**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“JnanaSangama”, Belgaum -590014, Karnataka.**



LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

***Submitted by***

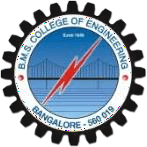
SANCHIT MEHTA (1BM23CS299)

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING

**(Autonomous Institution under VTU)**

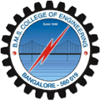
BENGALURU-560019 Sep 2024-Jan 2025

B. M. S. College of Engineering,

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that the Lab work entitled “**OBJECT ORIENTED JAVA PROGRAMMING**” carried out by **SANCHIT MEHTA(1BM23CS299),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

**Dr. Nandhini Vineeth Dr. Kavitha Sooda**

Associate Professor, Professor and Head,

Department of CSE, Department of CSE

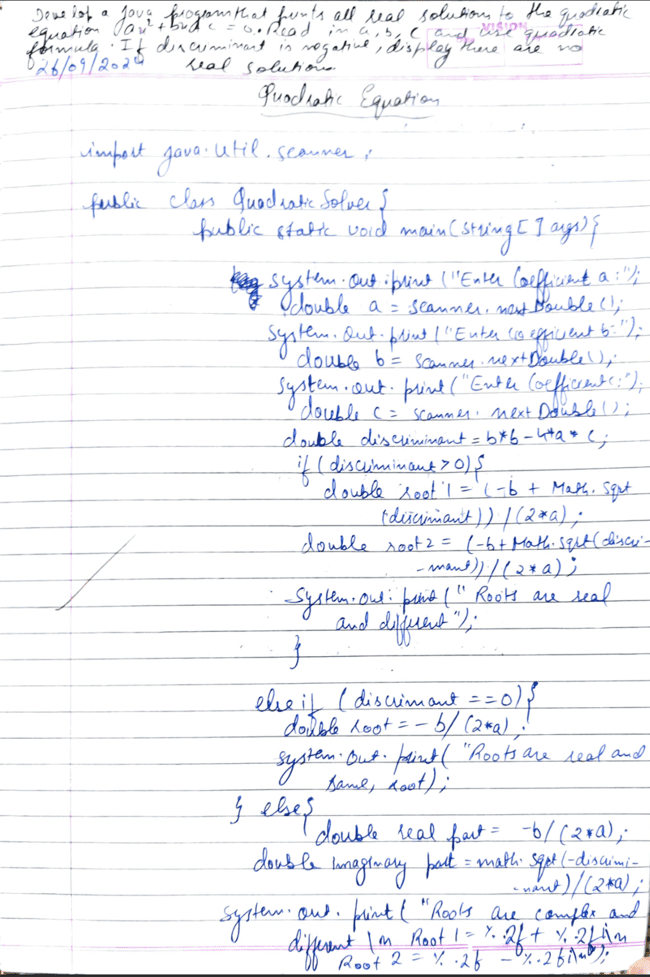
BMSCE, Bengaluru BMSCE, Bengaluru

**INDEX**

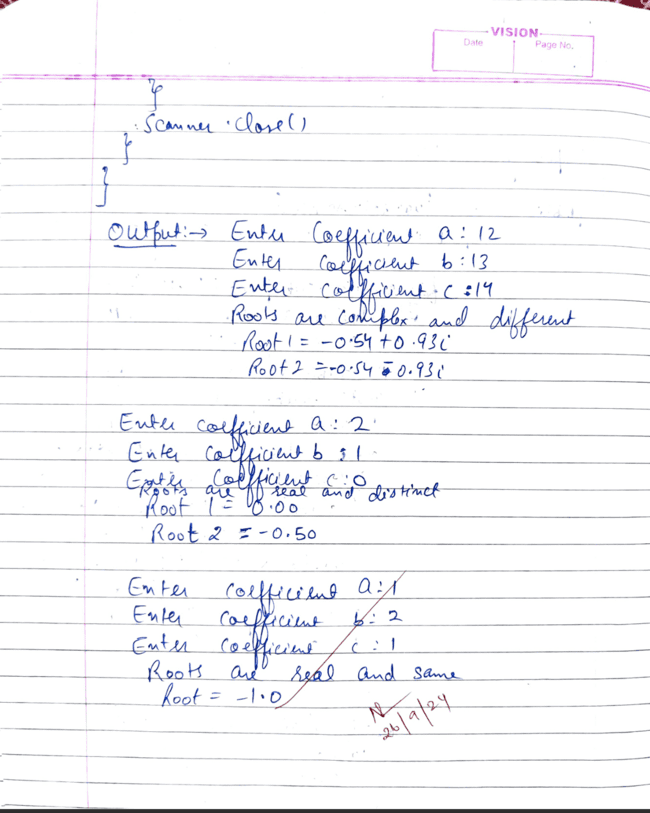
|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 | 20/09/24 | Quadratic Equation | 1 |
| 2 | 03/10/24 | SGPA Calculator | 5 |
| 3 | 19/10/24 | Book Details | 10 |
| 4 | 24/10/24 | Abstract Shape | 17 |
| 5 | 07/10/24 | Bank Operations | 21 |
| 6 | 14/11/24 | CIE/SEE Packages | 30 |
| 7 | 21/11/24 | Father-Son Age Exception | 39 |
| 8 | 05/12/24 | Multi-Threading | 43 |
| 9 | 12/12/24 | Custom division using awt | 46 |
| 10 | 19/12/24 | Demonstrate Inter process Communication and deadlock | 49 |

**LABORATORY PROGRAM – 1**

Develop a Java program that prints all real solutions to the quadratic equation ax2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions



1



2

Code:

import java.util.Scanner;

public class lab1 {

public static void main(String[] args) {

try (Scanner scanner = new Scanner(System.in)) {

System.out.print("Enter coefficient a: ");

double a = scanner.nextDouble();

System.out.print("Enter coefficient b: ");

double b = scanner.nextDouble();

System.out.print("Enter coefficient c: ");

double c = scanner.nextDouble();

if (a == 0) {

System.out.println("Coefficient a cannot be zero for a quadratic equation.");

return;

}

double discriminant = b \* b - 4 \* a \* c;

if (discriminant > 0) {

double sqrtDiscriminant = Math.sqrt(discriminant);

double root1 = (-b + sqrtDiscriminant) / (2 \* a);

double root2 = (-b - sqrtDiscriminant) / (2 \* a);

System.out.println("The roots are real and different.");

System.out.println("Root 1: " + root1);

System.out.println("Root 2: " + root2);

} else if (discriminant == 0) {

double root = -b / (2 \* a);

3

System.out.println("The root is real and repeated.");

System.out.println("Root: " + root);

} else {

System.out.println("There are no real solutions.");

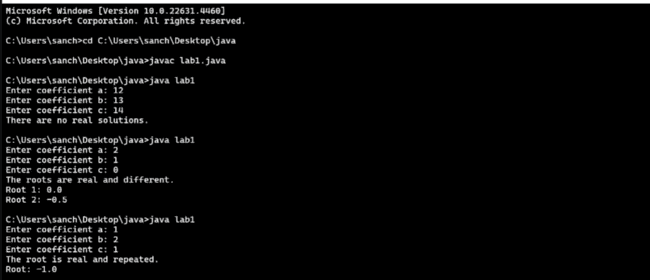
}

}

}

}

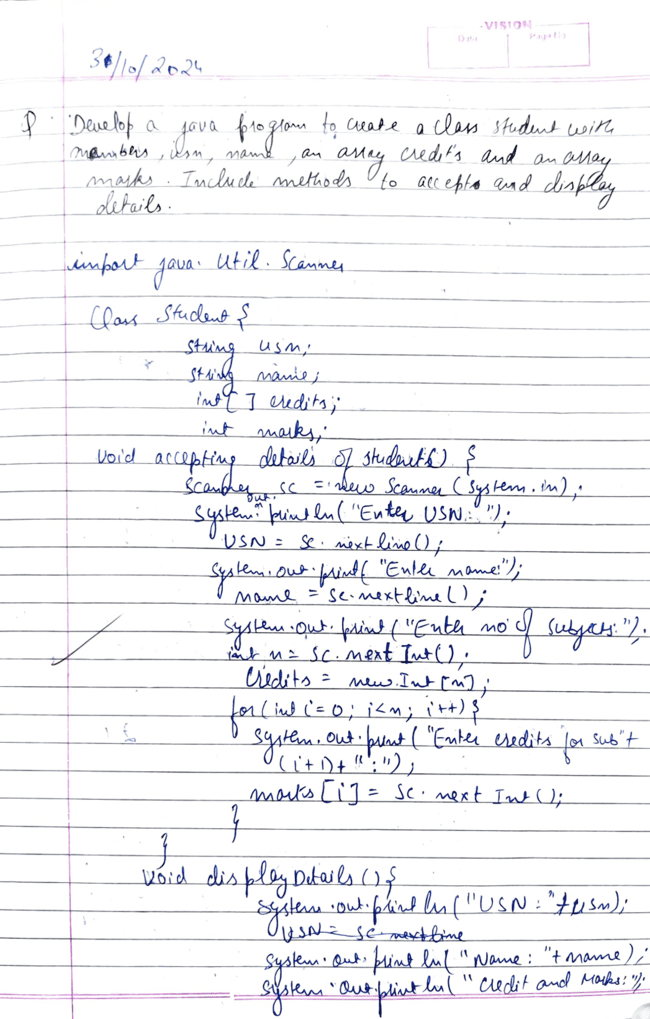
OUTPUT



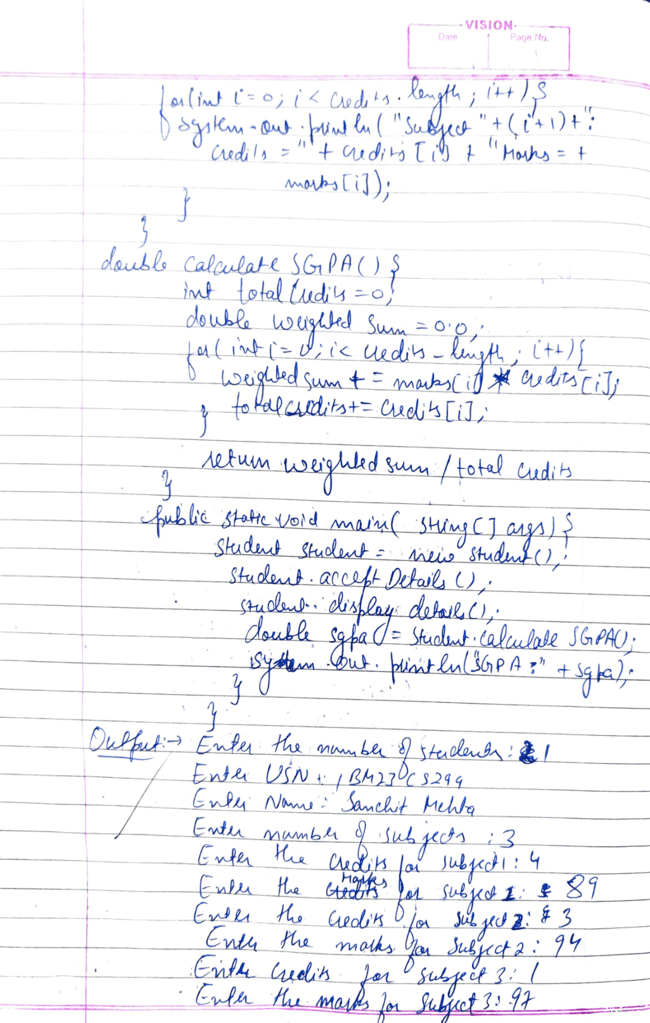
4

**LABORATORY PROGRAM – 2**

Develop a Java program to create a class Student withmembers usn, name, an array credits and an array marks.Include methods to accept and display details and amethod to calculate SGPA of a student



