DEPSTAR

Devang Patel Institute of Advance Technology and Research

(A Constitute Institute of CHARUSAT)

Certificate

This is to certify that

Mr. / Mrs. GARALA DHRUVI

of DEPSTAR (CSE) *Class,*

ID. No. 23DCS029 *has satisfactorily completed*

his/ her term work in JAVA PROGRAMMING *for*

the ending in NOV *2024 /2025*

Date : 17/10/24

[illegible]

Sign. of Faculty

[illegible]

Head of Department

CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY (CHARUSAT)
DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY AND RESEARCH
DEPSTAR

Subject : JAVA PROGRAMMING **Semester:** 3

Subject Code: CSE201 **Academic Year :**2024-25

Course Outcome (COs):

At the end of the course, the students will be able to:

CO1	Comprehend Java Virtual Machine architecture and Java Programming Fundamentals.
CO2	Demonstrate basic problem-solving skills: analyzing problems, modelling a problem as a system of objects, creating algorithms, and implementing models and algorithms in an object-oriented computer language (classes, objects, methods with parameters)
CO3	Design applications involving Object Oriented Programming concepts such as inheritance, polymorphism, abstract classes and interfaces.
CO4	Build and test program using exception handling
CO5	Design and build multi-threaded Java Applications.
CO6	Build software using concepts such as files and collection frameworks.

Bloom's Taxonomy:

Level 1- Remembering
Level 2- Understanding
Level 3- Applying
Level 4- Analyzing
Level 5- Evaluating
Level 6- Creating

CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY (CHARUSAT)
DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY AND RESEARCH
DEPSTAR

Practical List

Sr No.	AIM	Hrs.	CO	Bloom's Taxonomy
PART-I Data Types, Variables, String, Control Statements, Operators, Arrays				
1	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.	**2**	**1**	**1**
2	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.	**1**	**1**	**2,3,4**
3	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).	**1**	**1**	**2,3,4**
4	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. **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.	**1**	**1, 2**	**2,3**
5	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.	**1**	**1, 2**	**2**
6	Create a Java program that prompts the user to enter the	**1**	**1, 2**	**2,3,4**

	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. **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.			
	PART-II Strings			
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'	**1**	**1, 2**	**2,3,4**
8	Given an array of ints, return the number of 9's in the array. array_count9([1, 2, 9]) → 1 array_count9([1, 9, 9]) → 2 array_count9([1, 9, 9, 3, 9]) → 3 **Supplementary Experiment:** 1. Write a Java program to replace each substring of a given string that matches the given regular expression with the given replacement. Sample string : "The quick brown fox jumps over the lazy dog." **In the above string replace all the fox with cat.**	**1**	**1, 2**	**2,3**
9	Given a string, return a string where for every char in the original, there are two chars. double_char('The') → 'TThhee' double_char('AAbb') → 'AAAAbbbb' double_char('Hi-There') → 'HHii--TThheerree'	**1**	**1, 2**	**2**
10	Perform following functionalities of the string: ● Find Length of the String ● Lowercase of the String ● Uppercase of the String ● Reverse String	**1**	**1, 2**	**2,3,4**

	Sort the string			
11	Perform following Functionalities of the string: "CHARUSAT UNIVERSITY" ● Find length ● Replace 'H' by 'FIRST LATTER OF YOUR NAME' ● Convert all character in lowercase **Supplementary Experiment:** 1. Write a Java program to count and print all duplicates in the input string. Sample Output: The given string is: resource The duplicate characters and counts are: e appears 2 times r appears 2 times	**1**	**1, 2**	**4**
PART-III Object Oriented Programming: Classes, Methods, Constructors				
12	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.	**1**	**2**	**3**
13	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.	**2**	**1, 2**	**3**
14	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.	**2**	**1, 2**	**3**

15	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. **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]	**1**	**1, 2**	**3**
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.	**1**	**1, 2**	**2,3**
	PART-IV Inheritance, Interface, Package			
17	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	**1**	**1, 2, 3**	**3**
18	Create a class named 'Member' having the following members: Data members 1 - Name 2 - Age 3 - Phone number 4 - Address 5 – Salary It also has a method named 'printSalary' which prints the salary of the members. Two classes 'Employee' and 'Manager' inherits the 'Member' class. 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.	**2**	**1, 2, 3**	**3**
19	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	**1**	**2,3**	**3**

	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. **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]			
20	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.	**2**	**2,3**	**3**
21	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.	**1**	**2,3**	**3**
22	Write a java that implements an interface AdvancedArithmetic which contains amethod signature int divisor_sum(int n). You need to write a class calledMyCalculator which implements the interface. divisorSum 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 divisor_sum should return 12. The value of n will be at most 1000. **Supplementary Experiment:** 1.Write a Java programming to create a banking system with three classes - Bank, Account, SavingsAccount, and CurrentAccount. The bank should have a list of accounts and methods for adding them. Accounts should be an interface with methods to deposit, withdraw,	**2**	**2,3**	**2,3**

	calculate interest, and view balances. SavingsAccount and CurrentAccount should implement the Account interface and have their own unique methods. [L:A]			
23	Assume you want to capture shapes, which can be either circles (with a radiusand 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.	**2**	**2,3**	**6**
PART-V Exception Handling				
24	Write a java program which takes two integers x & y as input, you have to compute x/y. If x and y are not integers or if y is zero, exception will occur and you have to report it.	**1**	**4**	**3**
25	Write a Java program that throws an exception and catch it using a try-catch block.	**1**	**4**	**3**
26	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). **Supplementary Experiment:** 1.Write a Java program that reads a list of integers from the user and throws an exception if any numbers are duplicates. [L:M]	**2**	**4**	**2,3**
	PART-VI File Handling & Streams			
27	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.	**1**	**4,6**	**3**
28	Write an example that counts the number of times a	**1**	**4,6**	**3**

	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.			
29	Write a Java Program to Search for a given word in a File. Also show use of Wrapper Class with an example.	**2**	**4,6**	**3**
30	Write a program to copy data from one file to another file. If the destination file does not exist, it is created automatically. **Supplementary Experiment:** 1.Write a Java program to sort a list of strings in alphabetical order, ascending and descending using streams.	**2**	**4,6**	**3**
31	Write a program to show use of character and byte stream. Also show use of BufferedReader/BufferedWriter to read console input and write them into a file.	**2**	**4,6**	**2,3**
	PART-VII Multithreading			
32	Write a program to create thread which display “Hello World” message. A. by extending Thread class B. by using Runnable interface.	**1**	**5,6**	**3**
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.	**1**	**5,6**	**3**
34	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.	**2**	**5,6**	**3**
35	Write a program to increment the value of one variable by one and display it after one second using thread using sleep() method.	**2**	**5,6**	**2,3**
36	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.	**2**	**5,6**	**2,3**
37	Write a program to solve producer-consumer problem using thread synchronization.	**2**	**5,6**	**3**
	PART-VIII Collection Framework and Generic			
38	Design a Custom Stack using ArrayList class, which	**2**	**5**	**3**

	implements following functionalities of stack. My Stack -list ArrayList<Object>: 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.			
39	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.	**2**	**5**	**6**
40	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.	**2**	**5**	**3**
41	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.	**2**	**5**	**2,3**

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: 2024-25

Part - 1

No.	Aim of the Practical
1.	Demonstrate 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. 1. 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. • Demonstration of installation steps of Java. 1. Download: Get the Java installer from the official website. 2. Run Installer: Execute the downloaded file. 3. Follow Steps: Accept the license agreement and choose the installation folder. 4. Optional: Set environment variables (like JAVA_HOME). • Introduction to Object Oriented Concepts

1.Encapsulation:

o Encapsulation bundles data (attributes) and methods (functions) into a single unit called a class.It restricts direct access to data, ensuring that data integrity is maintained.

2.Inheritance:

o Inheritance models the relationship between a base class (parent) and a derived class (child). The derived class inherits properties and behaviors from the base class

3.Polymorphism:

o Polymorphism allows objects of different classes to be treated uniformly.

o It enables flexibility by providing a single interface for various data types.

4.Abstraction:

o Abstraction focuses on essential features while hiding unnecessary details.

o It simplifies complex systems by providing a high-level view.

o Let us compare JAVA with object oriented programming language C++

1. Platform Dependency.
2. Memory Management.
3. Syntax and Complexity.
4. Object-Oriented Features.

• Introduction to JDK, JRE, JVM, Javadoc and command line argument.

JDK: The Java Development Kit (JDK) is a cross-platformed software development environment that offers a collection of tools and libraries necessary for developing Java-based software applications and applets.

JRE: Java Runtime Environment (JRE) is an open-access software distribution that has a Java class library, specific tools, and a separate JVM. In Java, JRE is one of the interrelated components in the Java Development Kit (JDK). It is the most common environment available on devices for running Java programs.

JVM: JVM (Java Virtual Machine) acts as a run-time engine to run Java

applications. JVM is the one that actually calls the main method present in a Java code. JVM is a part of JRE. When we compile a .java file, .class file with the same class names present in .java file are generated by the Java compiler. This .class file goes into various steps when we run it.

- Introduction to NetBeans and console programming:

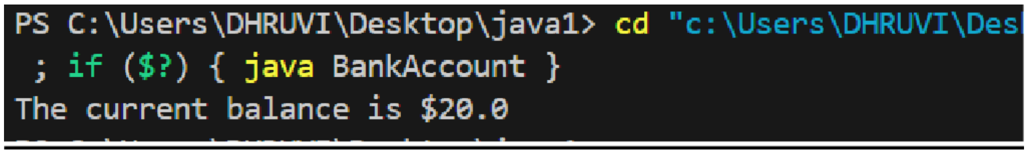
o NetBeans IDE is a free, open source, integrated development environment (IDE) that enables you to develop desktop, mobile and web applications. .The IDE provides comprehensive support for JDK 8 technologies and the most recent Java enhancements. It is the first IDE that provides support for JDK 8, Java EE 7, and JavaFX The IDE fully supports Java EE using the latest standards for Java, XML, Web services, and SQL and fully supports the GlassFish Server, the reference implementation of Java EE.

2. 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.

