**VISVESVARAYA TECHNOLOGICAL**

**UNIVERSITY**

“JnanaSangama”, Belgaum -590014, Karnataka.



**Object Oriented Java Programming**

**(23CS3PCOOJ)**

***Submitted by***

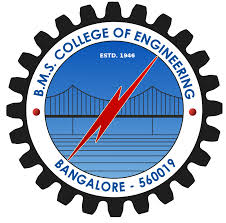
Krithika H Kotian (1BM23CS159)

***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**

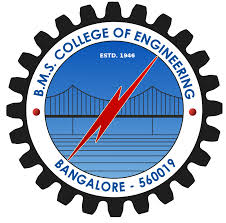
**Sep-2024 to Jan-2025**

**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 Krithika H Kotian(1BM23CS159), 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. 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.

|  |  |
| --- | --- |
| Prof. Ambuja  Assistant Professor  Department of CSE, BMSCE | Dr. Jyothi S Nayak  Professor &amp; HOD  Department of CSE, BMSCE |

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment title** | **Page No.** |
| 1 | 23/09/2024 | Implement Quadratic Equation | 3-5 |
| 2 | 30/09/2024 | Implement SGPA Calculator | 6-10 |
| 3 | 07/10/2024 | Implement ToString | 11-14 |
| 4 | 14/10/2024 | Implement Shape Area | 15-18 |
| 5 | 21/10/2024 | Implement Saving and Current Account | 19-26 |
| 6 | 28/10/2024 | Implement Package | 27-30 |
| 7 | 11/11/2024 | Implement Age Exception | 31-34 |
| 8 | 28/11/2024 | Implement Threads | 35-36 |
| 9 | 28/11/2024 | Implement Swing demo | 37-40 |
| 10(a) | 28/11/2024 | ImplementDeadLock | 41-44 |
| 10(b) | 28/11/2024 | ImplementDeadLock | 44-47 |

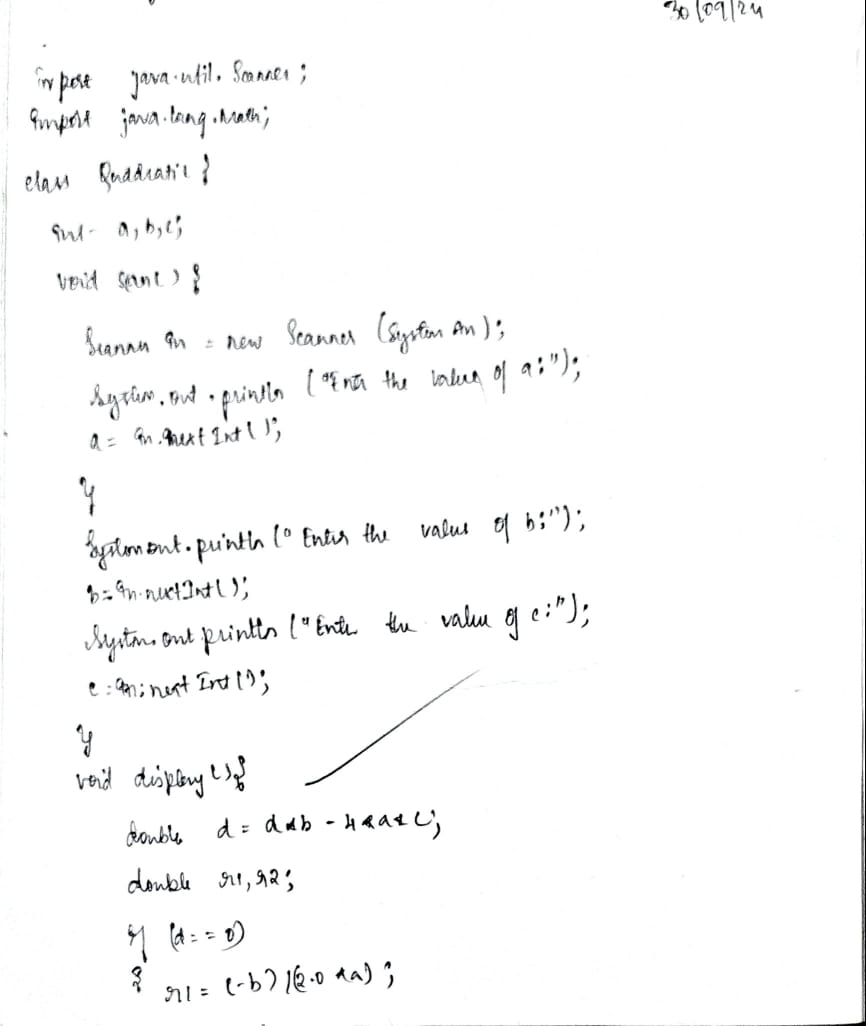
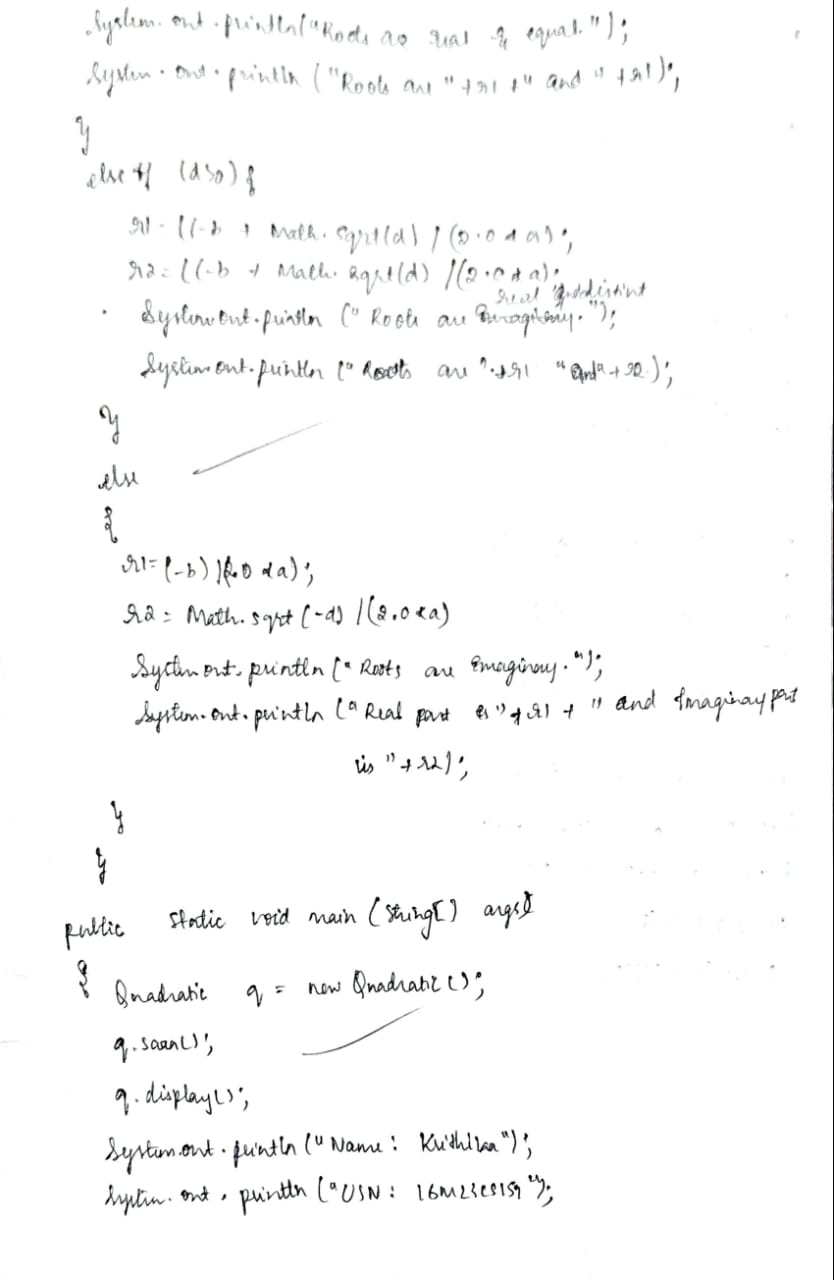
**Github Link:**