5



6



Code:

import java.util.Scanner;

class Student {

String usn;

String name;

int[] credits;

int[] marks;

public Student(int numSubjects) {

credits = new int[numSubjects];

marks = new int[numSubjects];

}

public void acceptDetails() {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter USN: ");

usn = scanner.nextLine();

System.out.print("Enter Name: ");

name = scanner.nextLine();

for (int i = 0; i < credits.length; i++) {

System.out.print("Enter credits for subject " + (i + 1) + ": ");

credits[i] = scanner.nextInt();

System.out.print("Enter marks for subject " + (i + 1) + ": ");

marks[i] = scanner.nextInt();

}

7

}

public void displayDetails() {

System.out.println("USN: " + usn);

System.out.println("Name: " + name);

for (int i = 0; i < credits.length; i++) {

System.out.println("Subject " + (i + 1) + " - Credits: " + credits[i] + ", Marks: " + marks[i]);

}

}

public double calculateSGPA() {

double totalCredits = 0;

double totalGradePoints = 0;

for (int i = 0; i < credits.length; i++) {

double gradePoint = getGradePoint(marks[i]);

totalGradePoints += gradePoint \* credits[i];

totalCredits += credits[i];

}

return totalCredits == 0 ? 0 : totalGradePoints / totalCredits;

}

public double getGradePoint(int mark) {

if (mark >= 90) return 10.0;

else if (mark >= 80) return 9.0;

else if (mark >= 70) return 8.0;

else if (mark >= 60) return 7.0;

else if (mark >= 50) return 6.0;

else if (mark >= 40) return 5.0;

else return 0.0; // Fail

}

}

public class lab2 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter number of subjects: ");

int numSubjects = scanner.nextInt();

Student student = new Student(numSubjects);

student.acceptDetails();

student.displayDetails();

double sgpa = student.calculateSGPA();

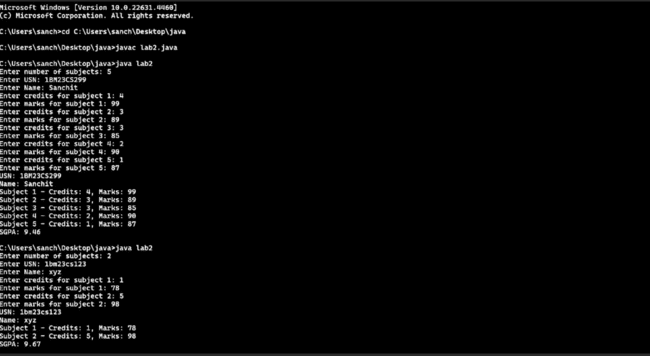
System.out.printf("SGPA: %.2f\n", sgpa);

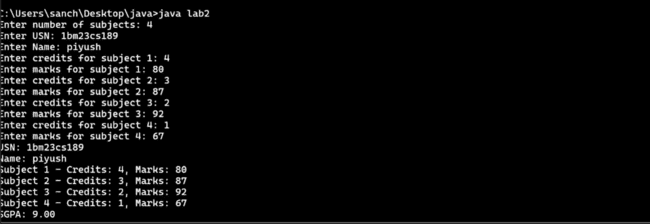
}

}

8

**OUTPUT**

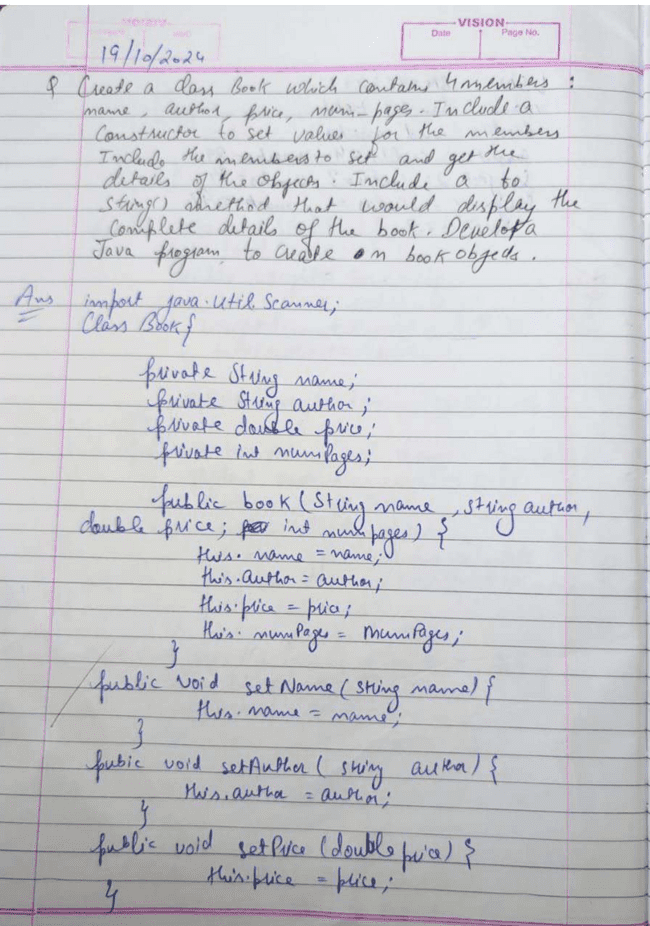
****

****

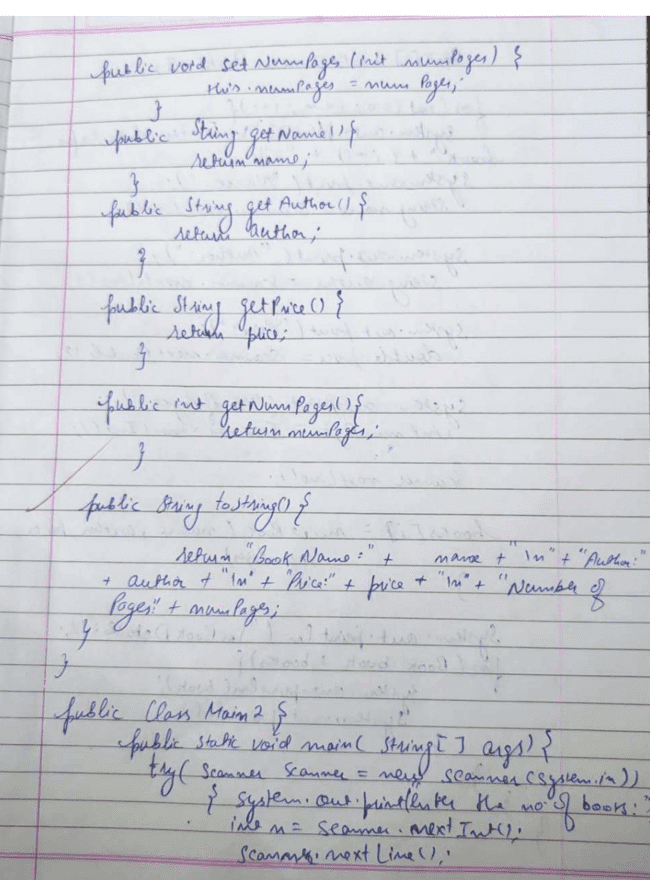
9

**LABORATORY PROGRAM – 3**

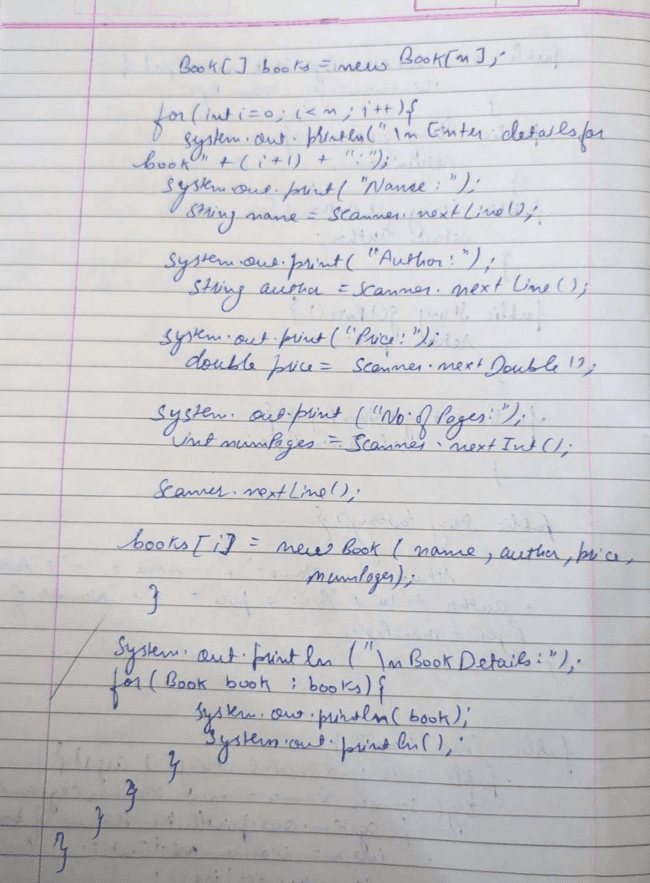
Create a class Book which contains four members: name, author, price, num\_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to create n book objects.