PROGRAM CODE:

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

        double currentBalance = 20.0;



        System.out.println("The current balance is $" + currentBalance);
    }
}
```

OUTPUT:



```
PS C:\Users\DHRUVI\Desktop\java1> cd "c:\Users\DHRUVI\Des
 ; if ($?) { java BankAccount }
The current balance is $20.0
```

CONCLUSION:

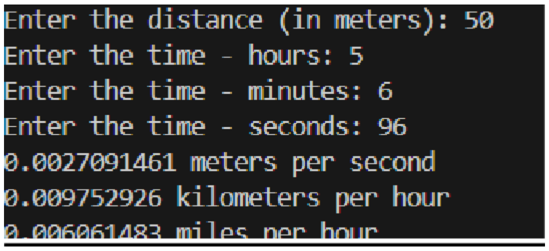
Here in this pratical we learned how to store this balance in a variable and then display it to the user.

3 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).

PROGRAM CODE:

```
import java.util.Scanner;

class pra3 {
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);



        System.out.print("Enter the distance (in meters): ");
        float distance = sc.nextFloat();



        System.out.print("Enter the time - hours: ");
        int hours = sc.nextInt();

        System.out.print("Enter the time - minutes: ");
        int minutes = sc.nextInt();

        System.out.print("Enter the time - seconds: ");
        int seconds = sc.nextInt();



        sc.close();



        int totalTimeInSeconds = hours * 3600 + minutes * 60 + seconds;



        float speedMetersPerSecond = distance / totalTimeInSeconds;
        float totalHours = totalTimeInSeconds / 3600.0f;
        float distanceInKilometers = distance / 1000.0f;
        float speedKilometersPerHour = distanceInKilometers / totalHours;
        float distanceInMiles = distance / 1609.0f;
        float speedMilesPerHour = distanceInMiles / totalHours;



        System.out.println(speedMetersPerSecond + " meters per second");
        System.out.println(speedKilometersPerHour + " kilometers per hour");
        System.out.println(speedMilesPerHour + " miles per hour");
    }
}
```

OUTPUT:



```
Enter the distance (in meters): 50
Enter the time - hours: 5
Enter the time - minutes: 6
Enter the time - seconds: 96
0.0027091461 meters per second
0.009752926 kilometers per hour
0.006061483 miles per hour
```

CONCLUSION:

here in this pratical we learned about 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.

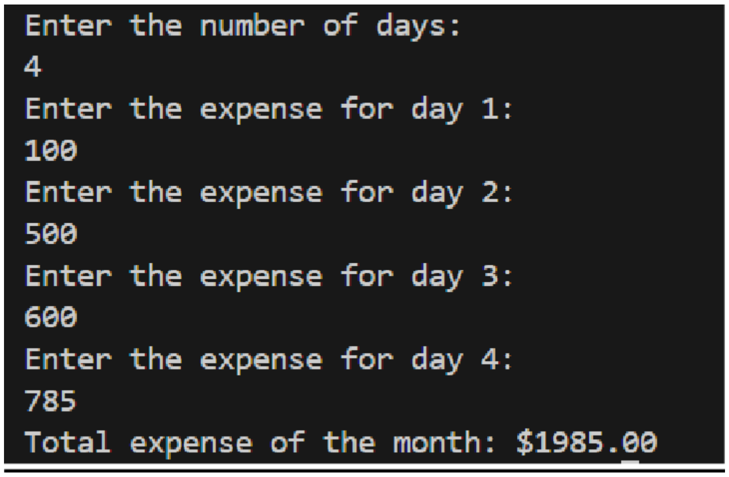
4

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.

PROGRAM CODE:

```
import java.util.Scanner;

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

        System.out.println("Enter the number of days:");
        int numDays = sc.nextInt();


        if (numDays <= 0) {
            System.out.println("Number of days must be a positive integer.");
            sc.close();
            return;
        }

        double[] expenses = new double[numDays];

        for (int i = 0; i < numDays; i++) {
            System.out.println("Enter the expense for day " + (i + 1) + ":");
            expenses[i] = sc.nextDouble();
        }

        double totalExpense = 0;
        for (int i = 0; i < numDays; i++) {
            totalExpense += expenses[i];
        }
        System.out.printf("Total expense of the month: $%.2f%n", totalExpense);
        sc.close();
    }
}
```

OUTPUT:



```
Enter the number of days:
4
Enter the expense for day 1:
100
Enter the expense for day 2:
500
Enter the expense for day 3:
600
Enter the expense for day 4:
785
Total expense of the month: $1985.00
```

CONCLUSION:

Here we learned to calculate the total expenses for the month.

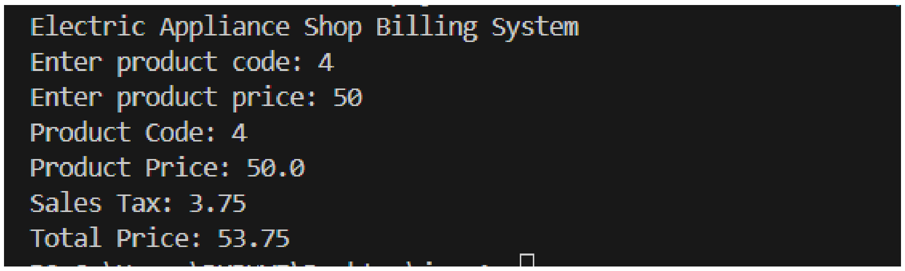
5

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.

PROGRAM CODE:

```
import java.util.Scanner;

 class pra5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        int[] productCodes = {1, 2, 3, 4, 5};
        double[] prices = {100.0, 200.0, 150.0, 50.0, 30.0};

        System.out.println("Electric Appliance Shop Billing System");


        System.out.print("Enter product code: ");
        int code = sc.nextInt();
        System.out.print("Enter product price: ");
        double price = sc.nextDouble();

        double taxRate = 0.0;


        switch (code) {
            case 1:
                taxRate = 0.08;
                break;
            case 2:
                taxRate = 0.12;
                break;
            case 3:
                taxRate = 0.05;
                break;
            case 4:
                taxRate = 0.075;
                break;
            default:
                taxRate = 0.03;
                break;
        }


        double taxAmount = price * taxRate;
        double totalPrice = price + taxAmount;


        System.out.println("Product Code: " + code);
        System.out.println("Product Price: " + price);
        System.out.println("Sales Tax: " + taxAmount);
        System.out.println("Total Price: " + totalPrice);

        sc.close();
    }
}
```

OUTPUT:



```
Electric Appliance Shop Billing System
Enter product code: 4
Enter product price: 50
Product Code: 4
Product Price: 50.0
Sales Tax: 3.75
Total Price: 53.75
```

CONCLUSION:

Here we learned to prepare bil using switch case.

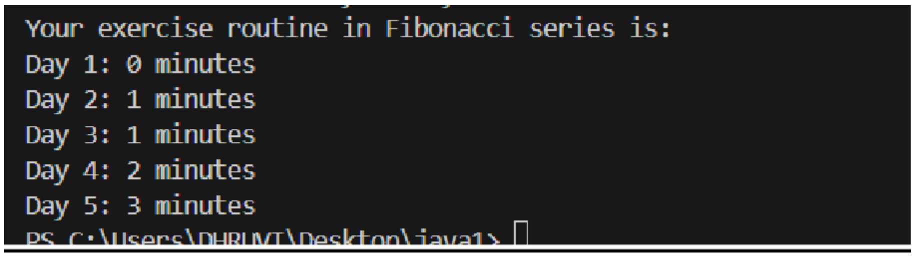
6

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 pra6 {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

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

        scanner.close();

        System.out.println("Your exercise routine in Fibonacci series is:");
        generateFibonacci(n);
    }


    public static void generateFibonacci(int n) {
        int firstTerm = 0;
        int secondTerm = 1;

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


            int nextTerm = firstTerm + secondTerm;
            firstTerm = secondTerm;
            secondTerm = nextTerm;
        }
    }
}
```

OUTPUT:



```
Your exercise routine in Fibonacci series is:
Day 1: 0 minutes
Day 2: 1 minutes
Day 3: 1 minutes
Day 4: 2 minutes
Day 5: 3 minutes
PS C:\Users\DHRUVI\Desktop\java1>
```

CONCLUSION:

Here we learn to display the first n terms of the Fibonacci series.

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 - 25

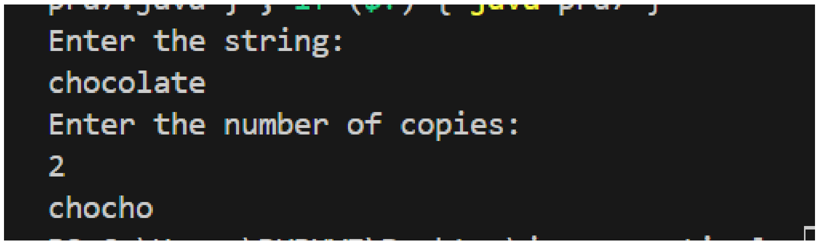
PART – 2 (STRINGS)

No.	Aim of the Practical

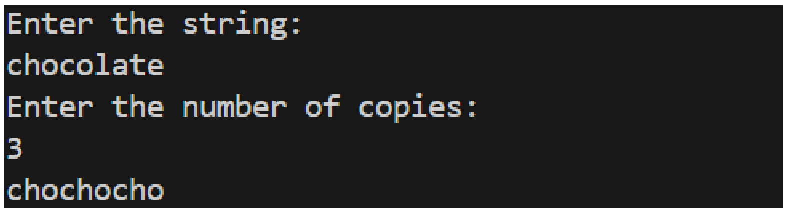
7.	Given a string and a non-negative int n, we'll say that the front of the string is the first 3 chars, or whatever is there if the string is less than length 3. Return n copies of the front;
	front_times('Chocolate', 2) → 'ChoCho'
	front_times('Chocolate', 3) → 'ChoChoCho'
	front_times('Abc', 3) → 'AbcAbcAbc'
	
	PROGRAM CODE:
	
	`import java.util.*;`
	`public class Pra7 {`
	
	`    public static void main(String[] args) {`
	
	`        Scanner sc = new Scanner(System.in);`
	`        System.out.print("Enter a string: ");`
	`        String str = sc.nextLine();`
	`        System.out.print("Enter the number of copies: ");`

```
        int n = sc.nextInt();

for(int i=0;i<n;i++)
{
System.out.print(str.substring(0,3));
}


   }
};
```

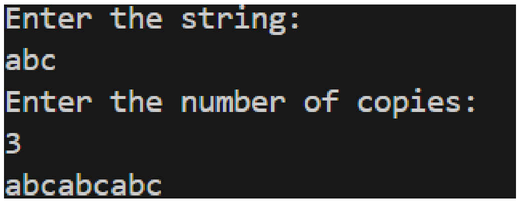
<u>OUTPUT:</u>



```
Enter the string:
chocolate
Enter the number of copies:
2
chocho
```



```
Enter the string:
chocolate
Enter the number of copies:
3
chochocho
```



```
Enter the string:
abc
Enter the number of copies:
3
abcabcabc
```

CONCLUSION:

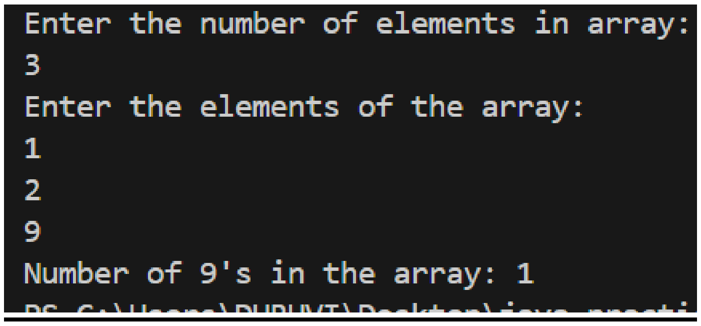
The provided Java code takes a string and a non-negative integer as input from the user. It then prints the first three characters of the input string repeatedly for the number of times specified by the user.