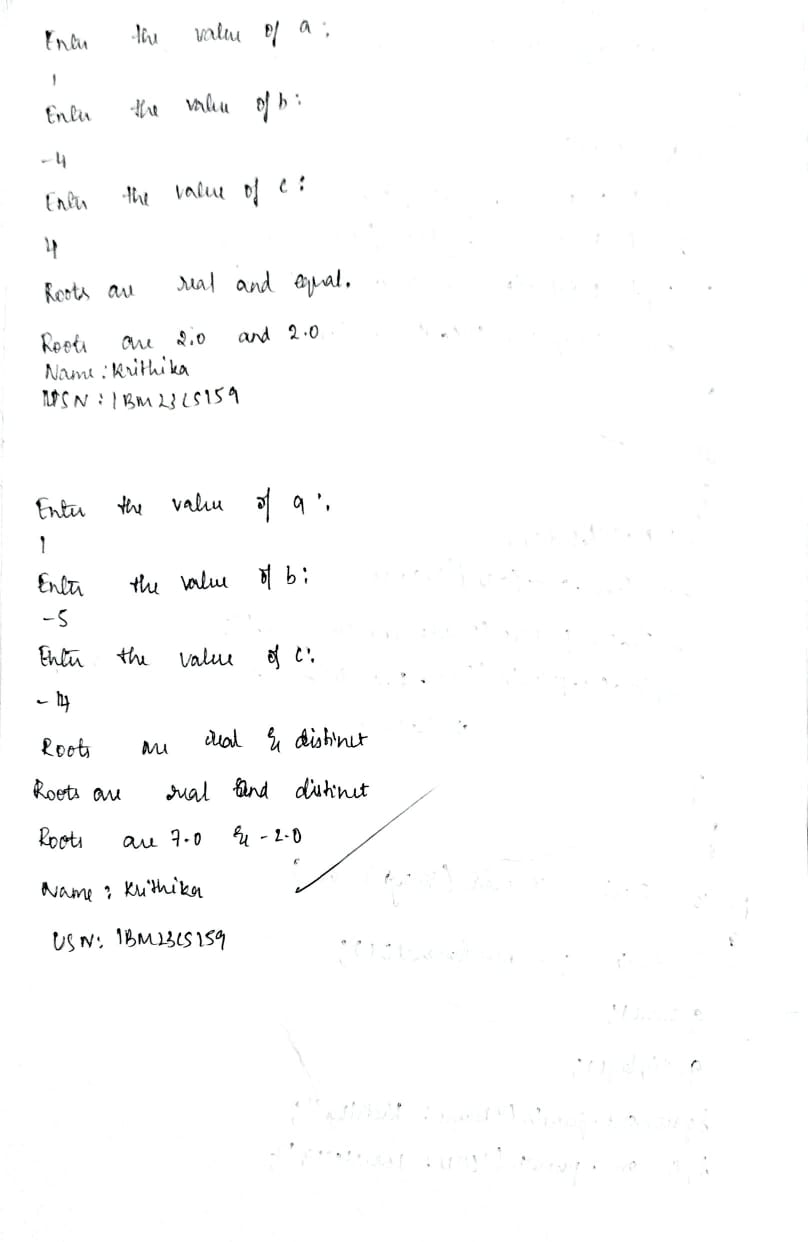
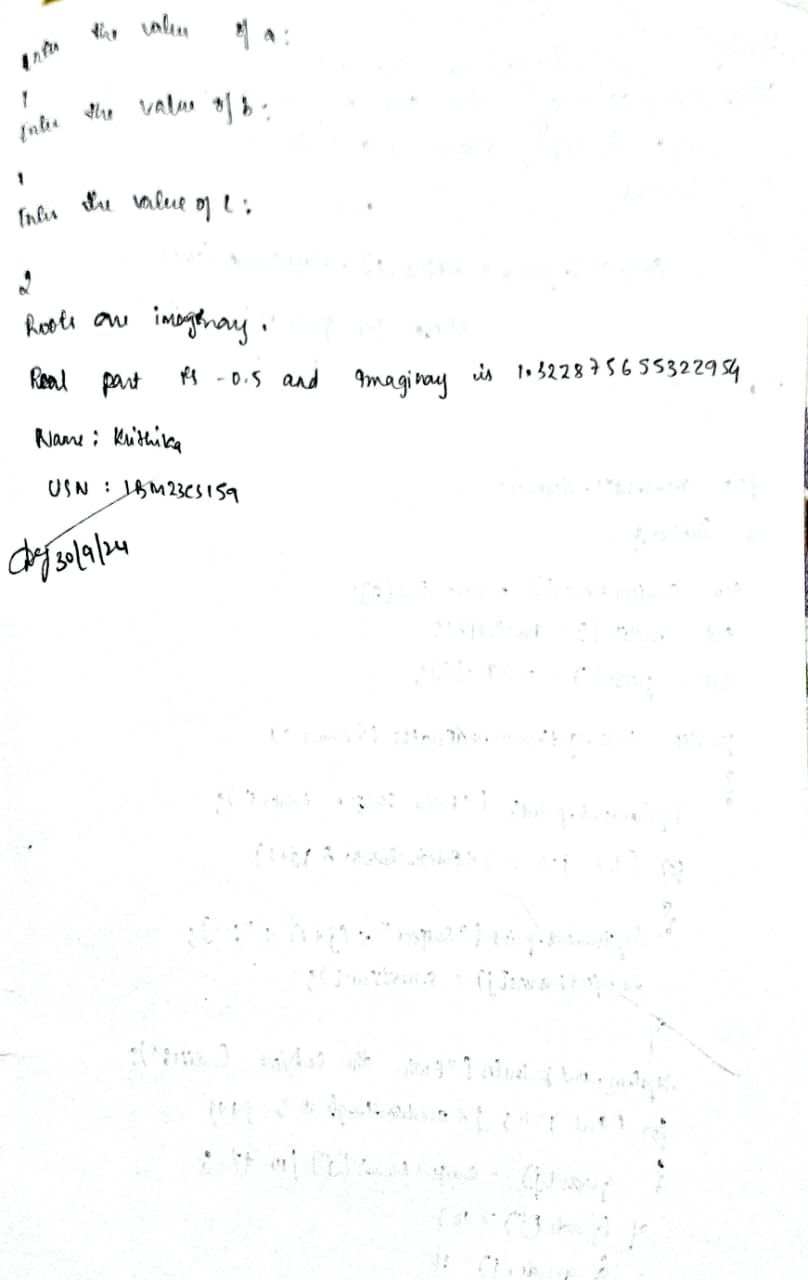
https://github.com/krithikahkotian/java\_lab

**Program 1**

**Implement Quadratic Equation**

Algorithm:





**Code:**

import java.util.Scanner;

import java.lang.Math;

class Quadratic {

int a, b, c;

void scan() {

Scanner in = new Scanner(System.in);

System.out.println("Enter the value of a:");

a = in.nextInt();

if (a == 0) {

System.out.println("Value of a is invalid. Enter a non-zero value:");

a = in.nextInt();

}

System.out.println("Enter the value of b:");

b = in.nextInt();

System.out.println("Enter the value of c:");

c = in.nextInt();

}

void display() {

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

double r1, r2;

if (d == 0) {

r1 = (-b) / (2.0 \* a);

System.out.println("Roots are real and equal.");

System.out.println("Roots are " + r1 + " and " + r1);