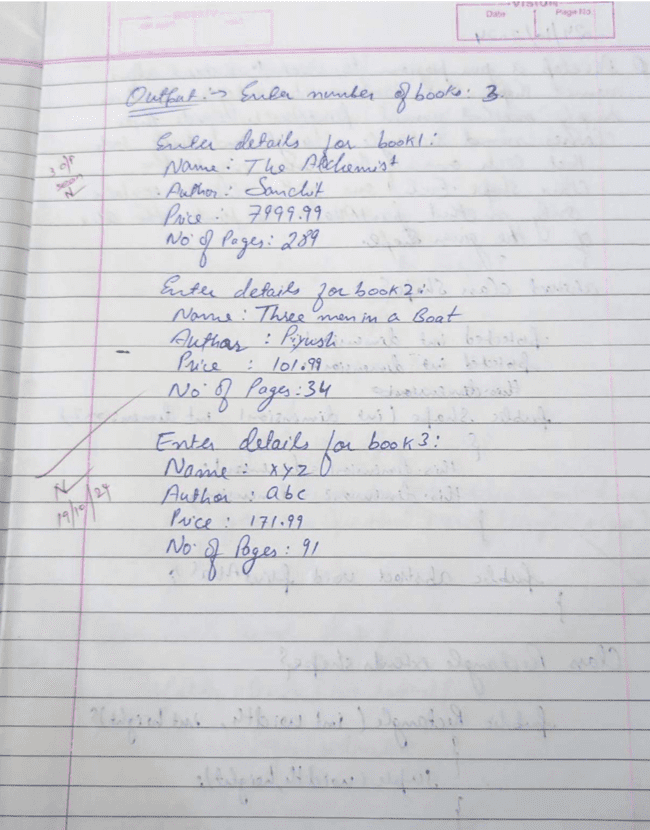
10

****

11

****

12

****

13

Code:

import java.util.Scanner;

class Book {

private String name;

private String author;

private double price;

private int numPages;

public Book(String name, String author, double price, int numPages) {

this.name = name;

this.author = author;

this.price = price;

this.numPages = numPages;

}

public void setName(String name) {

this.name = name;

}

public void setAuthor(String author) {

this.author = author;

}

public void setPrice(double price) {

this.price = price;

}

public void setNumPages(int numPages) {

this.numPages = numPages;

}

public String getName() {

return name;

}

public String getAuthor() {

return author;

}

public double getPrice() {

return price;

}

public int getNumPages() {

return numPages;

14

}

public String toString() {

return "Book Name: " + name + "\n" +

"Author: " + author + "\n" +

"Price: $" + price + "\n" +

"Number of Pages: " + numPages;

}

}

public class LAB3{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of books: ");

int n = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

Book[] books = new Book[n];

for (int i = 0; i < n; i++) {

System.out.println("\nEnter details for book " + (i + 1) + ":");

System.out.print("Name: ");

String name = scanner.nextLine();

System.out.print("Author: ");

String author = scanner.nextLine();

System.out.print("Price: ");

double price = scanner.nextDouble();

System.out.print("Number of Pages: ");

int numPages = scanner.nextInt();

scanner.nextLine();

books[i] = new Book(name, author, price, numPages);

}

System.out.println("\nBook Details:");

for (Book book : books) {

System.out.println(book);

System.out.println();

}

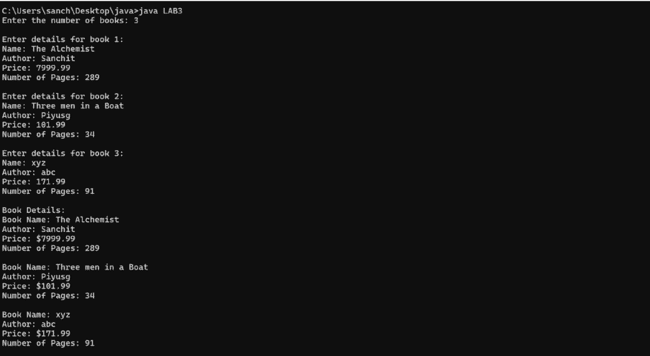
scanner.close();

15

}

}

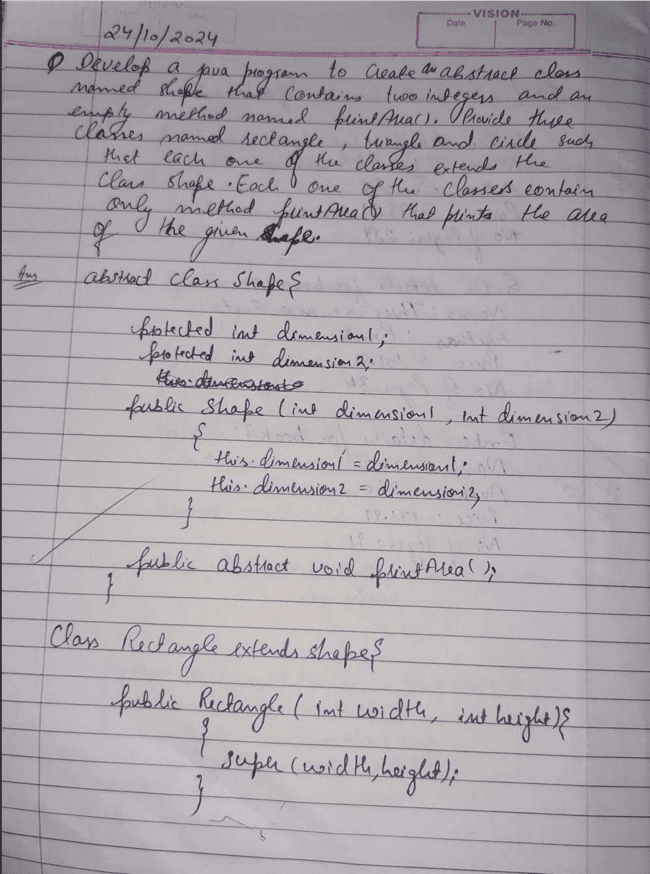
**OUTPUT**

****

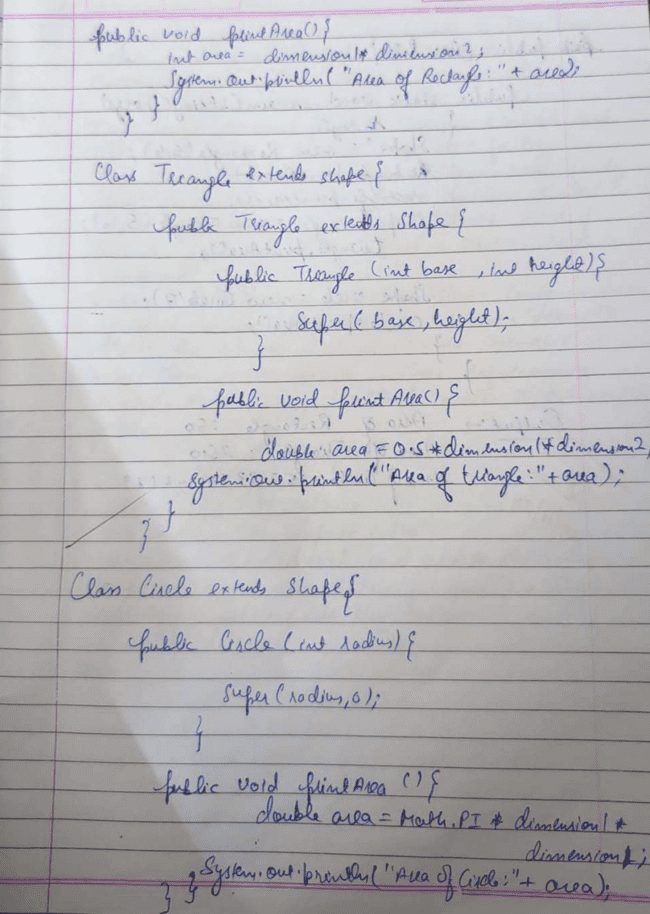
16

**LABORATORY PROGRAM – 4**

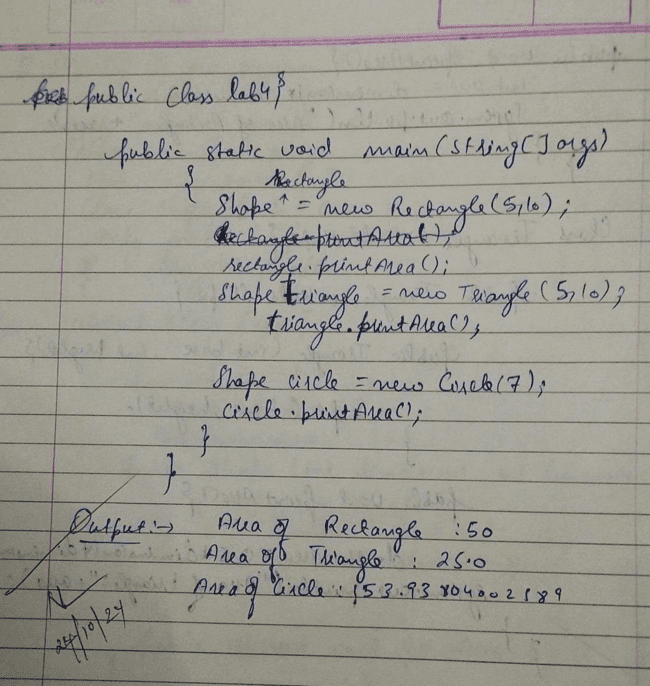
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given shape.



