**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

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

****

**LAB REPORT**

**on**

**Object Oriented Java Programming (23CS3PCOOJ)**

***Submitted by***

**Dhanush K (24BECS405)**

***in partial fulfilment 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 to 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 (23CS3PCOOJ)” carried out by **Dhanush K (24BECS405),** who is Bonafide student of **B.M.S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

|  |  |
| --- | --- |
| Lab faculty Incharge Name  Assistant Professor  Department of CSE, BMSCE | Prof. Ambuja  Department of CSE, BMSCE |

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 |  | Quadratic Equation Program | 4-5 |
| 2 |  | Calculate SGPA Of A Student. Program | 5-8 |
| 3 |  | Book Program | 8-11 |
| 4 |  | Abstract Program | 11-14 |
| 5 |  | Bank Program | 14-18 |
| 6 |  | Package CIE Program | 18-22 |
| 7 |  | Interface 5 program | 22-26 |
| 8 |  | Exceptions Program | 26-28 |
| 9 |  | Threads Program | 28-30 |
| 10 |  |  |  |

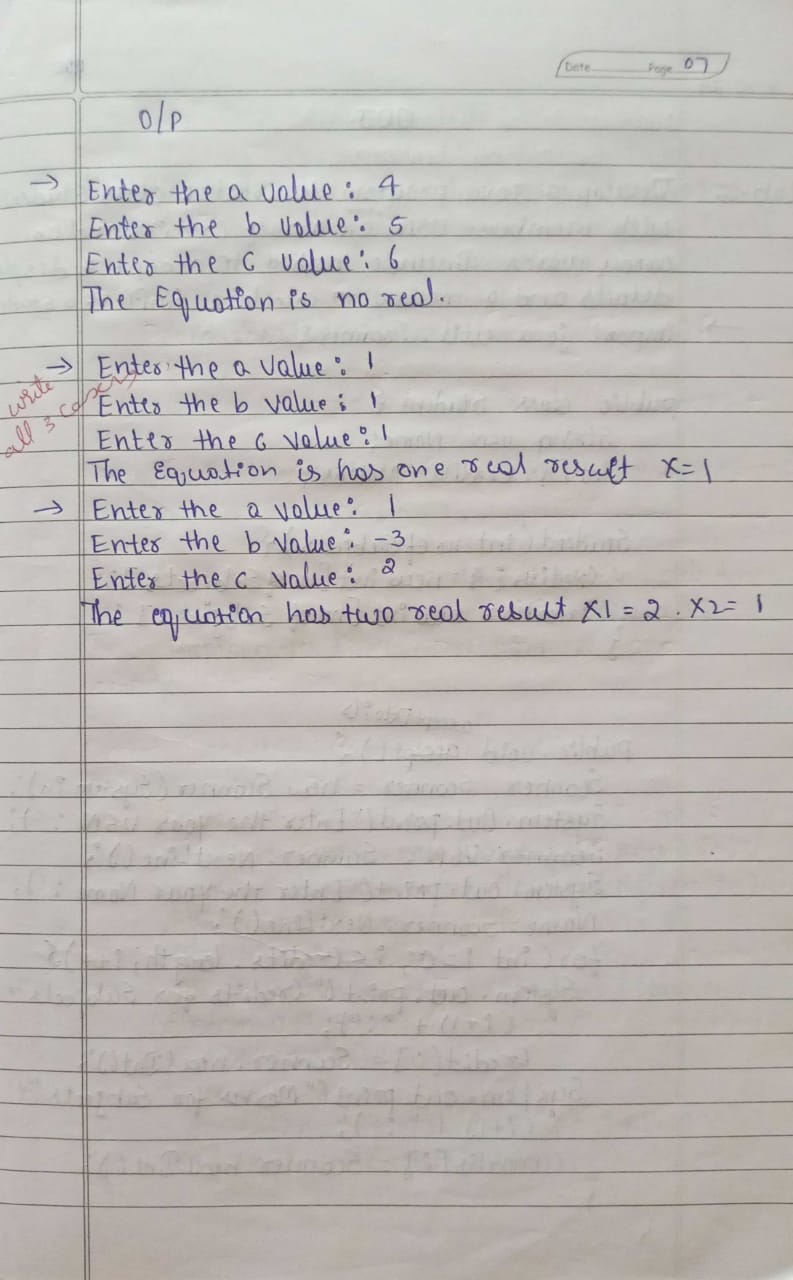
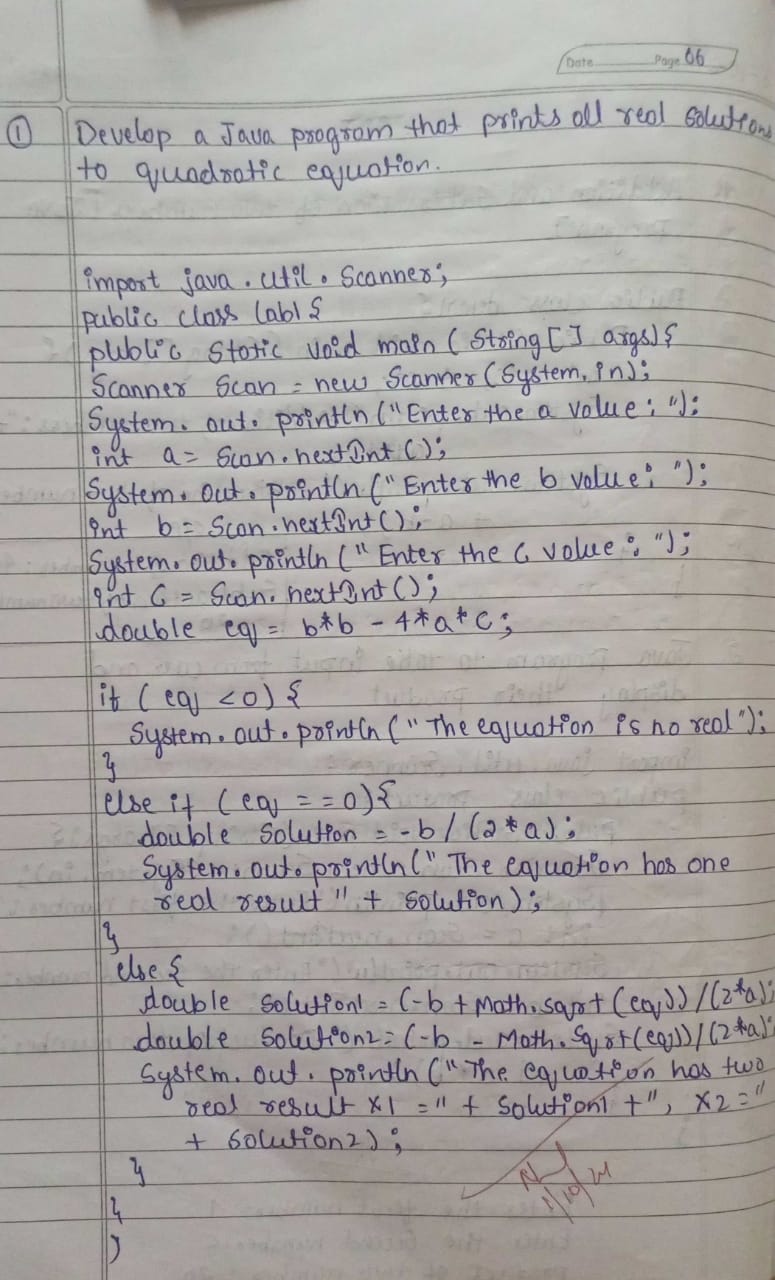
**GitHub Link:**

<https://github.com/dhanushk240206/OOJ-lab>

**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.

**Algorithm:**



**Code:**

import java.util.Scanner;

public class QuadraticEquation {

public static void main(String[] args) {

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();

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

if (discriminant > 0) {

double root1 = (-b + Math.sqrt(discriminant)) / (2 \* a);

double root2 = (-b - Math.sqrt(discriminant)) / (2 \* a);

System.out.println("The equation has two real solutions: " + root1 + " and " + root2);

} else if (discriminant == 0) {

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

System.out.println("The equation has one real solution: " + root);

} else {

System.out.println("The equation has no real solutions.");

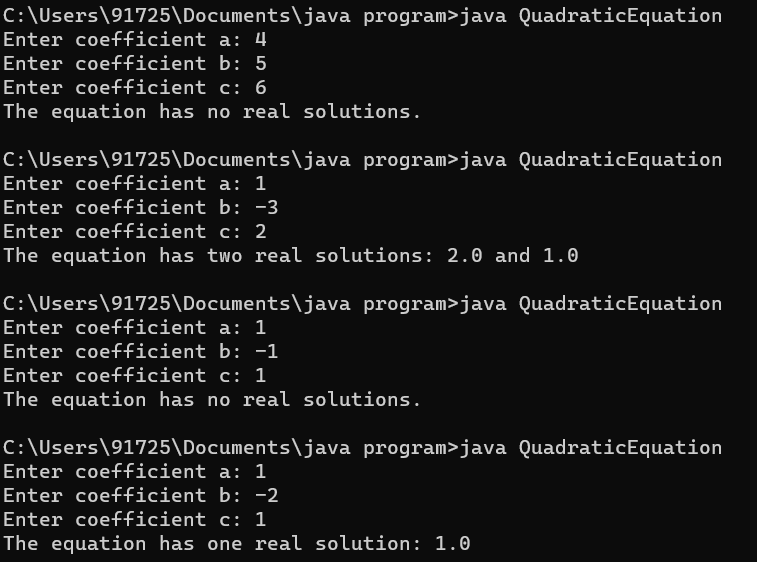
}

scanner.close();

}

}

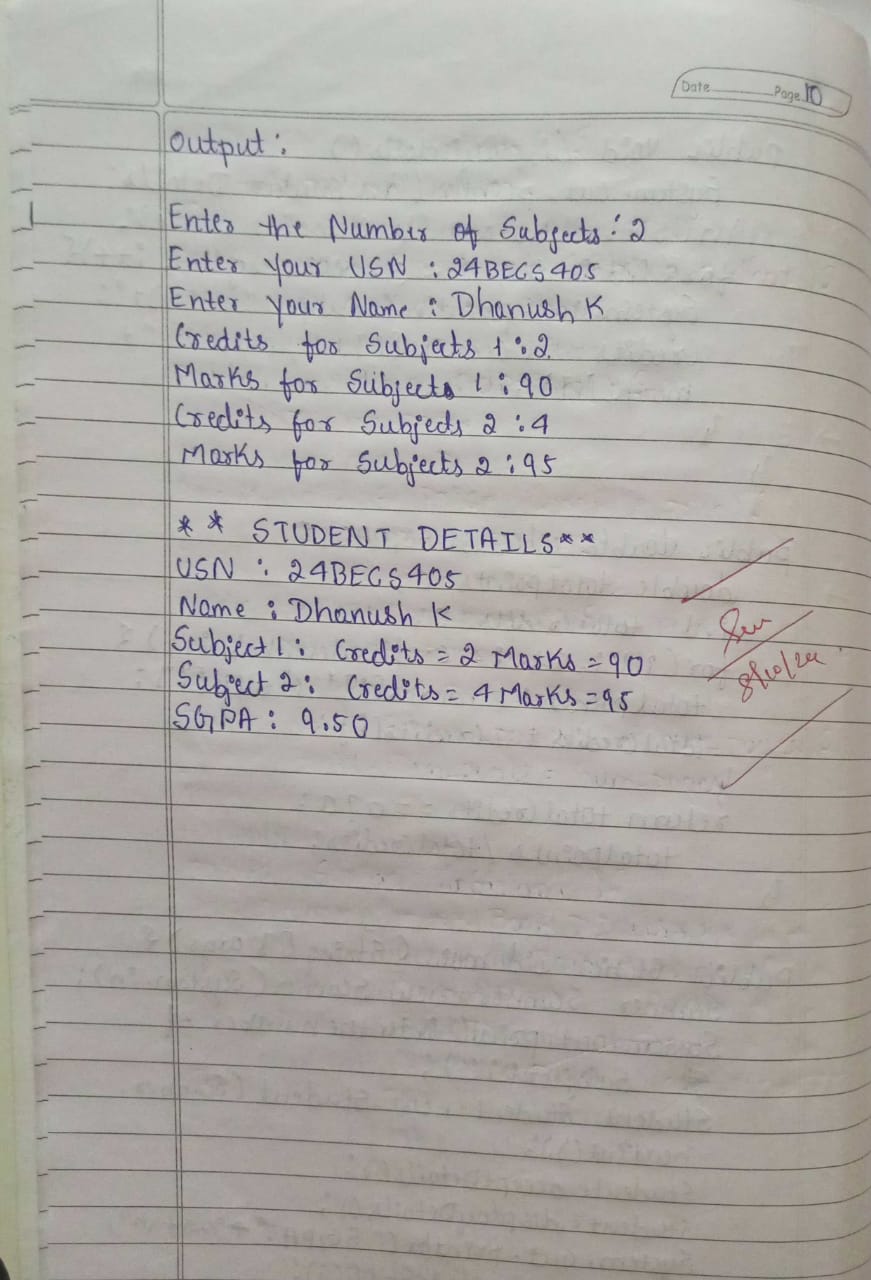
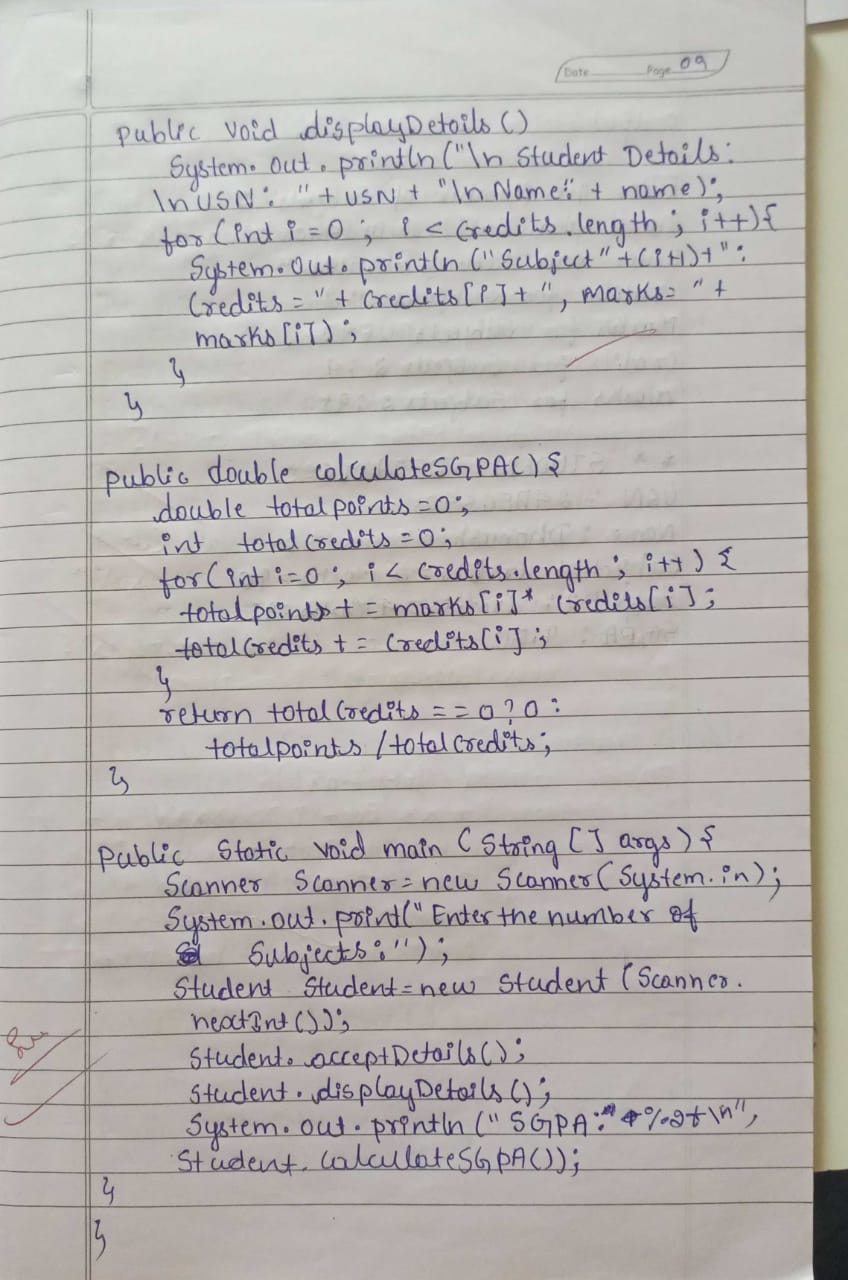
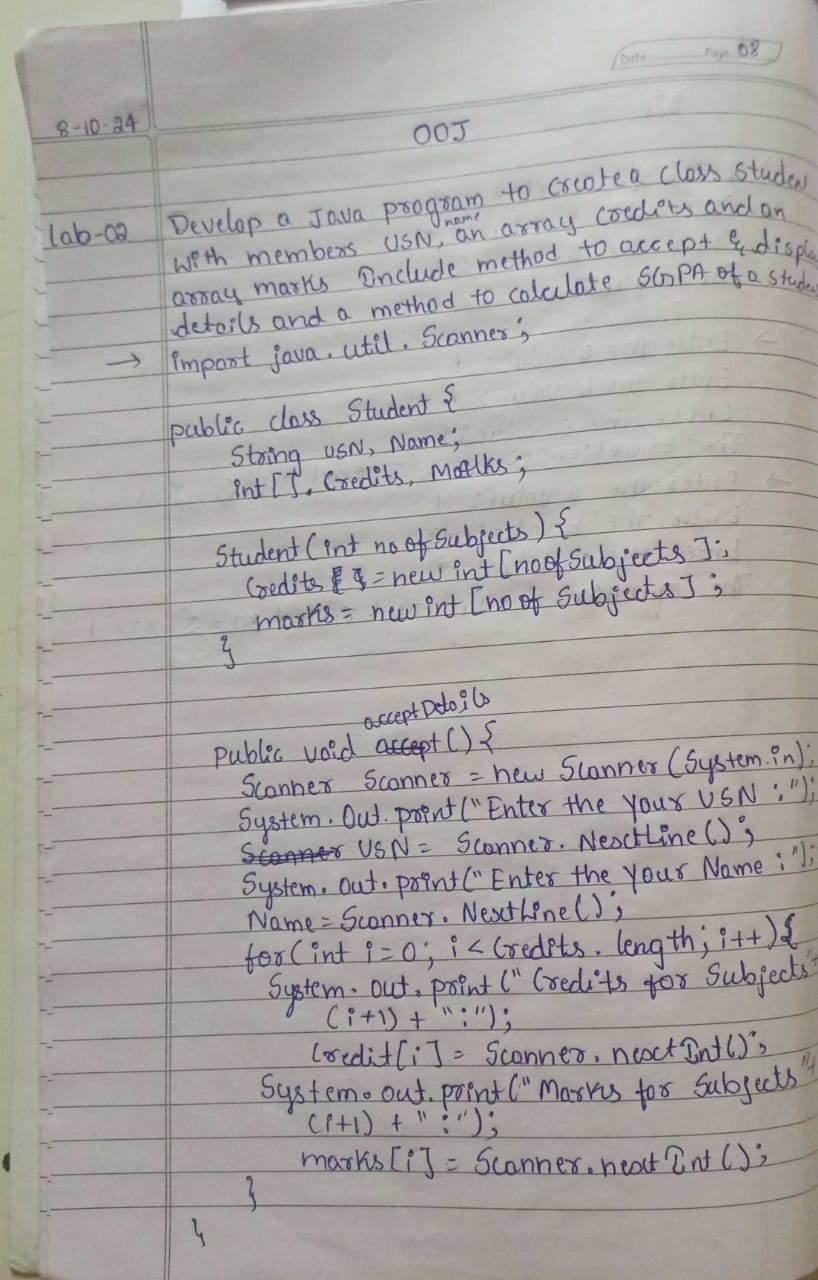
**Output**