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

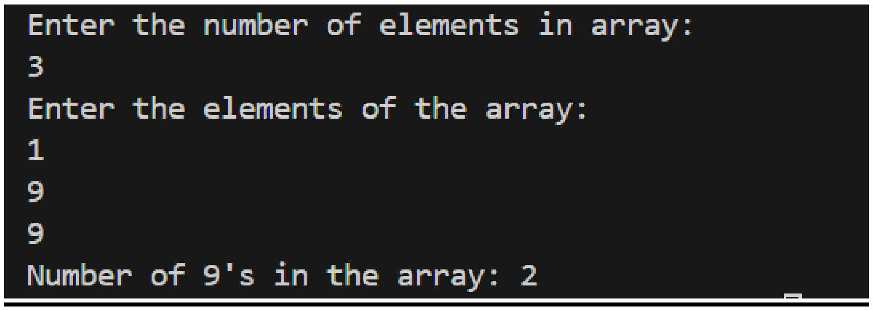
PROGRAM CODE :

```
import java.util.Scanner;

class pra8 {
    public static int arrayCounter9(int[] nums) {
        int count = 0;
        for (int num : nums) {
            if (num == 9) {
                count++;
            }
        }
        return count;
    }

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

        System.out.println("Enter the number of
elements in array:");
        int n = sc.nextInt();

        int[] array = new int[n];

        System.out.println("Enter the elements of
the array:");
        for (int i = 0; i < n; i++) {
            array[i] = sc.nextInt();
        }
        int result = arrayCounter9(array);
        System.out.println("Number of 9's in the
array: " + result);

        sc.close();
    }
}
```

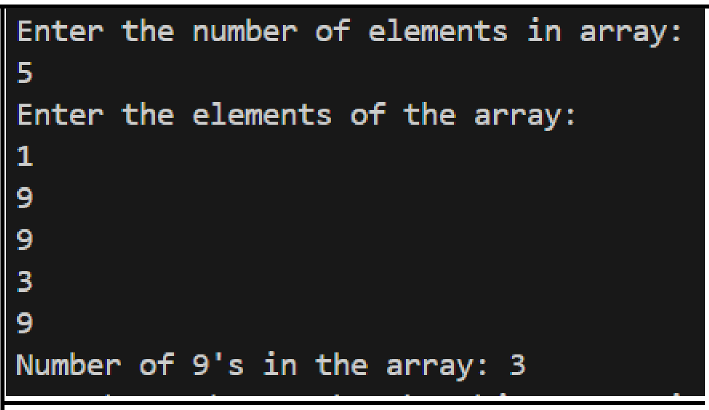
OUTPUT:



```
Enter the number of elements in array:
3
Enter the elements of the array:
1
2
9
Number of 9's in the array: 1
PS C:\Users\DHRUVI\Desktop\java practi
```



```
Enter the number of elements in array:
3
Enter the elements of the array:
1
9
9
Number of 9's in the array: 2
```



```
Enter the number of elements in array:
5
Enter the elements of the array:
1
9
9
3
9
Number of 9's in the array: 3
```

CONCLUSION:

This Java program reads three integers from the user into an array and counts how many times the number 9 appears in the array. It demonstrates basic array operations, user input handling, and simple counting logic. The program outputs the count of the number 9 after processing the input.

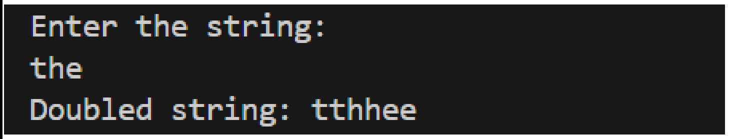
9

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

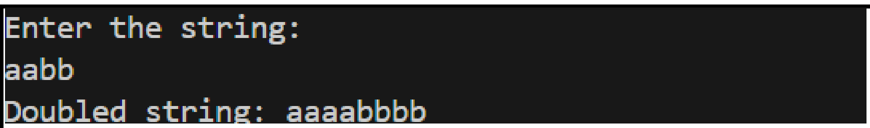
PROGRAM CODE :

```
import java.util.*;
public class Pra9 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
         System.out.print("Enter a string: ");
         String str = sc.nextLine();

for(int i=0;i<str.length();i++)
{
char letter = str.charAt(i);
int count =0;
while(count != 2)
{
System.out.print( letter);
count++;
}
}
}
};
```

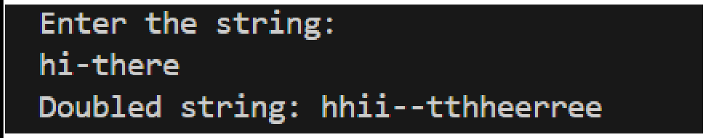
OUTPUT:



```
Enter the string:
the
Doubled string: tthhee
```



```
Enter the string:
aabb
Doubled string: aaaabbbb
```



```
Enter the string:
hi-there
Doubled string: hhii--tthheerree
```

CONCLUSION:

This Java program takes a string input from the user and a number specifying how many times each character in the string should be repeated. It then prints the modified string with each character repeated the specified number of times. The program demonstrates the use of StringBuffer for efficient string manipulation and handling user input.

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: 2024-25

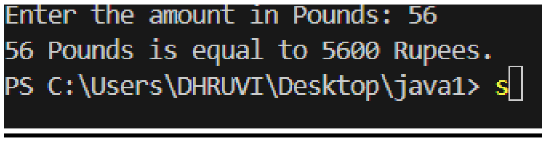
Part - 3

No.	Aim of the Practical

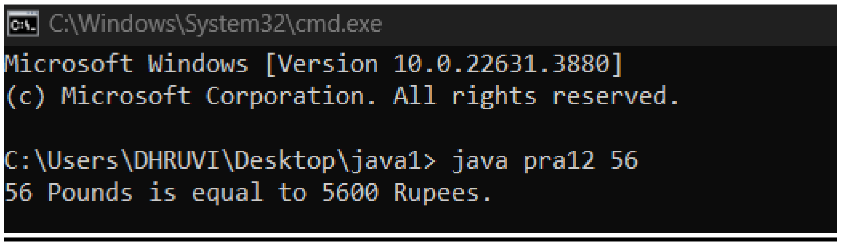
12.	Imagine you are developing a currency conversion tool for a travel agency. This tool should be able to convert an amount in Pounds to Rupees. For simplicity, we assume the conversion rate is fixed: 1 Pound = 100 Rupees. The tool should be able to take input both from command-line arguments and interactively from the user.
	
	PROGRAM CODE :
	
	`import java.util.Scanner;`
	
	`class pra12 {`
	
	`    public static void main(String[] args) {`
	
	`        int pounds = 0;`
	
	`        if (args.length > 0) {`
	
	`            pounds = Integer.parseInt(args[0]);`
	
	`        } else {`

```
            Scanner sc = new Scanner(System.in);

            System.out.print("Enter the amount in

Pounds: ");

            pounds = sc.nextInt();

            sc.close();

        }

        int rupees = pounds * 100;

        System.out.println(pounds + " Pounds is

equal to " + rupees + " Rupees.");

    }

}
```

OUTPUT:



```
Enter the amount in Pounds: 56
56 Pounds is equal to 5600 Rupees.
PS C:\Users\DHRUVI\Desktop\java1> s
```



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22631.3880]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DHRUVI\Desktop\java1> java pra12 56
56 Pounds is equal to 5600 Rupees.
```

CONCLUSION:
Here we learned to convert an amount in Pounds to Rupees.

13. 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 Employee test 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.

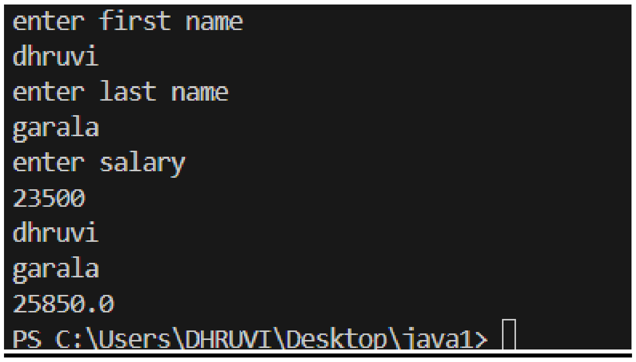
PROGRAM CODE:

```
 import java.util.*;

class Employee
{
 Scanner sc= new Scanner(System.in);
 String fs=" ";
 String ls=" ";
 double sal;

Employee(){}
Employee(String f,String l,double sa)
{
 fs=f;
ls=l;
sal=sa;
}
void setfs()
{
System.out.println("enter first name");
 fs= sc.nextLine();
}
void setls()
{
```

```
System.out.println("enter last name");
 ls= sc.nextLine();
}
void setsal()
{
System.out.println("enter salary");
 sal= sc.nextDouble();
if(sal<0)
{
sal=0.0;
}
else
{
 sal= sal + (sal*0.1);
}
}
String getfs()
{
return fs;
}
String getls()
{
return ls;
}
double getsal()
{
return sal;
}

};

class pra13
```

```
{
public static void main(String args[])
{
Employee e1 = new Employee();
e1.setfs();
e1.setls();
e1.setsal();
String c = e1.getfs();
System.out.println(c);
String b = e1.getls();
System.out.println(b);
Double a= e1.getsal();
System.out.println(a);

}
};
```

OUTPUT:



```
enter first name
dhruvi
enter last name
garala
enter salary
23500
dhruvi
garala
25850.0
PS C:\Users\DHRUVI\Desktop\java1>
```

CONCLUSION:

Here we learned to display the salary of employe.

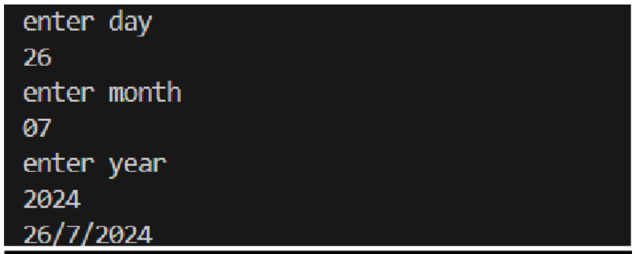
14. Create a class called Date that includes three pieces of information as instance variables—a month (type int), a day (type int) and a year (type int). Your class should have a constructor that initializes the three instance variables and assumes that the values provided are correct. Provide a set and a get method for each instance variable. Provide a method display Date that displays the month, day and year separated by forward slashes (/). Write a test application named Date Test that demonstrates class Date's capabilities.

PROGRAM CODE:

```
import java.util.Scanner;
class Date
{
int d,m,y;
Scanner sc = new Scanner(System.in);
Date(){}
Date(int day,int mon,int year)
{
d=day;
m=mon;
y=year;
}
void setday()
{
System.out.println("enter day");
 d= sc.nextInt();
}
void setmonth()
{
System.out.println("enter month");
 m= sc.nextInt();
}
void setyear()
{
```

```
System.out.println("enter year");
 y= sc.nextInt();
}
void displaydate()
{
System.out.println(d+"/"+m+"/"+y);
}
};
class Pra14
{
public static void main(String args[])
{
Date d=new Date();
d.setday();
d.setmonth();
d.setyear();
d.displaydate();
}
};
```

OUTPUT:



```
enter day
26
enter month
07
enter year
2024
26/7/2024
```

CONCLUSION:

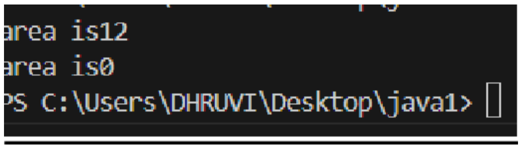
here we learned to display date.

15. 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.

PROGRAM CODE:

```
import java.util.*;
class Rect
{
int a,b,c;
Rect(){}
Rect(int l,int m)
{
a=l;
b=m;
}
int returnArea()
{
 int c=(a*b);
System.out.println("area is"+c);
return c;
}
};

class Pra15
{
public static void main(String args[])
{
Rect r1 = new Rect(3,4);
Rect r2= new Rect();
r1.returnArea();
r2.returnArea();
}
};
```

OUTPUT:



```
area is12
area is0
PS C:\Users\DHRUVI\Desktop\java1>
```

CONCLUSION:

here we learn to created a program to calculate rectangle area. Length and breadth of rectangle are entered through keyboard.

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.

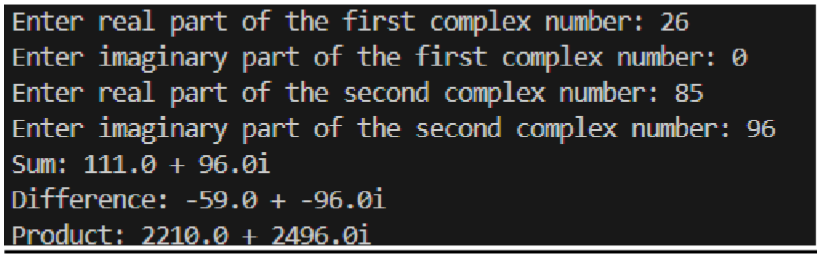
<u>PROGRAM CODE:</u>

```
import java.util.Scanner;

class Complex {
    double real;
    double imaginary;

    Complex(double r, double i) {
        real = r;
        imaginary = i;
    }

    Complex add(Complex other) {
        double r = real + other.real;
        double i = imaginary + other.imaginary;
        return new Complex(r, i);
    }
    Complex subtract(Complex other) {
        double r = real - other.real;
        double i = imaginary - other.imaginary;
        return new Complex(r, i);
    }
    Complex multiply(Complex other) {
        double r = (real * other.real) - (imaginary * other.imaginary);
        double i = (real * other.imaginary) + (imaginary * other.real);
        return new Complex(r, i);
    }
```

```
public String toString() {
    return real + " + " + imaginary + "i";
}

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

    System.out.print("Enter real part of the first complex number: ");
    double real1 = sc.nextDouble();
    System.out.print("Enter imaginary part of the first complex number: ");
    double imaginary1 = sc.nextDouble();
    Complex complex1 = new Complex(real1, imaginary1);


    System.out.print("Enter real part of the second complex number: ");
    double real2 = sc.nextDouble();
    System.out.print("Enter imaginary part of the second complex number: ");
    double imaginary2 = sc.nextDouble();
    Complex complex2 = new Complex(real2, imaginary2);


    Complex sum = complex1.add(complex2);
    Complex difference = complex1.subtract(complex2);
    Complex product = complex1.multiply(complex2);

    System.out.println("Sum: " + sum);
    System.out.println("Difference: " + difference);
    System.out.println("Product: " + product);

    sc.close();
}
```

}

OUTPUT:



```
Enter real part of the first complex number: 26
Enter imaginary part of the first complex number: 0
Enter real part of the second complex number: 85
Enter imaginary part of the second complex number: 96
Sum: 111.0 + 96.0i
Difference: -59.0 + -96.0i
Product: 2210.0 + 2496.0i
```

CONCLUSION:

Here we learned to print the sum, difference and product of two complex numbers.

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 - 25

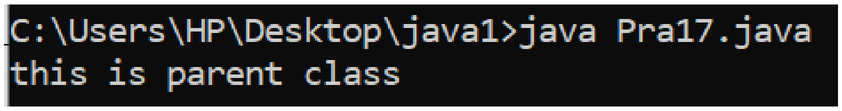
Part - 4

No.	**Aim of the Practical**
1.	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.

PROGRAM CODE:

```
import java.util.*;

class Parent
{
public void print1()
{
System.out.println("this is parent class");
}
};

class Child extends Parent
{
public void print2()
{
System.out.println("this is child class");
}
};

class Pra17
{
public static void main(String args[])
{
Parent p1 = new Parent();
p1.print1();

Child c1 = new Child();
}
};
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra17.java
this is parent class
```

CONCLUSION:

In this practical I learnt about various types of Inheritance.

18. Create a class named 'Member' having the following members: Data members 1 -Name 2 - Age 3 - Phone number 4 - Address 5 – Salary It also has a method named 'printSalary' which prints the salary of the members. Two classes 'Employee' and 'Manager' inherits the 'Member' class. 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.

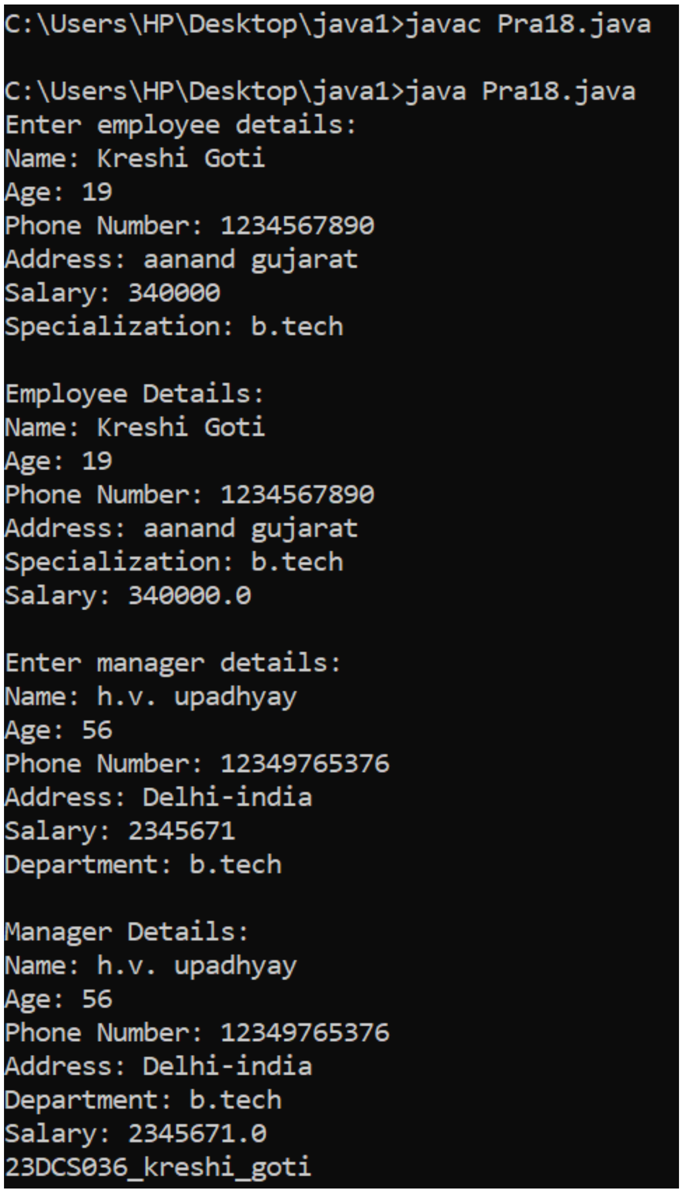
PROGRAM CODE:

```
import java.util.Scanner;
class Member {
    String name;
    int age;
    String phoneNumber;
    String address;
    double salary;

    void printSalary() {
        System.out.println("Salary: " + salary);
    }
}



class Employee extends Member {
    String specialization;

    void displayEmployeeDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Phone Number: " +
phoneNumber);
        System.out.println("Address: " + address);
        System.out.println("Specialization: " +
specialization);
        printSalary();
    }
```

```
}



class Manager extends Member {
    String department;

    void displayManagerDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Phone Number: " +
phoneNumber);
        System.out.println("Address: " + address);
        System.out.println("Department: " +
department);
        printSalary();
    }
}

public class Pra18 {
    public static void main(String[] args) {
        Scanner scanner = new
Scanner(System.in);



        Employee employee = new Employee();

        System.out.println("Enter employee
details:");
        System.out.print("Name: ");
        employee.name = scanner.nextLine();
        System.out.print("Age: ");
        employee.age = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Phone Number: ");
        employee.phoneNumber =
scanner.nextLine();
```

```
        System.out.print("Address: ");
        employee.address = scanner.nextLine();
        System.out.print("Salary: ");
        employee.salary = scanner.nextDouble();
        scanner.nextLine();
        System.out.print("Specialization: ");
        employee.specialization =
scanner.nextLine();

        System.out.println("\nEmployee Details:");
        employee.displayEmployeeDetails();



        Manager manager = new Manager();

        System.out.println("\nEnter manager
details:");
        System.out.print("Name: ");
        manager.name = scanner.nextLine();
        System.out.print("Age: ");
        manager.age = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Phone Number: ");
        manager.phoneNumber =
scanner.nextLine();
        System.out.print("Address: ");
        manager.address = scanner.nextLine();
        System.out.print("Salary: ");
        manager.salary = scanner.nextDouble();
        scanner.nextLine();
        System.out.print("Department: ");
        manager.department = scanner.nextLine();

        System.out.println("\nManager Details:");
        manager.displayManagerDetails();
        System.out.print("23DCS036_kreshi_goti
");
   }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra18.java

C:\Users\HP\Desktop\java1>java Pra18.java
Enter employee details:
Name: Kreshi Goti
Age: 19
Phone Number: 1234567890
Address: aanand gujarat
Salary: 340000
Specialization: b.tech

Employee Details:
Name: Kreshi Goti
Age: 19
Phone Number: 1234567890
Address: aanand gujarat
Specialization: b.tech
Salary: 340000.0

Enter manager details:
Name: h.v. upadhyay
Age: 56
Phone Number: 12349765376
Address: Delhi-india
Salary: 2345671
Department: b.tech

Manager Details:
Name: h.v. upadhyay
Age: 56
Phone Number: 12349765376
Address: Delhi-india
Department: b.tech
Salary: 2345671.0
23DCS036_kreshi_goti
```

CONCLUSION:

In this practical I learnt about various types of Inheritance.

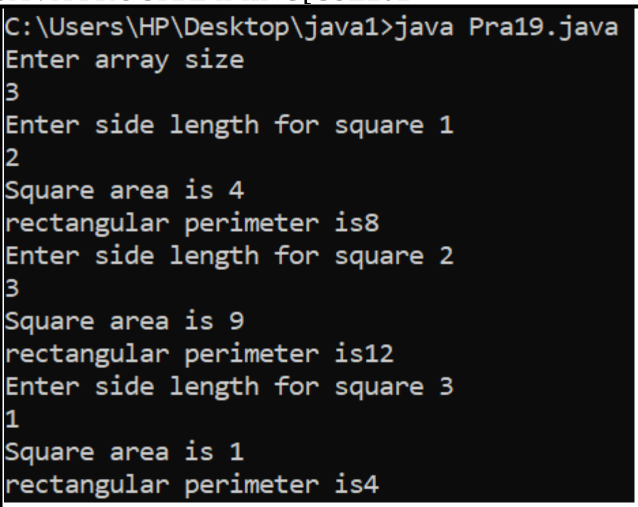
19. Create a class named 'Rectangle' with two data members 'length' and 'breadth' and two methods to print the area and perimeter of the rectangle respectively. Its constructor having parameters for length and breadth is used to initialize length and breadth of the rectangle. Let class 'Square' inherit 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.

PROGRAM CODE:

```
import java.util.*;

class Rectangle
{
public int l;
public int b;

Rectangle(int l, int b)
{
this.l = l;
this.b = b;
}

public void calarea()
{
System.out.println("rectangular area is"+(l*b));
}

public void calperimeter()
{
System.out.println("rectangular perimeter
is"+(2*(l+b)));
}
};

class Square extends Rectangle
{
```

```
public Square(int s)
{
super(s,s);
}
public void calareasq()
{
 System.out.println("Square area is " + (l * l));
}
}
class Pra19
{
public static void main(String args[])
{
 Scanner sc = new Scanner(System.in);
        System.out.println("Enter array size");
        int n = sc.nextInt();