17



18



Code:  
abstract class Shape {

int dimension1;

int dimension2;

Shape(int dimension1, int dimension2) {

this.dimension1 = dimension1;

this.dimension2 = dimension2;

}

public abstract void printArea();

}

class Rectangle extends Shape {

public Rectangle(int width, int height) {

19

super(width, height);

}

public void printArea() {

int area = dimension1 \* dimension2;

System.out.println("Area of Rectangle: " + area);

}

}

class Triangle extends Shape {

public Triangle(int base, int height) {

super(base, height);

}

public void printArea() {

double area = 0.5 \* dimension1 \* dimension2;

System.out.println("Area of Triangle: " + area);

}

}

class Circle extends Shape {

public Circle(int radius) {

super(radius, 0);

}

public void printArea() {

double area = Math.PI \* dimension1 \* dimension1;

System.out.println("Area of Circle: " + area);

}

}

public class lab4 {

public static void main(String[] args) {

Shape rectangle = new Rectangle(5, 10);

rectangle.printArea();

Shape triangle = new Triangle(5, 10);

triangle.printArea();

Shape circle = new Circle(7);

circle.printArea();

}

}

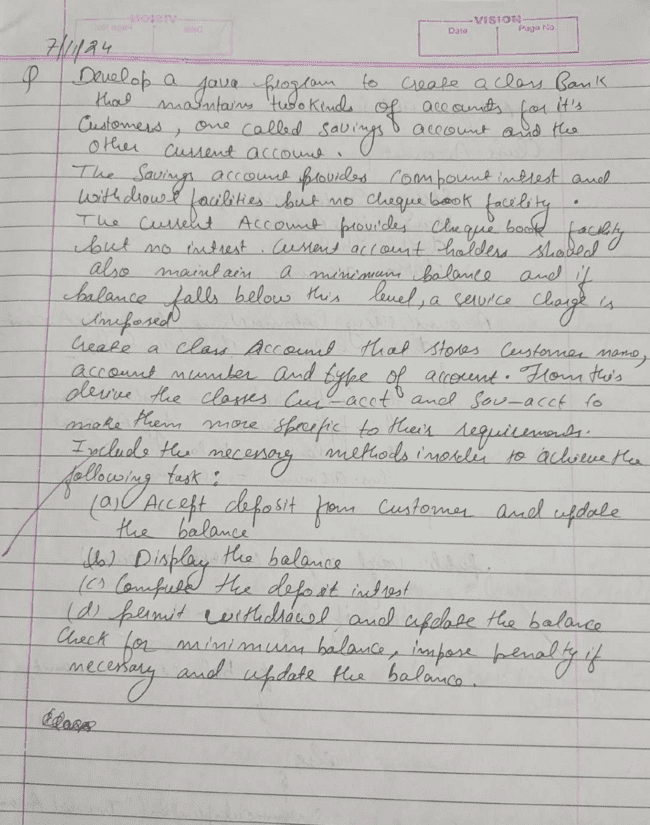
**OUTPUT**

****

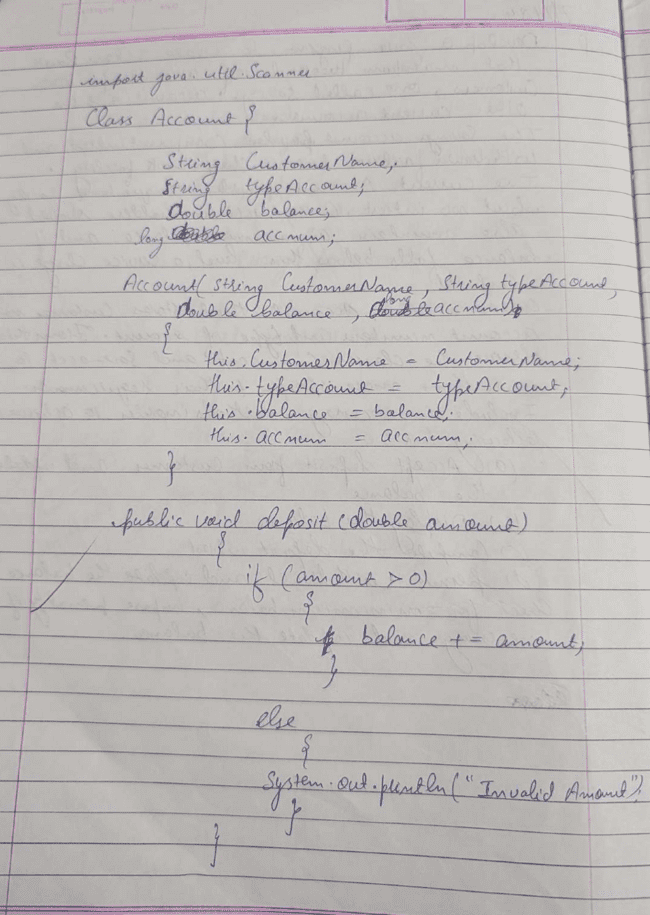
20

**LABORATORY PROGRAM – 5**

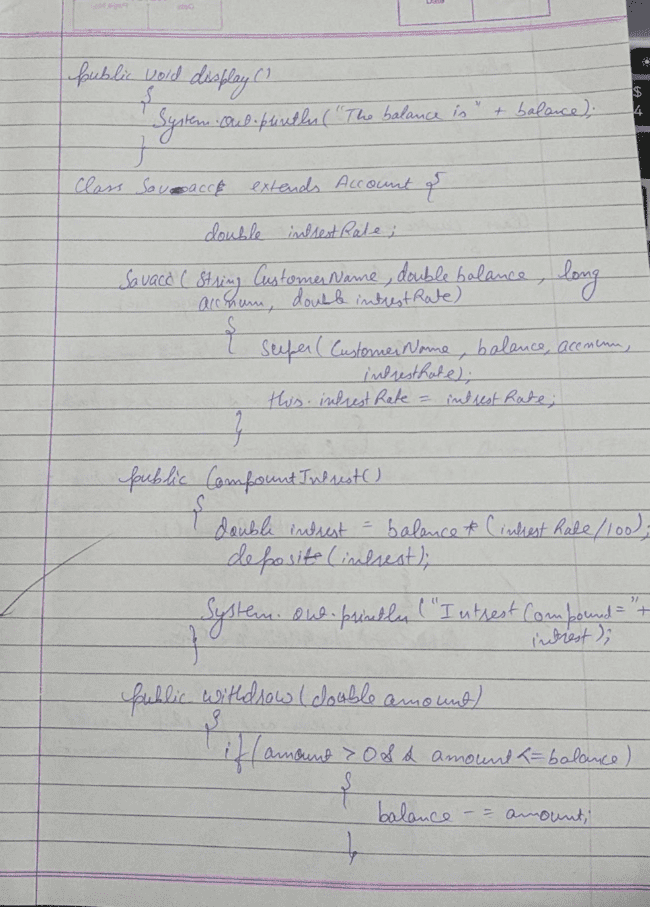
Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks: a) Accept deposit from customer and update the balance. b) Display the balance. c) Compute and deposit interest d) Permit withdrawal and update the balance Check for the minimum balance, impose penalty if necessary and update the balance.

****

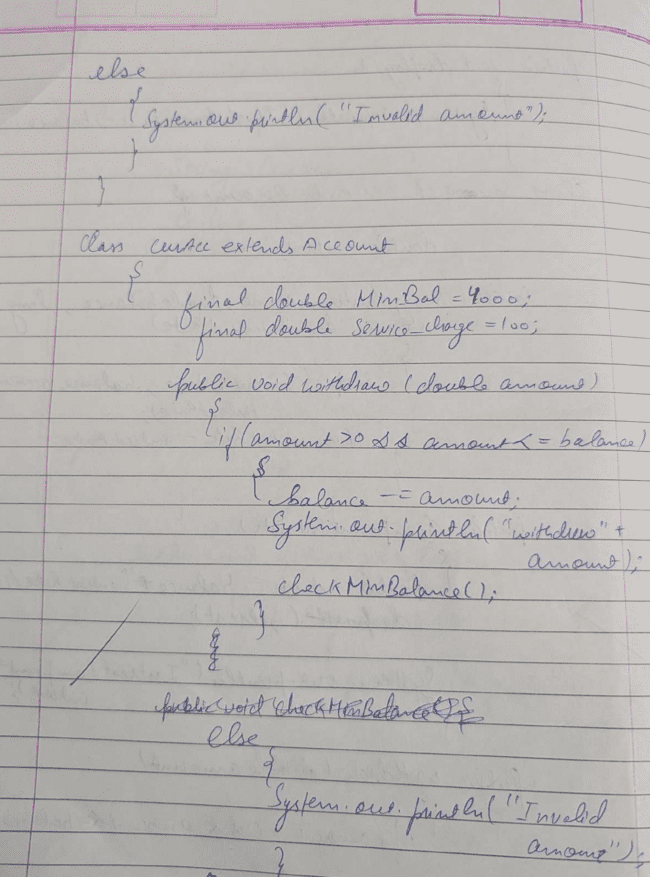
21



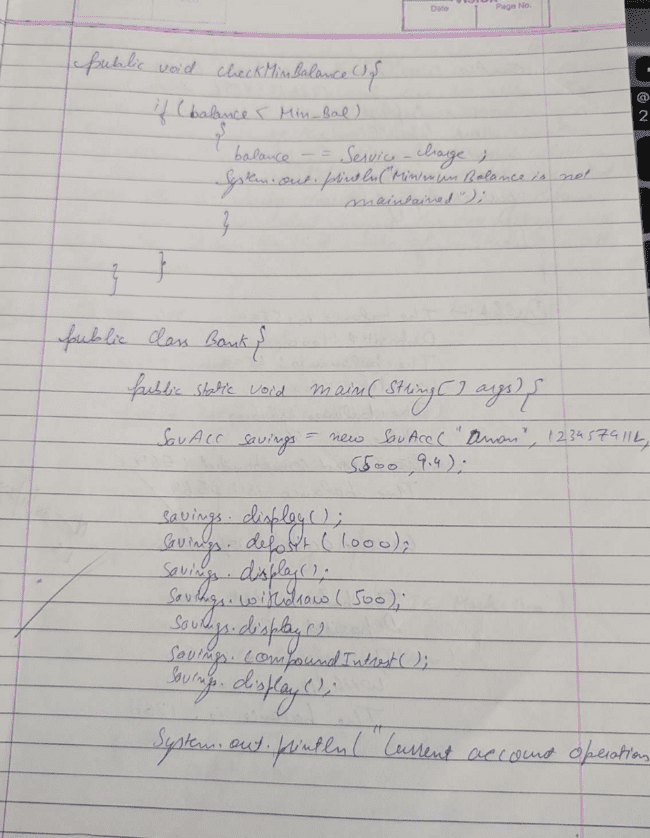
22

****

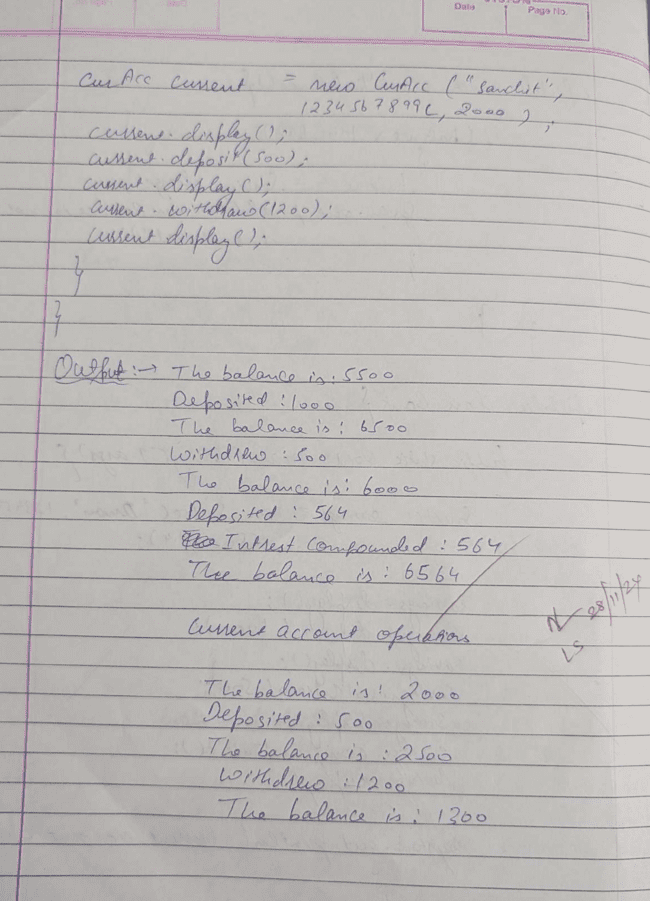
23

****

24

****

25

****

26

Code:

import java.util.Scanner;

class Account {

String customerName;

long accountNum;

double balance;

public Account(String customerName, long accountNum, double balance) {

this.customerName = customerName;

this.accountNum = accountNum;

this.balance = balance;