**Program 2**

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

**Algorithm:**



**Code:**

import java.util.Scanner;

class Student {

String usn;

String name;

int[] credits;

int[] marks;

int numSubjects;

public void acceptDetails() {

Scanner sc = new Scanner(System.in);

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

usn = sc.nextLine();

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

name = sc.nextLine();

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

numSubjects = sc.nextInt();

credits = new int[numSubjects];

marks = new int[numSubjects];

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

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

credits[i] = sc.nextInt();

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

marks[i] = sc.nextInt();

}

}

public void displayDetails() {

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

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

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

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

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

}

}

public double calculateSGPA() {

int totalCredits = 0;

int totalPoints = 0;

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

totalCredits += credits[i];

totalPoints += credits[i] ;

}

return (double) totalPoints / totalCredits;

}

public static void main(String[] args) {

Student student = new Student();

student.acceptDetails();

student.displayDetails();

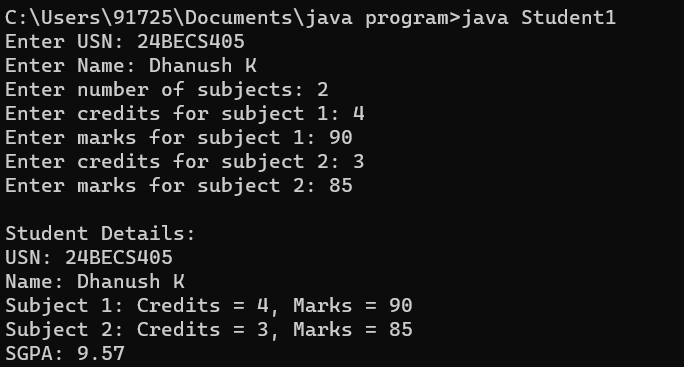
double sgpa = student.calculateSGPA();

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

}

}

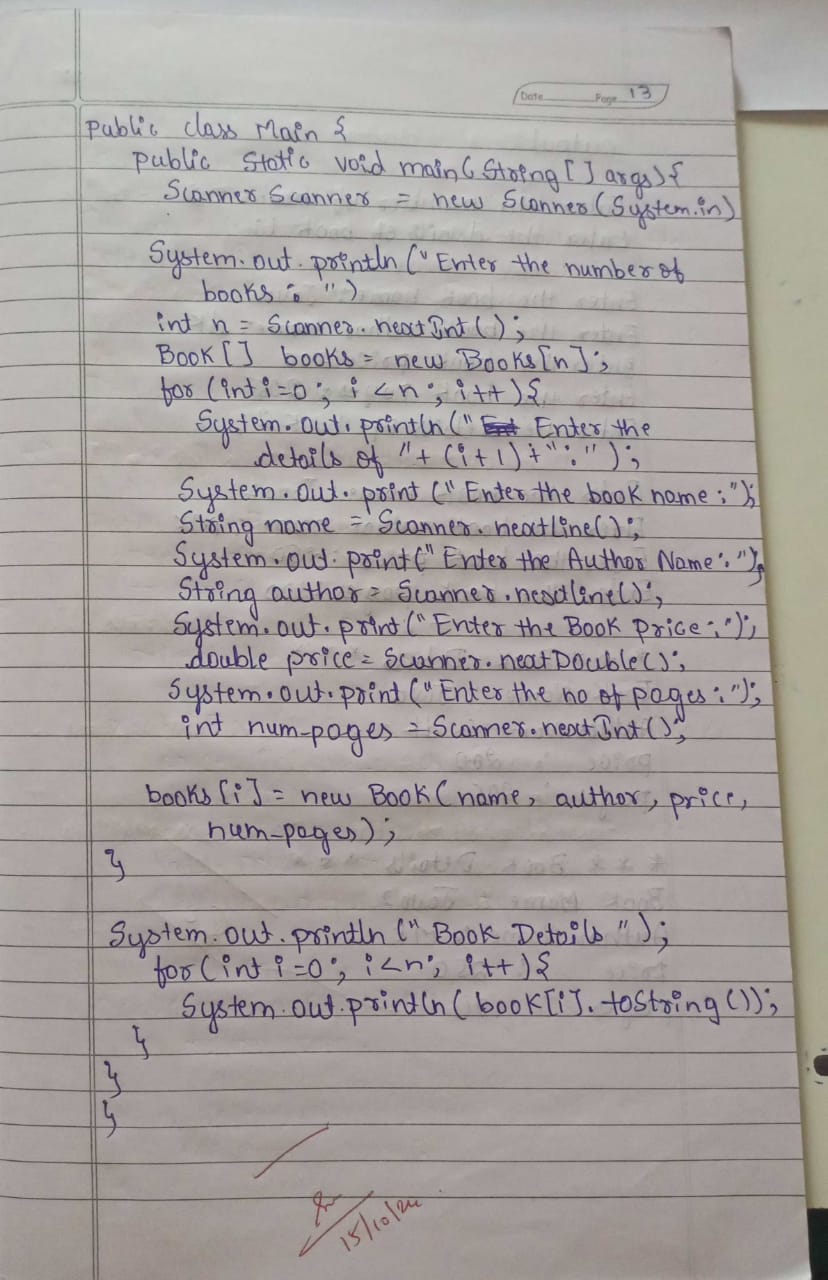
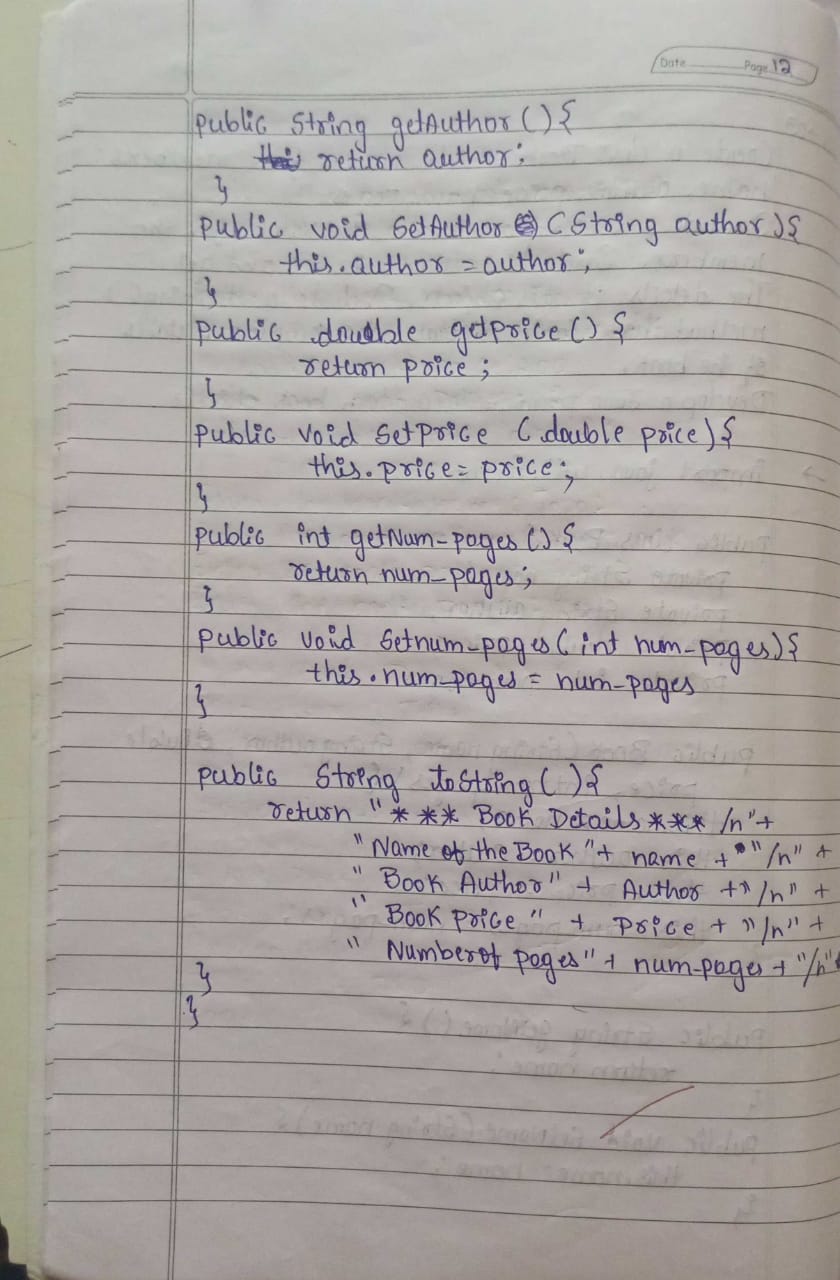
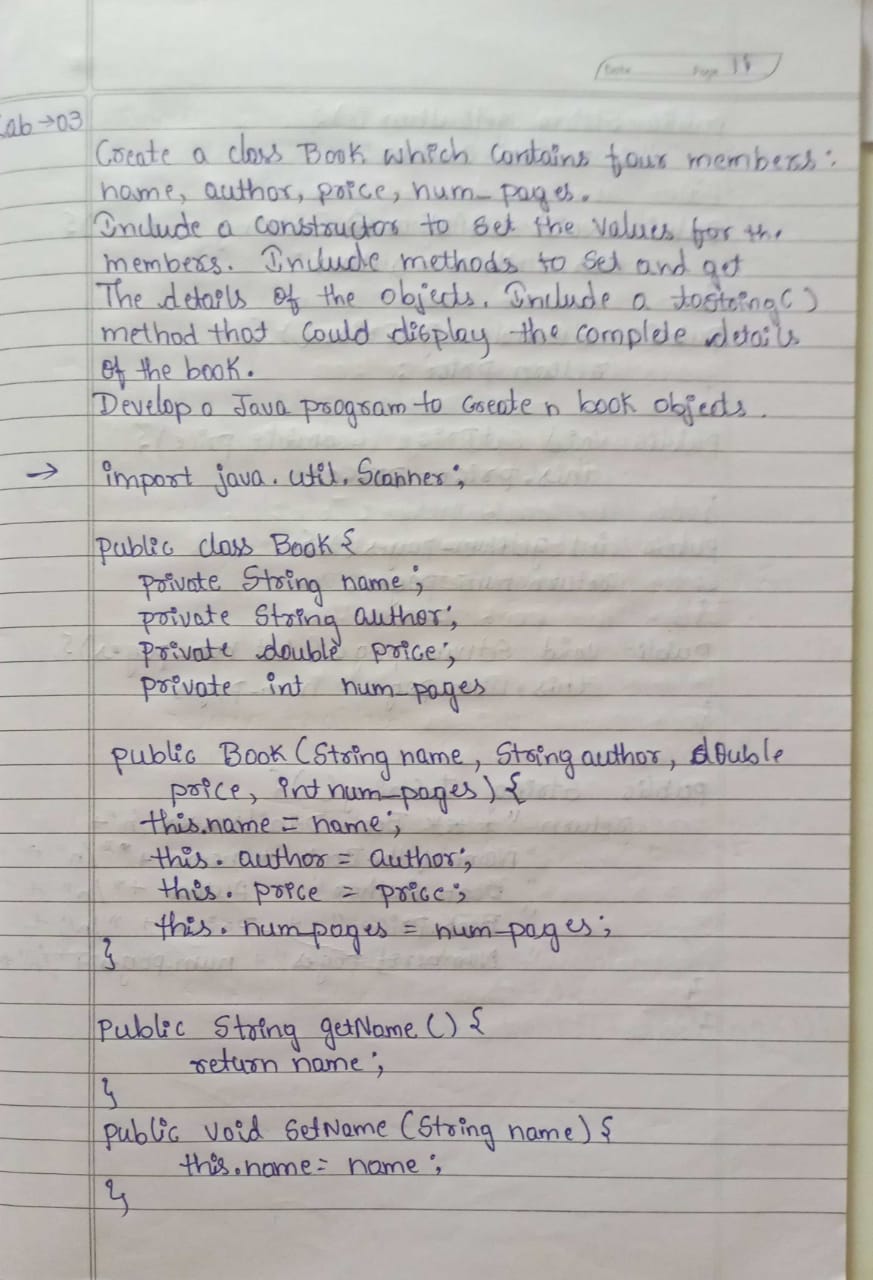
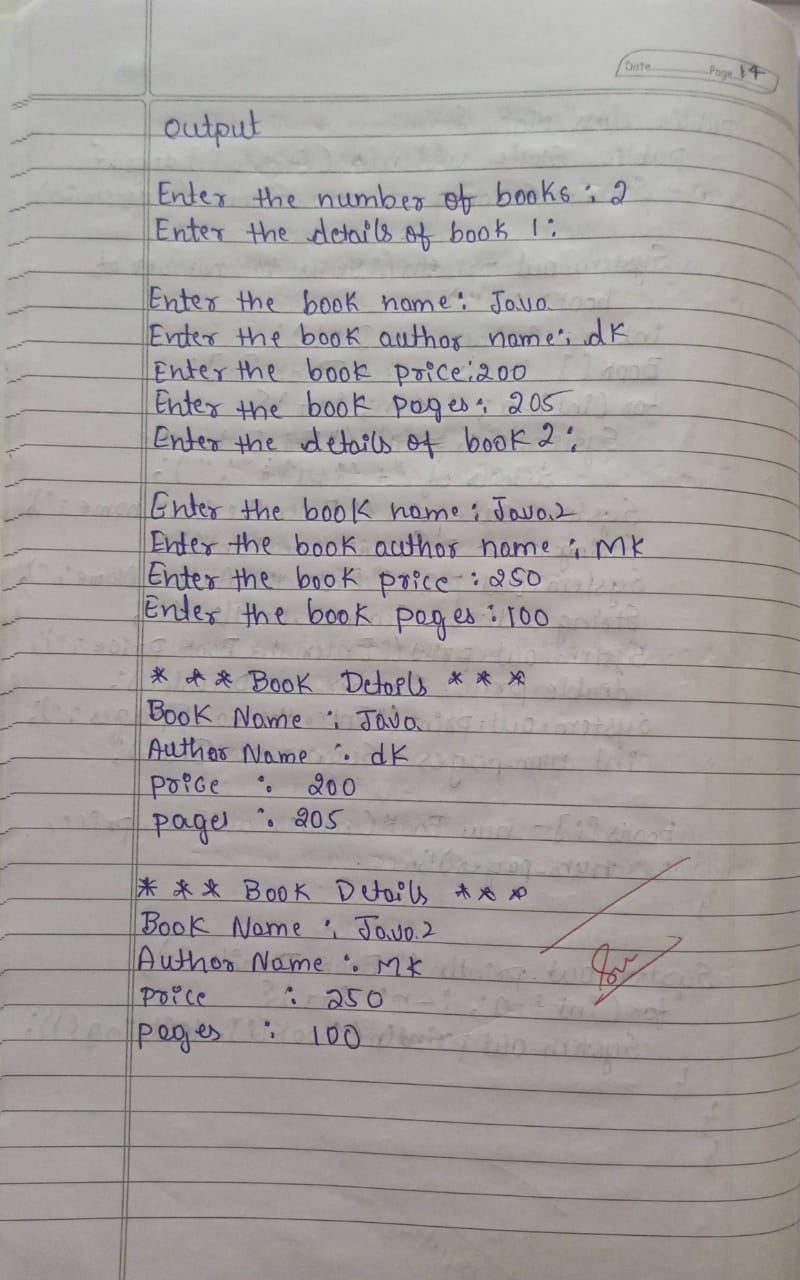
**Output**