        Square[] s1 = new Square[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter side length for
square " + (i + 1));
            int side = sc.nextInt();
            s1[i] = new Square(side);
            s1[i].calareasq();
            s1[i].calperimeter();
}
}
};
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra19.java
Enter array size
3
Enter side length for square 1
2
Square area is 4
rectangular perimeter is8
Enter side length for square 2
3
Square area is 9
rectangular perimeter is12
Enter side length for square 3
1
Square area is 1
rectangular perimeter is4
```

CONCLUSION:

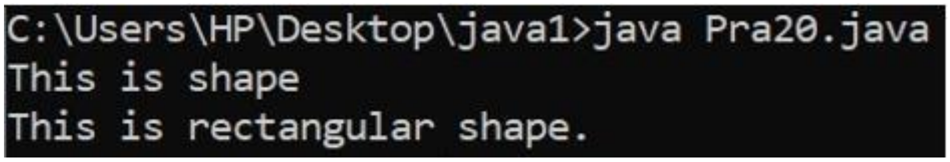
In this practical I learnt about various types of Inheritance.

20. 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.

<u>PROGRAM CODE:</u>

```
import java.util.*;
class Shape
{
public void printshape()
{
System.out.println("This is shape");
}
};
class Rectangle extends Shape
{
public void printrec()
{
System.out.println("This is rectangular shape.
");
}
};

class Square extends Rectangle
{
public void printsuq()
{
System.out.println("Square is a rectangle. ");
}
};
class Circle extends Shape
{
public void printcir()
{
System.out.println("This is circular shape.");
}
};
class Pra20
{
public static void main(String args[])
{
Square s1 = new Square();
s1.printshape();
s1.printrec();

}
};
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra20.java
This is shape
This is rectangular shape.
```

CONCLUSION:

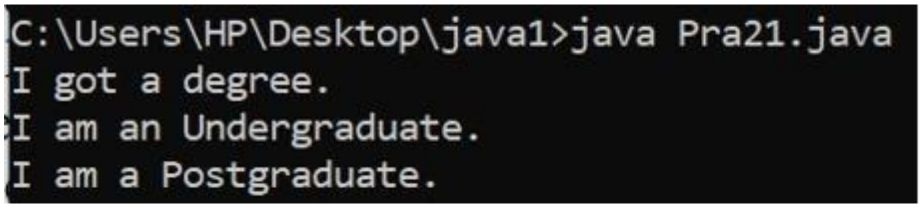
In this practical I learnt about various types of Inheritance.

21. 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:

```
import java.util.*;
class Degree
{
public void getDegree()
{
System.out.println("I got a degree.");
}
};

class Undergraduate
{
public void getDegree()
{
System.out.println("I am an Undergraduate.");
}
};
class Postgraduate
{
public void getDegree()
{
System.out.println("I am a Postgraduate.");
}
};
class Pra21
{
public static void main(String args[])
{
Degree d1 = new Degree();
d1.getDegree();
Undergraduate u1 = new Undergraduate();
u1.getDegree();
Postgraduate p1 = new Postgraduate();
p1.getDegree();

}
};
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra21.java
I got a degree.
I am an Undergraduate.
I am a Postgraduate.
```

CONCLUSION:

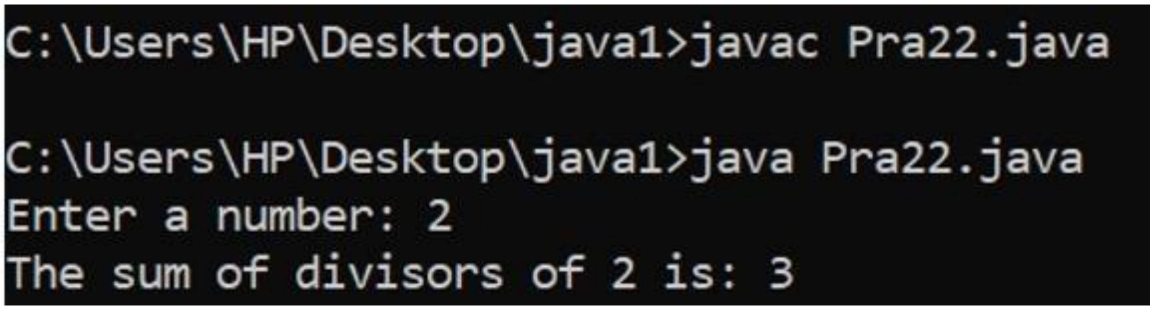
In this practical I learnt about various types of Inheritance.

22. Write a java that implements an interface AdvancedArithmetic which contains amethod signature int divisor_sum(int n). You need to write a class calledMyCalculator which implements the interface. divisorSum 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 divisor_sum should return 12. The value of n will be at most 1000.

PROGRAM CODE:

```
import java.util.Scanner;
interface AdvancedArithmetic{

   int divisor_sum(int n);
}

class MyCalculator implements
AdvancedArithmetic{
       @Override
   public int divisor_sum(int n){
   int sum = 0;
       for(int i=1; i<=n; i++){
          if(n % i == 0){
                 sum = sum + i;
              }
       }
       return sum;
  }

}
class Pra22{
   public static void main(String[]args){
              Scanner sc = new
Scanner(System.in);
              System.out.print("Enter a number:
");
              int number = sc.nextInt();

          MyCalculator c1 = new
MyCalculator();
        int result = c1.divisor_sum(number);
        System.out.println("The sum of divisors of
" + number + " is: " + result);

         }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra22.java

C:\Users\HP\Desktop\java1>java Pra22.java
Enter a number: 2
The sum of divisors of 2 is: 3
```

CONCLUSION:

In this practical I learnt about various methods in Interface.

23. Assume you want to capture shapes, which can be either circles (with a radiusand 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.

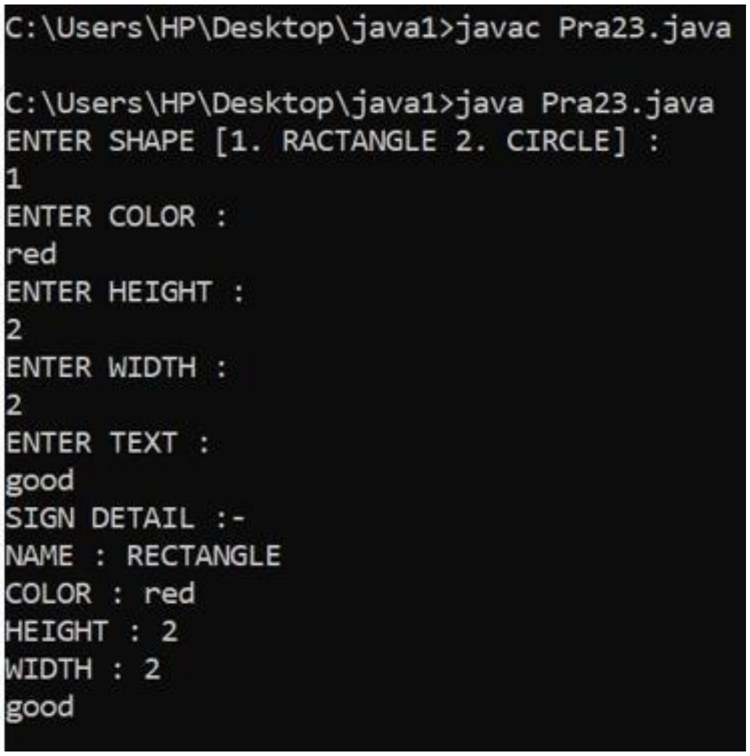
PROGRAM CODE:

```
import java.util.*;

public class Pra23 {
   public static void main(String args[]) {
      Scanner in = new Scanner(System.in);

      sign s = new sign();
      s.print();

System.out.println("\n23dcs036_kreshi_goti");
      in.close();
   }
}

interface shape {
   public String shap_name = "";

   public String getColor();

   public void setColor(String c);

   public String getShapename();

   default void printdata() {
      System.out.println("NAME : " +
getShapename());
      System.out.println("COLOR : " +
getColor());
```

```
  }
}

class circle implements shape {
  protected String color;
  protected int radius;
  public String shap_name = "CIRCLE";

  public String getColor() {
    return color;
  }

  public void setColor(String c) {
    color = c;
  }

  public int getRadius() {
    return radius;
  }

  public void setRadius(int r) {
    radius = r;
  }

  public String getShapename() {
    return shap_name;
  }

  public void printdata() {
    System.out.println("NAME : " +
getShapename());
    System.out.println("COLOR : " +
getColor());
    System.out.println("RADIUS : " +
getRadius());
  }
  }

class rectangle implements shape {
  public String shap_name = "RECTANGLE";
  protected String color;
  protected int height, width;

  public String getColor() {
    return color;
  }

  public void setColor(String c) {
    color = c;
  }

  public int getHeight() {
    return height;
  }

  public void setHeight(int r) {
    height = r;
  }

  public int getWidth() {
    return width;
  }

  public void setWidth(int r) {
    width = r;
  }

  public String getShapename() {
    return shap_name;
  }

  public void printdata() {
    System.out.println("NAME : " +
getShapename());
    System.out.println("COLOR : " +
getColor());
    System.out.println("HEIGHT : " +
getHeight());
    System.out.println("WIDTH : " +
getWidth());
  }
}

class sign {
  Scanner in = new Scanner(System.in);
  private String t;

  public void print() {
    System.out.println("ENTER SHAPE [1.
RACTANGLE 2. CIRCLE] : ");
    int n = in.nextInt();
    rectangle r = new rectangle();
    circle c = new circle();

    if (n == 1) {
      System.out.println("ENTER COLOR : ");
      r.setColor(in.next());
      System.out.println("ENTER HEIGHT : ");
      r.setHeight(in.nextInt());
      System.out.println("ENTER WIDTH : ");
      r.setWidth(in.nextInt());

    } else {
      System.out.println("ENTER COLOR : ");
      c.setColor(in.next());
      System.out.println("ENTER RADIUS : ");
      c.setRadius(in.nextInt());
```

```
        }
        System.out.println("ENTER TEXT : ");
        in.nextLine();
        t = in.nextLine();

        if (n == 1) {
            System.out.println("SIGN DETAIL :- ");
            r.printdata();
            System.out.println(t);
        } else {
            System.out.println("SIGN DETAIL :- ");
            c.printdata();
            System.out.println(t);
        }
        in.close();
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra23.java

C:\Users\HP\Desktop\java1>java Pra23.java
ENTER SHAPE [1. RACTANGLE 2. CIRCLE] :
1
ENTER COLOR :
red
ENTER HEIGHT :
2
ENTER WIDTH :
2
ENTER TEXT :
good
SIGN DETAIL :-
NAME : RECTANGLE
COLOR : red
HEIGHT : 2
WIDTH : 2
good
```

CONCLUSION:

In this practical I learnt about various methods in Interface.

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: 2024-25

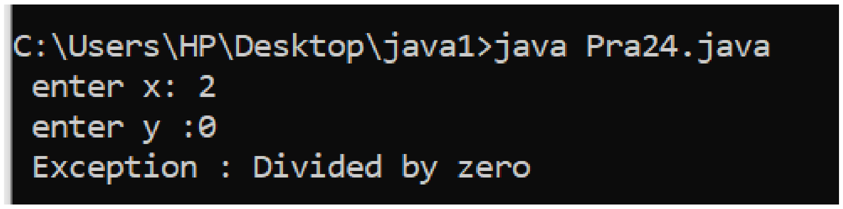
Part - 5

No.	Aim of the Practical

24.	Write a java program which takes two integers x & y as input, you have to compute x/y. If x and y are not integers or if y is zero, exception will occur and you have to report it.
	
	PROGRAM CODE:
	
	import java.util.*;
	public class Pra24
	{
	public static void main(String args[])
	{
	Scanner sc = new Scanner(System.in); int
	x=0,y=0;
	int result;
	try
	{
	System.out.print(" enter x: ");
	x = sc.nextInt();
	System.out.print(" enter y :");