}

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

System.out.println("Deposited: " + amount);

} else {

System.out.println("Invalid amount");

}

}

public void withdraw(double amount) {

if (amount > 0 && amount <= balance) {

balance -= amount;

System.out.println("Withdrew: " + amount);

} else {

System.out.println("Invalid request or insufficient funds");

}

}

public void display() {

System.out.println("The balance is: " + balance);

}

}

class SavAcc extends Account {

double interestRate;

public SavAcc(String customerName, long accountNum, double balance, double interestRate) {

super(customerName, accountNum, balance);

this.interestRate = interestRate;

}

public void compoundInterest() {

double interest = balance \* (interestRate / 100);

deposit(interest);

System.out.println("Interest compounded: " + interest);

27

}

}

class CurAcc extends Account {

static final double minbal = 1000;

static final double servicecharge = 100;

public CurAcc(String customerName, long accountNum, double balance) {

super(customerName, accountNum, balance);

}

public void withdraw(double amount) {

if (amount > 0 && amount <= balance) {

balance -= amount;

System.out.println("Withdrew: " + amount);

checkMinBalance();

} else {

System.out.println("Invalid request or insufficient balance");

}

}

public void checkMinBalance() {

if (balance < minbal) {

balance -= servicecharge;

System.out.println("Minimum balance not maintained. Service charge imposed: " + servicecharge);

}

}

}

public class lab5 {

public static void main(String[] args)

{

SavAcc savings = new SavAcc("aman", 123457911, 5500.0, 9.4);

savings.display();

savings.deposit(1000.0);

savings.display();

savings.withdraw(500.0);

savings.display();

savings.compoundInterest();

savings.display();

System.out.println("current account operations\n");

CurAcc current = new CurAcc("sanchit", 1234567899, 2000.0);

current.display();

current.deposit(500.0);

current.display();

28

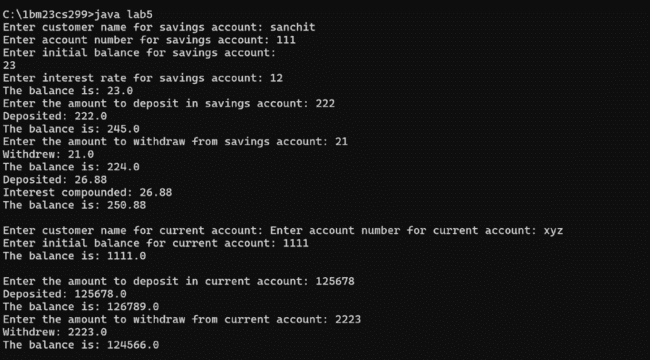
current.withdraw(1200.0);

current.display();

}

}

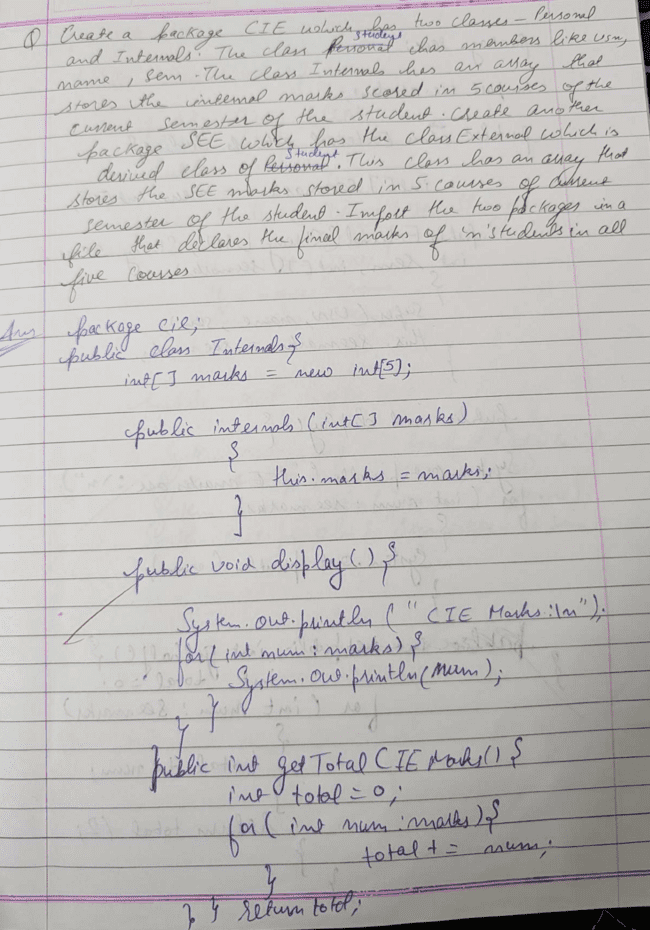
**OUTPUT**

****

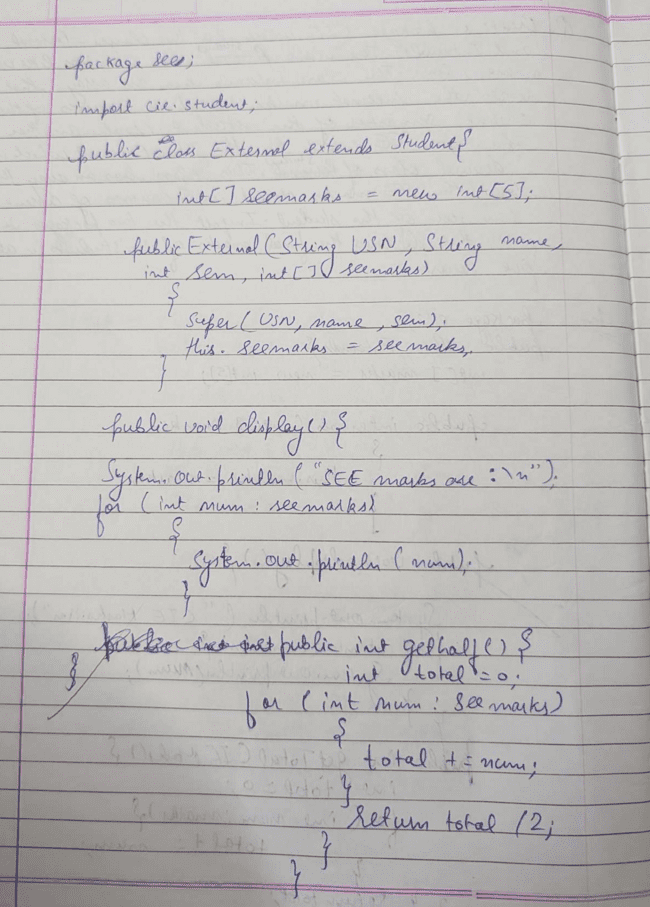
**LABORATORY PROGRAM – 6**

29

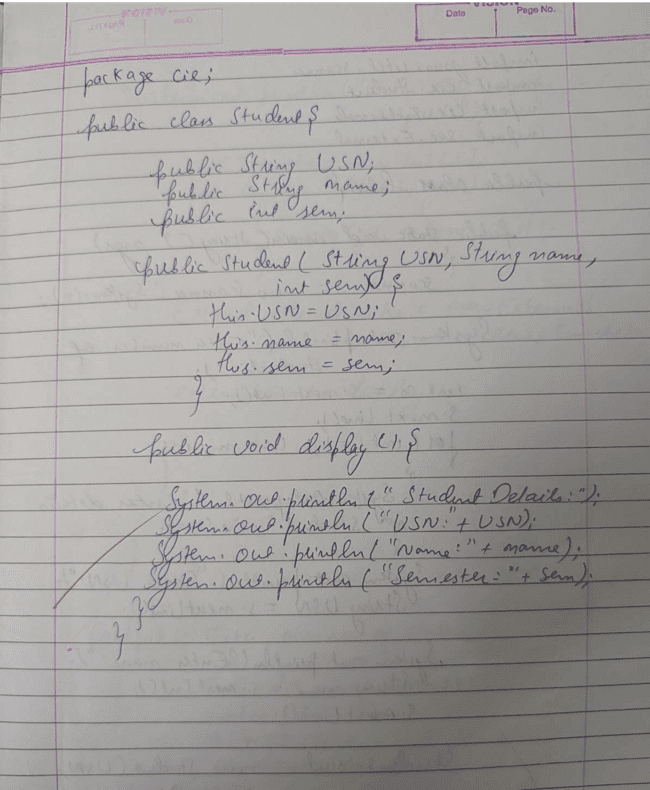
Create a package CIE which has two classes - Personal and Internals. The class Personal has members like usn, name, sem. The class Internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Personal. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.



