname = fname;  
    this.lname = lname;  
    setsalary(salary);  
  
}  
public String get_fname() {  
    return fname;  
}  
  
public void set_fname(String fname) {  
    this.fname = fname;  
}  
  
public void setsalary(Double salary) {  
    if(salary <= 0) {  
        this.salary = 0.0;  
    }  
    else {  
        this.salary = salary;  
    }  
}  
  
public Double getsalary() {  
    return salary;  
  
}  
public String get_lname() {  
    return lname;  
}  
  
public void set_lname(String lname) {  
    this.lname = lname;  
}  
public double get_annual_income() {  
    return salary*12;  
}
```

```
public void raise(double per) {  
    salary += salary * (per/100);  
}  
}
```

EMPLOYEE MAIN FILE:

```
package part3;
```

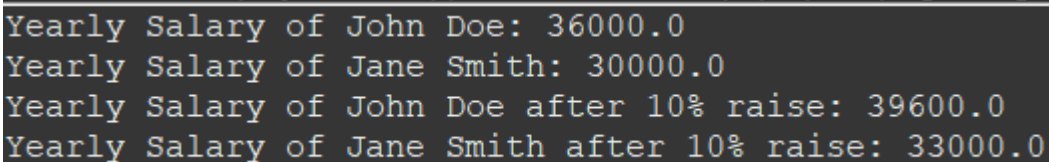
```
public class Employee {  
    public static void main(String[] args) {  
        Employ employee1 = new Employ("John", "Doe", 3000.00);  
        Employ employee2 = new Employ("Jane", "Smith", 2500.00);
```

```
        System.out.println("Yearly Salary of " + employee1.get_fname() + " " +  
            employee1.get_lname() + ": " + employee1.get_annual_income());  
        System.out.println("Yearly Salary of " + employee2.get_fname() + " " +  
            employee2.get_lname() + ": " + employee2.get_annual_income());
```

```
        employee1.raise(10.0);  
        employee2.raise(10.0);
```

```
        System.out.println("Yearly Salary of " + employee1.get_fname() + " " +  
            employee1.get_lname() + " after 10% raise: " + employee1.get_annual_income());  
        System.out.println("Yearly Salary of " + employee2.get_fname() + " " +  
            employee2.get_lname() + " after 10% raise: " + employee2.get_annual_income());  
    }  
}
```

OUTPUT:



```
Yearly Salary of John Doe: 36000.0  
Yearly Salary of Jane Smith: 30000.0  
Yearly Salary of John Doe after 10% raise: 39600.0  
Yearly Salary of Jane Smith after 10% raise: 33000.0
```

CONCLUSION:

This implementation of the `Employee` class successfully demonstrates the use of encapsulation and provides a simple way to manage employee data with salary calculations. The `EmployeeTest` class shows how we can create instances, manipulate data through setter methods, and calculate yearly salaries, while also applying raises and ensuring the integrity of the salary data.

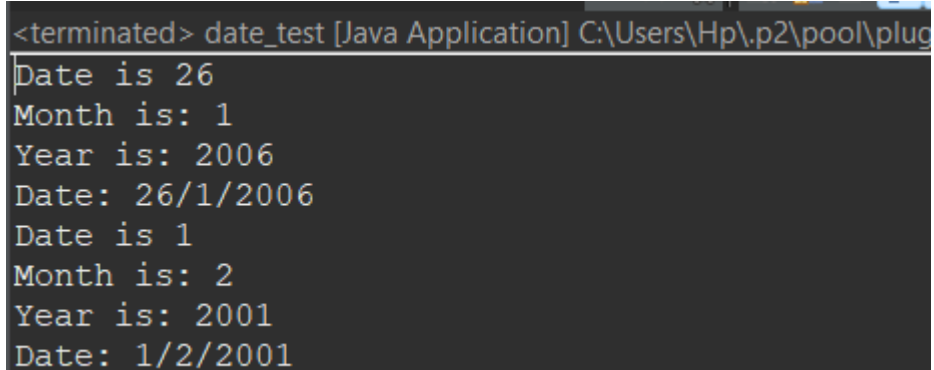
14	<p>Create a class called Date that includes three pieces of information as instance variables—a month (type int), a day (type int) and a year (type int). Your class should have a constructor that initializes the three instance variables and assumes that the values provided are correct. Provide a set and a get method for each instance variable. Provide a method displayDate that displays the month, day and year separated by forward slashes (/). Write a test application named DateTest that demonstrates class Date's capabilities.</p> <p>PROGRAM CODE :</p> <p>DATE FILE :</p> <pre>package part3; public class Date { private int day; private int month; private int year; public Date(int day , int month , int year) { this.day = day; this.month = month; this.year = year; } public int get_day() { return day; } public void set_day(int day) { this.day = day; } public int get_month() { return month; } public void set_month(int month) { this.month = month; } public int get_year() { return year; } public void set_year(int year) { this.year = year; } public void display(){ System.out.println("Date: " +day + "/" +month+ "/" +year); } }</pre>
----	---

```
}  
}
```

DATE_TEST FILE:

```
package part3;  
  
public class date_test {  
    public static void main(String[] args) {  
        Date d = new Date(26 , 01 , 2006);  
        System.out.println("Date is " + d.get_day());  
        System.out.println("Month is: " +d.get_month());  
        System.out.println("Year is: " +d.get_year());  
        d.display();  
        d.set_day(01);  
        d.set_month(02);  
        d.set_year(2001);  
        System.out.println("Date is " + d.get_day());  
        System.out.println("Month is: " +d.get_month());  
        System.out.println("Year is: " +d.get_year());  
        d.display();  
    }  
}
```

OUTPUT:



```
<terminated> date_test [Java Application] C:\Users\Hp\.p2\pool\plug  
Date is 26  
Month is: 1  
Year is: 2006  
Date: 26/1/2006  
Date is 1  
Month is: 2  
Year is: 2001  
Date: 1/2/2001
```

CONCLUSION:

The `Date` class successfully encapsulates month, day, and year as instance variables with get/set methods and a `display` method that formats the date. The `Date_Test` application demonstrates the class's functionality, showcasing basic object-oriented programming practices in Java.

15	<p>Write a program to print the area of a rectangle by creating a class named 'Area' taking the values of its length and breadth as parameters of its constructor and having a method named 'returnArea' which returns the area of the rectangle. Length and breadth of rectangle are entered through keyboard.</p> <p>PROGRAM CODE :</p> <pre>package part3 ; import java.util.Scanner; class recArea { private double length; private double breadth; public recArea(double length, double breadth) { this.length = length; this.breadth = breadth; } public double returnArea() { return length * breadth; } } public class area { public static void main(String[] args) { Scanner scanner = new Scanner(System.in); System.out.println("Enter the length of the rectangle: "); double length = scanner.nextDouble(); System.out.println("Enter the breadth of the rectangle: "); double breadth = scanner.nextDouble(); recArea rectangle = new recArea(length, breadth); double area = rectangle.returnArea(); System.out.println("The area of the rectangle is: " + area); } } OUTPUT:</pre>
----	---

```
<terminated> area [Java Application] C:\Users\Hp\.p2\pool\plug
Enter the length of the rectangle:

4
Enter the breadth of the rectangle:

4
The area of the rectangle is: 16.0
```

CONCLUSION:

The **Area** class calculates the area of a rectangle using length and breadth provided by the user, demonstrating basic object-oriented programming concepts such as constructors and methods in Java. The program efficiently captures input, processes it, and outputs the computed area.

Supplementary Experiment:

1. Write a Java program to create a class called "Airplane" with a flight number, destination, and departure time attributes, and methods to check flight status and delay.

[L:M]

PROGRAM CODE:

```
public class Airplane {
    private String flightNumber;
    private String destination;
    private String departureTime;
    private boolean isDelayed;

    public Airplane(String flightNumber, String destination, String departureTime) {
        this.flightNumber = flightNumber;
        this.destination = destination;
        this.departureTime = departureTime;
        this.isDelayed = false;
    }

    public String checkFlightStatus() {
        return isDelayed ? "Delayed" : "On Time";
    }

    public void delayFlight() {
        this.isDelayed = true;
    }
}
```

```

public void displayFlightInfo() {
    System.out.println("Flight Number: " + flightNumber);
    System.out.println("Destination: " + destination);
    System.out.println("Departure Time: " + departureTime);
    System.out.println("Status: " + checkFlightStatus());
}

public static void main(String[] args) {
    Airplane flight1 = new Airplane("AI123", "New York", "10:00 AM");
    flight1.displayFlightInfo();

    flight1.delayFlight();

    System.out.println("\nAfter delay:");
    flight1.displayFlightInfo();
}
}

```

OUTPUT:

```

<terminated> Airplane [Java Application] C:\Users\Hp\.p2\pool\plugins\org.eclipse.
Flight Number: AI123
Destination: New York
Departure Time: 10:00 AM
Status: On Time

After delay:
Flight Number: AI123
Destination: New York
Departure Time: 10:00 AM
Status: Delayed

```

- 16 Print the sum, difference and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by user.

PROGRAM CODE:

```

package part3;

import java.util.Scanner;

```

```
class Complex {
    private double real;
    private double imaginary;

    public Complex(double real, double imaginary) {
        this.real = real;
        this.imaginary = imaginary;
    }

    public Complex add(Complex other) {
        return new Complex(this.real + other.real, this.imaginary + other.imaginary);
    }

    public Complex subtract(Complex other) {
        return new Complex(this.real - other.real, this.imaginary - other.imaginary);
    }

    public Complex multiply(Complex other) {
        double realPart = this.real * other.real - this.imaginary * other.imaginary;
        double imaginaryPart = this.real * other.imaginary + this.imaginary * other.real;
        return new Complex(realPart, imaginaryPart);
    }

    public void display() {
        System.out.println(this.real + " + " + this.imaginary + "i");
    }
}

public class ComplexNumberOperations {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter real and imaginary parts of the first complex number:");
        double real1 = scanner.nextDouble();
        double imaginary1 = scanner.nextDouble();

        System.out.println("Enter real and imaginary parts of the second complex number:");
        double real2 = scanner.nextDouble();
        double imaginary2 = scanner.nextDouble();
    }
}
```

```
Complex c1 = new Complex(real1, imaginary1);
Complex c2 = new Complex(real2, imaginary2);

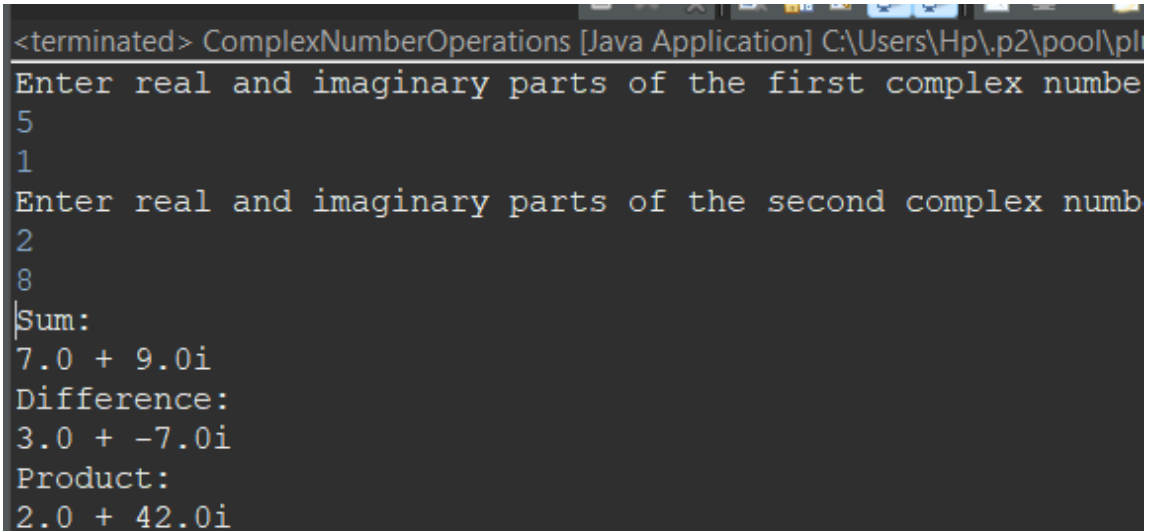
Complex sum = c1.add(c2);
Complex difference = c1.subtract(c2);
Complex product = c1.multiply(c2);

System.out.println("Sum:");
sum.display();

System.out.println("Difference:");
difference.display();

System.out.println("Product:");
product.display();
}
}
```

OUTPUT:

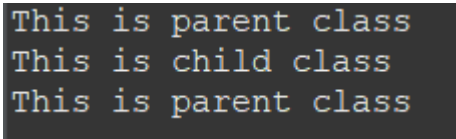


```
<terminated> ComplexNumberOperations [Java Application] C:\Users\Hp\.p2\pool\pl
Enter real and imaginary parts of the first complex number
5
1
Enter real and imaginary parts of the second complex number
2
8
Sum:
7.0 + 9.0i
Difference:
3.0 + -7.0i
Product:
2.0 + 42.0i
```

CONCLUSION:

The complex class handles operations on complex numbers, including addition, subtraction, and multiplication. The program reads input for two complex numbers, performs the operations, and displays the results.

Part - 4

17	<p>Create a class with a method that prints "This is parent class" and its subclass with another method that prints "This is child class". Now, create an object for each of the class and call 1 - method of parent class by object of parent</p> <p>PROGRAM CODE:</p> <pre>package part4; class Parent { void display() { System.out.println("This is parent class"); }} class Childd extends Parent { void show() { System.out.println("This is child class"); }} public class p17 { public static void main(String[] args) { Parent parentObj = new Parent(); Childd childObj = new Childd(); parentObj.display(); childObj.show(); childObj.display(); }}</pre> <p>OUTPUT:</p>  <p>CONCLUSION:</p> <p>In this practical, we created a class with a method that prints "This is parent class" and its subclass with another method that prints "This is child class" and an object for each of the class and call 1 - method of parent class by object of parent.</p>
18	<p>Create a class named 'Member' having the following members: Data members</p> <ul style="list-style-type: none"> 1 - Name 2 - Age 3 - Phone number 4 - Address 5 – Salary <p>It also has a method named 'printSalary' which prints the salary of the members. Two classes 'Employee' and 'Manager' inherits the 'Member' class.</p>

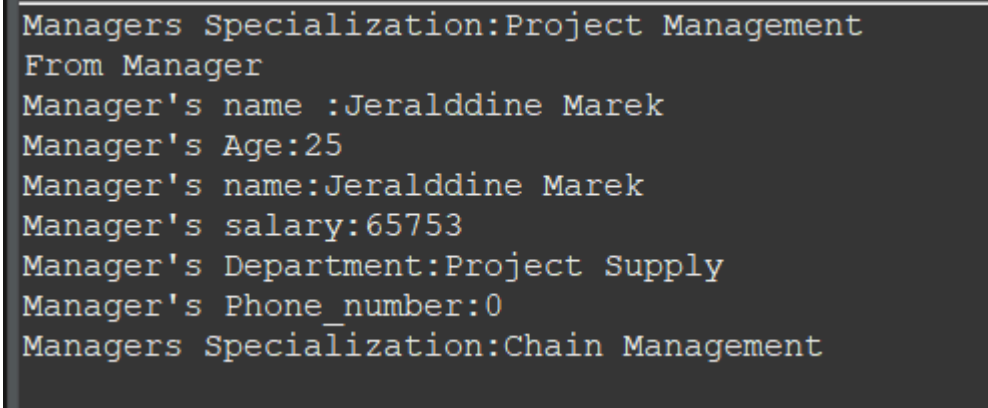
The 'Employee' and 'Manager' classes have data members 'specialization' and 'department' respectively. Now, assign name, age, phone number, address and salary to an employee and a manager by making an object of both of these classes and print the same.

PROGRAM CODE:

```
package part4;
```

```
public class p18 {  
    public static void main(String[] args) {  
        System.out.println("From Employee");  
        Employee e = new Employee();  
        e.address = "101,Old Town Road";  
        e.age = 21;  
        e.name="Den Josh";  
        e.salary = 150000;  
        e.Department = "Management";  
        e.phone_number = 66987844;  
        e.Specialization = "Project Management";  
        e.Salary();  
        System.out.println("Manager's name :"+ e.name);  
        System.out.println("Manager's Age:"+e.age);  
        System.out.println("Manager's name:"+e.name);  
        System.out.println("Manager's salary:"+e.salary);  
        System.out.println("Manager's Department:"+e.Department);  
        System.out.println("Manager's Phone_number:"+e.phone_number);  
        System.out.println("Managers Specialization:" + e.Specialization);
```

```
        System.out.println("From Manager");  
        Manager m = new Manager();  
        m.address = "New City Near Library ";  
        m.age = 25;  
        m.name = "Jeralddine Marek";  
        m.salary = 65753;  
        m.Department = "Project Supply";  
        m.Specialization = "Chain Management";  
        m.Salary();  
        System.out.println("Manager's name :"+ m.name);  
        System.out.println("Manager's Age:"+m.age);
```

	<pre> System.out.println("Manager's name:"+m.name); System.out.println("Manager's salary:"+m.salary); System.out.println("Manager's Department:"+m.Department); System.out.println("Manager's Phone_number:"+m.phone_number); System.out.println("Managers Specialization:" + m.Specialization); } } </pre> <p>OUTPUT:</p>  <p>CONCLUSION: In this practical I learnt implementation of hierarchical inheritance, single parent class member and 2 child classes manager and employee.</p>
19	<p>Create a class named 'Rectangle' with two data members 'length' and 'breadth' and two methods to print the area and perimeter of the rectangle respectively. Its constructor having parameters for length and breadth is used to initialize length and breadth of the rectangle. Let class 'Square' inherit the 'Rectangle' class with its constructor having a parameter for its side (suppose s) calling the constructor of its parent class as 'super(s,s)'. Print the area and perimeter of a rectangle and a square. Also use array of objects.</p> <p>PROGRAM CODE:</p> <pre> rectangle class: package part4; public class rectangle { double l; double b; public rectangle(double l ,double b){ this.l = l; this.b = b; } } </pre>

```
}  
public double calc_area(){  
    return l*b;  
}  
public double calc_peri() {  
    return 2*(l+b);  
  
}  
public void area() {  
    System.out.println("Area : " + calc_area());  
}  
public void perimeter() {  
    System.out.println("Perimeter : "+calc_peri());  
}  
}  
square:  
package part4;  
  
public class square extends rectangle {  
    public square(double side) {  
        super(side,side);  
    }  
}  
main file:  
package part4;  
  
public class mainn {  
    public static void main (String[] args) {  
        rectangle r = new rectangle(55.0,60);  
        System.out.println("Rectangle :-- ");  
        r.area();  
        r.perimeter();  
  
        square sq = new square(4.0);  
        System.out.println();  
        System.out.println("Square :-- ");  
        sq.area();  
        sq.perimeter();  
  
        rectangle[] rr = new rectangle[3];
```

```
rr[0] = new rectangle(44.9,211.0);
rr[1] = new rectangle(51.9,71.0);
rr[2] = new rectangle(54.7,217);
System.out.println();
System.out.println("Array of Rectangles : ");

for(int i = 0 ; i < rr.length; i++) {
System.out.println();
System.out.println("Rectangle " + (i+1) + ":");
System.out.println();
rr[i].area();
rr[i].perimeter();
}
}
}
```

OUTPUT:

```
Rectangle :--
Area : 3300.0
Perimeter : 230.0

Square :--
Area : 16.0
Perimeter : 16.0

Array of Rectangles :

Rectangle 1:

Area : 9473.9
Perimeter : 511.8

Rectangle 2:

Area : 3684.9
Perimeter : 245.8

Rectangle 3:

Area : 11869.900000000001
Perimeter : 543.4
```

Supplementary Experiment:

1. Write a Java program to create a vehicle class hierarchy. The base class should be Vehicle, with subclasses Truck, Car and Motorcycle. Each subclass should have properties such as make, model, year, and fuel type. Implement methods for calculating fuel efficiency, distance traveled, and maximum speed. [L:A]

PROGRAM CODE:

vehicle class:

package part4;

```
public abstract class vehicle {
```

```
String make;
```

```
String model;
```

```
int year;
```

```
String fuel_type;
```

```
public vehicle(String make, String model , int year , String fuel_type) {
```

```
this.make = make;
```

```
this.model = model;
```

```
this.year = year;
```

```
this.fuel_type = fuel_type;
```

```
}
```

```
public abstract double fuel_efficiency();
```

```
public abstract double distance(double fuel) ;
```

```
public abstract int speed();
```

```
public void display_info() {
```

```
System.out.println("Make : " +make);
```

```
System.out.println("Model :"+model);
```

```
System.out.println("Year : " +year);
```

```
System.out.println("fuel_type : " +fuel_type);
```

```
}
```

```
}
```

car class:

package part4;

```
public class car extends vehicle {
```

```
public car(String make, String model , int year , String fuel_type) {
    super(make,model,year,fuel_type);

}
public double fuel_efficiency(){
    return 15.0;
}
public double distance(double fuel) {
    return fuel * fuel_efficiency();

}
public int speed() {
    return 180;

}

}

truck:
package part4;

public class truck extends vehicle{
    public truck(String make, String model , int year , String fuel_type) {
        super(make,model,year,fuel_type);

    }
    public double fuel_efficiency(){
        return 8.0;
    }
    public double distance(double fuel) {
        return fuel * fuel_efficiency();

    }
    public int speed() {
        return 90;

    }

}
```

```
}  
motorcycle:  
package part4;  
  
public class motorcycle extends truck {  
    public motorcycle(String make, String model , int year , String fuel_type) {  
        super(make,model,year,fuel_type);  
    }  
    public double fuel_efficiency(){  
        return 30.0;  
    }  
    public double distance(double fuel) {  
        return fuel * fuel_efficiency();  
    }  
    public int speed() {  
        return 150;  
    }  
}  
  
main file:  
package part4;  
  
public class vehicle_test {  
    public static void main(String[] args) {  
        System.out.println(":: -- Truck -- ::");  
        System.out.println();  
        truck t = new truck("Toyata","F15h",2013,"deisel");  
        System.out.println("Fuel Efficiency : " + t.fuel_efficiency() + " kmph");  
        System.out.println("Distnce Travelled : " + t.distance(50));  
        System.out.println("Maximum Speed : " + t.speed());  
        System.out.println();  
        System.out.println("Truck Details :");  
        t.display_info();  
        System.out.println();  
        System.out.println(":: -- Car -- ::");  
        System.out.println();  
        car c = new car("Maruti","C66e",2020,"deisel");
```

```
System.out.println("Fuel Efficiency : " + c.fuel_efficiency() + " kmph");
System.out.println("Distnce Travelled : " + c.distance(5));
System.out.println("Maximum Speed : " + c.speed());
System.out.println();
System.out.println("Car Details :");
c.display_info();
System.out.println();
System.out.println(":: -- Motorcycle -- ::");
System.out.println();
motorcycle mc = new motorcycle("Honda","B887H",2003,"Petrol");
System.out.println("Fuel Efficiency : " + mc.fuel_efficiency() + " kmph");
System.out.println("Distnce Travelled : " + mc.distance(50));
System.out.println("Maximum Speed : " + mc.speed());
System.out.println();
System.out.println("Motorcycle Details :");
mc.display_info();