```
y = sc.nextInt();
}
catch(InputMismatchException e)
{
System.out.println(" Enter valid integers ");
}
try
{
result = x/y;
System.out.println("x/y = " +result);
}
catch(ArithmeticException e)
{
System.out.println(" Exception : Divided by
zero");
}
}
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra24.java
 enter x: 2
 enter y :0
 Exception : Divided by zero
```

CONCLUSION:

In this practical learnt the simple exception handling code by using the Arithmetic Exception.

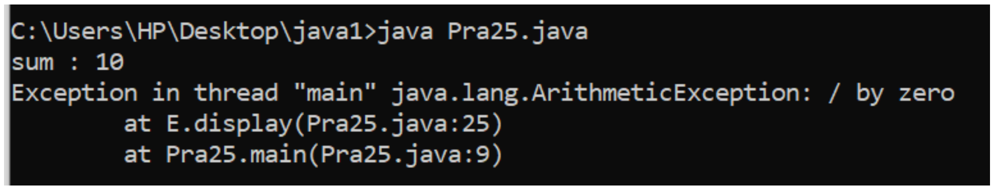
25.

Write a Java program that throws an exception and catch it using a try-catch block.

PROGRAM CODE:

```
import java.io.*;

import java.lang.*;

public class Pra25
{
   public static void main(String[] args)
   {

       try

       {
           E e1 = new E();
           e1.display();
       }catch (IOException e)

       {

        throw new ArithmeticException("Divided by zero");
       }

   }
}

class E
{

   void display() throws IOException
   {

      int a = 10;

      int b = 0;
      int sum = a+b;
      System.out.println("sum : "+sum);
      int div = a/b;
      System.out.println("div : "+div);
   }

}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra25.java
sum : 10
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at E.display(Pra25.java:25)
        at Pra25.main(Pra25.java:9)
```

CONCLUSION:

In this practical I learnt how to throw the exception using throws keyword.

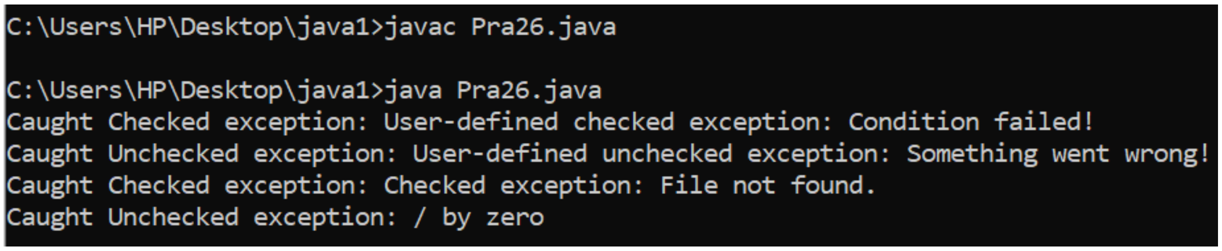
26. Write a java program to generate user defined exception using “throw” and “throws” keyword. Also Write a java that differentiates checked and unchecked exceptions. (Mention at least two checked and two unchecked exceptions in program).

PROGRAM CODE:

```
import java.io.*;
class mycheckedException extends Exception
{
public mycheckedException(String s1)
{
super(s1);
}
}
class MyUncheckedException extends RuntimeException
{
public MyUncheckedException(String s2)
{
super(s2);
}
}
public class Pra26
{
public static void main(String[] args)
 {
try
{
myexception.checkCondition(false);
}
```

```
catch (mycheckedException e)
{
System.out.println("Caught Checked exception: " +
e.getMessage());
}
try
{
myexception.riskyOperation();
}
catch (MyUncheckedException e)
{
System.out.println("Caught Unchecked exception: " + e.getMessage());
}
try
{
myexception.readFile();
}
catch (IOException e)
{
System.out.println("Caught Checked exception: " + e.getMessage());
}
try
{
myexception.divide(10, 0);
}
catch (ArithmeticException e)
{
System.out.println("Caught Unchecked exception: " + e.getMessage());
}
}
}
class myexception
{
public static void checkCondition(boolean condition) throws mycheckedException
{
if (!condition)
{
throw new mycheckedException("User-defined checked exception: Condition failed!");
}
}
public static void riskyOperation()
{
throw new MyUncheckedException("User-defined unchecked exception: Something
went wrong!");
}
public static void readFile() throws IOException
{
throw new IOException("Checked exception: File not found.");
}
public static void divide(int a, int b)
{
System.out.println("Result: " + (a / b));
}
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra26.java

C:\Users\HP\Desktop\java1>java Pra26.java
Caught Checked exception: User-defined checked exception: Condition failed!
Caught Unchecked exception: User-defined unchecked exception: Something went wrong!
Caught Checked exception: Checked exception: File not found.
Caught Unchecked exception: / by zero
```

<u>CONCLUSION:</u>

In this program we implement the user defined exception by extending the Exception class and using checked exception and unchecked exception.

CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH

Department of Computer Science & Engineering

Subject Name: JAVA PROGRAMMING

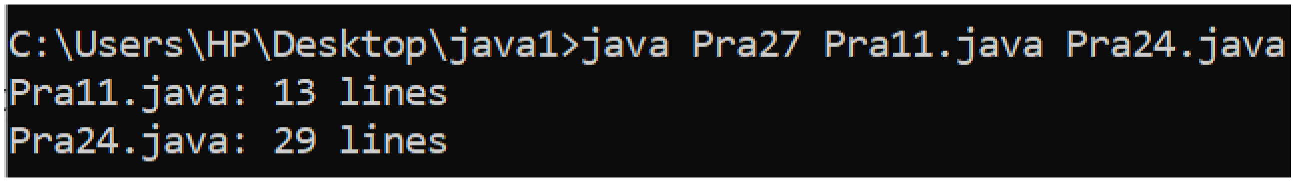
Semester: 3
Subject Code: CSE201
Academic year: 2024-25

Part - 6

No.	Aim of the Practical
27.	Design a Custom Stack using ArrayList class, which implements following functionalities of stack. My Stack -list ArrayList <Object>: 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. **PROGRAM CODE:**

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

public class Pra27 {
    public static void main(String[] args) {
        if (args.length == 0) {
            System.out.println("Please specify one or more files.");
            return;
        }

        for (String fileName : args) {
            try {
                int lineCount = countLinesInFile(fileName);
                System.out.println(fileName + ": " + lineCount + " lines");
            } catch (IOException e) {
                System.err.println("Error reading file: " + fileName + " (" +
e.getMessage() + ")");
            }
        }
    }
    private static int countLinesInFile(String fileName) throws IOException {
        int lines = 0;
        try (BufferedReader reader = new BufferedReader(new
FileReader(fileName))) {
            while (reader.readLine() != null) {
                lines++;
            }
        }
        return lines;
    }
    public void push(Object o) {
        System.out.println("Element added to the stack: " + o);
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra27 Pra11.java Pra24.java
Pra11.java: 13 lines
Pra24.java: 29 lines
```

CONCLUSION:
In this practical learnt how file handling helps us to count lines in any file.

28.

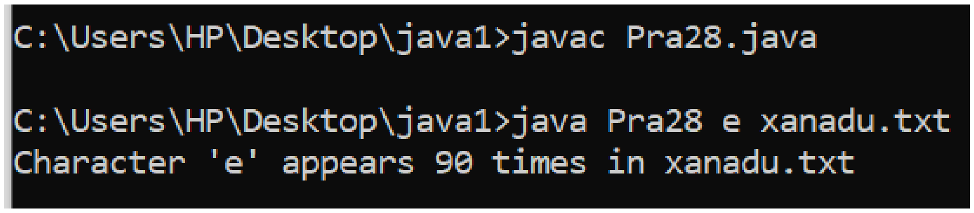
Write an example that counts the number of times a particular character, such as e, appears in a file. The character can be specified at the command line. You can use xanadu.txt as the input file.

PROGRAM CODE:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;


public class Pra28 {


public static void main(String[] args) {
        if (args.length != 2) {
           System.out.println("Usage: java Pra28 <character> <filename>");
           return;
        }


        char targetChar = args[0].charAt(0); // The character to search for
        String fileName = args[1]; // The file name to process


        int charCount = 0; // To store the count of the target character
        try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {
           int currentChar;


           while ((currentChar = reader.read()) != -1) {
              if (currentChar == targetChar) {
                 charCount++;
              }
           }
```

```
            System.out.println("Character '" + targetChar + "' appears " + charCount + " times in
" + fileName);


        } catch (IOException e)
        {
            System.err.println("Error reading file: " + fileName);
        }
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra28.java

C:\Users\HP\Desktop\java1>java Pra28 e xanadu.txt
Character 'e' appears 90 times in xanadu.txt
```

CONCLUSION:

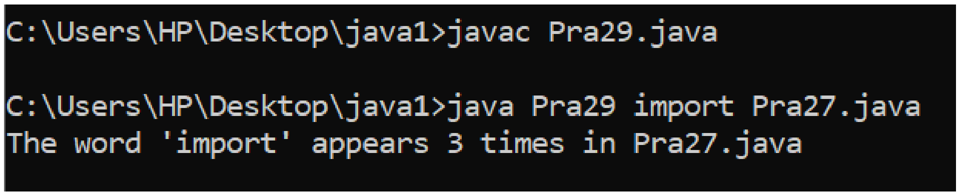
In this practical I learnt how to search a character using file handling in java.

29. Write a Java Program to Search for a given word in a File. Also show use of Wrapper Class with an example.

PROGRAM CODE:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

public class Pra29 {
    public static void main(String[] args) {
        if (args.length != 2) {
            System.out.println("Usage: java Pra29 <word> <filename>");
            return;
        }

        String searchWord = args[0];
        String fileName = args[1];
        int occurrences = searchWordInFile(searchWord, fileName);

        Integer result = Integer.valueOf(occurrences);

        if (result > 0) {
            System.out.println("The word '" + searchWord + "' appears " + result + " times
in " + fileName);
        } else {
            System.out.println("The  word '"  + searchWord +  "'  was  not found  in  "  +
fileName);
        }
    }

    public static int searchWordInFile(String word, String fileName) {
        int count = 0;

        try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {
            String line;
            while ((line = reader.readLine()) != null) {
                String[] words = line.split("\\s+");
                for (String w : words) {
                    if (w.equals(word)) {
                        count++;
                    }
                }
            }

        } catch (IOException e) {
            System.err.println("Error reading file: " + fileName);
        }

        return count;
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra29.java

C:\Users\HP\Desktop\java1>java Pra29 import Pra27.java
The word 'import' appears 3 times in Pra27.java
```

CONCLUSION:

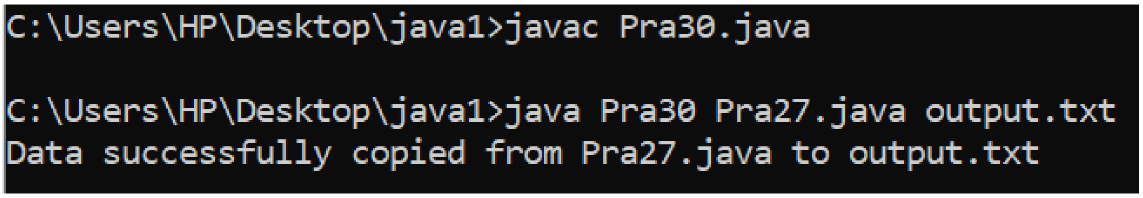
In this program I learnt about how to count searched word in specific file by file handling.

30. Write a program to copy data from one file to another file. If the destination file does not exist, it is created automatically.