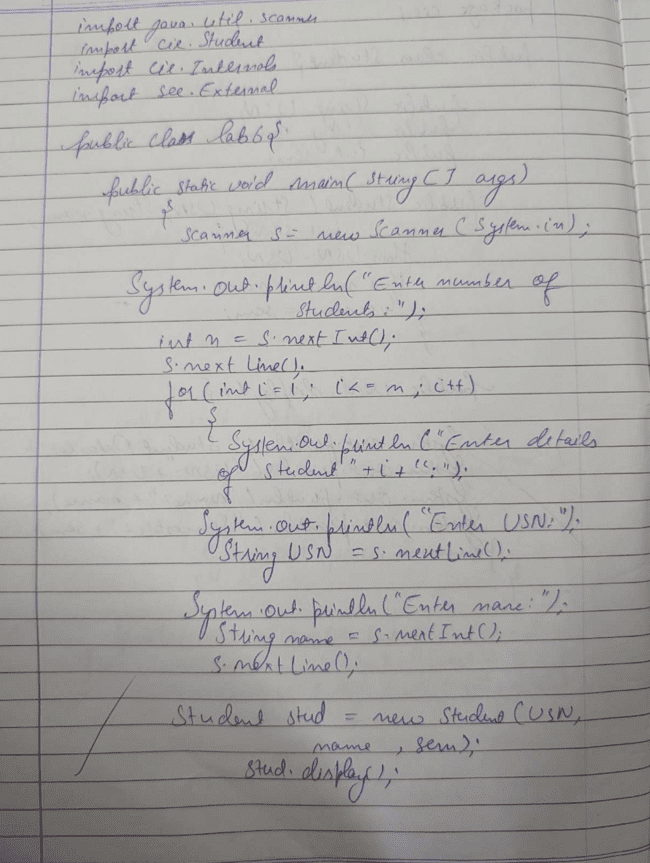
30

****

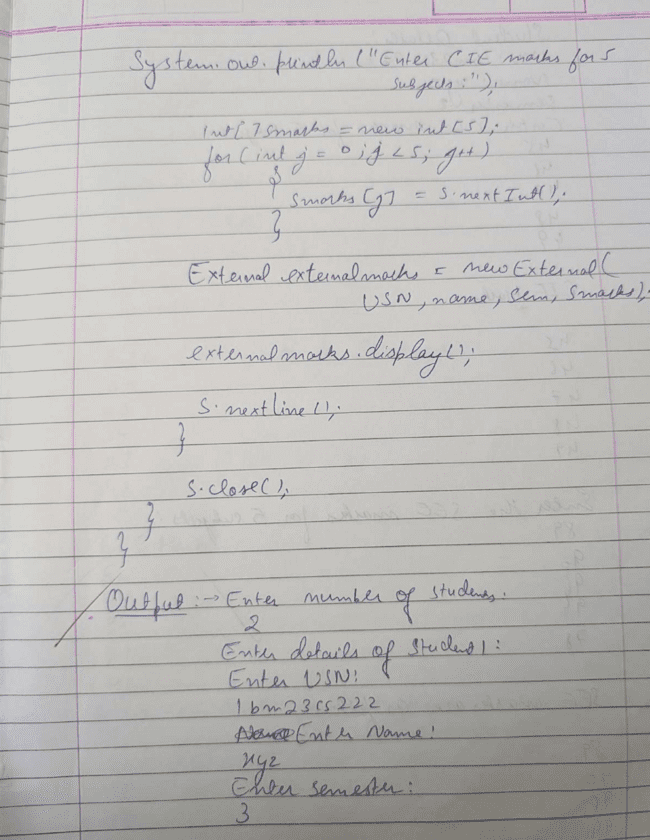
31

****

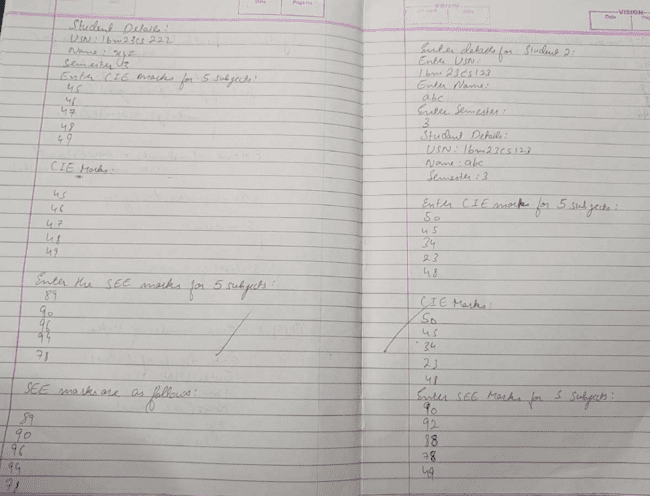
32

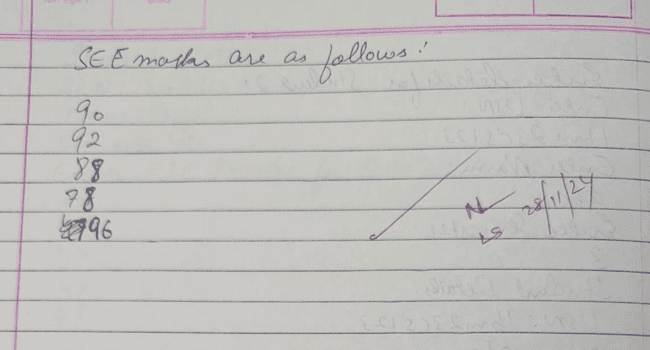
****

33

****

34

****

****

35

Code:

package cie;  
public class Internals {  
    int[] marks = new int[5];  
  
    public Internals(int[] marks) {  
        this.marks = marks;  
    }  
  
    public void display() {  
        System.out.println("CIE Marks: \n");  
        for (int num : marks) {  
            System.out.println(num);  
        }  
    }  
}  
  
  
package cie;  
  
public class Student {  
    public String USN;  
    public String name;  
    public  int sem;  
  
    public Student(String USN, String name, int sem) {  
        this.USN = USN;  
        [this.name](http://this.name/) = name;  
        this.sem = sem;  
    }  
  
    public void display() {  
        System.out.println("Student Details:");  
        System.out.println("USN: " + USN);  
        System.out.println("Name: " + name);  
        System.out.println("Semester: " + sem);  
    }  
}  
  
  
package see;  
import cie.Student;  
  
public class External extends Student {  
    int[] seemarks = new int[5];  
  
    public External(String USN, String name, int sem, int[] seemarks) {  
        super(USN, name, sem);  
        this.seemarks = seemarks;  
    }  
  
    public void display() {  
        System.out.println("SEE Marks are: \n");  
        for (int num : seemarks) {  
            System.out.println(num);  
        }  
    }  
}

import java.util.Scanner;

import cie.Student;

36

import cie.Internals;

import see.External;

public class lab6 {

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

System.out.println("Enter the number of students:");

int n = s.nextInt();

s.nextLine();

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

System.out.println("\nEnter details for Student " + i + ":");

System.out.println("Enter USN: ");

String USN = s.nextLine();

System.out.println("Enter Name: ");

String name = s.nextLine();

System.out.println("Enter Semester: ");

int sem = s.nextInt();

s.nextLine();

Student stud = new Student(USN, name, sem);

stud.display();

int[] marks = readMarks(s, "CIE");

Internals internalMarks = new Internals(marks);

internalMarks.display();

int[] smarks = readMarks(s, "SEE");

External externalMarks = new External(USN, name, sem, smarks);

externalMarks.display();

int[] finalMarks = new int[5];

System.out.println("Final Marks (CIE + 1/2 SEE) for each subject:");

for (int j = 0; j < 5; j++) {

finalMarks[j] = marks[j] + (smarks[j] / 2);

System.out.println("Subject " + (j + 1) + ": " + finalMarks[j]);

}

}

s.close();

}

public static int[] readMarks(Scanner s, String examType) {

System.out.println("Enter the " + examType + " marks (5 subjects): ");

37

int[] marks = new int[5];

for (int i = 0; i < 5; i++) {

marks[i] = s.nextInt();

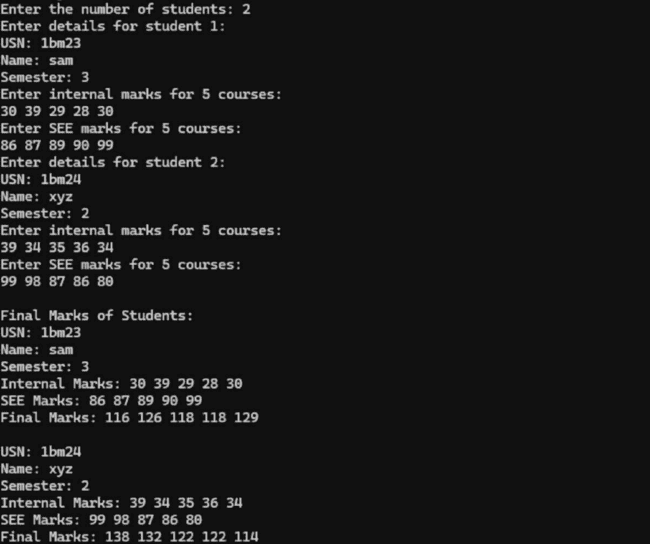
}

return marks;