} else if (d > 0) {

r1 = (-b + Math.sqrt(d)) / (2.0 \* a);

r2 = (-b - Math.sqrt(d)) / (2.0 \* a);

System.out.println("Roots are real and distinct.");

System.out.println("Roots are " + r1 + " and " + r2);

} else {

r1 = (-b) / (2.0 \* a);

r2 = Math.sqrt(-d) / (2.0 \* a);

System.out.println("Roots are imaginary.");

System.out.println("Real part is " + r1 + " and imaginary part is " + r2);

}

}

public static void main(String[] args) {

Quadratic q = new Quadratic();

q.scan();

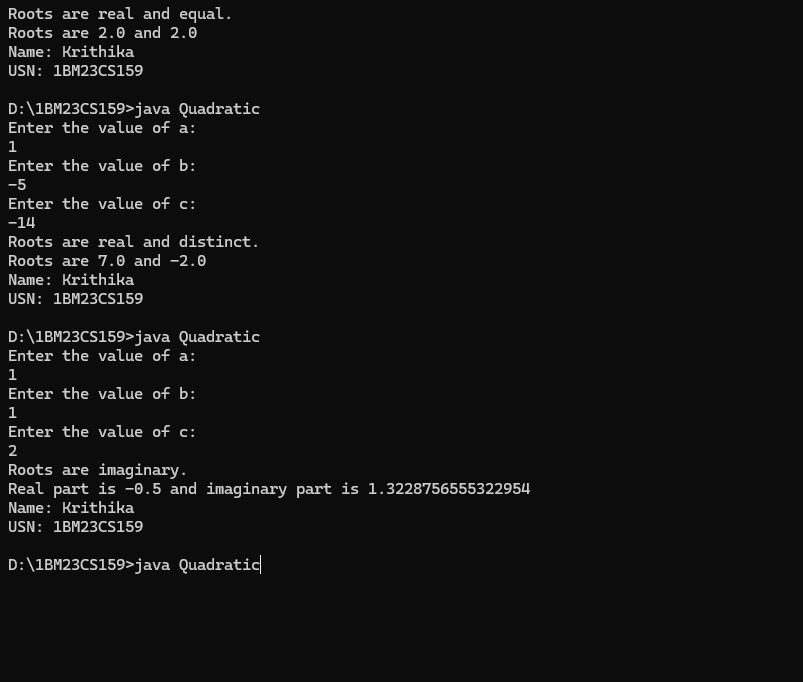
q.display();

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

System.out.println("USN: 1BM23CS159");

}

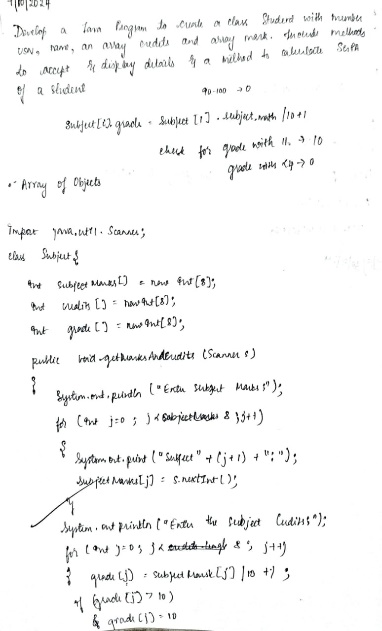
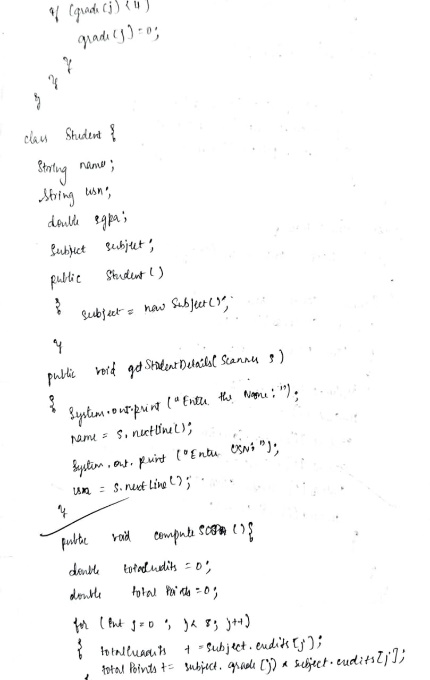
}

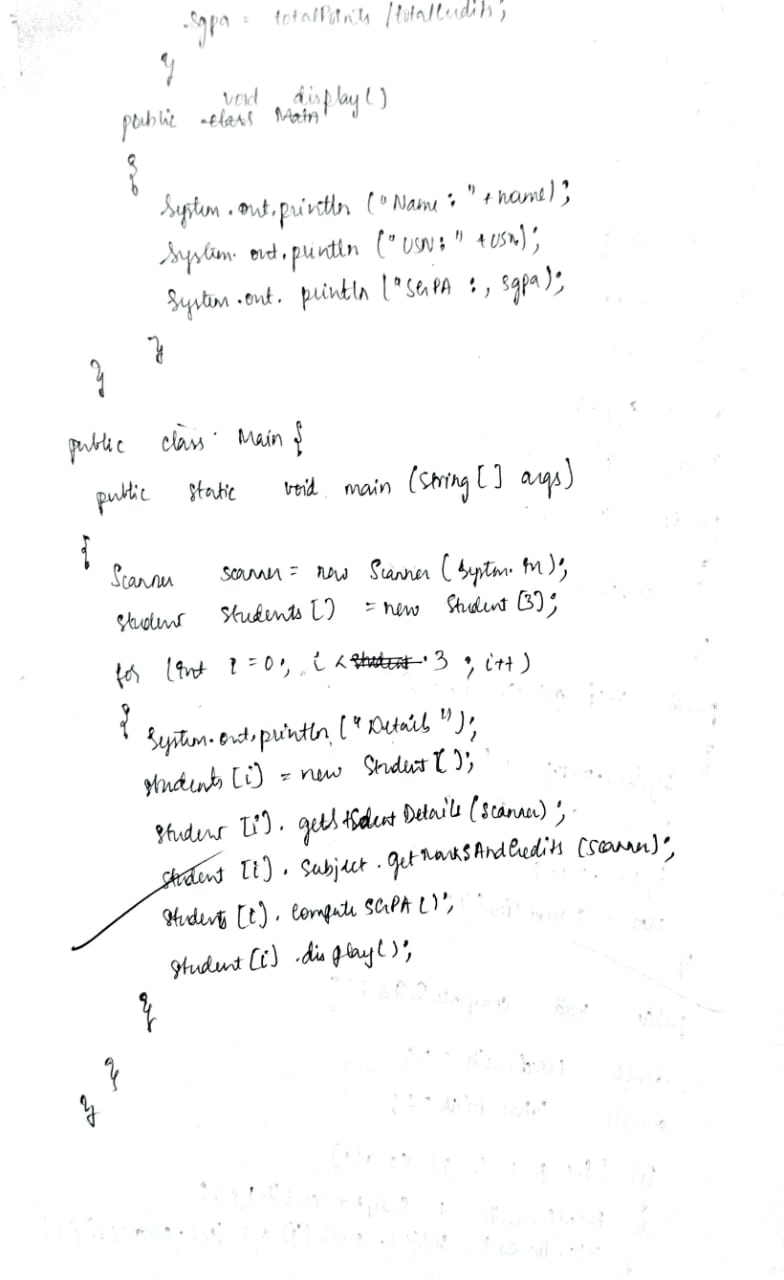
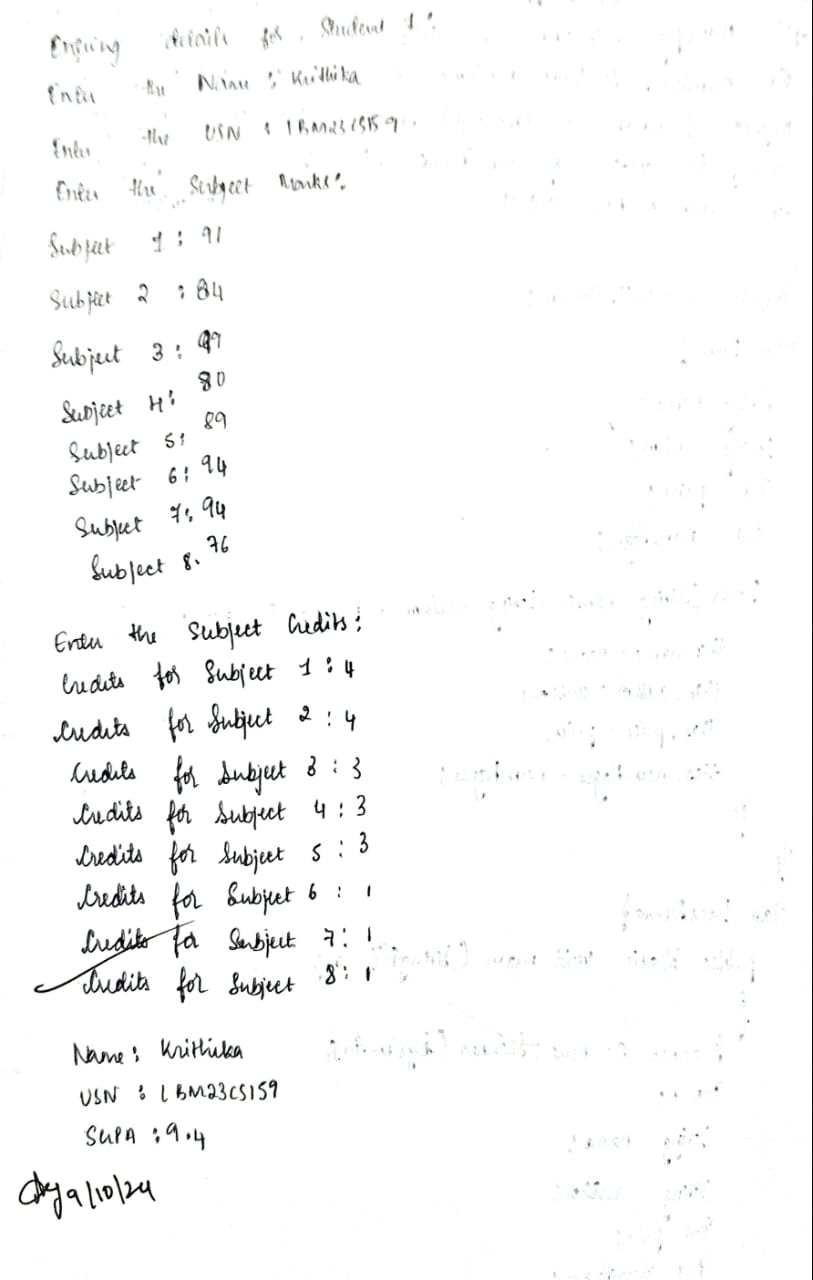