}}
```

OUTPUT:


```

:: -- Truck -- ::

Fuel Efficiency : 8.0 kmph
Distance Travelled : 400.0
Maximum Speed : 90

Truck Details :
Make : Toyata
Model :F15h
Year : 2013
fuel_type : deisel

:: -- Car -- ::

Fuel Efficiency : 15.0 kmph
Distance Travelled : 75.0
Maximum Speed : 180

Car Details :
Make : Maruti
Model :C66e
Year : 2020
fuel_type : deisel

:: -- Motorcycle -- ::

Fuel Efficiency : 30.0 kmph
Distance Travelled : 1500.0
Maximum Speed : 150

Motorcycle Details :
Make : Honda
Model :B887H
Year : 2003
fuel_type : Petrol

```

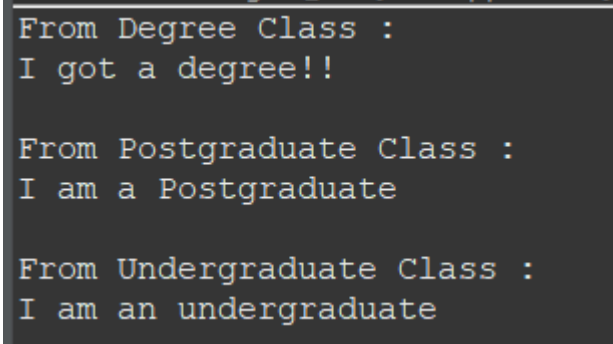
CONCLUSION: In this practical I learnt implementation of creating a parent class Rectangle inheriting its properties using super keyword in square class.

- 20 Create a class 'Degree' having a method 'getDegree' that prints "I got a degree". It has two subclasses namely 'Undergraduate' and 'Postgraduate' each having a method with the same name that prints "I am an Undergraduate" and "I am a Postgraduate" respectively. Call the method by creating an object of each of the three classes.
- PROGRAM CODE:
- ```

package part4;

public class Degree {

```

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre> public void getDegree() {     System.out.println("I got a degree!!"); } } package part4;  public class Degree_test {     public static void main(String[] args) {         System.out.println("From Degree Class : ");         Degree d = new Degree();         d.getDegree();         System.out.println();         System.out.println("From Postgraduate Class : ");         postgraduate pd = new postgraduate();         pd.getDegree();         System.out.println();         System.out.println("From Undergraduate Class : ");         Undergraduate ud = new Undergraduate();         ud.getDegree();     } } </pre> <p>OUTPUT:</p>  <p>CONCLUSION:</p> <p>In this practical I created one parent class and two child classes.</p> |
| 21 | <p>Create a class named 'Shape' with a method to print "This is This is shape". Then create two other classes named 'Rectangle', 'Circle' inheriting the Shape class, both having a method to print "This is rectangular shape" and "This is circular shape" respectively. Create a subclass 'Square' of 'Rectangle' having a method to print "Square is a rectangle". Now call the method of 'Shape' and 'Rectangle' class by the object of 'Square' class.</p>                                                                                                                                                                                                                                                                                                                                                                   |

**PROGRAM CODE:**

```
package part4;

class Shape {
 void printShape() {
 System.out.println("This is shape");
 }
}

class R extends Shape {
 void printRectangle() {
 System.out.println("This is rectangular shape");
 }
}

class Circle extends Shape {
 void printCircle() {
 System.out.println("This is circular shape");
 }
}

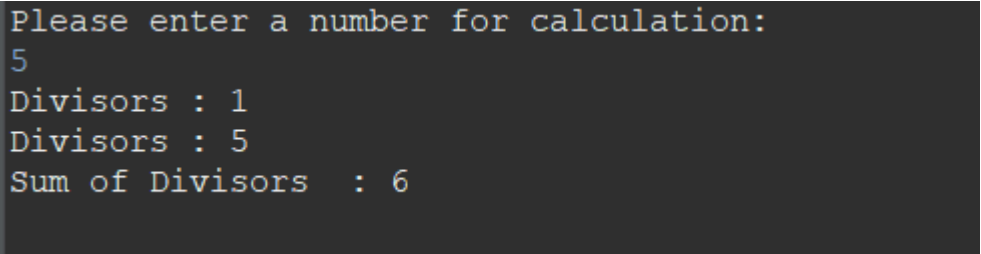
class sq extends R {
 void printSquare() {
 System.out.println("Square is a rectangle");
 }
}

public class p20 {
 public static void main(String[] args) {
 sq square = new sq();
 square.printShape();
 square.printRectangle();
 square.printSquare();
 }
}
```

**OUTPUT:**

```
This is shape
This is rectangular shape
Square is a rectangle
```

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <p>CONCLUSION: This program demonstrates inheritance in Java, where a subclass inherits and extends the behavior of its parent classes.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 22 | <p>Write a java that implements an interface AdvancedArithmetic which contains a method signature <code>int divisor_sum(int n)</code>. You need to write a class called MyCalculator which implements the interface. <code>divisorSum</code> function just takes an integer as input and return the sum of all its divisors. For example, divisors of 6 are 1, 2, 3 and 6, so <code>divisor_sum</code> should return 12. The value of <code>n</code> will be at most 1000.</p> <p>PROGRAM CODE:</p> <pre>package part4; import java.util.*;  interface AdvancedArithmetic{     public void divisor_sum(int n); }  public class Mycalculator {     public static void main(String[] args) {         calc c = new calc();         Scanner sc = new Scanner(System.in);         System.out.println("Please enter a number for calculation: ");         int k = sc.nextInt();         c.divisor_sum(k);     }  }  class calc extends Mycalculator implements AdvancedArithmetic{     public void divisor_sum(int n) {         int d = 0;         for(int i = 1; i &lt;= n ; i++){             if(n % i == 0){                 System.out.println("Divisors : " +i);                 d = d + i;}         }     } }</pre> |

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre>System.out.println("Sum of Divisors : "+ d);  } }</pre> <p>OUTPUT:</p>  <p>CONCLUSION: In this practical I implemented the AdvancedArithmetic Interface. MyCalculator class implemented the interface.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 23 | <p>Assume you want to capture shapes, which can be either circles (with a radius and a color) or rectangles (with a length, width, and color). You also want to be able to create signs (to post in the campus center, for example), each of which has a shape (for the background of the sign) and the text (a String) to put on the sign. Create classes and interfaces for circles, rectangles, shapes, and signs. Write a program that illustrates the significance of interface default method.</p> <p>PROGRAM CODE:</p> <pre>package part4;  interface S {     String getColor();     double getArea();     default String describeShape() {         return "This is a shape with color: " + getColor() + " and area: " + getArea();     } }  class cir implements S {     private double radius;     private String color;      public cir(double radius, String color) {         this.radius = radius;</pre> |

```
this.color = color;
}

public String getColor() {
return color;
}

public double getArea() {
return Math.PI * radius * radius;
}

public double getRadius() {
return radius;
}
}

class Re implements S {
private double length;
private double width;
private String color;

public Re(double length, double width, String color) {
this.length = length;
this.width = width;
this.color = color;
}

@Override
public String getColor() {
return color;
}

@Override
public double getArea() {
return length * width;
}

public double getLength() {
return length;
}
```

```
public double getWidth() {
 return width;
}
}

class sn {
 private S shape;
 private String text;

 public sn(S shape, String text) {
 this.shape = shape;
 this.text = text;
 }

 public void displaySign() {
 System.out.println(shape.describeShape());
 System.out.println("Sign text: " + text);
 }
}

public class p23 {
 public static void main(String[] args) {
 cir circle = new cir(5.0, "Red");
 Re rectangle = new Re(4.0, 6.0, "Blue");

 sn sign1 = new sn(circle, "Welcome to the Campus Center");
 sn sign2 = new sn(rectangle, "Library Hours");