****

**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

**Algorithm:**

**Code:**

import java.util.Scanner;

class Book {

private String name, author;

private double price;

private int num\_pages;

public Book(String name, String author, double price, int num\_pages) {

this.name = name;

this.author = author;

this.price = price;

this.num\_pages = num\_pages;

}

public String toString() {

return "Book Details: \n" +

"Name: " + name + "\n" +

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

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

"Pages: " + num\_pages + "\n";

}

}

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int n = scanner.nextInt();

scanner.nextLine();

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("Pages: ");

int num\_pages = scanner.nextInt();

scanner.nextLine();

books[i] = new Book(name, author, price, num\_pages);

}

System.out.println("\n--- Book Details ---");

for (Book book : books) {

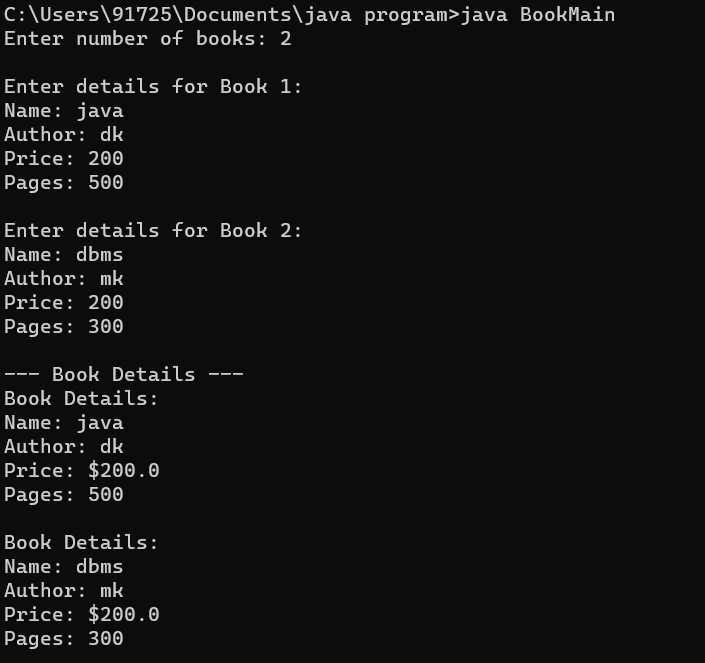
System.out.println(book);

}

scanner.close()

}

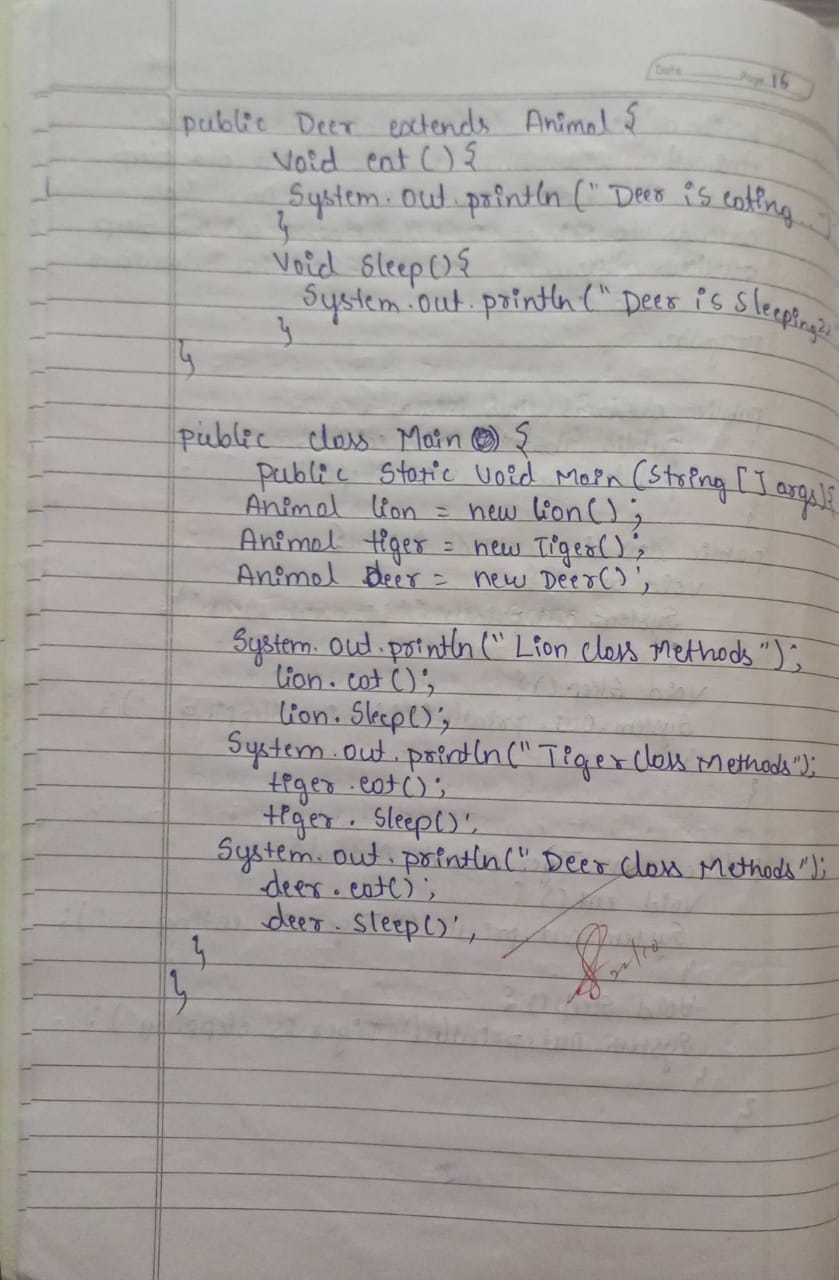
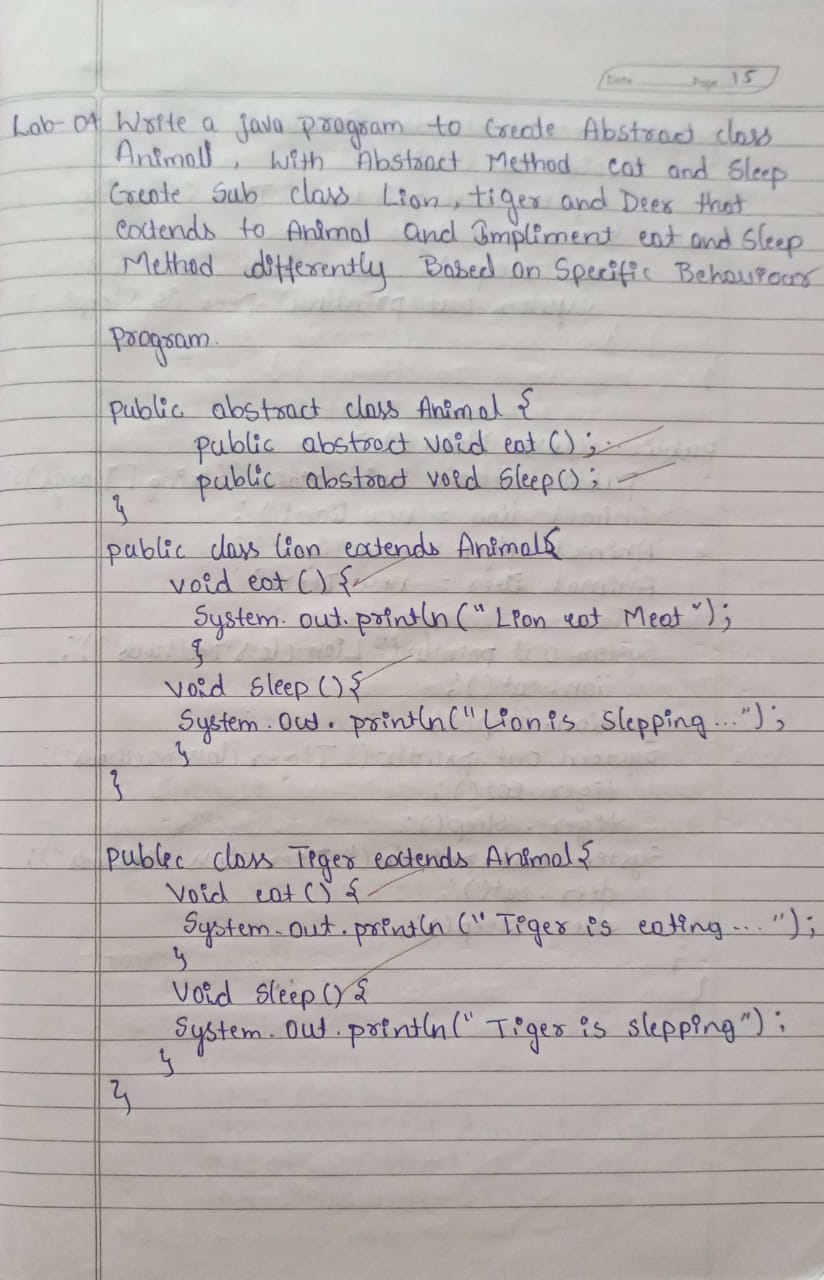
**Output:**

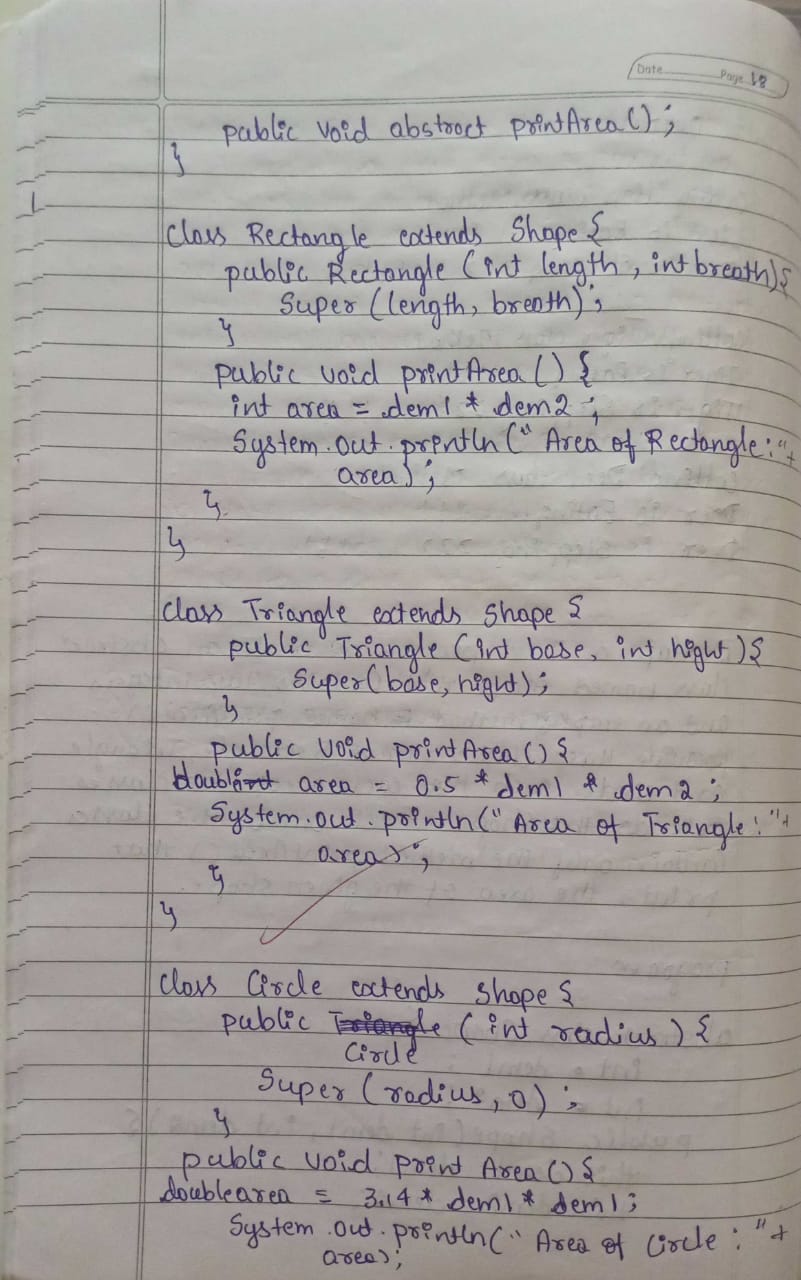
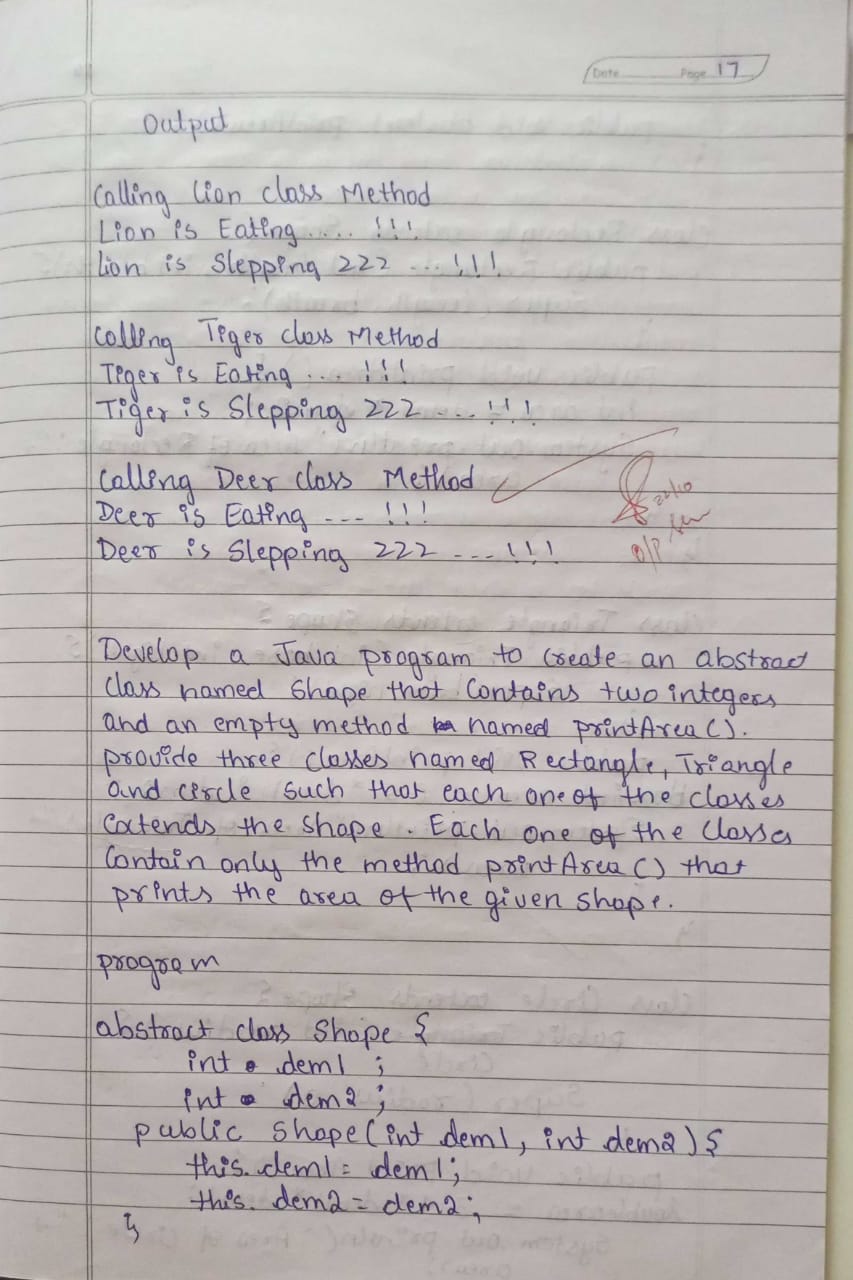
****