**Program 2**

**Implement SGPA Calculator**

**Algorithm:**





**Code:**

import java.util.Scanner;

class Subject {

int subjectMarks[] = new int[8];

int credits[] = new int[8];

int grades[] = new int[8];

public void getMarksAndCredits(Scanner s) {

System.out.println("Enter the subject Marks:");

for (int j = 0; j < 8; j++) {

System.out.print("Subject " + (j + 1) + ": ");

subjectMarks[j] = s.nextInt();

}

System.out.println("Enter the subject Credits:");

for (int j = 0; j < 8; j++) {

System.out.print("Credits for Subject " + (j + 1) + ": ");

credits[j] = s.nextInt();

}

}

public void computeGrades() {

for (int j = 0; j < 8; j++) {

grades[j] = subjectMarks[j] / 10 + 1;

if (grades[j] > 10) {

grades[j] = 10;

}

if (grades[j] < 4) {

grades[j] = 0;

}

}

}

}

class Student {

String name;

String usn;

double sgpa;

Subject subject;

public Student() {

subject = new Subject();

}

public void getStudentDetails(Scanner s) {

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

name =s.nextLine();

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

usn=s.nextLine();

}

public void computeSGPA() {

double totalCredits = 0;

double totalPoints = 0;

for (int j = 0; j < 8; j++) {

totalCredits += subject.credits[j];

totalPoints += subject.grades[j] \* subject.credits[j];

}

sgpa = totalPoints / totalCredits;

}

public void display() {

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

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

System.out.println("SGPA: " + sgpa);

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

Student[] students = new Student[3];

System.out.println("Name: Krithika H Kotian");

System.out.println("USN: 1BM23CS159");

for (int i = 0; i < 3; i++) {

System.out.println("Entering details for Student " + (i + 1) + ":");

students[i] = new Student();

students[i].getStudentDetails(sc);

students[i].subject.getMarksAndCredits(sc);

students[i].subject.computeGrades();

students[i].computeSGPA();

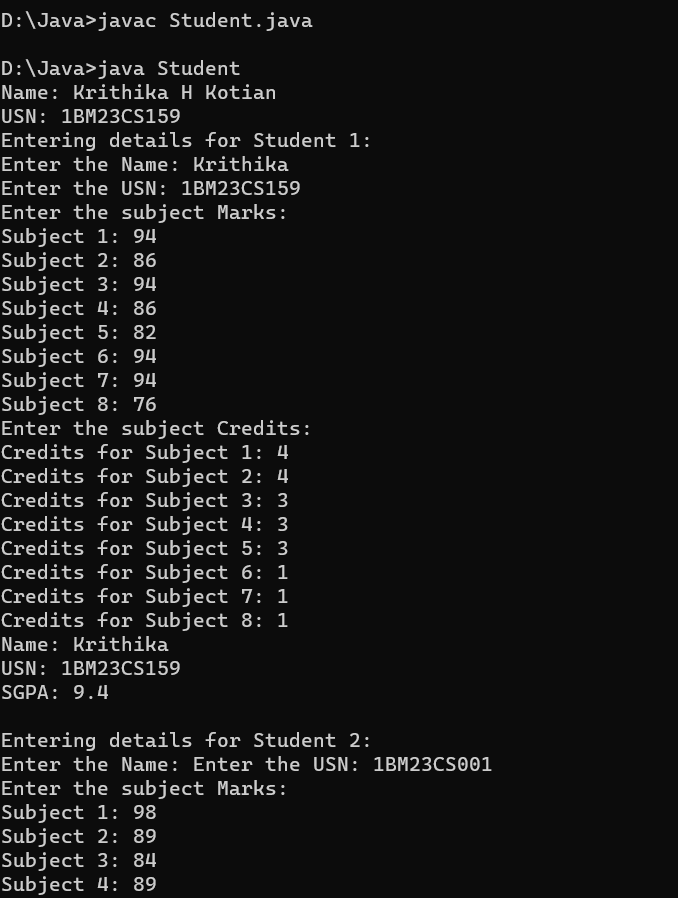
students[i].display();