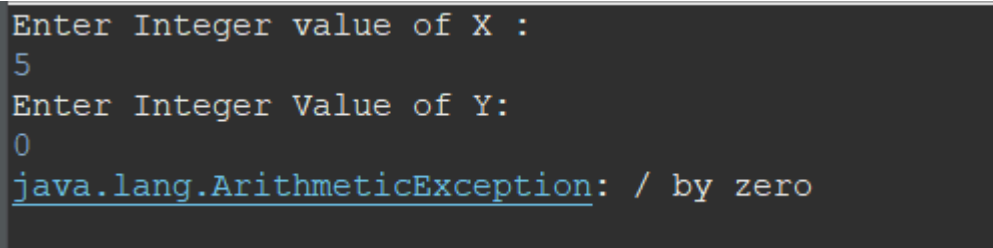
 sign1.displaySign();
 sign2.displaySign();
 }
}
```

OUTPUT:

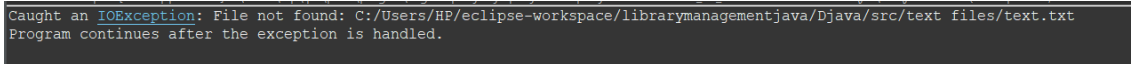
```
This is a shape with color: Red and area: 78.53981633974483
Sign text: Welcome to the Campus Center
This is a shape with color: Blue and area: 24.0
Sign text: Library Hours
```

|  |                                                                                                                                                                                                 |
|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | CONCLUSION: The program shows how interface default methods can provide shared functionality across classes. It also demonstrates how different shapes can be managed using a common interface. |
|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Part - 5

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 24 | <p>The program shows how interface default methods can provide shared functionality across classes. It also demonstrates how different shapes can be managed using a common interface.</p> <p>PROGRAM CODE:</p> <pre>package part5; import java.util.*; public class p24 {     public static void main(String[] args) {         try {             Scanner sc = new Scanner(System.in);             System.out.println("Enter Integer value of X : ");             int x = sc.nextInt();             System.out.println("Enter Integer Value of Y: " );             int y = sc.nextInt();             int d = (x/y);             System.out.println(x+ " Divided by " + y + " is : " + d);             sc.close();          }         catch(Exception e) {             System.out.println(e);         }      } }</pre> <p>OUTPUT:</p>  <pre>Enter Integer value of X : 5 Enter Integer Value of Y: 0 java.lang.ArithmeticException: / by zero</pre> |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

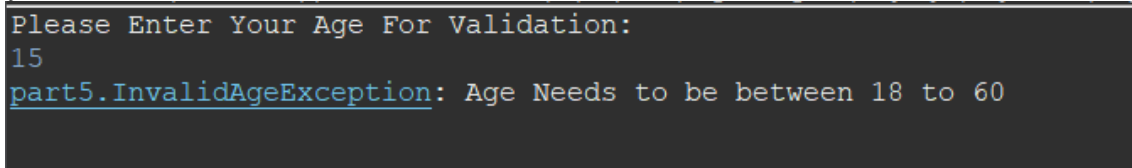


|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <p><b>CONCLUSION:</b>In this practical I implemented interface default methods that provide shared functionality across classes. It also demonstrates how different shapes can be managed using a common interface.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 25 | <p>Write a Java program that throws an exception and catch it using a try-catch block.</p> <p><b>PROGRAM CODE:</b></p> <pre>package part5; import java.io.File; import java.io.FileInputStream; import java.io.IOException;  public class p25 {     public static void main(String[] args) {         try {             readFile("C:/Users/HP/eclipse- workspace/librarymanagementjava/Djava/src/text files/text.txt");             System.out.println("File Reading Successfull!! ");         } catch (IOException e) {             System.out.println("Caught an IOException: " + e.getMessage());         }          System.out.println("Program continues after the exception is handled.");     }      public static void readFile(String fileName) throws IOException {         File file = new File(fileName);          if (!file.exists()) {             throw new IOException("File not found: " + fileName);         }          FileInputStream fis = new FileInputStream(file);          fis.close();     } }</pre> <p><b>OUTPUT:</b></p>  <p><b>CONCLUSION:</b> In this practical I implemented try and catch block.</p> |
| 26 | <p>Write a java program to generate user defined exception using “throw” and “throws” keyword. Also Write a java that differentiates checked and unchecked exceptions. (Mention at least two checked and two unchecked exceptions in program).</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**PROGRAM CODE:**

Throw and Throws:

```
package part5;
import java.util.*;
class InvalidAgeException extends Exception{
 InvalidAgeException(String msg){
 super(msg);
 }
}
public class p26 {
 public static void check_age(int age)throws InvalidAgeException{
 if(age < 18 || age>60) {
 throw new InvalidAgeException("Age Needs to be between 18 to 60");
 }
 else {
 System.out.println("Age is Valid");
 }
 }
 public static void main(String[] args) {
 try {
 Scanner sc = new Scanner(System.in);
 System.out.println("Please Enter Your Age For Validation: ");
 int i = sc.nextInt();
 check_age(i);
 }
 catch(InvalidAgeException e){
 System.out.println(e);
 }
 }
}
```

A screenshot of a terminal window with a dark background. It shows the prompt 'Please Enter Your Age For Validation:' followed by the user input '15'. The program then throws an exception, which is printed as 'part5.InvalidAgeException: Age Needs to be between 18 to 60'.

15

part5.InvalidAgeException: Age Needs to be between 18 to 60

Checked and Unchecked Exception:

```
package part5;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
```

```
import java.io.IOException;
import java.lang.reflect.Method;
public class p26final {
 public void readFile() throws IOException {
 try {
 BufferedReader reader = new BufferedReader(new FileReader("somefile.txt"));
 String line = reader.readLine();
 System.out.println(line);
 reader.close();
 } catch (IOException e) {
 throw new IOException("The specified file was not found");
 }
 }

 public void checkMethod() throws NoSuchMethodException {
 try {
 Method method = String.class.getMethod("nonExistentMethod");
 } catch (NoSuchMethodException e) {
 throw new NoSuchMethodException("The specified method does not exist");
 }
 }

 public void uncheckedExceptionsDemo() {
 int[] arr = {1, 2, 3};
 System.out.println(arr[5]);

 String str = null;
 System.out.println(str.length());
 }

 public static void main(String[] args) {
 p26final demo = new p26final();

 try {
 demo.readFile();
 } catch (IOException e) {
 System.out.println("Caught IOException: " + e.getMessage());
 }

 try {
 demo.checkMethod();
 } catch (NoSuchMethodException e) {
 System.out.println("Caught NoSuchMethodException: " + e.getMessage());
 }