PROGRAM CODE:

```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class Pra30 {

    public static void main(String[] args) {
        if (args.length != 2) {
            System.out.println("Usage: java Pra30 <sourceFile> <destinationFile>");
            return;
        }

        String sourceFile = args[0];
        String destinationFile = args[1];

        copyFile(sourceFile, destinationFile);
    }

    public static void copyFile(String sourceFile, String destinationFile) {
        try (FileReader fileReader = new FileReader(sourceFile);
             FileWriter fileWriter = new FileWriter(destinationFile)) {

            int character;

            while ((character = fileReader.read()) != -1) {
                fileWriter.write(character);
            }
            System.out.println("Data successfully copied from " + sourceFile + " to " + destinationFile);

        } catch (IOException e) {
            System.err.println("Error occurred while copying data: " + e.getMessage());
        }
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra30.java

C:\Users\HP\Desktop\java1>java Pra30 Pra27.java output.txt
Data successfully copied from Pra27.java to output.txt
```

CONCLUSION:

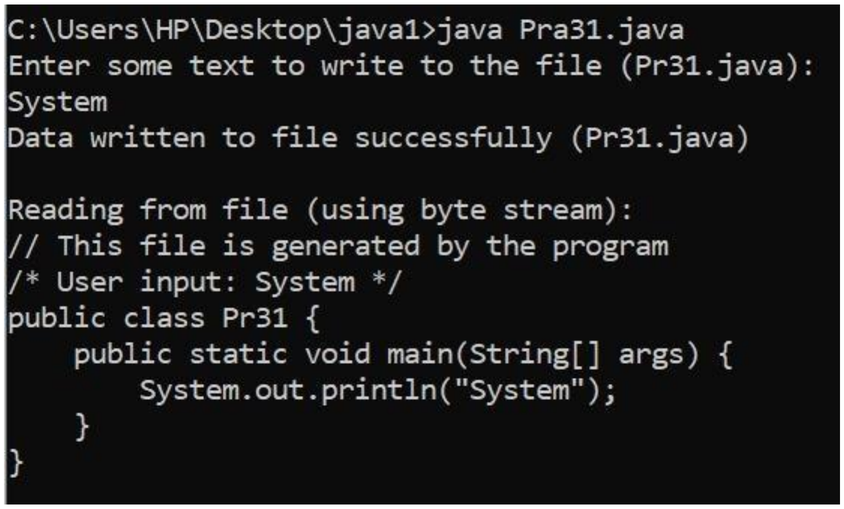
In this program I learnt about how to transfer data from one file to other file using file handling concept.

31. Write a program to show use of character and byte stream. Also show use of BufferedReader/BufferedWriter to read console input and write them into a file.

PROGRAM CODE:

```
import java.io.*;
public class Pra31 {
    public static void main(String[] args) {
        try {
            BufferedReader consoleReader = new BufferedReader(new
InputStreamReader(System.in));

            System.out.println("Enter some text to write to the file (Pr31.java):");
            String userInput = consoleReader.readLine();

            BufferedWriter fileWriter = new BufferedWriter(new FileWriter("Pr31.java"));
            fileWriter.write("// This file is generated by the program\n");
            fileWriter.write("/* User input: " + userInput + " */\n");
            fileWriter.write("public class Pra31 {\n");
            fileWriter.write("    public static void main(String[] args) {\n");
            fileWriter.write("        System.out.println(\"" + userInput + "\");\n");
            fileWriter.write("    }\n");
            fileWriter.write("}\n");

            fileWriter.close();

            System.out.println("Data written to file successfully (Pr31.java)");



            FileInputStream fileInputStream = new FileInputStream("Pr31.java");
            System.out.println("\nReading from file (using byte stream):");

int byteData;
            while ((byteData = fileInputStream.read()) != -1) {
                System.out.print((char) byteData);
            }
            fileInputStream.close();

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra31.java
Enter some text to write to the file (Pr31.java):
System
Data written to file successfully (Pr31.java)

Reading from file (using byte stream):
// This file is generated by the program
/* User input: System */
public class Pr31 {
    public static void main(String[] args) {
        System.out.println("System");
    }
}
```

CONCLUSION:
In this practical I have learnt about BufferedReader/BufferedWriter to read console input and write them into a file.

CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH

Department of Computer Science & Engineering

Subject Name: JAVA PROGRAMMING

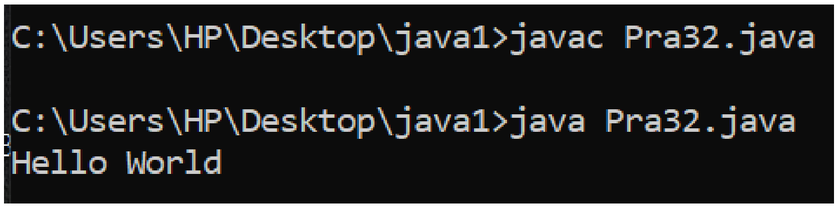
Semester: 3
Subject Code: CSE201
Academic year: 2024-25

Part - 7

No.	Aim of the Practical

32.	Write a program to create thread which display "Hello World" message. A. by extending Thread class B. by using Runnable interface.
	
	PROGRAM CODE:
	
	`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();`
	`}`
	`}`

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra32.java

C:\Users\HP\Desktop\java1>java Pra32.java
Hello World
```

CONCLUSION:

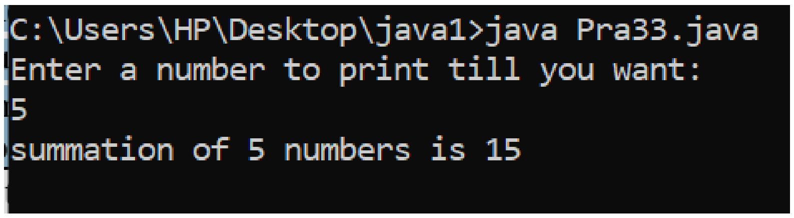
In this practical learnt how to create a thread using runnable interface.

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();

        Thread thread = new Thread(p1);

        thread.start();

    }

}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra33.java
Enter a number to print till you want:
5
summation of 5 numbers is 15
```

CONCLUSION:

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.

34.

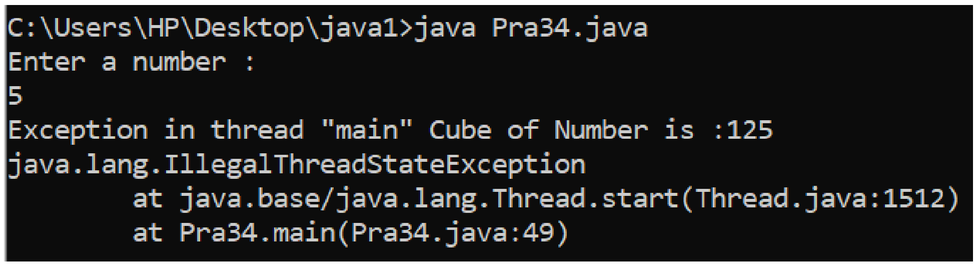
Write a java program that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.

PROGRAM CODE:

```
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);
```

```
        }
    }

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:



```
C:\Users\HP\Desktop\java1>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)
```

CONCLUSION:

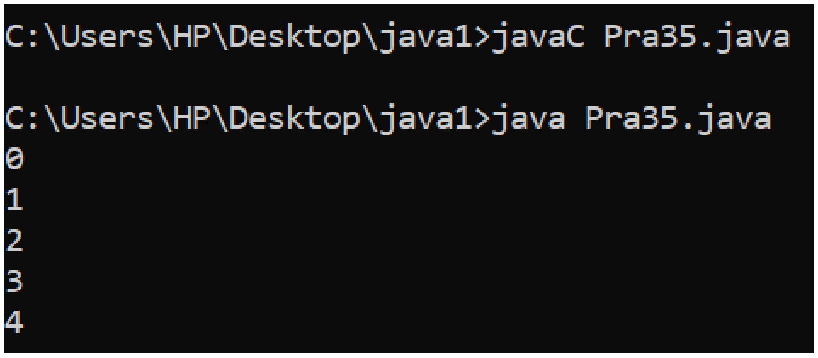
In this program I learnt about multiple threads that each thread can have it`s own void run method so every thread can do different task.

35. Write a program to increment the value of one variable by one and display it after one second using thread using sleep() method.

PROGRAM:

```
import java.io.*;
import java.lang.Thread;

class Pra35{
        public static void main(String[] args)
        {
try {
for (int i = 0; i < 5; i++)
        {
        Thread.sleep(1000);
        System.out.println(i);
        }
        }
catch (Exception e)
{
        System.out.println(e);
}
        }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javaC Pra35.java

C:\Users\HP\Desktop\java1>java Pra35.java
0
1
2
3
4
```

CONCLUSION:

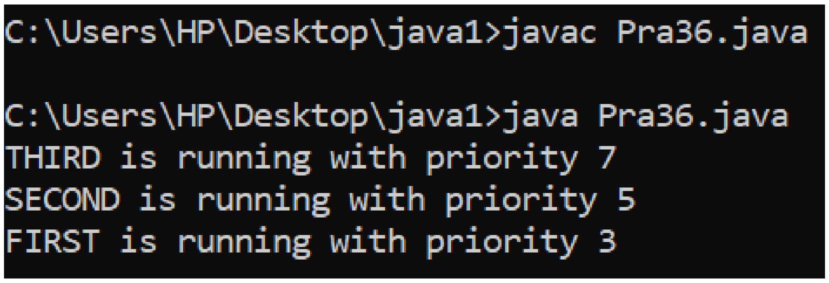
In this program I learnt about sleep method to delay the execution of any thread so that other threads perform their task without disturbance.

36. 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.

PROGRAM:

```
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);


        firstThread.setPriority(3);
        secondThread.setPriority(5);
        thirdThread.setPriority(7);

        // Start the threads
        firstThread.start();
        secondThread.start();
        thirdThread.start();
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra36.java

C:\Users\HP\Desktop\java1>java Pra36.java
THIRD is running with priority 7
SECOND is running with priority 5
FIRST is running with priority 3
```

CONCLUSION:

In this program I learnt about thread working priorities.

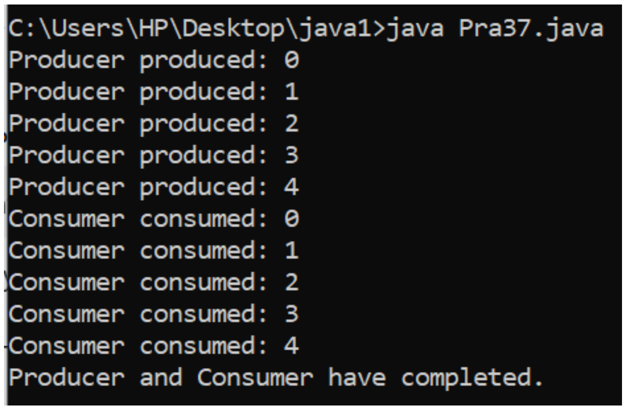
37. Write a program to solve producer-consumer problem using thread synchronization.

PROGRAM:

```
import java.util.LinkedList;

class Buffer {
    private LinkedList<Integer> list = new LinkedList<>();
    private int capacity = 5


    public synchronized void produce() throws InterruptedException {
        int value = 0;
        for (int i = 0; i < 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 {
        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();