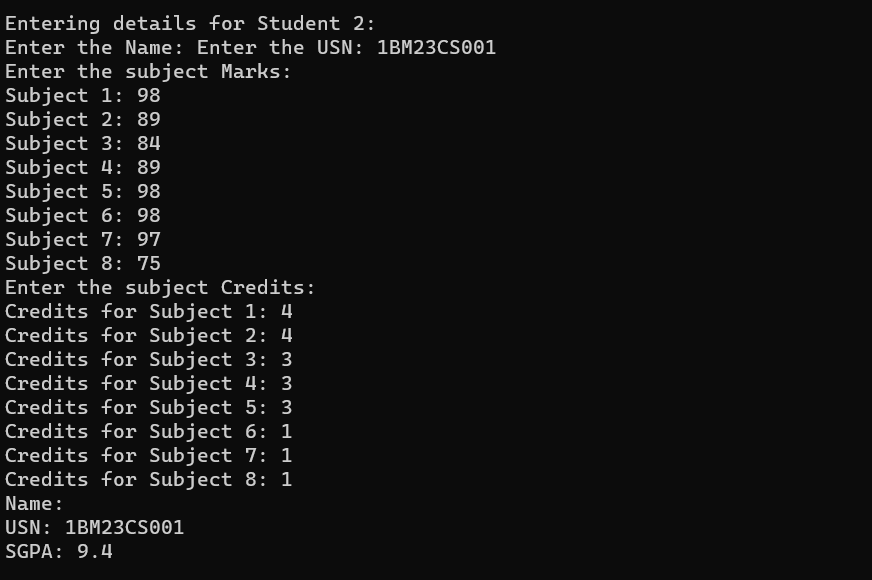
System.out.println();

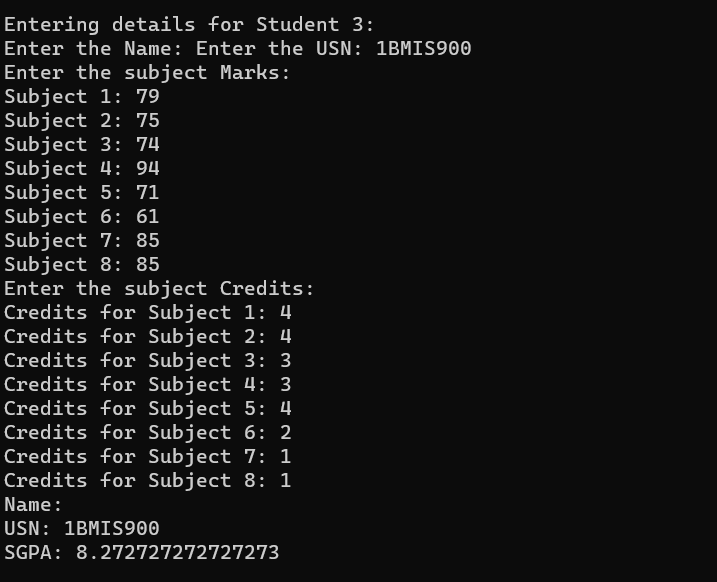
}

}

}



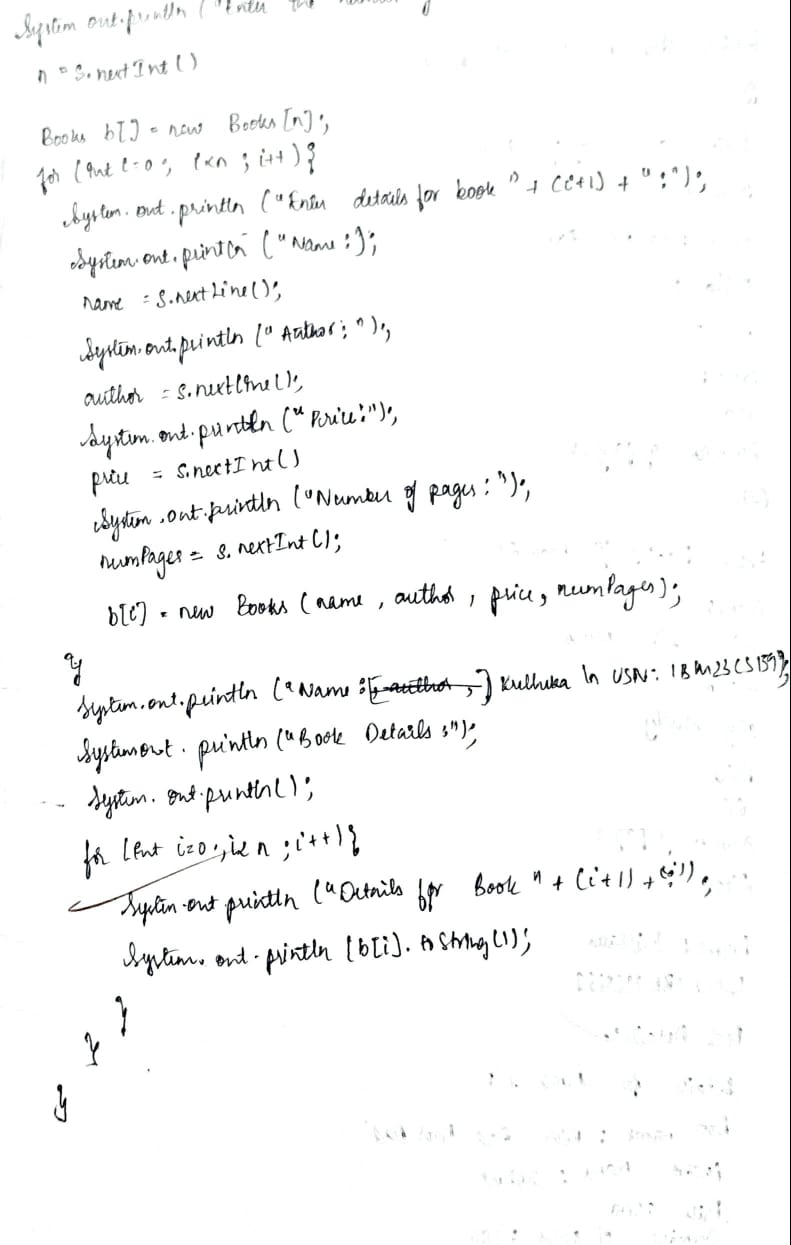
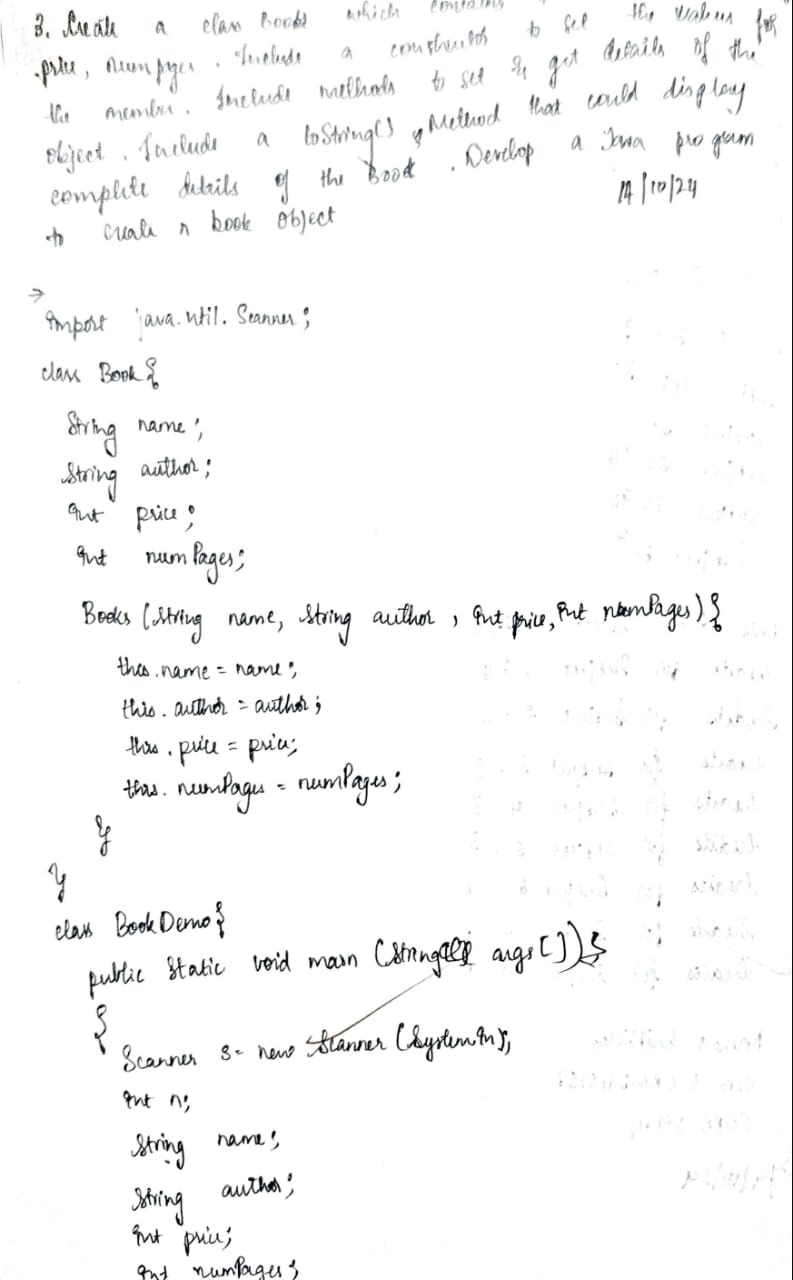