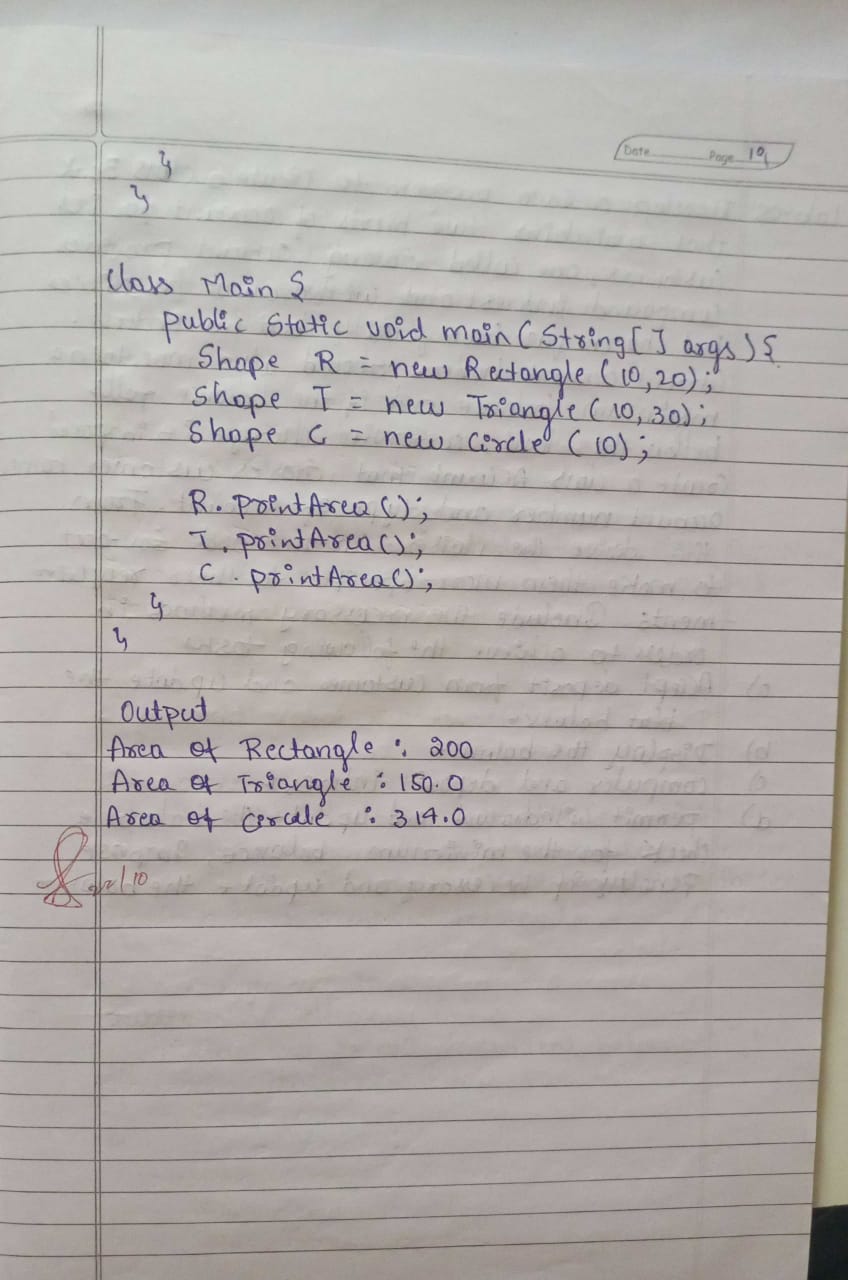
**Program 4**

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.

**Algorithm:**







**Code:**

abstract class Shape {

int dimension1;

int dimension2;

Shape(int d1, int d2) {

this.dimension1 = d1;

this.dimension2 = d2;

}

abstract void printArea();

}

class Rectangle extends Shape {

Rectangle(int length, int breadth) {

super(length, breadth);

}

void printArea() {

int area = dimension1 \* dimension2;

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

}

}

class Triangle extends Shape {

Triangle(int base, int height) {

super(base, height);

}

void printArea() {

double area = 0.5 \* dimension1 \* dimension2;

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

}

}

class Circle extends Shape {

Circle(int radius) {

super(radius, 0);

}

void printArea() {

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

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

}

}

public class Main {

public static void main(String[] args) {

Shape rectangle = new Rectangle(10, 5);

Shape triangle = new Triangle(8, 4);

Shape circle = new Circle(7);

rectangle.printArea();

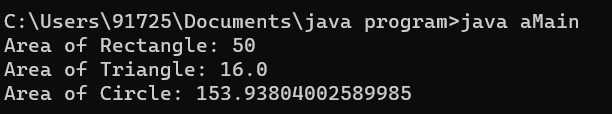
triangle.printArea();

circle.printArea();

    }

}

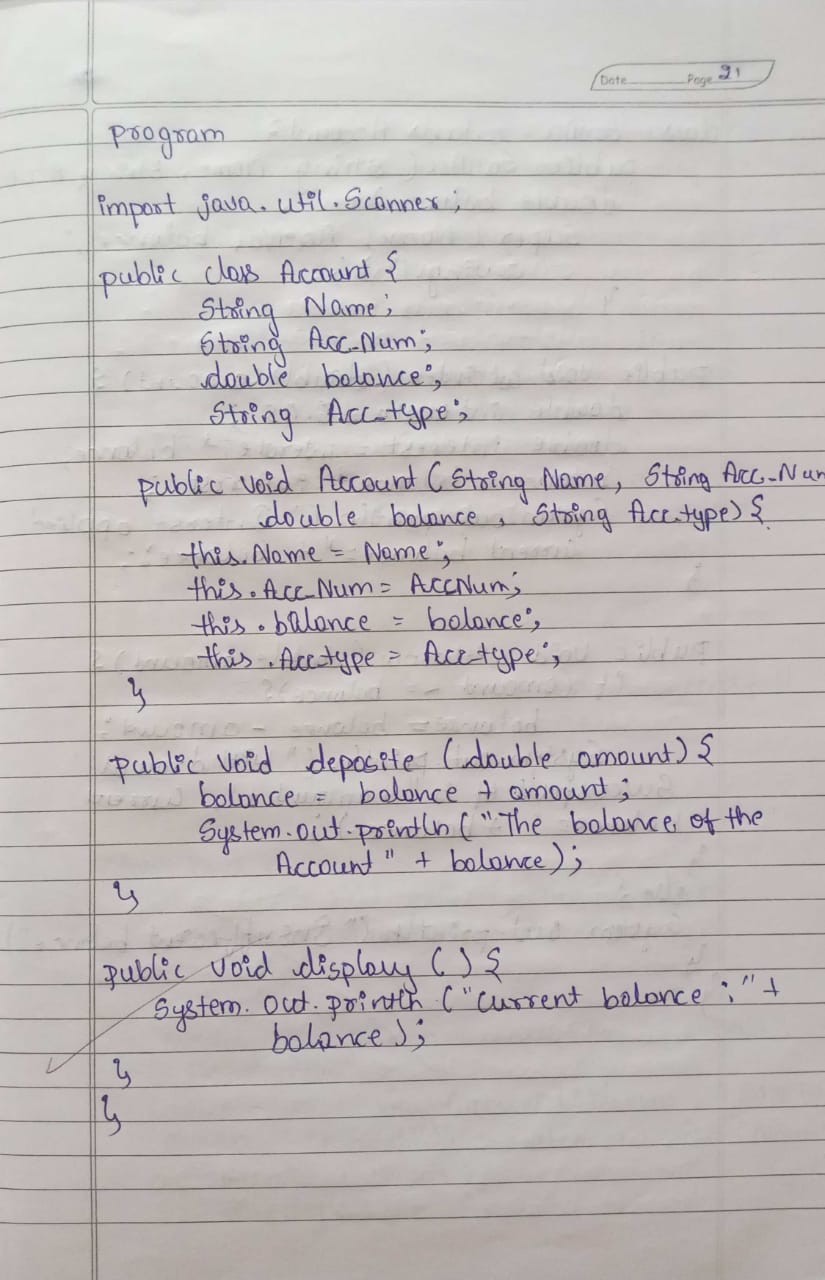
**Output:**

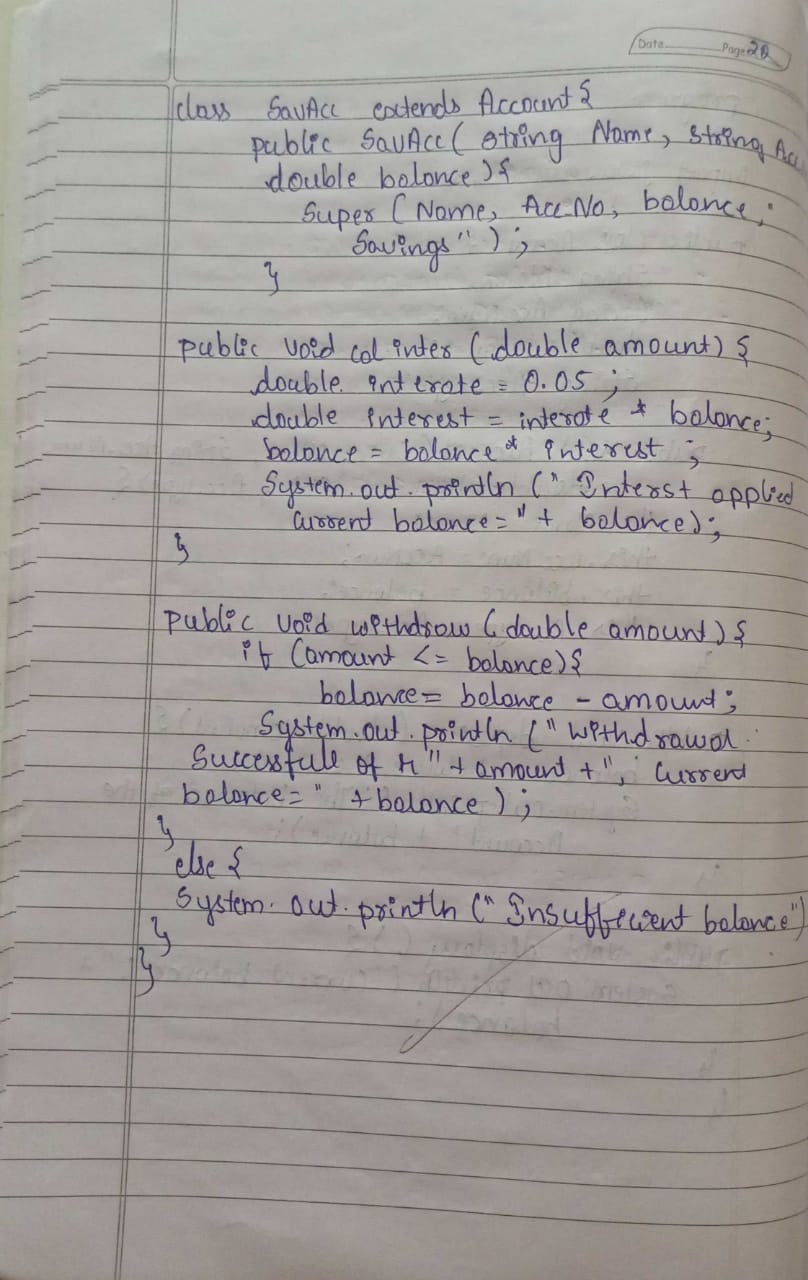
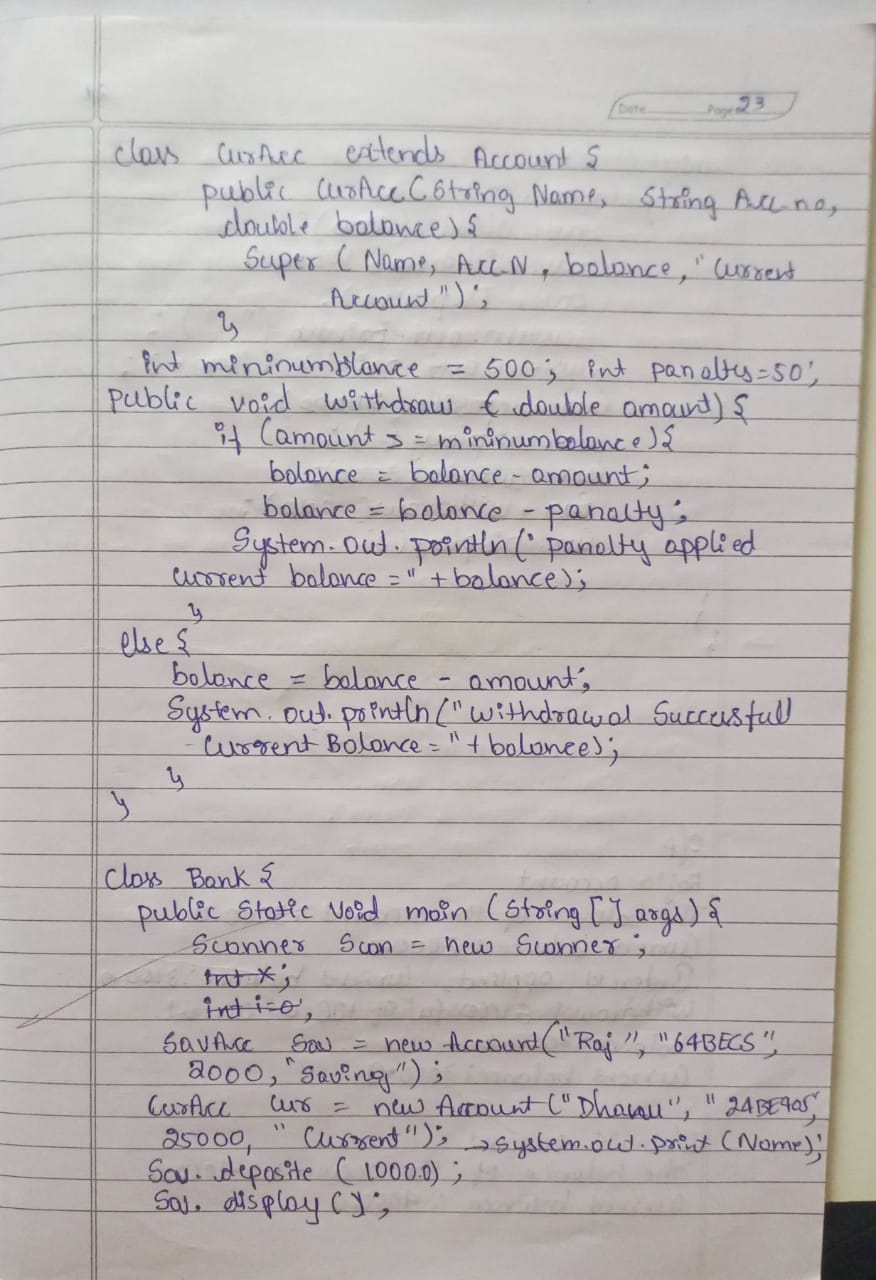