 try {
 demo.uncheckedExceptionsDemo();
 } catch (Exception e) {
 System.out.println("Caught Exception: " + e.getMessage());
 }
 }
}
```

|  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>         } catch (ArrayIndexOutOfBoundsException e) {             System.out.println("Caught ArrayIndexOutOfBoundsException: " + e.getMessage());         } catch (NullPointerException e) {             System.out.println("Caught NullPointerException: " + e.getMessage());         }     } } </pre> <p><b>OUTPUT:</b></p> <pre> Caught <u>IOException</u>: The specified file was not found Caught <u>NoSuchMethodException</u>: The specified method does not exist Caught <u>ArrayIndexOutOfBoundsException</u>: Index 5 out of bounds for length 3 </pre> <p><b>CONCLUSION:</b>In this practical I implemented throw , throws , checked and unchecked exception.</p> |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Part - 6

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 27 | <p>Write a program that will count the number of lines in each file that is specified on the command line. Assume that the files are text files. Note that multiple files can be specified, as in "java Line Counts file1.txt file2.txt file3.txt". Write each file name, along with the number of lines in that file, to standard output. If an error occurs while trying to read from one of the files, you should print an error message for that file, but you should still process all the remaining files.</p> <p><b>PROGRAM CODE:</b></p> <pre> package part6; import java.io.BufferedReader; import java.io.FileReader; import java.io.IOException; public class p27 {     public static void main(String[] args) {         if (args.length == 0) {             System.out.println("Usage: java LineCounts &lt;file1&gt; &lt;file2&gt; ...");             return;         }          for (String fileName : args) {             int lineCount = 0;              try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {                 while (reader.readLine() != null) {                     lineCount++;                 }             }             System.out.println(fileName + ": " + lineCount + " lines");         }     } } </pre> |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre>     } catch (IOException e) {         System.out.println("Error reading file " + fileName + ": " + e.getMessage());     }     }     }  } </pre> <p><b>OUTPUT:</b></p> <pre> C:\Users\Hp\workspace\librarymanagementjava\Djava\src&gt;javac part6/p27.java C:\Users\Hp\workspace\librarymanagementjava\Djava\src&gt;java part6.p27 source.txt destination.txt source.txt: 1 lines destination.txt: 1 lines </pre> <p><b>CONCLUSION:</b>In this practical we displayed number of lines in each file specified in command line.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 28 | <p>Write an example that counts the number of times a particular character, such as e, appears in a file. The character can be specified at the command line. You can use xanadu.txt as the input file.</p> <p><b>PROGRAM CODE:</b></p> <pre> package part6; import java.io.BufferedReader; import java.io.FileReader; import java.io.IOException; public class p28 {      public static void main(String[] args) {         if (args.length != 2) {             System.out.println("Usage: java CharacterCount &lt;character&gt; &lt;filename&gt;");             return;         }          char targetChar = args[0].charAt(0);         String fileName = args[1];         int count = 0;          try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {             int ch;             while ((ch = reader.read()) != -1) {                 if ((char) ch == targetChar) {                     count++;                 }             }             System.out.println("The character '" + targetChar + "' appears " + count + " times in the file " + fileName);         } catch (IOException e) {             System.out.println("Error reading file " + fileName + ": " + e.getMessage());         }     } } </pre> |

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre> } } </pre> <p><b>OUTPUT:</b></p> <pre> C:\Users\Hp\eclipse-workspace\librarymanagementjava\Djava\src&gt;java part6.p28 e ./xanadu.txt.txt The character 'e' appears 3 times in the file ./xanadu.txt.txt </pre> <p><b>CONCLUSION:</b> In this practical we passed file as arg. and counted number of times e appeared in file.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 29 | <p>Write a Java Program to Search for a given word in a File. Also show use of Wrapper Class with an example.</p> <p><b>PROGRAM CODE:</b></p> <pre> import java.io.BufferedReader; import java.io.FileReader; import java.io.IOException; public class WordSearch {     public static void main(String[] args) {         String targetWord = "the"; // Specify the word to search         String fileName = "C:/Users/Hp/eclipse- workspace/librarymanagementjava/Djava/src/xanadu.txt"; // Specify the file path         Integer count = 0;          try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {             String line;             while ((line = reader.readLine()) != null) {                 String[] words = line.split("\\W+");                 for (String word : words) {                     if (word.equalsIgnoreCase(targetWord)) {                         count++;                     }                 }             }             System.out.println("The word " + targetWord + " appears " + count + " times in the file " + fileName);         } catch (IOException e) {             System.out.println("Error reading file " + fileName + ": " + e.getMessage());         }     } } </pre> <p><b>OUTPUT:</b></p> <pre> The word 'the' appears 3 times in the file C:/Users/Hp/eclipse-workspace/librarymanagementjava/Djava/src/xanadu.txt.txt </pre> |
| 30 | <p>Write a program to copy data from one file to another file. If the destination file does not exist, it is created automatically.</p> <p><b>PROGRAM CODE:</b></p> <pre> package part6; </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

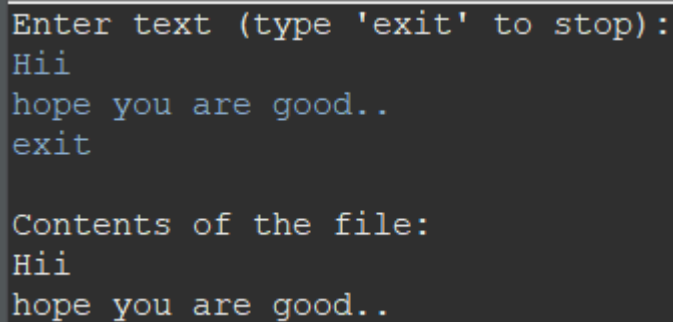
|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre> import java.io.FileInputStream; import java.io.FileOutputStream; import java.io.IOException; public class p30 {     public static void main(String[] args) {          String        sourceFile        =        "C:/Users/Hp/eclipse- workspace/librarymanagementjava/Djava/src/source.txt";          String destinationFile = "destination.txt";          try (FileInputStream fis = new FileInputStream(sourceFile);             FileOutputStream fos = new FileOutputStream(destinationFile)) {              byte[] buffer = new byte[1024];             int bytesRead;              while ((bytesRead = fis.read(buffer)) != -1) {                 fos.write(buffer, 0, bytesRead);             }              System.out.println("File copied successfully!");          } catch (IOException e) {             e.printStackTrace();         }     } }  OUTPUT: File copied successfully! </pre> |
| 31 | <p>Write a program to show use of character and byte stream.<br/>Also show use of BufferedReader/BufferedWriter to read console input and write them into a file.</p> <p><b>PROGRAM CODE:</b></p> <pre> package part6; import java.io.BufferedReader; import java.io.BufferedWriter; import java.io.FileInputStream; import java.io.FileWriter; import java.io.IOException; public class p31 {      public static void main(String[] args) {         String fileName = "output.txt"; </pre>                                                                                                                                                                                                                                                                                                                                                                                                              |

```
try (BufferedReader consoleReader = new BufferedReader(new
java.io.InputStreamReader(System.in));
 BufferedWriter fileWriter = new BufferedWriter(new FileWriter(fileName))) {

 System.out.println("Enter text (type 'exit' to stop):");
 String inputLine;
 while (!(inputLine = consoleReader.readLine()).equals("exit")) {
 fileWriter.write(inputLine);
 fileWriter.newLine();
 }
} catch (IOException e) {
 System.out.println("Error: " + e.getMessage());
}

try (FileInputStream fileInputStream = new FileInputStream(fileName)) {
 int byteData;
 System.out.println("\nContents of the file:");
 while ((byteData = fileInputStream.read()) != -1) {
 System.out.print((char) byteData);
 }
} catch (IOException e) {
 System.out.println("Error reading file: " + e.getMessage());
}
}
```

OUTPUT:



```
Enter text (type 'exit' to stop):
Hii
hope you are good..
exit

Contents of the file:
Hii
hope you are good..
```

CONCLUSION: In this practical we used character , byte stream and we took contents of output file as input.



## Part - 7

33

Write a program which takes N and number of threads as an argument. Program should distribute the task of summation of N numbers amongst number of threads and final result to be displayed on the console.

**PROGRAM CODE:**

```
import java.util.*;

public class Pra33 implements
Runnable {

 Scanner sc = new
Scanner(System.in);

 public void run() {

 System.out.println("Enter a
number to print till you want:");

 int n = sc.nextInt();

 int sum = 0;

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

 {

 sum+=i;

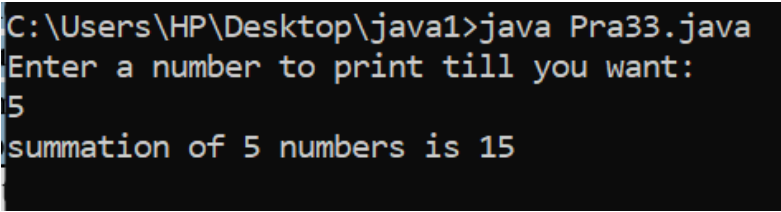
 }

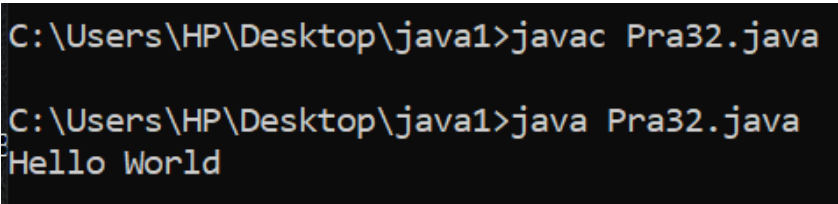
 System.out.println("summation of
"+n+" numbers is "+sum);