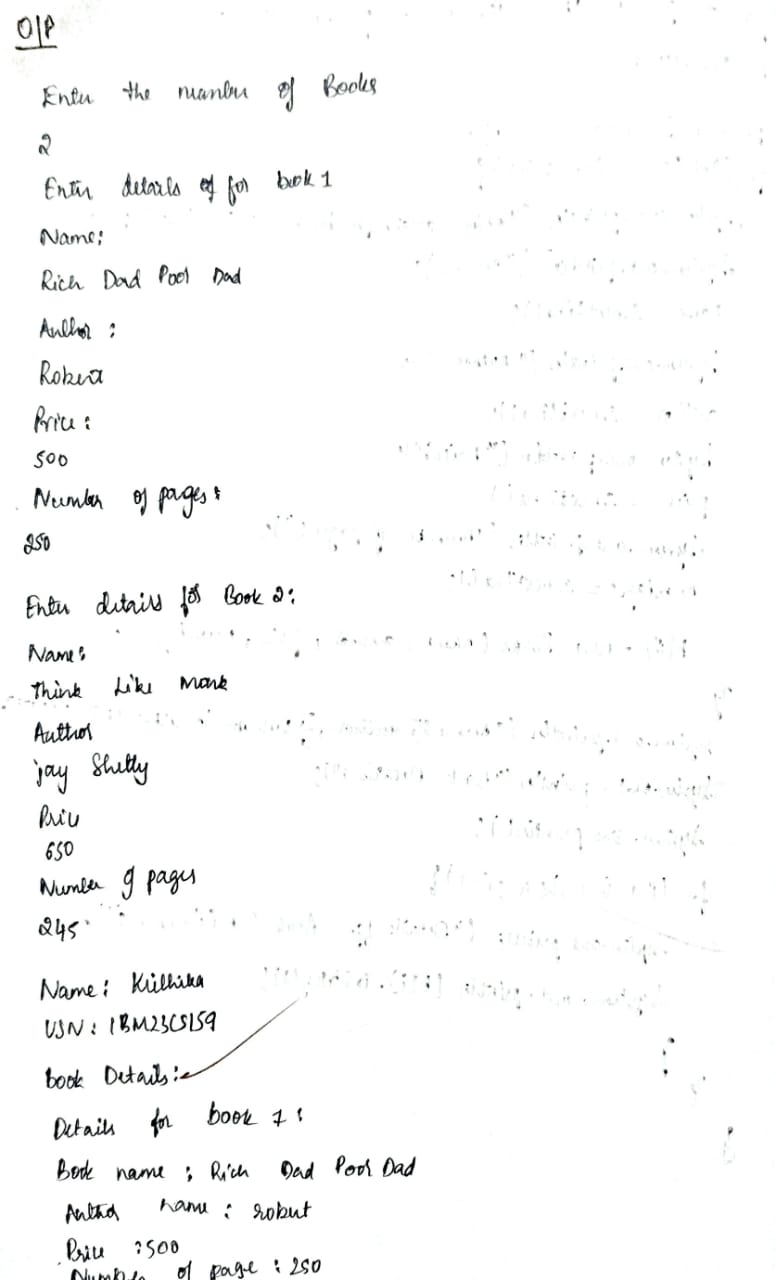
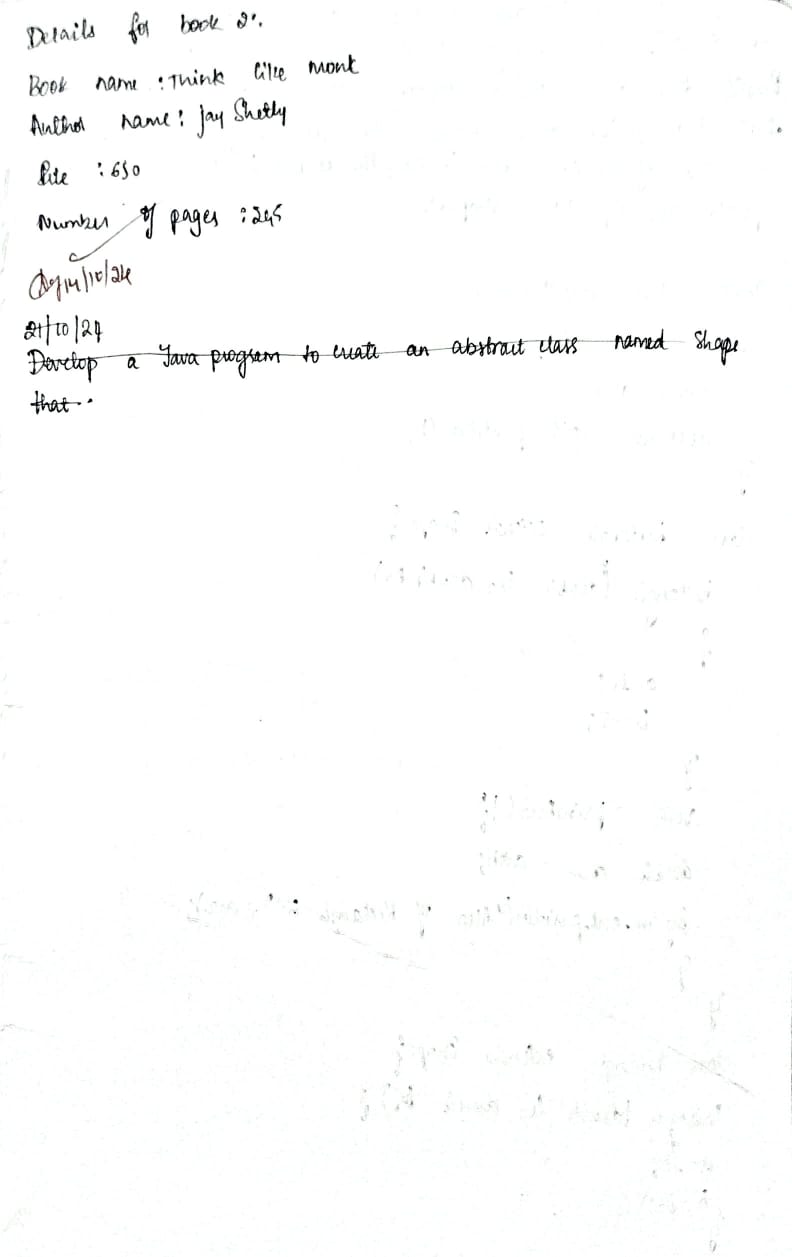


**Program 3**

**Implement ToString**

**Algorithm:**



**Code:**

import java.util.Scanner;

class Books {

String name;

String author;

int price;

int numPages;

Books(String name, String author, int price, int numPages) {

this.name = name;

this.author = author;

this.price = price;

this.numPages = numPages;