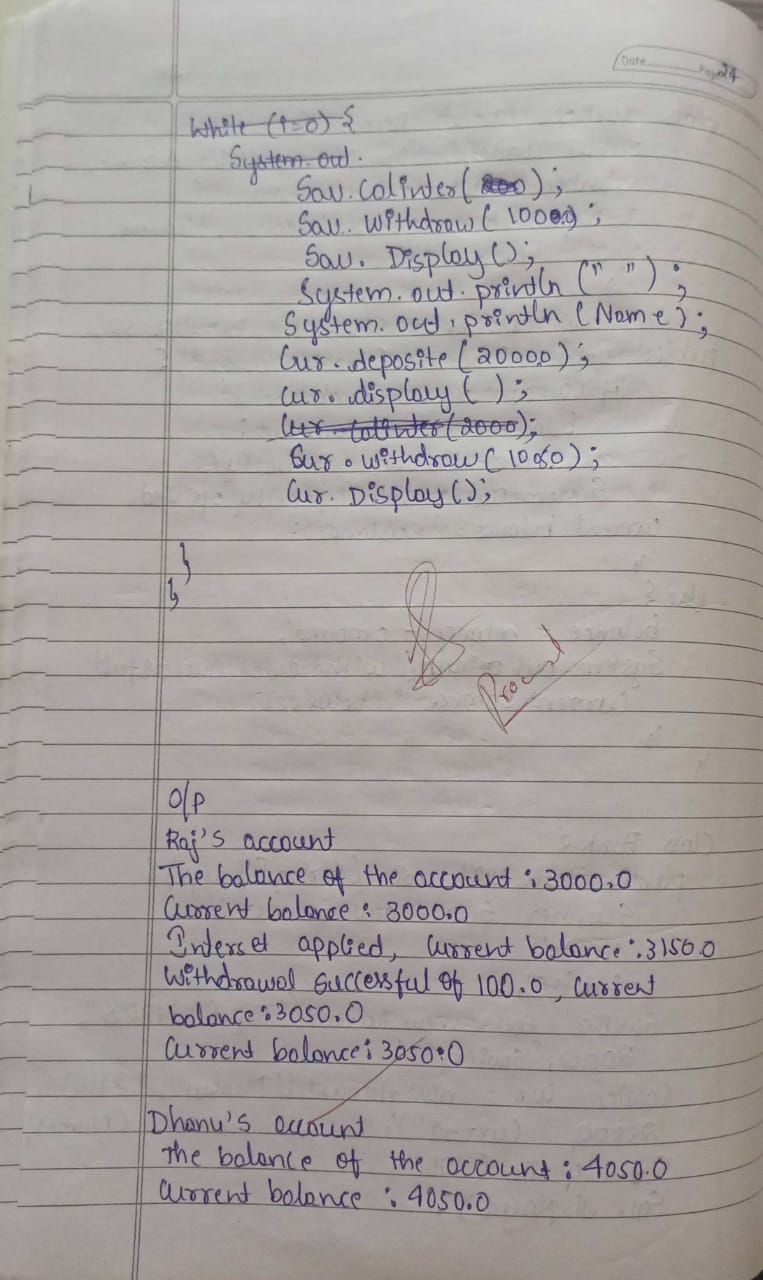
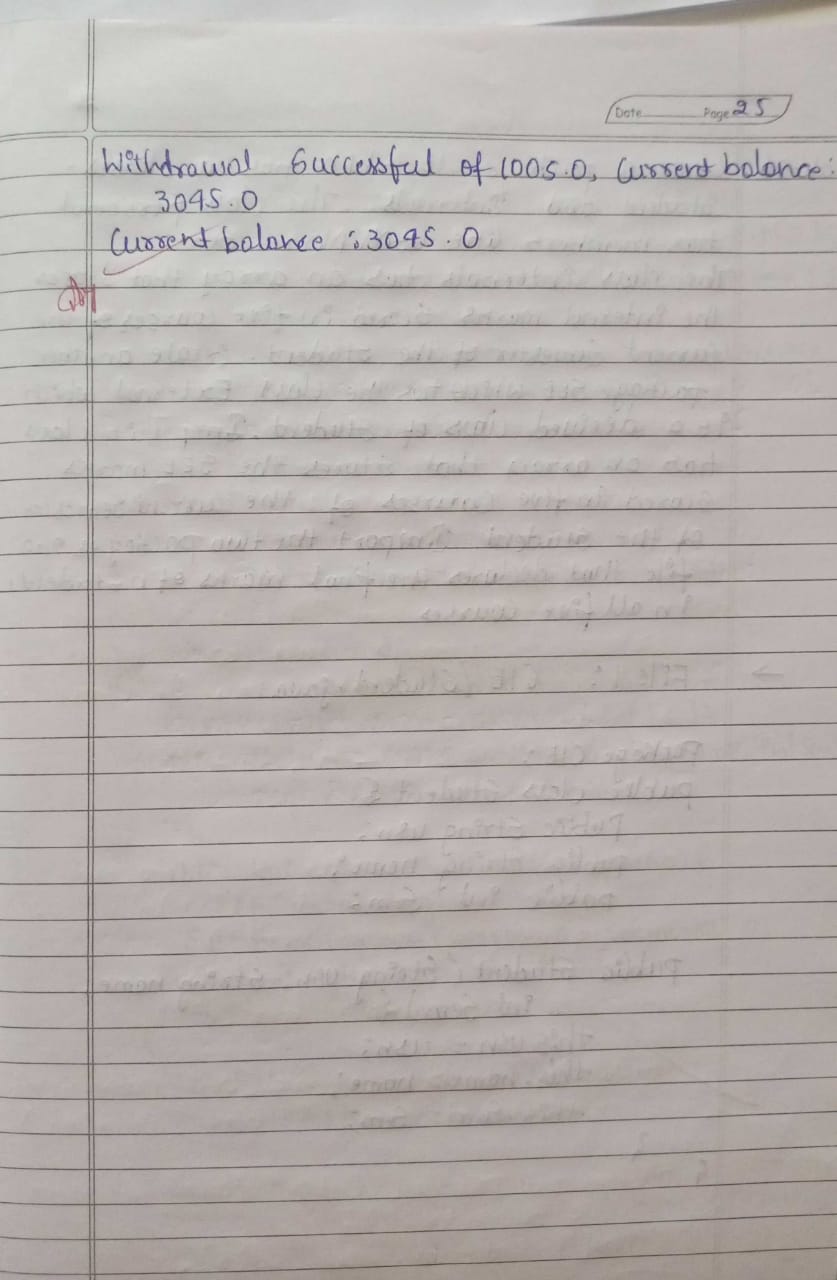
**Program 5**

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.

**Algorithm:**

**Code:**

import java.util.Scanner;

class Account {

String name;

String acc\_num;

double balance;

String acc\_type;

public Account(String name, String acc\_num, double balance, String acc\_type) {

this.name = name;

this.acc\_num = acc\_num;

this.balance = balance;

this.acc\_type = acc\_type;

}

public void deposit(double amount) {

balance += amount;

System.out.println("The balance of the account: " + balance);

}

public void display() {

System.out.println("Current balance: " + balance);

}

}

class SavAcc extends Account {

public SavAcc(String name, String acc\_num, double balance) {

super(name, acc\_num, balance, "Saving");

}

public void calInterest() {

double interestRate = 0.05;

double interest = interestRate \* balance;

balance += interest;

System.out.println("Interest applied, current balance: " + balance);

}

public void withdraw(double amount) {

if (amount <= balance) {

balance -= amount;

System.out.println("Withdrawal successful of " + amount + ", current balance: " + balance);

} else {

System.out.println("Insufficient balance");

}

}

}

class CurAcc extends Account {

int minBalance = 500;

int penalty = 50;

public CurAcc(String name, String acc\_num, double balance) {

super(name, acc\_num, balance, "Current");

}

public void withdraw(double amount) {

if (balance - amount < minBalance) {

balance -= (amount + penalty);

System.out.println("Penalty applied, current balance: " + balance);

} else {

balance -= amount;

System.out.println("Withdrawal successful of " + amount + ", current balance: " + balance);

}

}

}

class bank1 {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

SavAcc sav = new SavAcc("Raj", "64BE404", 2000);

CurAcc cur = new CurAcc("Dhanu", "64BE489", 2050);

System.out.println("Raj's account:");

sav.deposit(1000.0);

sav.display();

sav.calInterest();

sav.withdraw(100.0);

sav.display();

System.out.println("\nDhanu's account:");

cur.deposit(2000.0);

cur.display();

cur.withdraw(1005.0);

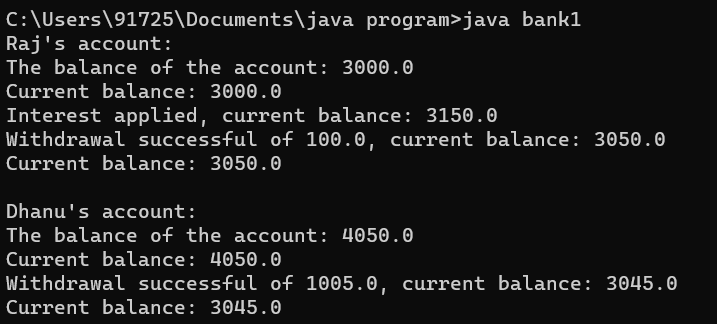
cur.display();

scan.close();

}

}

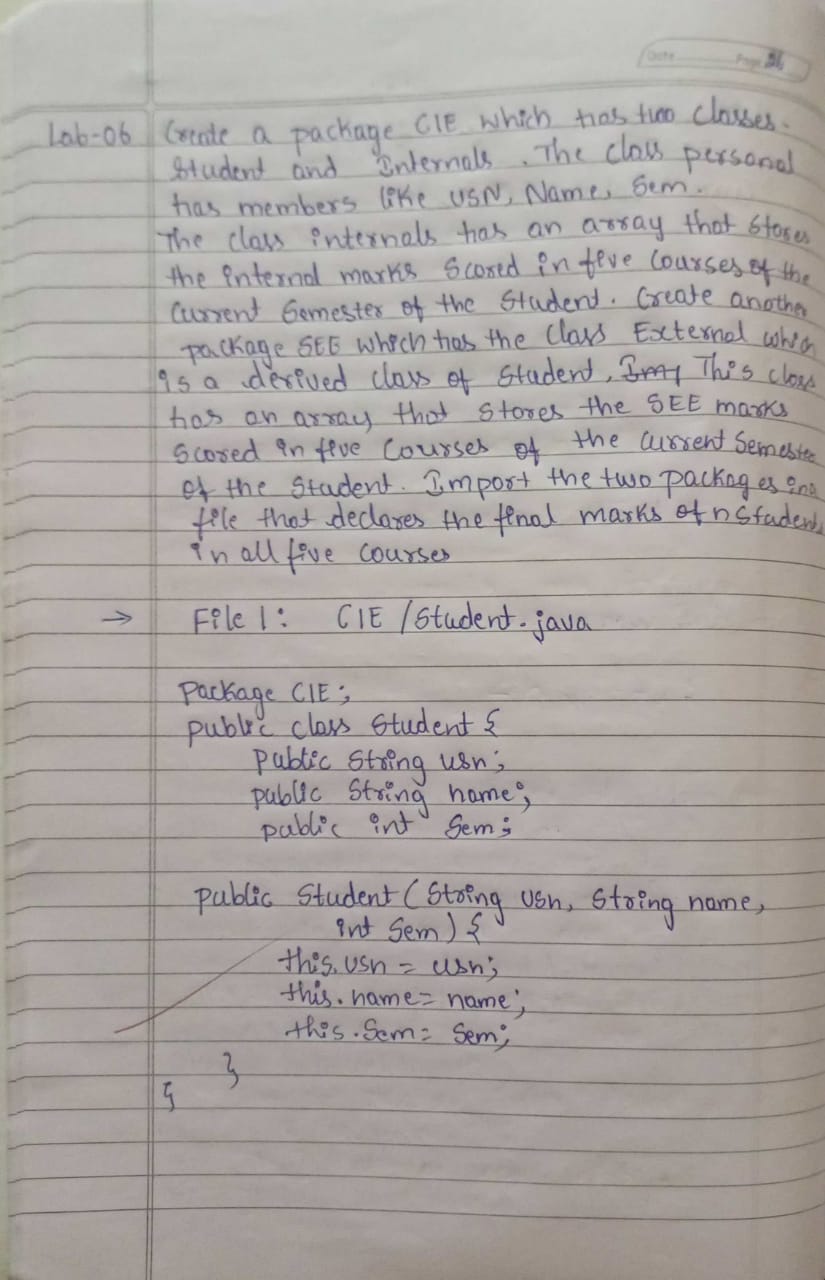
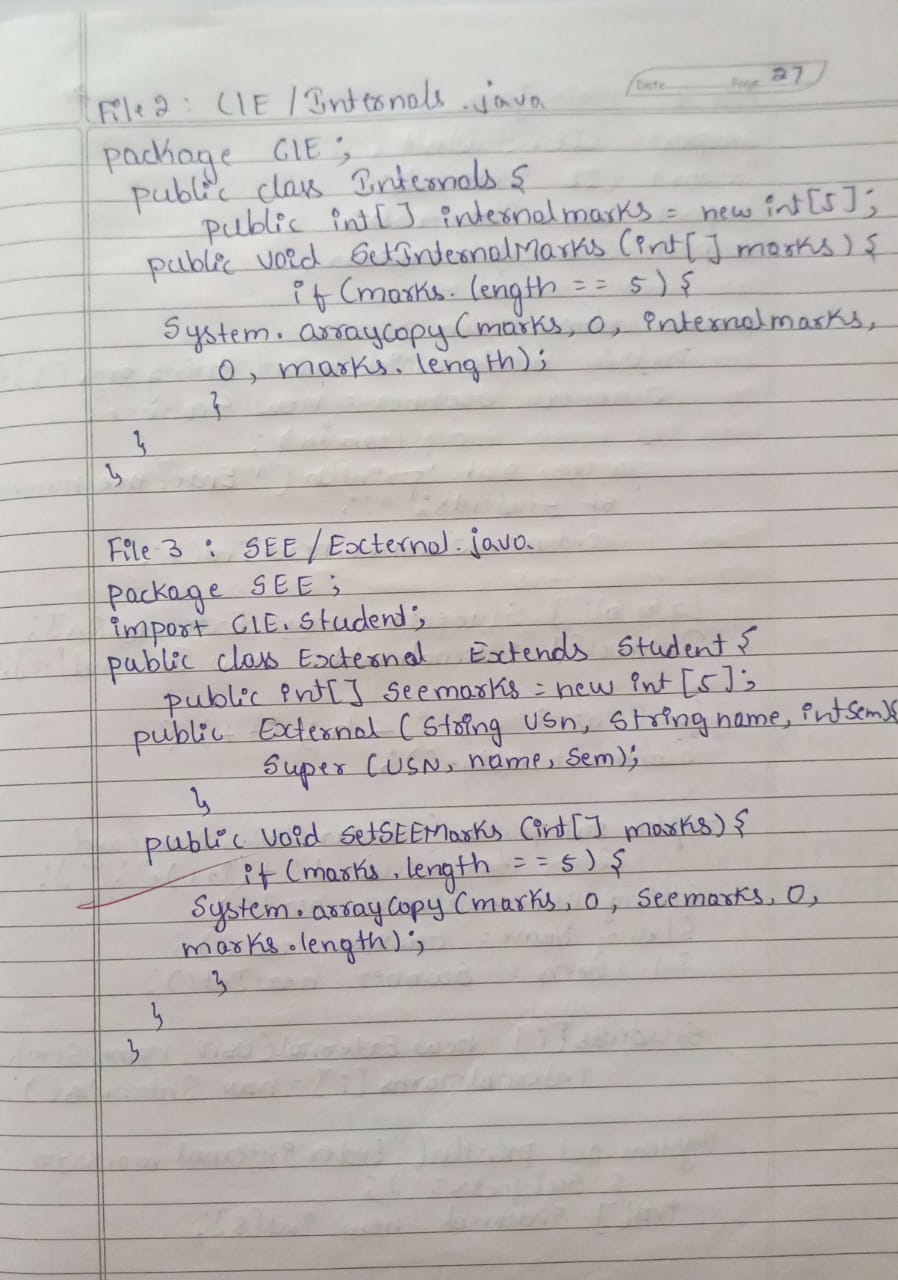
**Output:**