}

}

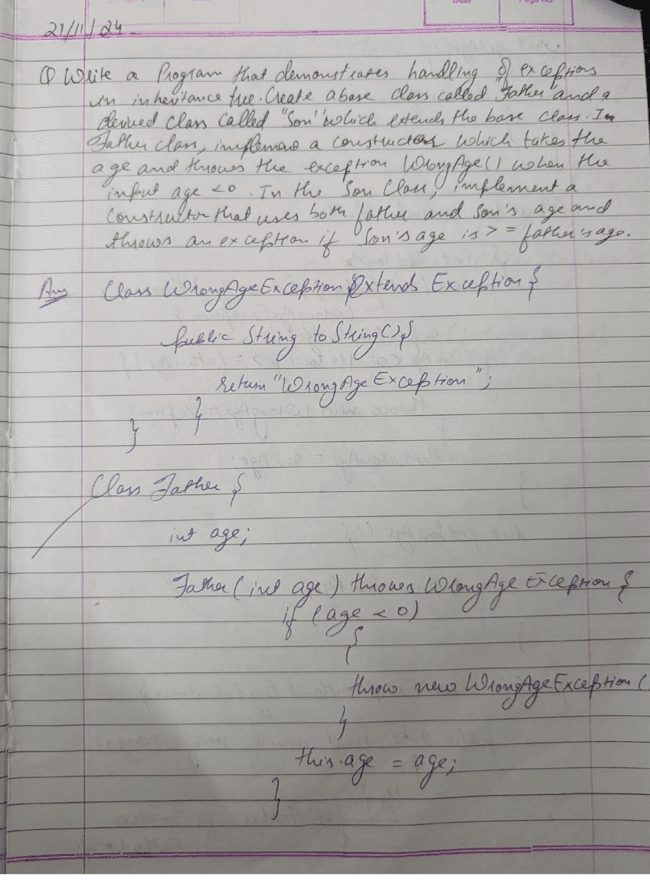
**OUTPUT**

****

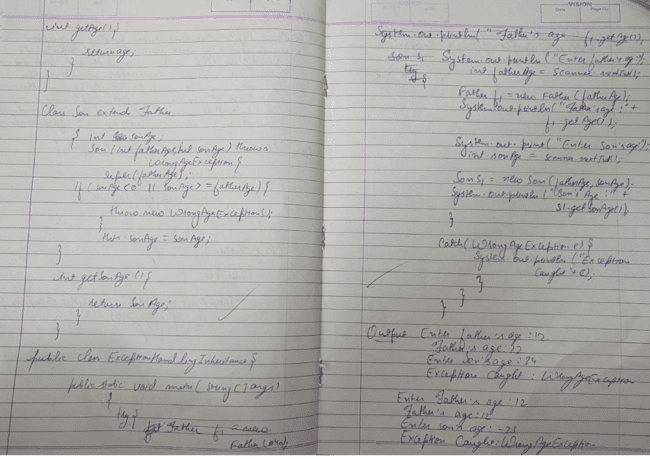
38

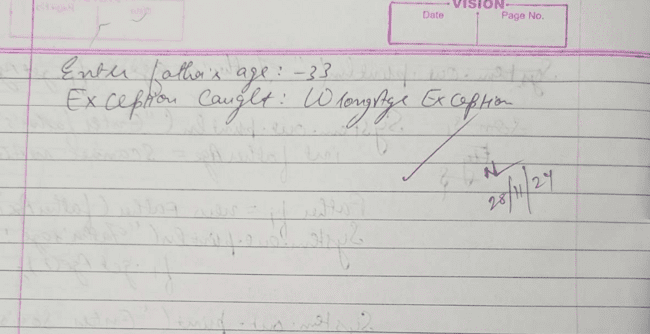
**LABORATORY PROGRAM – 7**

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age=father’s age



39

****

****

40

Code:

import java.util.Scanner;

class WrongAgeException extends Exception {

public String toString() {

return "WrongAgeException";

}

}

class Father {

int age;

Father(int age) throws WrongAgeException {

if (age < 0) {

throw new WrongAgeException();

}

this.age = age;

}

int getAge() {

return age;

}

}

class Son extends Father {

int sonAge;

Son(int fatherAge, int sonAge) throws WrongAgeException {

super(fatherAge);

if (sonAge < 0 || sonAge >= fatherAge) {

throw new WrongAgeException();

}

this.sonAge = sonAge;

}

int getSonAge() {

return sonAge;

}

}

public class lab7 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

try {

System.out.print("Enter Father's age: ");

int fatherAge = scanner.nextInt();

41

Father f1 = new Father(fatherAge);

System.out.println("Father's age: " + f1.getAge());

System.out.print("Enter Son's age: ");

int sonAge = scanner.nextInt();

Son s1 = new Son(fatherAge, sonAge);

System.out.println("Son's age: " + s1.getSonAge());

} catch (WrongAgeException e) {

System.out.println("Exception caught: " + e);

} finally {

scanner.close();

}

}

}

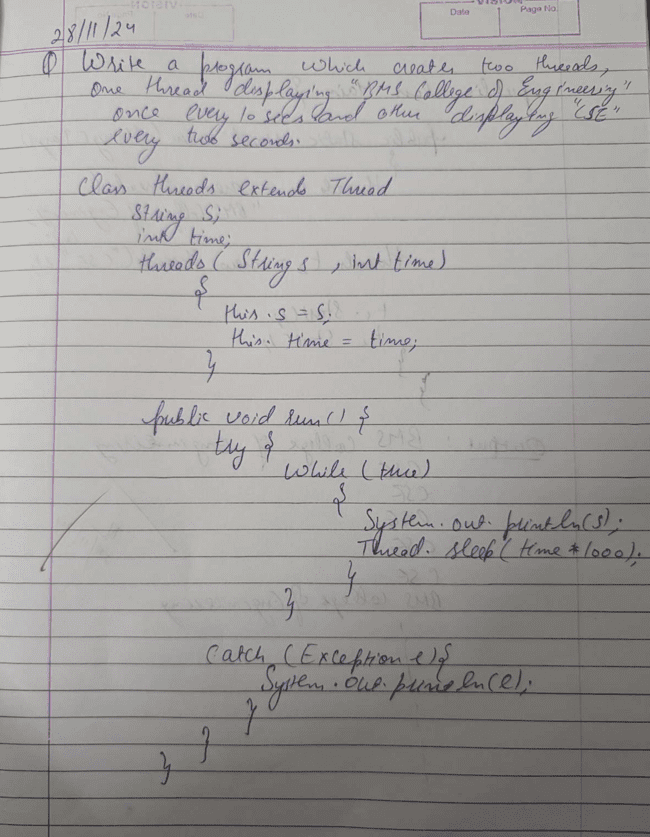
**OUTPUT**

****

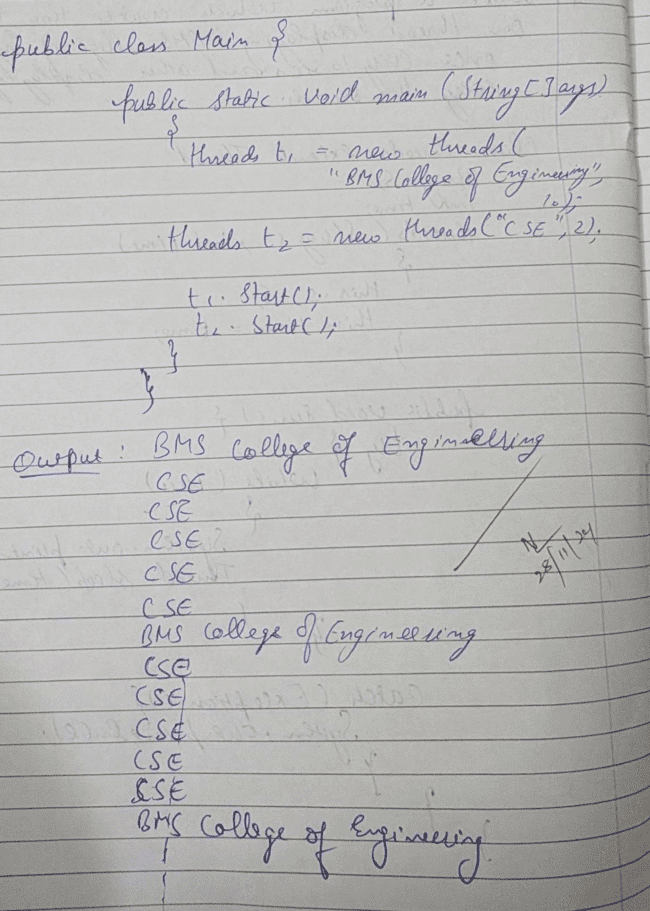
42

**LABORATORY PROGRAM – 8**

Write a program which creates two threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds

****

43

****

44

class threads extends Thread{

String s; int time;

threads(String s, int time){

this.s = s;

this.time = time;

}

public void run(){

try{

while(true){

System.out.println(s);

Thread.sleep(time \* 1000);

}

}

catch(Exception e){

System.out.println(e);

}

}

}

public class lab8{

public static void main(String[] args) {

threads t1 = new threads("BMS College of Engineering", 10);

threads t2 = new threads("CSE", 2);

t1.start();

t2.start();

}

}

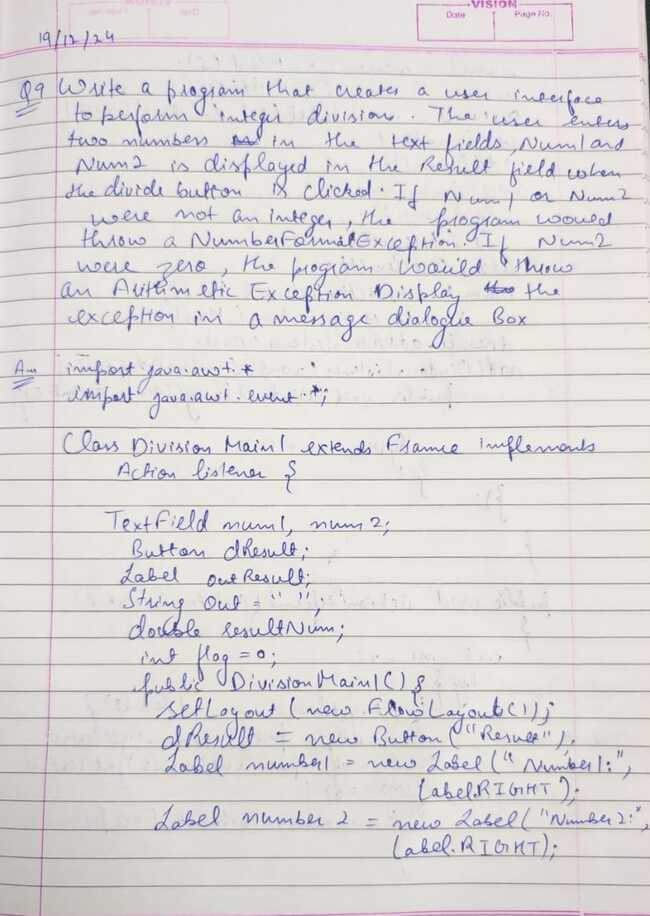
**OUTPUT**

****

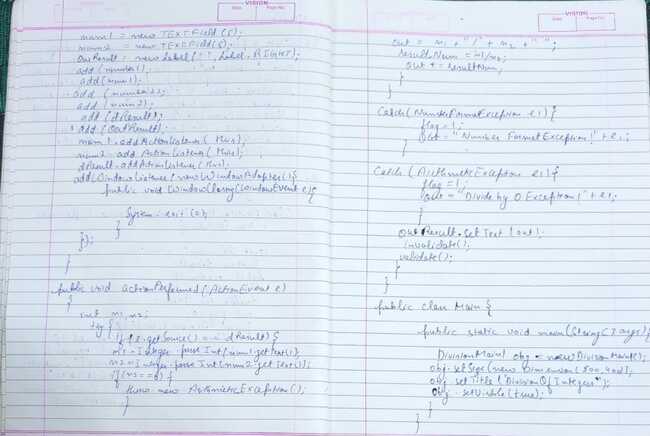
45

**LABORATORY PROGRAM - 9**

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.