}

public String toString() {

return "Book name: " + this.name + "\n" +

"Author name: " + this.author + "\n" +

"Price: " + this.price + "\n" +

"Number of pages: " + this.numPages + "\n";

}

}

class BookDemo {

public static void main(String args[]) {

Scanner s = new Scanner(System.in);

int n;

String name;

String author;

int price;

int numPages;

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

n = s.nextInt();

s.nextLine();

Books b[] = new Books[n];

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

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

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

name = s.nextLine();

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

author = s.nextLine();

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

price = s.nextInt();

System.out.println("Number of pages:");

numPages = s.nextInt();

s.nextLine();

b[i] = new Books(name, author, price, numPages);

}

System.out.println("Name: Krithika\nUSN:1BM23CS159");

System.out.println("Book Details:");

System.out.println();

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

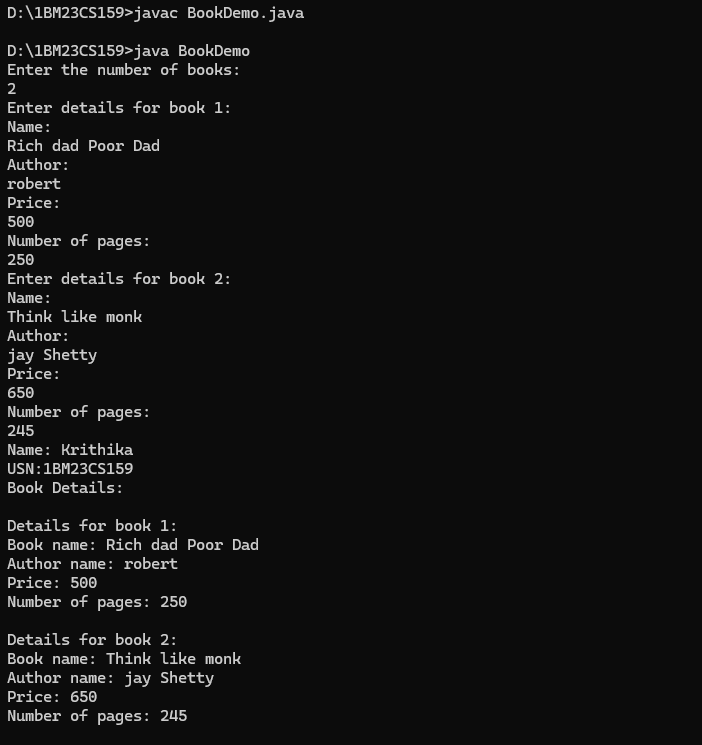
System.out.println("Details for book " + (i + 1) + ":");

System.out.println(b[i].toString());

}

}

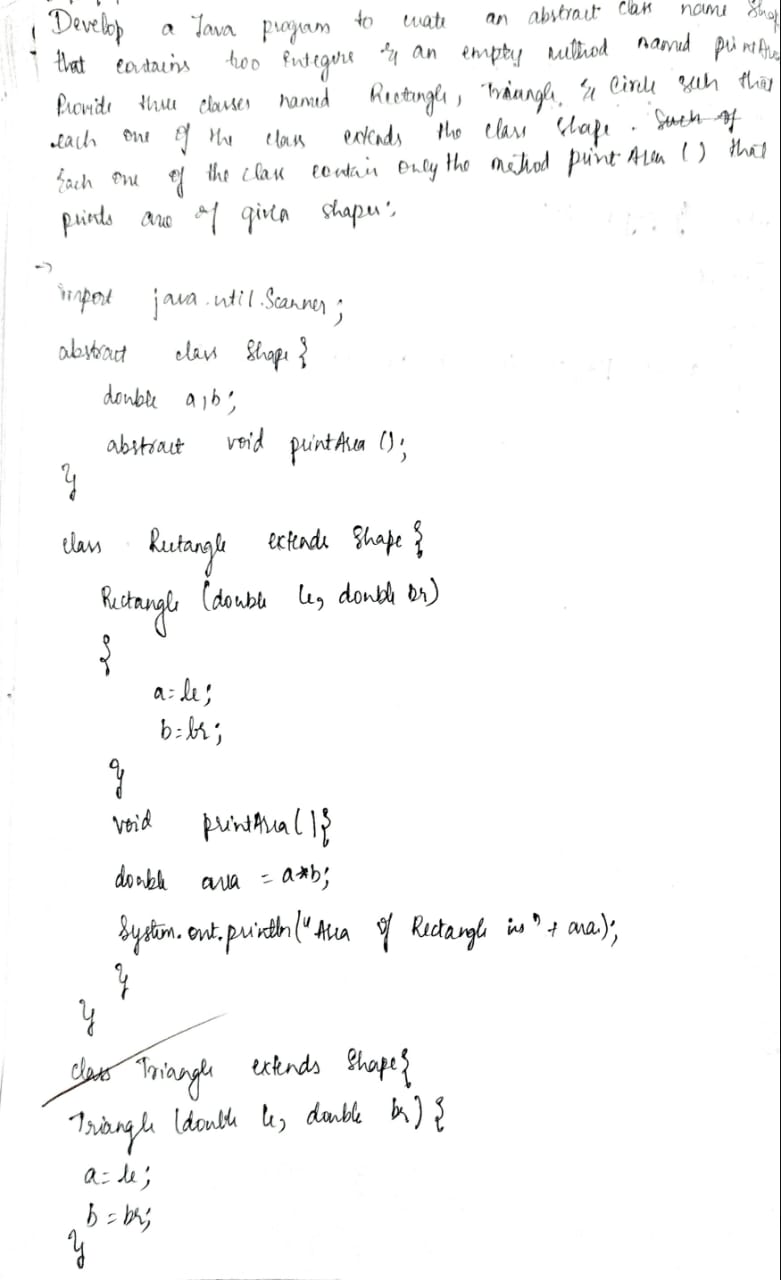
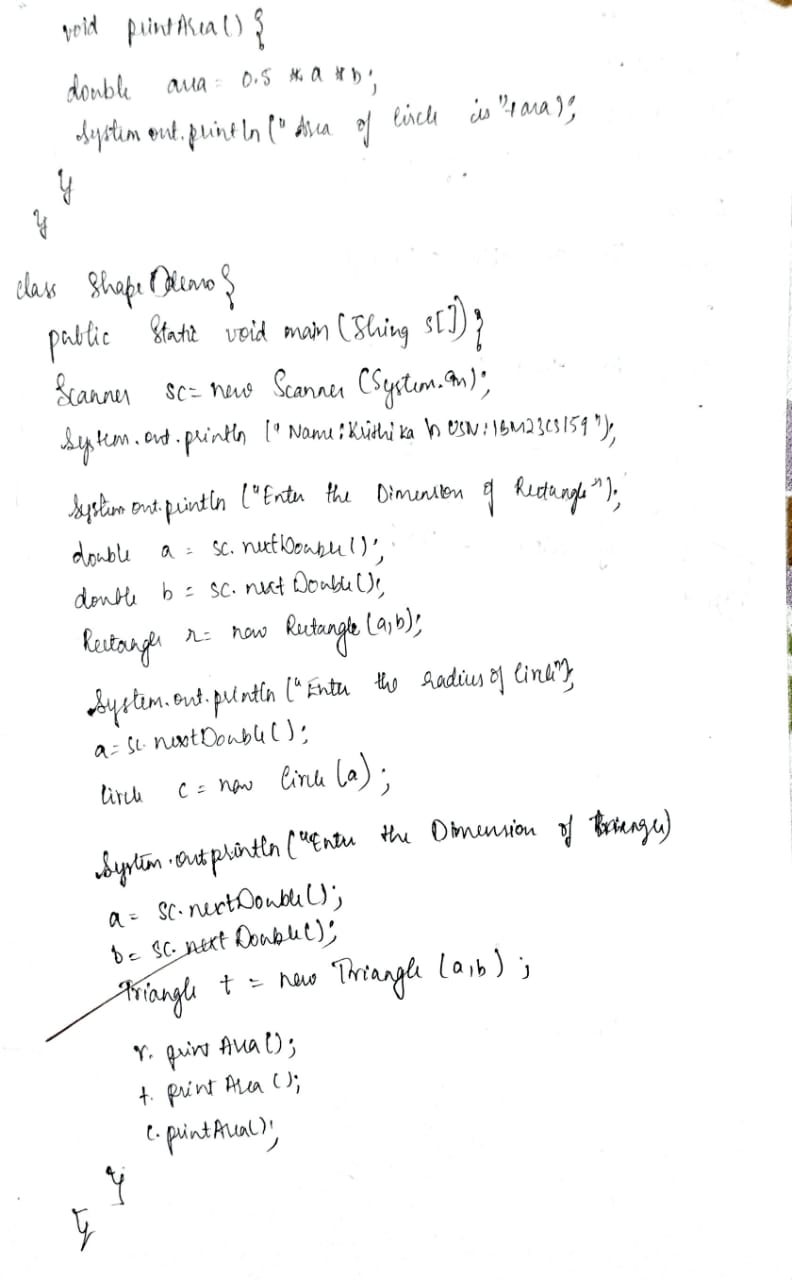
}