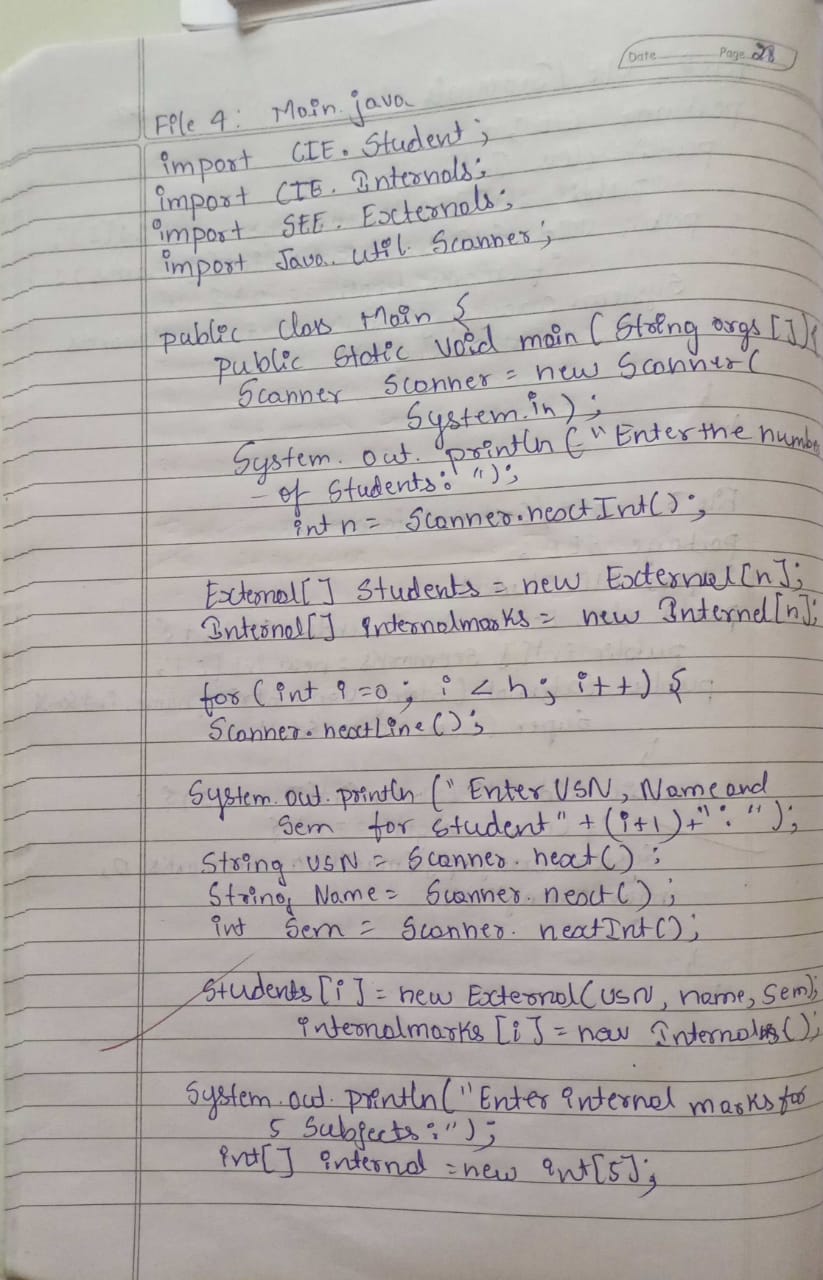
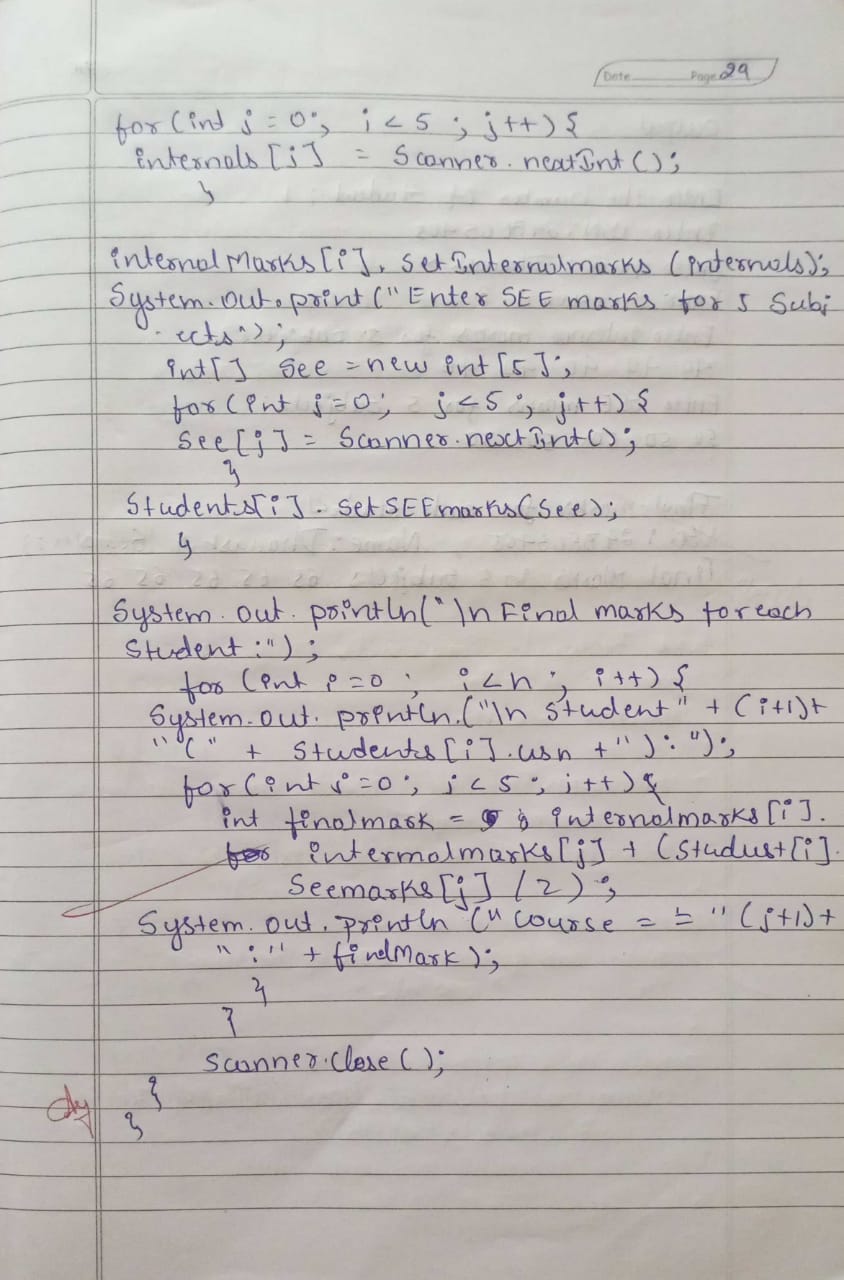


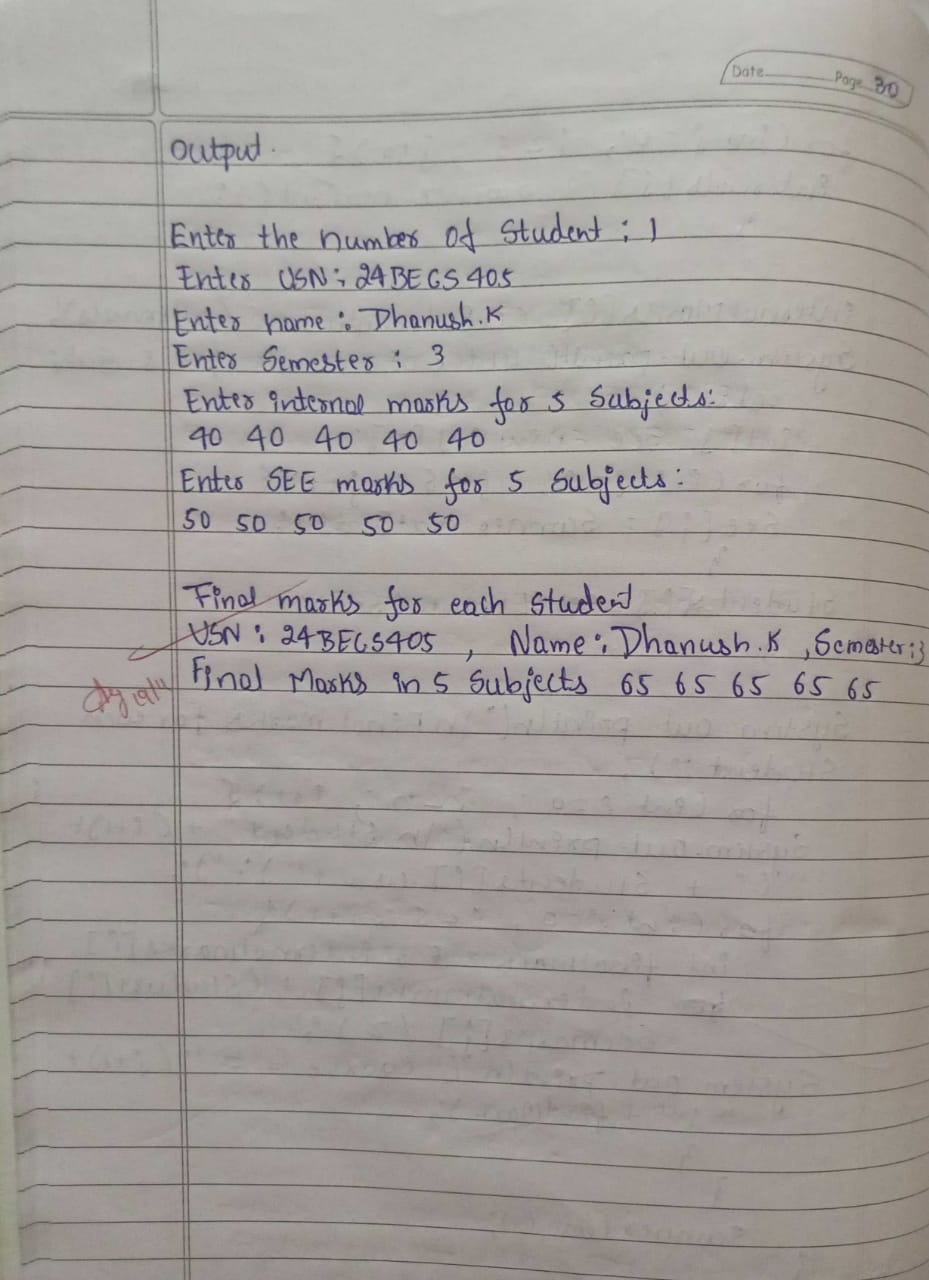
**Program 6**

Create a package CIE which has two classes- Student 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 Student. 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.

**Algorithm:**



**Code:**

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 = name;

this.sem = sem;

}

}

package CIE;

public class Internals {

private int[] internalMarks = new int[5];

public Internals(int[] marks) {

if (marks.length == 5) {

System.arraycopy(marks, 0, internalMarks, 0, 5);

} else {

throw new IllegalArgumentException("Internal marks array must have exactly 5 elements.");

}

}

public int[] getInternalMarks() {

return internalMarks;

}

}

package SEE;

import CIE.Student;

public class External extends Student {

private int[] seeMarks = new int[5];

public External(String usn, String name, int sem, int[] marks) {

super(usn, name, sem);

if (marks.length == 5) {

System.arraycopy(marks, 0, seeMarks, 0, 5);

} else {

throw new IllegalArgumentException("SEE marks array must have exactly 5 elements.");

}

}

public int[] getSeeMarks() {

return seeMarks;

}

}

import CIE.Internals;

import SEE.External;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int n = scanner.nextInt();

scanner.nextLine();

External[] students = new External[n];

Internals[] internalsArray = new Internals[n];

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

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

String usn = scanner.nextLine();

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

String name = scanner.nextLine();

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

int sem = scanner.nextInt();

System.out.println("Enter internal marks for 5 subjects:");

int[] internalMarks = new int[5];

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

internalMarks[j] = scanner.nextInt();

}

internalsArray[i] = new Internals(internalMarks);

System.out.println("Enter SEE marks for 5 subjects:");

int[] seeMarks = new int[5];

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

seeMarks[j] = scanner.nextInt();

}

scanner.nextLine();

students[i] = new External(usn, name, sem, seeMarks);

}

System.out.println("\nFinal marks for each student:");

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

System.out.println("USN: " + students[i].usn + ", Name: " + students[i].name + ", Semester: " + students[i].sem);

int[] finalMarks = calculateFinalMarks(internalsArray[i].getInternalMarks(), students[i].getSeeMarks());

System.out.print("Final Marks in 5 subjects: ");

for (int mark : finalMarks) {

System.out.print(mark + " ");

}

System.out.println();

}

}

public static int[] calculateFinalMarks(int[] internalMarks, int[] seeMarks) {

int[] finalMarks = new int[5];

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

finalMarks[i] = internalMarks[i] + (seeMarks[i] / 2);

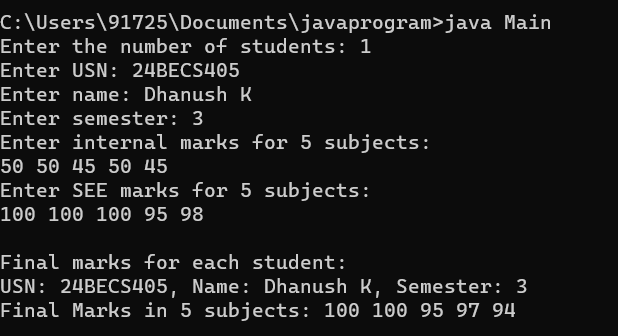
}

return finalMarks;