 }

 public static void main(String[]
args) {

 Pra33 p1 = new Pra33();
```

|    |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre> Thread thread = new Thread(p1);  thread.start();  }  } </pre> <p><b><u>OUTPUT:</u></b></p>  <p><b><u>CONCLUSION:</u></b></p> <p>In this practical I learnt how to create a thread and what is importance of void run method to process any thread and also distribute the task into n number of threads.</p>                          |
| 32 | <p>Write a program to create thread which display “Hello World” message. A. by extending Thread class B. by using Runnable interface.</p> <p><b><u>PROGRAM CODE:</u></b></p> <pre> import java.util.*;  public class Pra32 implements Runnable { public void run() { System.out.println("Hello World"); } public static void main(String args[]) { Pra32 p1 = new Pra32(); Thread th = new Thread(p1); th.start(); } } </pre> |

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre>     }     } </pre> <p><b><u>OUTPUT:</u></b></p>  <p><b><u>CONCLUSION:</u></b></p> <p>In this practical learnt how to create a thread using runnable interface.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 34 | <p>Write a java program that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.</p> <p><b><u>PROGRAM CODE:</u></b></p> <pre> import java.util.*;  public class Pra34 {     private static int n;      public static class Th1 implements Runnable {         Scanner sc = new Scanner(System.in);          public void run() {             System.out.println("Enter a number :");             n = sc.nextInt();         }     }      public static class Th2 implements Runnable {         public void run() {             int sq=n*n;             System.out.println("Square of Number is :"+sq);         }     } } </pre> |

```
public static class Th3 implements Runnable {
 public void run() {
 int cube= n*n*n;
 System.out.println("Cube of Number is :"+cube);
 }
}

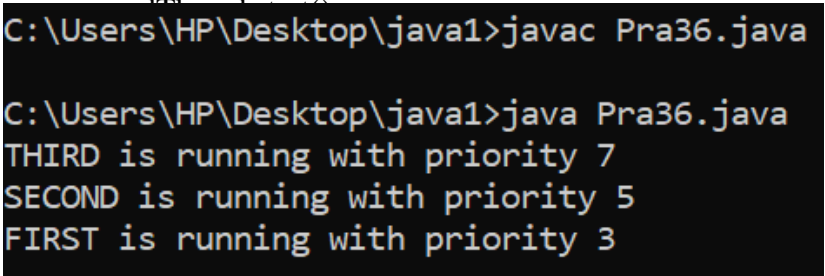
public static void main(String[] args) throws InterruptedException
{
 Th1 t1= new Th1();
 Thread thread1 = new Thread(t1);
 thread1.start();
 thread1.join();

 Th2 t2 = new Th2();
 Th3 t3 = new Th3();
 Thread thread2 = new Thread(t2);
 Thread thread3 = new Thread(t3);

 for(int i = 1; i <= n; i++)
 {
 if(n%2==0)
 {
 thread2.start();
 }
 else
 {
 thread3.start();
 }
 }
}
```

**OUTPUT:**

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre>C:\Users\HP\Desktop\java1&gt;java Pra34.java Enter a number : 5 Exception in thread "main" Cube of Number is :125 java.lang.IllegalThreadStateException     at java.base/java.lang.Thread.start(Thread.java:1512)     at Pra34.main(Pra34.java:49)</pre> <p><b><u>CONCLUSION:</u></b></p> <p>In this program I implemented multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.</p>                                                                                                                                                                                                                                                                             |
| 35 | <p>Write a program to create three threads 'FIRST', 'SECOND', 'THIRD'. Set the priority of the 'FIRST' thread to 3, the 'SECOND' thread to 5(default) and the 'THIRD' thread to 7.</p> <p><b><u>PROGRAM:</u></b></p> <pre>class Pra36 implements Runnable {      private String threadName;      public Pra36(String name) {         this.threadName = name;     }      @Override     public void run() {         System.out.println(threadName + " is running with priority " + Thread.currentThread().getPriority());     }      public static void main(String[] args) {          Pra36 firstTask = new Pra36("FIRST");         Pra36 secondTask = new Pra36("SECOND");         Pra36 thirdTask = new Pra36("THIRD");          Thread firstThread = new Thread(firstTask);         Thread secondThread = new Thread(secondTask);         Thread thirdThread = new Thread(thirdTask);</pre> |

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre> firstThread.setPriority(3); secondThread.setPriority(5); thirdThread.setPriority(7);  // Start the threads firstThread.start(); </pre>  <p><b><u>CONCLUSION:</u></b></p> <p>In this program I learnt about thread working priorities.</p>                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 36 | <p>Write a program to solve producer-consumer problem using thread synchronization.</p> <p><b><u>PROGRAM:</u></b></p> <pre> import java.util.LinkedList; class Buffer {     private LinkedList&lt;Integer&gt; list = new LinkedList&lt;&gt;();     private int capacity = 5     public synchronized void produce() throws InterruptedException {         int value = 0;         for (int i = 0; i &lt; 5; i++) {             while (list.size() == capacity) {                 wait();             }              System.out.println("Producer produced: " + value);             list.add(value++);             notify();             Thread.sleep(1000);         }     }      public synchronized void consume() throws InterruptedException { </pre> |

```
 for (int i = 0; i < 5; i++)
 while (list.isEmpty()) {
 wait();
 }

 int value = list.removeFirst();
 System.out.println("Consumer consumed: " + value);
 notify();
 Thread.sleep(1000);
 }
}
}

public class Pra37 {
 public static void main(String[] args) throws InterruptedException {
 Buffer buffer = new Buffer();
 Thread producerThread = new Thread(new Runnable() {
 @Override
 public void run() {
 try {
 buffer.produce();
 } catch (InterruptedException e) {
 Thread.currentThread().interrupt();
 }
 }
 });

 Thread consumerThread = new Thread(new Runnable() {
 @Override
 public void run() {
 try {
 buffer.consume();
 } catch (InterruptedException e) {
 Thread.currentThread().interrupt();
 }
 }
 });

 producerThread.start();
 consumerThread.start();
 producerThread.join();
 consumerThread.join();
 }
}
```

|  |                                                                                                                                                                                                                                                                                                                    |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> Consumer consumed: 0 Consumer consumed: 1 Consumer consumed: 2 Consumer consumed: 3 Consumer consumed: 4 Producer and Consumer have completed. </pre> <p><b>CONCLUSION:</b><br/>In this program I learnt about how to implement thread synchronization and what is importance of thread synchronization.</p> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### Part - 8

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 24 | <p>AIM: Design a Custom Stack using ArrayList class, which implements following functionalities of stack. My Stack -list ArrayList&lt;Object&gt;: A list to store elements. +isEmpty: boolean: Returns true if this stack is empty. +getSize(): int: Returns number of elements in this stack. +peek(): Object: Returns top element in this stack without removing it. +pop(): Object: Returns and Removes the top elements in this stack. +push(o: object): Adds new element to the top of this stack.</p> <p>PROGRAM CODE:</p> <pre> package part8; import java.util.*; public class p38 { private ArrayList&lt;Object&gt; list; private int maxSize; public p38(int maxSize) { list = new ArrayList&lt;&gt;(); this.maxSize = maxSize; } public boolean isEmpty() { System.out.println("IsEmpty? : "+list.isEmpty()); return list.isEmpty(); } public int getSize() { System.out.println("size: "+list.size()); return list.size(); } public Object peek() { if(isEmpty()) { System.out.println("Stack Underflow : Stack is Empty! "); return null; } System.out.println("Peeked: "+list.get(list.size()-1)); return list.get(list.size()-1); } } </pre> |
|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|