46



import java.awt.\*;

import java.awt.event.\*;

class DivisionMain1 extends Frame implements ActionListener

{

TextField num1,num2;

Button dResult;

Label outResult;

String out="";

double resultNum;

int flag=0;

public DivisionMain1()

{

setLayout(new FlowLayout());

dResult = new Button("Result:");

Label number1 = new Label("Number 1:",Label.RIGHT);

Label number2 = new Label("Number 2:",Label.RIGHT);

num1=new TextField(5);

num2=new TextField(5);

outResult = new Label("",Label.RIGHT);

add(number1);

add(num1);

add(number2);

add(num2);

add(dResult);

add(outResult);

num1.addActionListener(this);

num2.addActionListener(this);

dResult.addActionListener(this);

addWindowListener(new WindowAdapter(){

public void windowClosing(WindowEvent e)

{

47

System.exit(0);

}

});

}

public void actionPerformed(ActionEvent e)

{

int n1,n2;

try

{

if (e.getSource() == dResult)

{

n1=Integer.parseInt(num1.getText());

n2=Integer.parseInt(num2.getText());

if(n2==0)

{throw new ArithmeticException();}

out=n1+"/"+n2+" ";

resultNum=n1/n2;

out+=resultNum;

}

}

catch(NumberFormatException e1)

{

flag=1;

out="Number Format Exception!"+e1;

}

catch(ArithmeticException e1)

{

flag=1;

out="Divide by 0 Exception!"+e1;

}

outResult.setText(out);

invalidate();

validate();

}

}

public class Main

{

public static void main(String args[])

{

DivisionMain1 obj=new DivisionMain1();

obj.setSize(new Dimension(800,400));

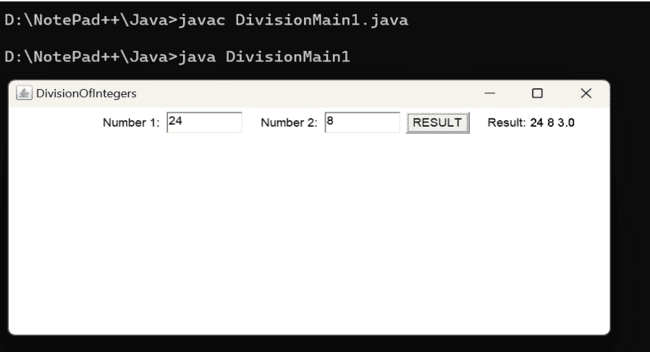
obj.setTitle("DivisionOfIntegers");

obj.setVisible(true);

}

}

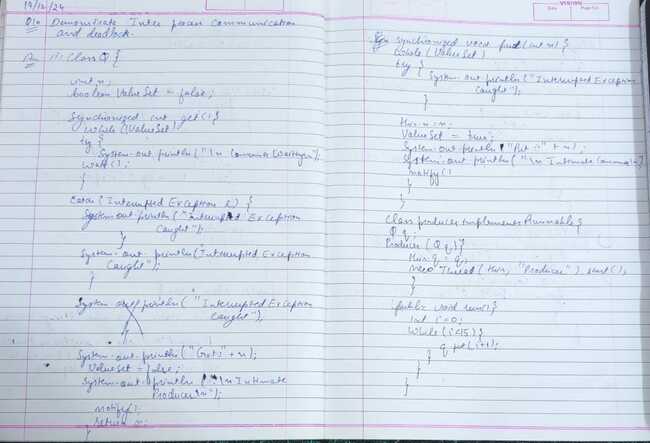
OUTPUT

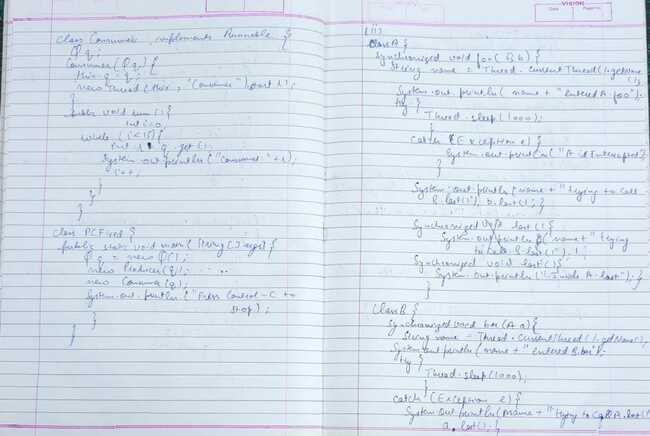


48

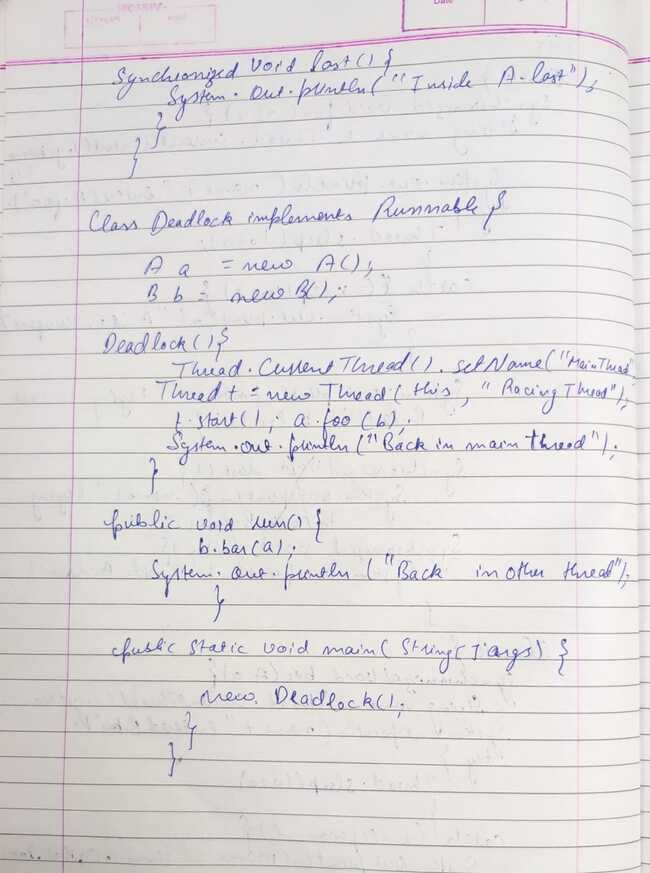
**LABORATORY PROGRAM – 10**

Demonstrate Inter process Communication and deadlock





49



50

Code(i):

class Q {

int n;

boolean valueSet = false;

synchronized int get() {

while(!valueSet)

try {

System.out.println("\nConsumer waiting\n");

wait();

} catch(InterruptedException e) {

System.out.println("InterruptedException caught");

}

System.out.println("Got: " + n);

valueSet = false;

System.out.println("\nIntimate Producer\n");

notify();

return n;

}

synchronized void put(int n) {

while(valueSet)

try {

System.out.println("\nProducer waiting\n");

wait();

} catch(InterruptedException e) {

System.out.println("InterruptedException caught");

}

this.n = n;

valueSet = true;

System.out.println("Put: " + n);

System.out.println("\nIntimate Consumer\n");

notify();

}

}

class Producer implements Runnable {

Q q;

Producer(Q q) {

this.q = q;

new Thread(this, "Producer").start();

}

public void run() {

int i = 0;

while(i<15) {

q.put(i++);

}

}

}

class Consumer implements Runnable {

Q q;

Consumer(Q q) {

this.q = q;

new Thread(this, "Consumer").start();

}

public void run() {

int i=0;

while(i<15) {

int r=q.get();

System.out.println("consumed:"+r);

i++;

}

}

}

class PCFixed {

51

public static void main(String args[]) {

Q q = new Q();

new Producer(q);

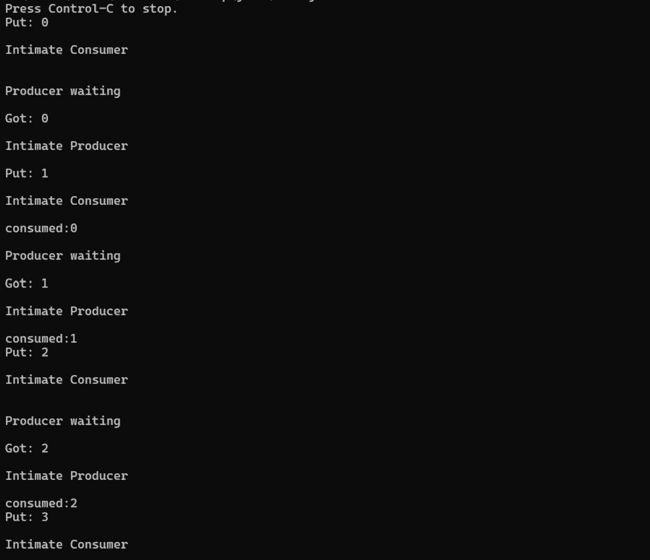
new Consumer(q);

System.out.println("Press Control-C to stop.");

}

}

OUTPUT



Code(ii):

class A

{

synchronized void foo(B b)

{ String name = Thread.currentThread().getName();

System.out.println(name + " entered A.foo");

try { Thread.sleep(1000); }

catch(Exception e) { System.out.println("A Interrupted"); }

System.out.println(name + " trying to call B.last()"); b.last(); }

synchronized void last() { System.out.println("Inside A.last"); }

}

class B {

synchronized void bar(A a) {

String name = Thread.currentThread().getName();

System.out.println(name + " entered B.bar");

try { Thread.sleep(1000); }

catch(Exception e) { System.out.println("B Interrupted"); }

System.out.println(name + " trying to call A.last()"); a.last(); }

synchronized void last() { System.out.println("Inside A.last"); }

52

}

class Deadlock implements Runnable

{

A a = new A(); B b = new B();

Deadlock( ) {

Thread.currentThread().setName("MainThread");

Thread t = new Thread(this, "RacingThread");

t.start(); a.foo(b); // get lock on a in this thread.

System.out.println("Back in main thread");

}

public void run() { b.bar(a); // get lock on b in other thread.

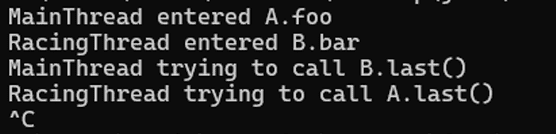
System.out.println("Back in other thread");

}

public static void main(String args[]) { new Deadlock(); }

}

OUTPUT



53