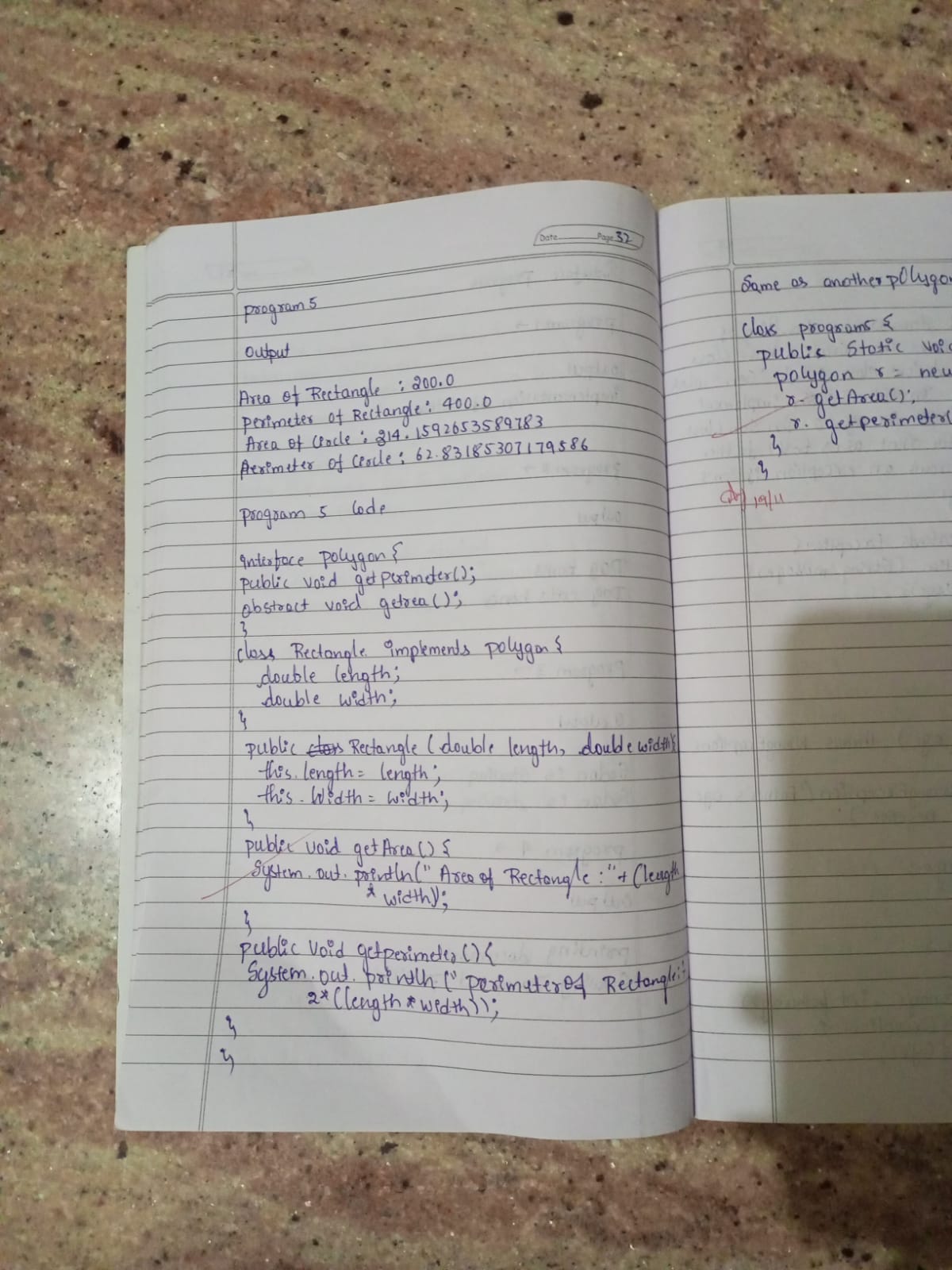
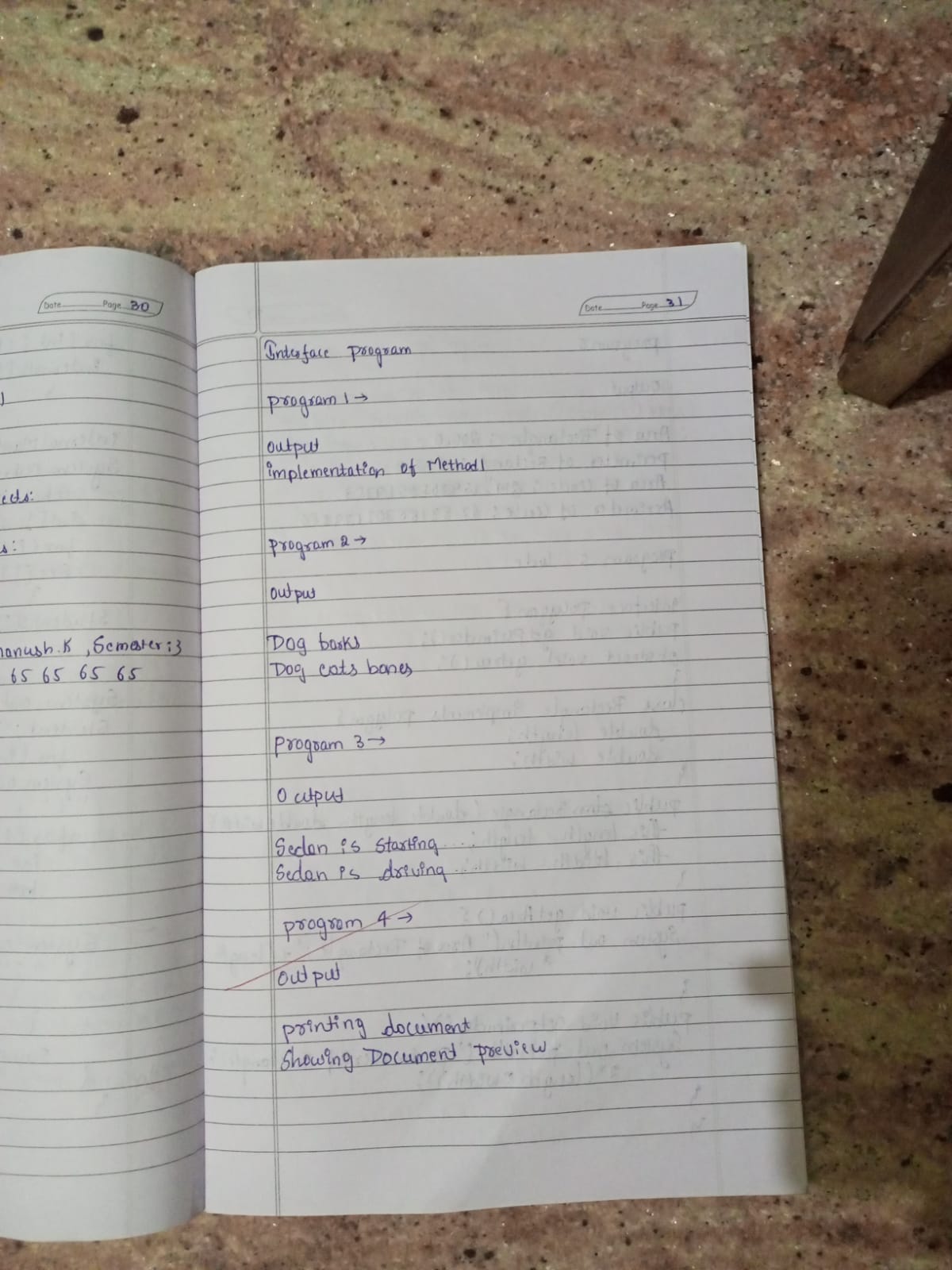
}

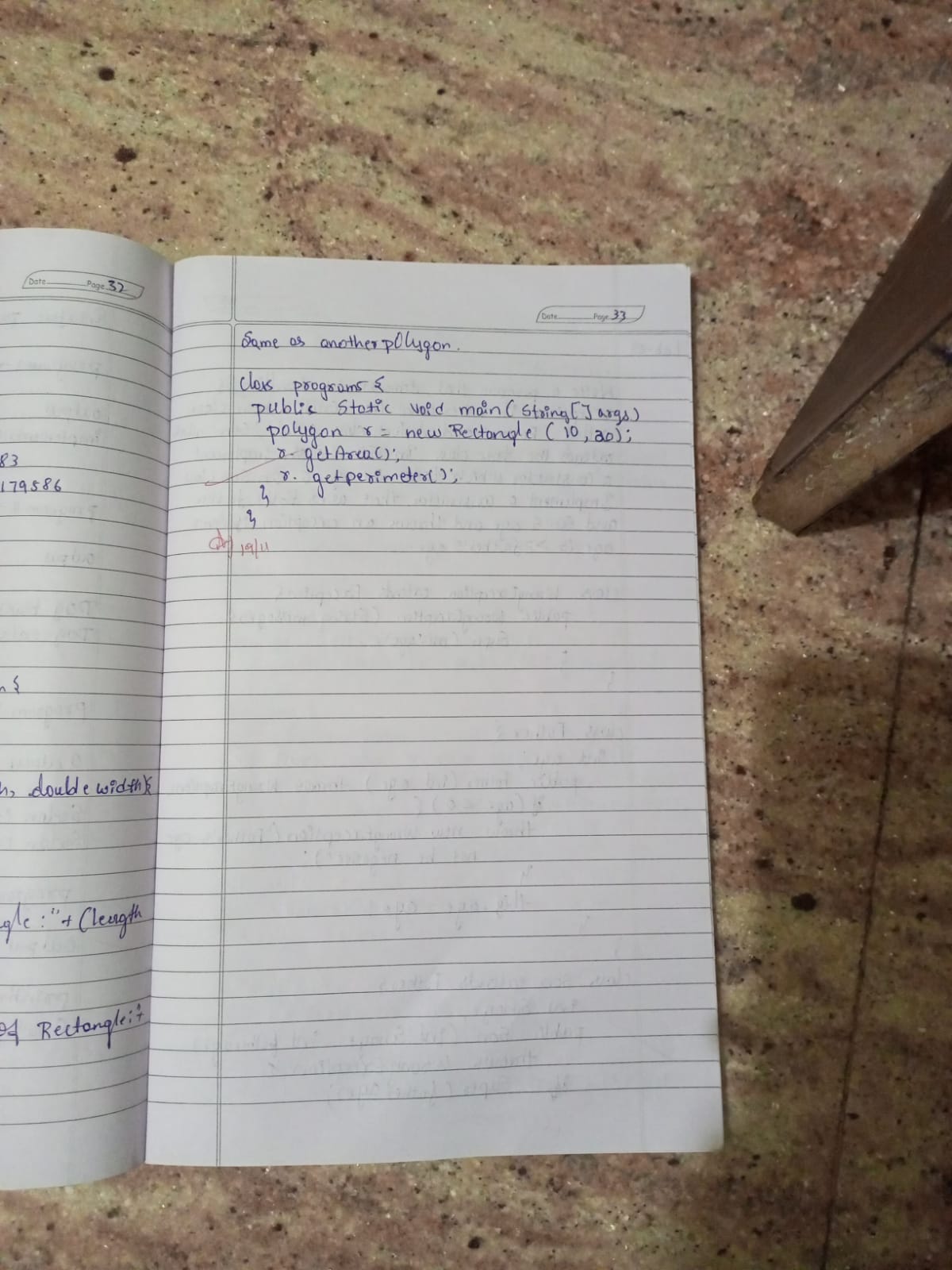
}

**Output:**



**Interface 5 programs:**





**Code:**

program 1\*\*\*\*\*\*\*

interface MyInterface {

public void method1();

public void method2();

}

class Demo implements MyInterface{

public void method1(){

System.out.println("implementation of method1");

}

public void method2(){

System.out.println("implementation of method2");

}

public static void main(String arg[])

{

MyInterface obj = new Demo();

obj.method1();

}

}

program 2\*\*\*\*

interface Animal{

void Sound();

void eat();

}

class Dog implements Animal{

public void Sound(){

System.out.println("Dog barks");

}

public void eat(){

System.out.println("Dog eats bones");

}

}

public class Maini{

public static void main(String[] args){

Animal myDog = new Dog();

myDog.Sound();

myDog.eat();

}

}

program 3\*\*\*\*\*\*

interface Vehicle{

void start();

}

interface Car extends Vehicle{

void drive();

}

class Sedan implements Car{

public void start(){

System.out.println("Sedan is starting...");

}

public void drive(){

System.out.println("Sedan is driving...");

}

}

public class Mainv{

public static void main(String[] args){

Car mycar = new Sedan();

mycar.start();

mycar.drive();

}

}

program 4 \*\*\*\*\*\*\*\*

interface Printable{

void print();

}

interface showable{

void show();

}

class Document implements Printable,showable{

public void print(){

System.out.println("Printing document");

}

public void show (){

System.out.println("Showing Document preview");

}

}

public class Mainp{

public static void main (String[] args){

Document doc = new Document();

doc.print();

doc.show();

}

}

program 5 \*\*\*\*\*\*\*

interface Polygon {

public void getPerimeter();

public void getArea();

}

class Rectangle implements Polygon {

double length;

double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

public void getArea() {

System.out.println("Area of Rectangle:"+ (length \* width));

}

public void getPerimeter() {

System.out.println("Perimeter of Rectangle:"+ 2\*(length \* width));

}

}

class Circle implements Polygon {

double radius;

public Circle(double radius) {

this.radius = radius;

}

public void getArea() {

System.out.println("Area of circle:"+ (Math.PI \* radius \* radius));

}

public void getPerimeter() {

System.out.println("Area of circle:"+ (2 \* Math.PI \* radius));

}

}

class program5 {

public static void main(String[] args){

Polygon r = new Rectangle(10,20);

r.getArea();

r.getPerimeter();

Polygon c = new Circle(10);

c.getArea();

c.getPerimeter();

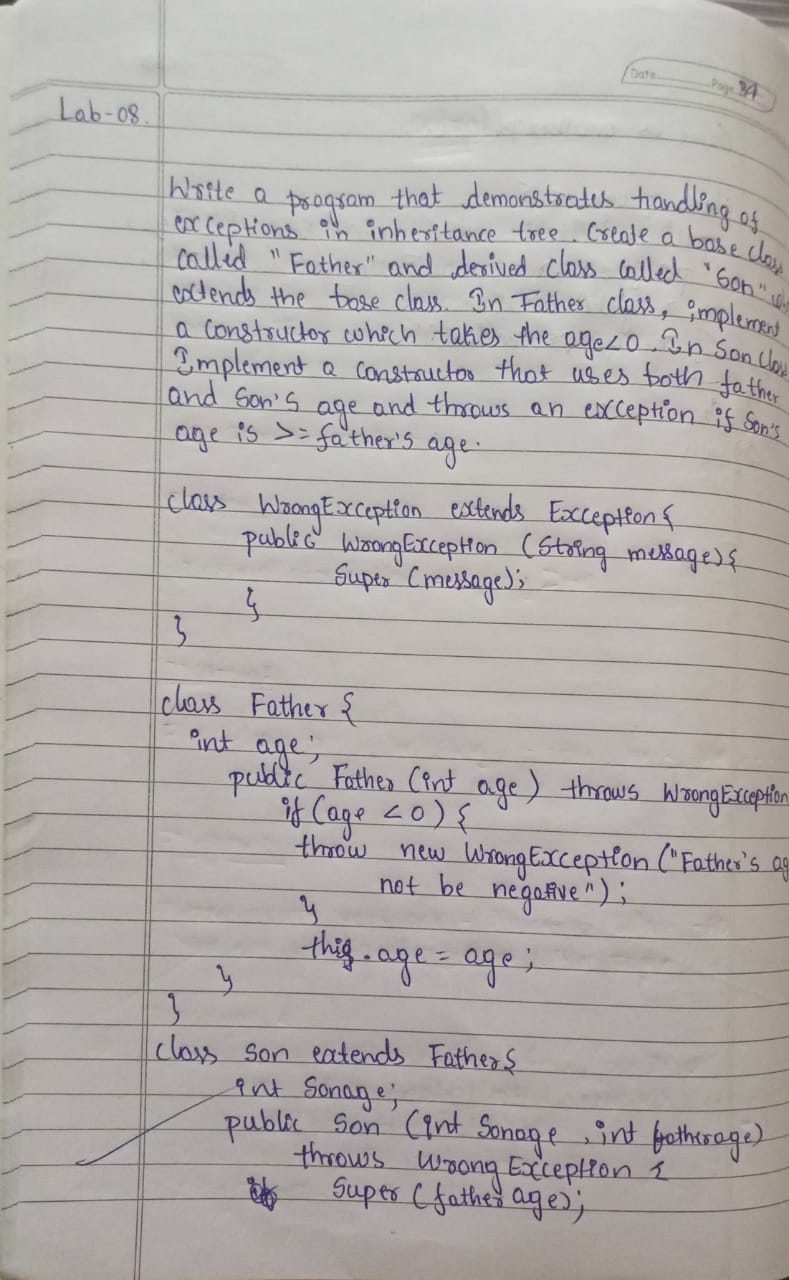
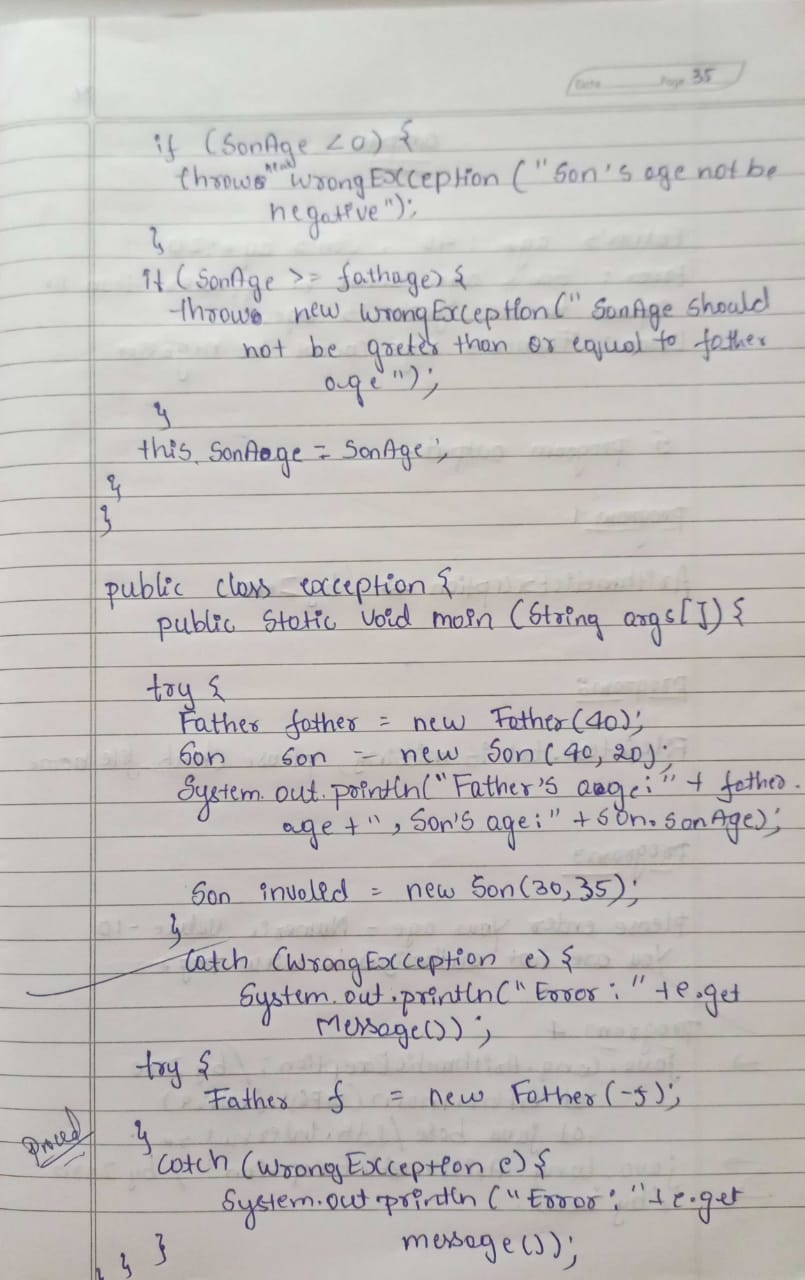
}