     System.out.println("Producer and Consumer have completed.");
  }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>java Pra37.java
Producer produced: 0
Producer produced: 1
Producer produced: 2
Producer produced: 3
Producer produced: 4
Consumer consumed: 0
Consumer consumed: 1
Consumer consumed: 2
Consumer consumed: 3
Consumer consumed: 4
Producer and Consumer have completed.
```

CONCLUSION:

In this program I learnt about how to implement thread synchronization and what is importance of thread synchronization.

CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH

Department of Computer Science & Engineering

Subject Name: JAVA

Semester: 3 RD
Subject Code:
Academic year: 2024-2025

Part - 8

No.	Aim of the Practical
1.	Design a Custom Stack using ArrayList class, which implements following functionalities of stack. My Stack -list ArrayList<Object>: 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. **PROGRAM CODE :** import java.util.ArrayList; import java.util.Scanner;

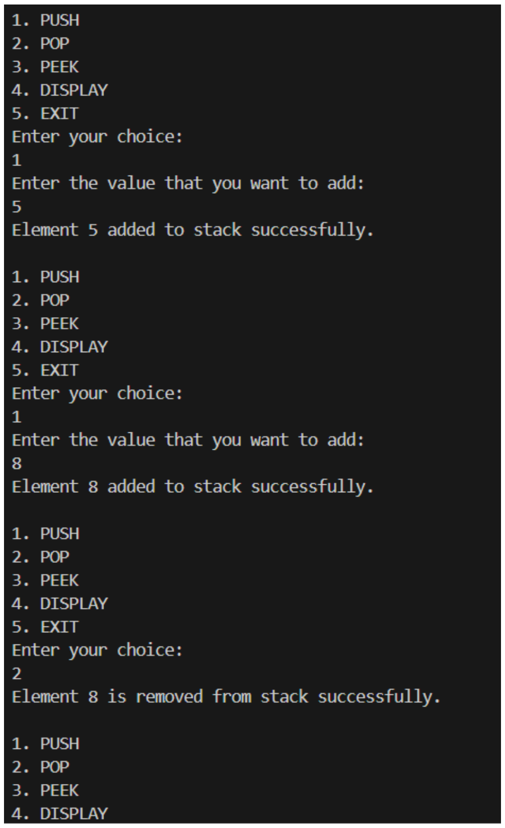
```
class Stack {


    ArrayList<Integer> array = new
ArrayList<>();
public boolean isEmpty() {
        return array.isEmpty();
    } public int getSize() {
        return array.size();
    }public void push(int x) {
        array.add(x);
        System.out.println("Element " + x + "
added to stack successfully.");
    } public void pop() {
        if (isEmpty()) {
            System.out.println("Stack is
underflow!!");
        } else {
            int y = array.remove(array.size() - 1);
            System.out.println("Element " + y + "
is removed from stack successfully.");
        }
    }
```

```
public int peek() {
        if (isEmpty()) {
            System.out.println("Stack is empty!");
            return -1;
        }
        return array.get(array.size() - 1);
    }
public void print() {
        if (isEmpty()) {
            System.out.println("Stack is empty.");
        } else {
            System.out.println("Stack elements: "
 + array);
        }
    }
 }

 public class Pra381 {
    public static void main(String[] args) {
        Stack stack = new Stack();
        Scanner sc = new Scanner(System.in);
        int choice;
```

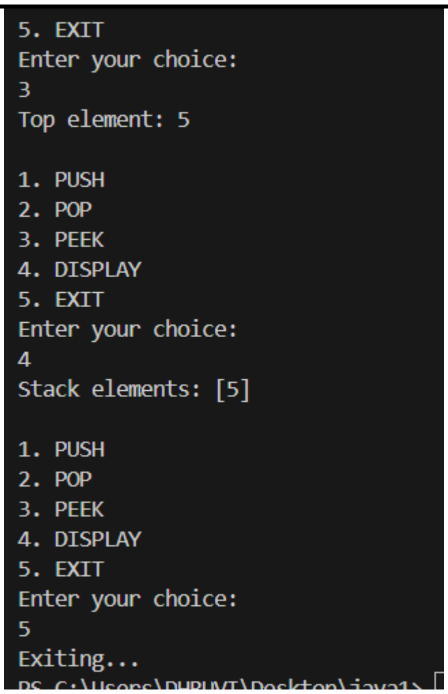
```
  do {
          System.out.println("\n1. PUSH\n2.
POP\n3. PEEK\n4. DISPLAY\n5. EXIT");
          System.out.println("Enter your choice:
");
          choice = sc.nextInt();

          switch (choice) {
            case 1:
              System.out.println("Enter the
value that you want to add: ");
              int x = sc.nextInt();
              stack.push(x);
              break;
 case 2:
              stack.pop();
              break;
case 3:
              int topElement = stack.peek();
              if (topElement != -1) {
                System.out.println("Top
element: " + topElement);
```

```
                    }

                    break;

case 4:

                    stack.print();

                    break;



                case 5:

                    System.out.println("Exiting...");

                    break;

default:

                    System.out.println("Invalid

input!!");

 break;

            }

        } while (choice != 5);

    }

}
```

OUTPUT:



```
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your choice:
1
Enter the value that you want to add:
5
Element 5 added to stack successfully.

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your choice:
1
Enter the value that you want to add:
8
Element 8 added to stack successfully.

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your choice:
2
Element 8 is removed from stack successfully.

1. PUSH
2. POP
3. PEEK
4. DISPLAY
```



```
5. EXIT
Enter your choice:
3
Top element: 5

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your choice:
4
Stack elements: [5]

1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your choice:
5
Exiting...
PS C:\Users\DHRUVI\Desktop\java1>
```

CONCLUSION:

Here in this practical we learned about Stack using ArrayList class.

39. 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.

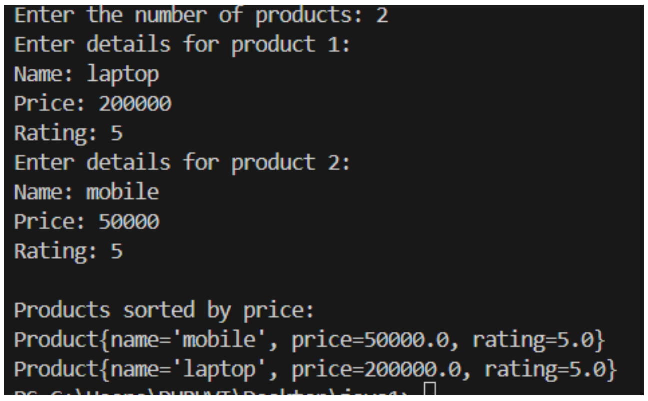
PROGRAM CODE :

```
import java.util.Arrays;

import java.util.Scanner;

class Product implements
```

	Comparable<Product> {
	
	    private String name;
	
	    private double price;
	
	    private double rating;
	
	public Product(String name, double price,
	
	double rating) {
	
	        this.name = name;
	
	        this.price = price;
	
	        this.rating = rating;
	
	    }
	
	@Override
	
	    public int compareTo(Product other) {
	
	        return Double.compare(this.price,
	
	other.price);
	
	    }
	
	@Override
	
	    public String toString() {
	
	        return "Product{name='" + name + "',
	
	price=" + price + ", rating=" + rating + "}";
	
	    }
	
	}
	
	public class Pra39 {

```
public static void sortArray(Comparable[]
array) {
        Arrays.sort(array);
    }
public static void main(String[] args) {
        Scanner scanner = new
Scanner(System.in);


        System.out.print("Enter the number of
products: ");
        int numProducts = scanner.nextInt();
        Product[] products = new
Product[numProducts];
 for (int i = 0; i < numProducts; i++) {
            System.out.println("Enter details for
product " + (i + 1) + ":");
            System.out.print("Name: ");
            String name = scanner.next();
            System.out.print("Price: ");
            double price = scanner.nextDouble();
            System.out.print("Rating: ");
            double rating = scanner.nextDouble();
products[i] = new Product(name, price, rating);

        }

 sortArray(products);

 System.out.println("\nProducts sorted by

 price:");

        for (Product product : products) {

            System.out.println(product);

        } scanner.close();

    }

 }
```

OUTPUT:



```
Enter the number of products: 2
Enter details for product 1:
Name: laptop
Price: 200000
Rating: 5
Enter details for product 2:
Name: mobile
Price: 50000
Rating: 5

Products sorted by price:
Product{name='mobile', price=50000.0, rating=5.0}
Product{name='laptop', price=200000.0, rating=5.0}
```

CONCLUSION:

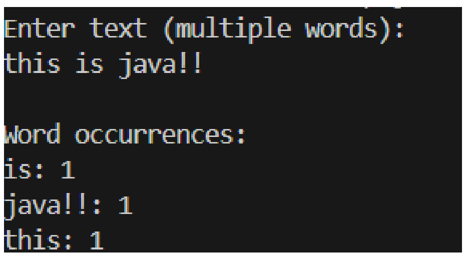
Here in this practical we learned about sorting.

40. 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.

PROGRAM CODE :

```
import java.util.*;


public class Pra40 {
public static void main(String[] args) {
  Scanner scanner = new Scanner(System.in);
System.out.println("Enter text (multiple words):");
  String text = scanner.nextLine();
String[] words = text.split("\\s+");
 Map<String, Integer> wordCountMap = new HashMap<>();
for (String word : words) {
            word = word.toLowerCase();
            wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);
        }

        Set<String> wordSet = new TreeSet<>(wordCountMap.keySet());

        System.out.println("\nWord occurrences:");
        for (String word : wordSet) {
            System.out.println(word + ": " + wordCountMap.get(word));
        }

        scanner.close();
    }
}
```

OUTPUT:



```
Enter text (multiple words):
this is java!!

Word occurrences:
is: 1
java!!: 1
this: 1
```

CONCLUSION:

Here in this practical we learned about a program that counts the occurrences of words in a text and displays the words and their occurrences in alphabetical order of the words.

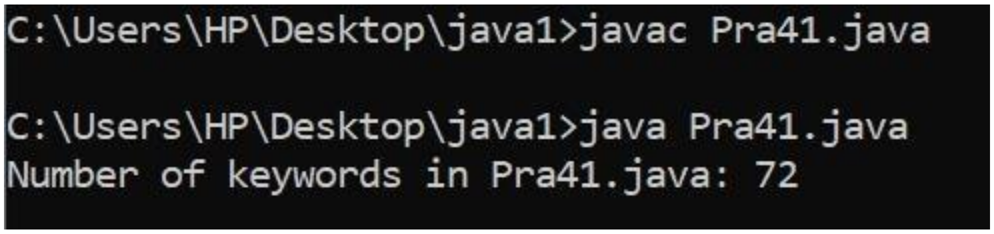
41.

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.

PROGRAM CODE:

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

public class Pra41 {

    private static final Set<String> JAVA_KEYWORDS = new HashSet<>(Arrays.asList(
        "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"
    ));

    public static void main(String[] args) {
        // File to analyze (replace with actual source file path)
        String fileName = "Pra41.java";

        try {

            BufferedReader fileReader = new BufferedReader(new FileReader(fileName));
            String line;
            int keywordCount = 0;


            while ((line = fileReader.readLine()) != null) {

                String[] words = line.split("\\W+");
                for (String word : words) {
                    if (JAVA_KEYWORDS.contains(word)) {
                        keywordCount++;
                    }
                }
            }

            fileReader.close();
            System.out.println("Number of keywords in " + fileName + ": " + keywordCount);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

OUTPUT:



```
C:\Users\HP\Desktop\java1>javac Pra41.java

C:\Users\HP\Desktop\java1>java Pra41.java
Number of keywords in Pra41.java: 72
```

CONCLUSION:

Here in this practical we learned about how to counts the number of the keywords in a Java source file.