```
public Object pop() {
 if(isEmpty()) {
 System.out.println("Stack Underflow : Stack is Empty! ");
 return null;
 }
 Object a = list.get(list.size()-1);
 System.out.println("Popped: "+a);
 return list.remove(list.size()-1);
}

public void push(Object o) {
 if (list.size() < maxSize) {
 list.add(o);
 System.out.println("Pushed: " + o);
 } else {
 System.out.println("Overflow: Can't insert values anymore!");
 }
}

public static void main(String[] args) {
 p38 stack = new p38(3);
 stack.isEmpty();
 stack.push(50);
 stack.push(10);
 stack.peek();
 stack.pop();
 stack.peek();
 stack.isEmpty();
 stack.getSize();
 stack.pop();
 stack.getSize();
 stack.isEmpty();
 stack.pop();
 stack.push(100);

 stack.push(550);
 stack.push(100);
 stack.push(110);
 stack.push(520);
 stack.getSize();
}
}
```

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <p>OUTPUT:</p> <pre> size: 1 IsEmpty? : false Popped: 50 size: 0 IsEmpty? : true IsEmpty? : true Stack Underflow : Stack is Empty! IsEmpty? : true Stack Underflow : Stack is Empty! Pushed: 100 Pushed: 550 Pushed: 100 Overflow: Can't insert values anymore! Overflow: Can't insert values anymore! size: 3 </pre> <p>CONCLUSION: In this practical I implemented Stack.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 39 | <p>AIM: Imagine you are developing an e-commerce application. The platform needs to sort lists of products based on different criteria, such as price, rating, or name. Each product object implements the Comparable interface to define the natural ordering. To ensure flexibility and reusability, you need a generic method that can sort any array of Comparable objects. Create a generic method in Java that sorts an array of Comparable objects. This method should be versatile enough to sort arrays of different types of objects (such as products, customers, or orders) as long as they implement the Comparable interface.</p> <p>PROGRAM CODE:</p> <pre> package part8; import java.util.Arrays;  public class p39 {      public static &lt;T extends Comparable&lt;T&gt;&gt; void sort(T[] array)     {         // Simple insertion sort algorithm         for (int i = 1; i &lt; array.length; i++)         {             T key = array[i]; </pre> |

```
int j = i - 1;

// Move elements that are greater than key, to one position ahead
while (j >= 0 && array[j].compareTo(key) > 0)
{
 array[j + 1] = array[j];
 j = j - 1;
}
array[j + 1] = key;
}
}

public static void main(String[] args)
{
 // Example: Sorting products by natural ordering (e.g., price or rating)
 Product[] products =
 {
 new Product("Laptop", 10000.0),
 new Product("Phone", 8000.0),
 new Product("Tablet", 6000.0)
 };

 System.out.println("Before sorting:");
 System.out.println(Arrays.toString(products));

 // Sort the products
 sort(products);

 System.out.println("After sorting:");
 System.out.println(Arrays.toString(products));
}

// Example Product class implementing Comparable
class Product implements Comparable<Product>
{
 private String name;
 private double price;

 public Product(String name, double price) {
 this.name = name;
```

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre> this.price = price; }  public String getName() { return name; }  public double getPrice() { return price; } @Override public int compareTo(Product other) { // Compare by price (natural ordering) return Double.compare(this.price, other.price); }  @Override public String toString() { return name + ": \$" + price; } }  OUTPUT: Before sorting: [Laptop: \$10000.0, Phone: \$8000.0, Tablet: \$6000.0] After sorting: [Tablet: \$6000.0, Phone: \$8000.0, Laptop: \$10000.0]  CONCLUSION: In this generic <code>sortArray</code> method uses the <code>Comparable</code> interface, making it versatile enough to sort any type of object that implements this interface. </pre> |
| 40 | <p>Aim: Write a program that counts the occurrences of words in a text and displays the words and their occurrences in alphabetical order of the words. Using Map and Set Classes.</p> <p>PROGRAM CODE:</p> <pre> import java.util.*;  public class p40 { </pre>                                                                                                                                                                                                                                                                                                                                                                                                                             |

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <pre> public static void main(String[] args) {     String text = "This is a sample String. This String have many words. Words are counted from this String.";     System.out.println("String: " +text);      String[] words = text.toLowerCase().replaceAll("[^a-zA-Z ]", "").split("\\s+");      Map&lt;String, Integer&gt; wordCountMap = new TreeMap&lt;&gt;();      for (String word : words) {         if (!word.isEmpty()) {             wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);         }     }      System.out.println("Word Occurrences:");     for (Map.Entry&lt;String, Integer&gt; entry : wordCountMap.entrySet()) {         System.out.println(entry.getKey() + ": " + entry.getValue());     } } </pre> <p><b>OUTPUT:</b></p> <pre> String: This is a sample String. This String have many words. Words are counted from this Word Occurrences: a: 1 are: 1 counted: 1 from: 1 have: 1 is: 1 many: 1 sample: 1 string: 3 this: 3 words: 2 </pre> <p><b>CONCLUSION:</b> In this practical we counted word occurrences.</p> |
| 41 | <p><b>AIM:</b> Write a code which counts the number of the keywords in a Java source file. Store all the keywords in a HashSet and use the contains () method to test if a word is in the keyword set.</p> <p><b>PROGRAM CODE:</b></p> <pre> import java.io.*; import java.util.*; </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

```
public class p41 {

 // Method to load all Java keywords into a HashSet
 public static Set<String> loadJavaKeywords() {
 // Java keywords list (as of Java 17)
 String[] keywords = {
 "abstract", "assert", "boolean", "break", "byte", "case", "catch",
 "char", "class",
 "const", "continue", "default", "do", "double", "else", "enum",
 "extends", "final",
 "finally", "float", "for", "goto", "if", "implements", "import",
 "instanceof",
 "int", "interface", "long", "native", "new", "null", "package",
 "private",
 "protected", "public", "return", "short", "static", "strictfp",
 "super",
 "switch", "synchronized", "this", "throw", "throws", "transient",
 "try",
 "void", "volatile", "while"
 };

 // Store keywords in a HashSet for quick lookup
 return new HashSet<>(Arrays.asList(keywords));
 }

 // Method to count the number of Java keywords in a source file
 public static int countKeywordsInFile(String filePath, Set<String>
keywordSet) throws IOException {
 int keywordCount = 0;

 // Read the file line by line
 BufferedReader reader = new BufferedReader(new
FileReader(filePath));
 String line;
 while ((line = reader.readLine()) != null) {
 // Split each line into words using non-word characters as delimiters
 String[] words = line.split("\\W+");

 // Check each word to see if it's a Java keyword
```

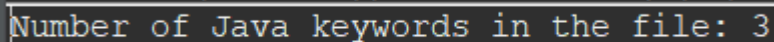
```
 for (String word : words) {
 if (keywordSet.contains(word)) {
 keywordCount++;
 }
 }
 }
 reader.close();
 return keywordCount;
}

public static void main(String[] args) {
 // Load the set of Java keywords
 Set<String> javaKeywords = loadJavaKeywords();

 // Path to the Java source file (modify with your file's path)
 String filePath = "src/javakeyword.java"; // Update this with the actual
path

 try {
 // Count the keywords in the file
 int keywordCount = countKeywordsInFile(filePath, javaKeywords);
 System.out.println("Number of Java keywords in the file: " +
keywordCount);
 } catch (IOException e) {
 System.out.println("An error occurred while reading the file: " +
e.getMessage());
 }
}
```

OUTPUT:

A screenshot of a terminal window showing the output of the Java program. The text "Number of Java keywords in the file: 3" is displayed in a monospaced font on a dark background.

CONCLUSION: In this program I implemented a code which counts the number of the keywords in a Java source file.