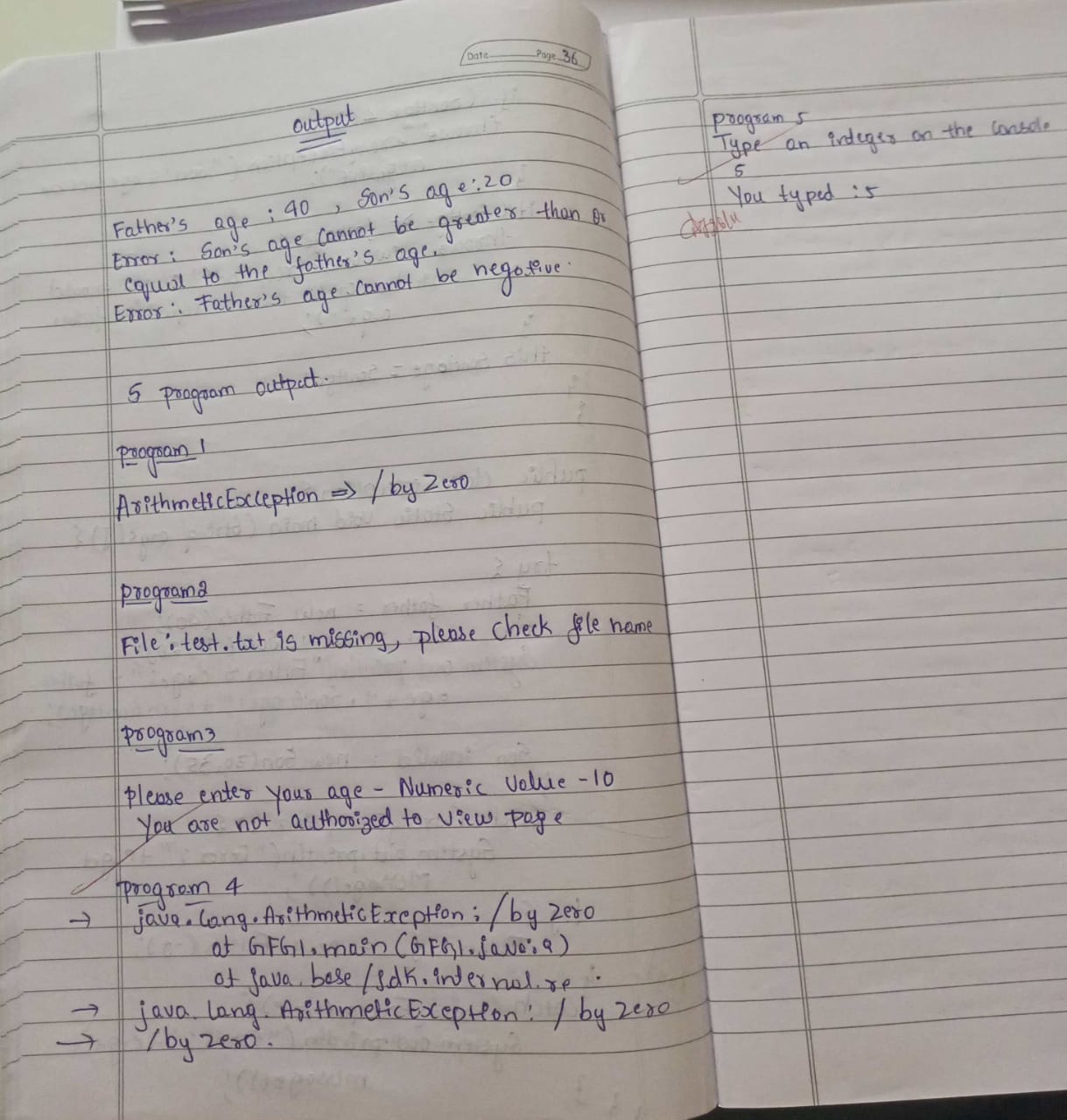
}

**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 Wrong Age( ) when the input age=father’s age

**Algorithm:**



**Code:**

class WrongException extends Exception {

public WrongException(String message) {

super(message);

}

}

class Father {

int age;

public Father(int age) throws WrongException {

if (age < 0) {

throw new WrongException("Father's age cannot be negative");

}

this.age = age;

}

}

class Son extends Father {

int sonage;

public Son(int fatherAge, int sonAge) throws WrongException {

super(fatherAge);

if (sonAge < 0) {

throw new WrongException("Son's age cannot be negative");

}

if (sonAge >= fatherAge) {

throw new WrongException("Son's age cannot be greater than or equal to the father's age");

}

this.sonage = sonAge;

}

}

public class Main {

public static void main(String args[]) {

System.out.println("Exception Handling Example:");

try {

Father father = new Father(40);

Son son = new Son(40, 20);

System.out.println("Father's age: " + father.age + ", Son's age: " + son.sonage);

Son invalidSon = new Son(30, 35);

} catch (WrongException e) {

System.out.println("Error: " + e.getMessage());

}

try {

Father invalidFather = new Father(-5);

} catch (WrongException e) {

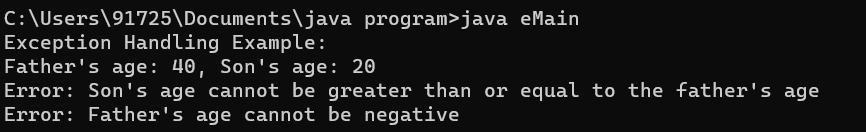
System.out.println("Error: " + e.getMessage());

}

}

}

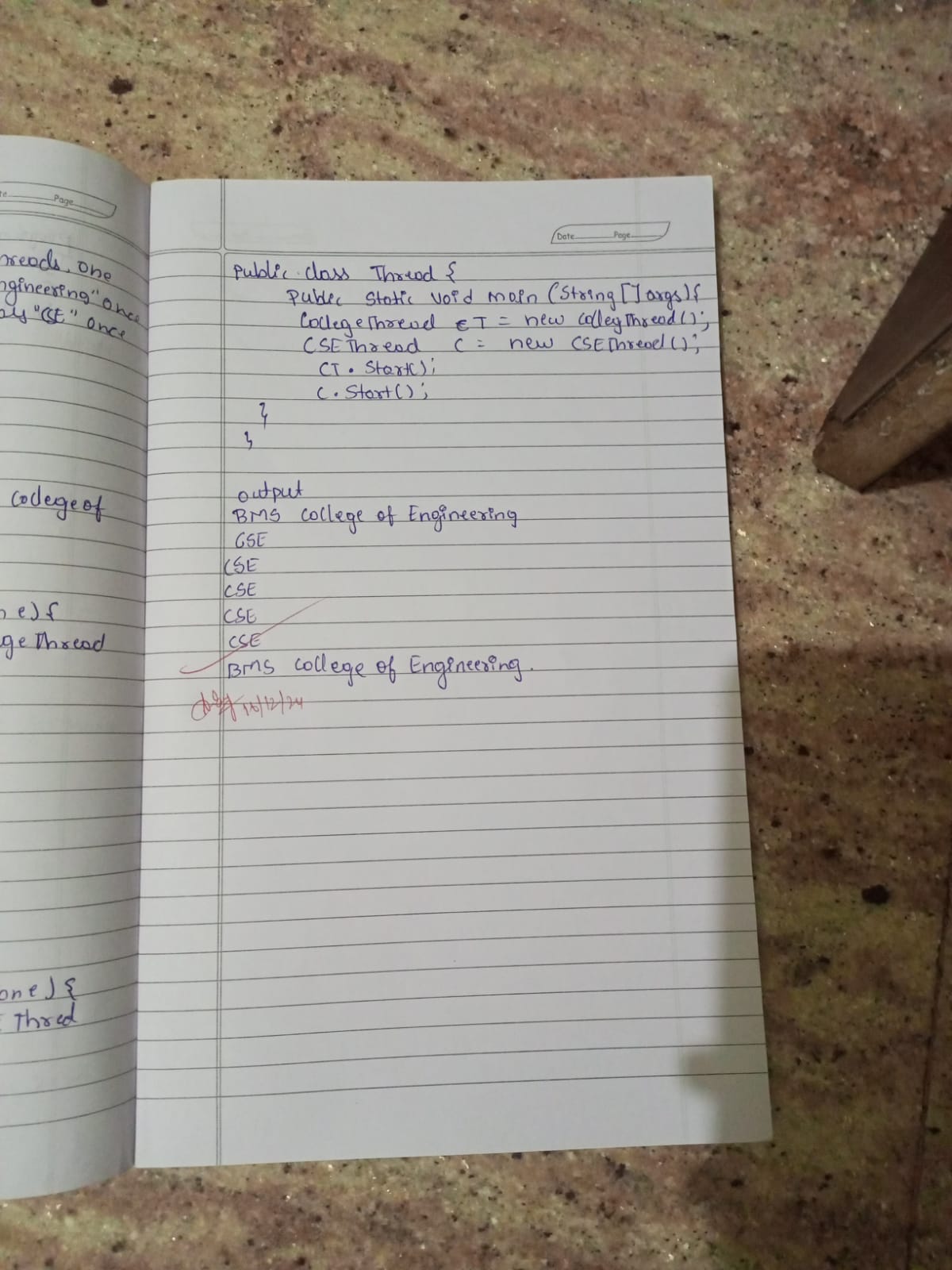
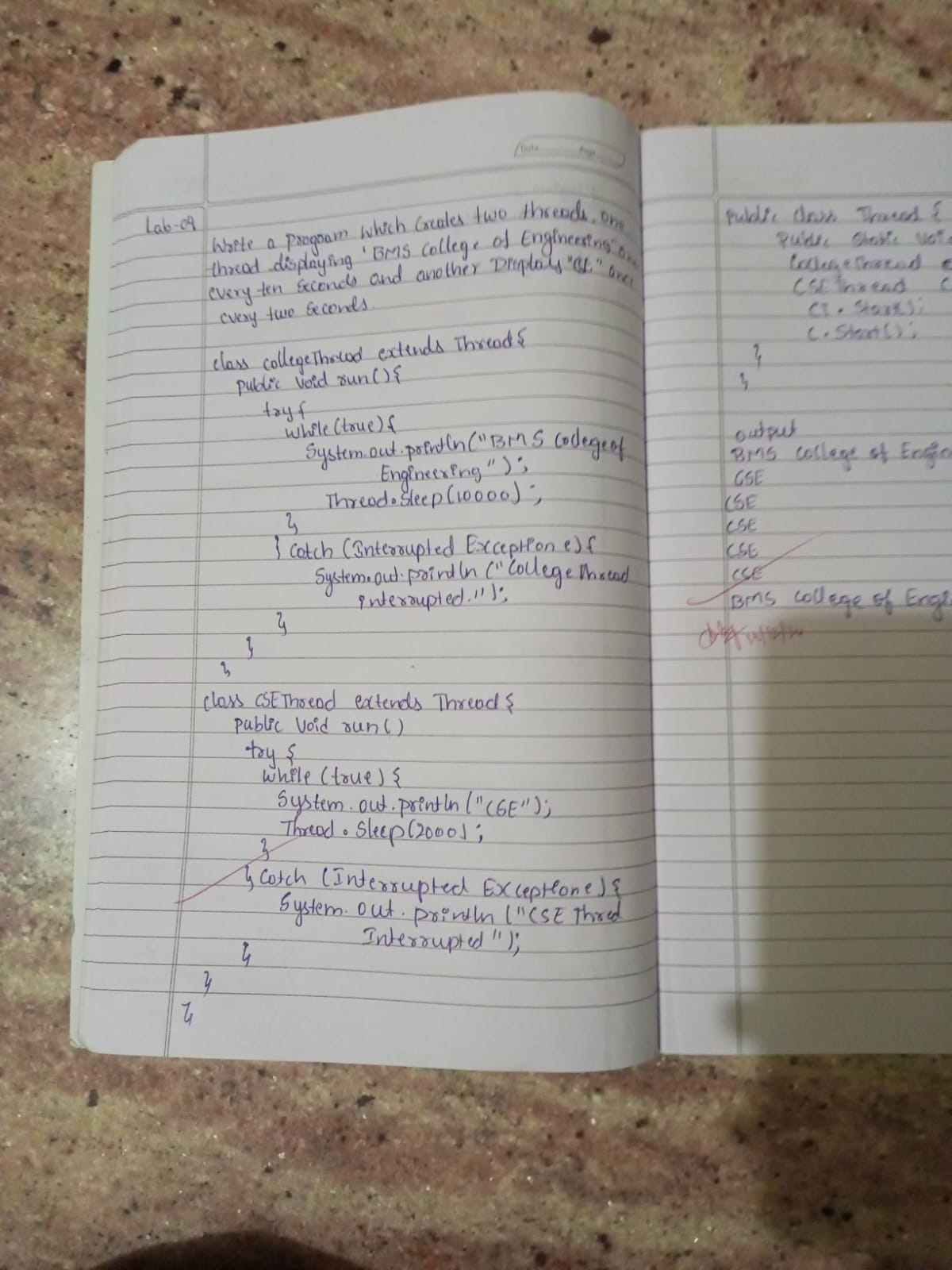
**Output:**



**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.

**Algorithm:**



**Code:**

class CollegeThread extends Thread {

public void run() {

try {

while (true) {

System.out.println("BMS College of Engineering");

Thread.sleep(10000);

}

} catch (InterruptedException e) {

System.out.println("CollegeThread interrupted.");

}

}

}

class CSEThread extends Thread {

public void run() {

try {

while (true) {

System.out.println("CSE");

Thread.sleep(2000);

}

} catch (InterruptedException e) {

System.out.println("CSEThread interrupted.");

}

}

}

public class ThreadDisplayProgram {

public static void main(String[] args) {

CollegeThread collegeThread = new CollegeThread();

CSEThread cseThread = new CSEThread();

collegeThread.start();

cseThread.start();

}

}

**Output:**