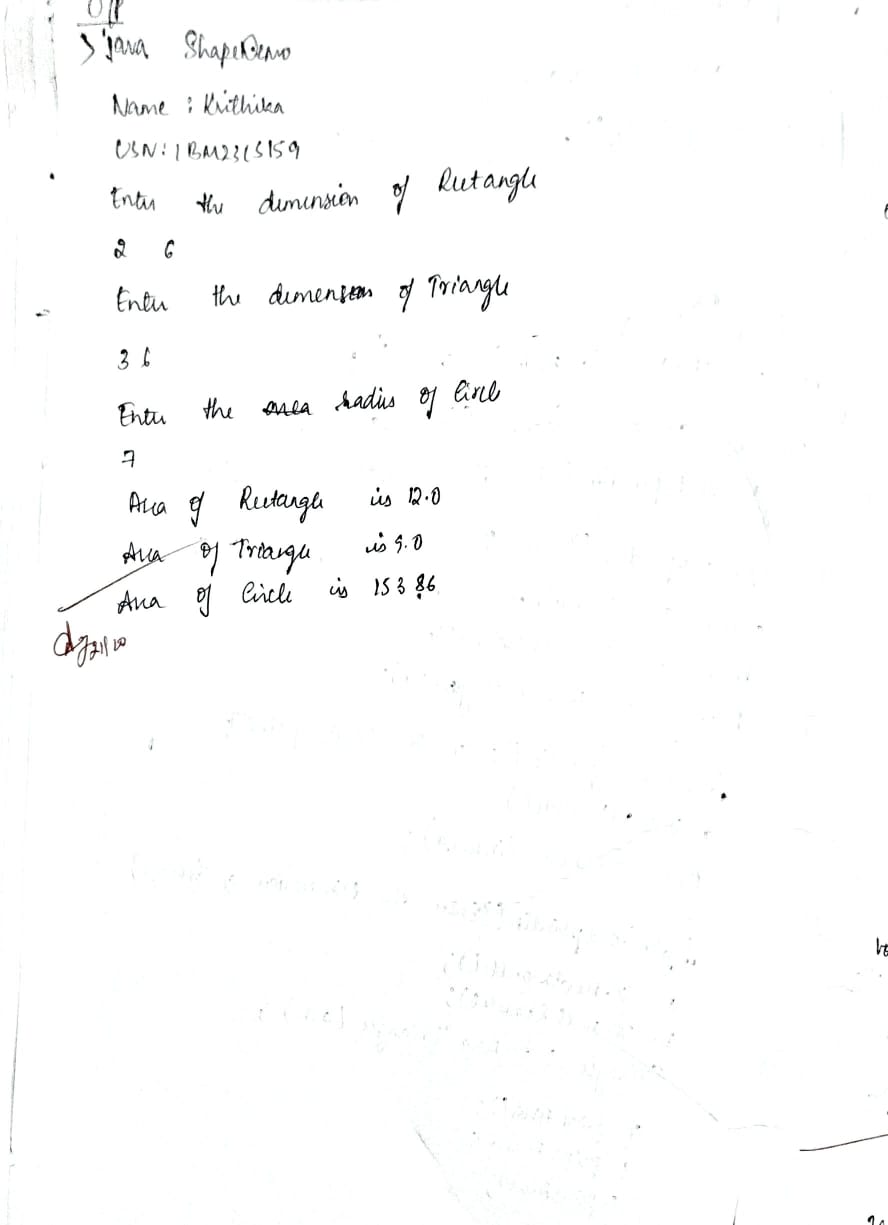


**Program 4**

**Implement Shape Area**

Algorithm:





**Code:**

import java.util.Scanner;

abstract class Shape {

double a, b;

abstract void printArea();

}

class Rectangle extends Shape {

Rectangle(double le, double br) {

a = le;

b = br;

}

void printArea() {

double area = a \* b;

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

}

}

class Triangle extends Shape {

Triangle(double le, double br) {

a = le;

b = br;

}

void printArea() {

double area = 0.5 \* a \* b;

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

}

}

class Circle extends Shape {

Circle(double le) {

a = le;

}

void printArea() {

double area = 3.14 \* a \* a;

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

}

}

class ShapeDemo {

public static void main(String s[]) {

Scanner sc = new Scanner(System.in);

System.out.println("Name:Krithika\nUSN:1BM23CS159");

System.out.println("Enter the dimensions of Rectangle");

double a = sc.nextDouble();

double b = sc.nextDouble();

Rectangle r = new Rectangle(a, b);

System.out.println("Enter the dimensions of Triangle");

a = sc.nextDouble();

b = sc.nextDouble();

Triangle t = new Triangle(a, b);

System.out.println("Enter the radius of Circle");

a = sc.nextDouble();

Circle c = new Circle(a);

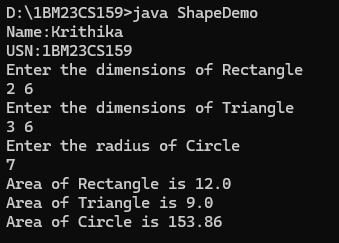
r.printArea();

t.printArea();

c.printArea();

}

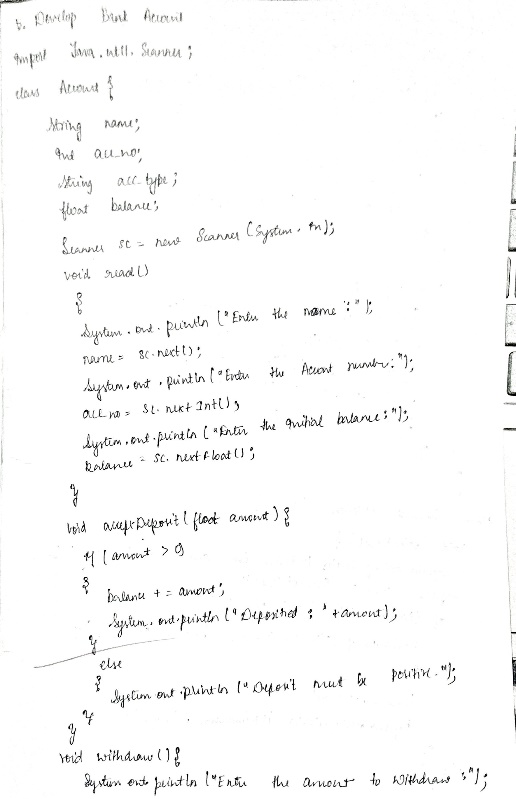
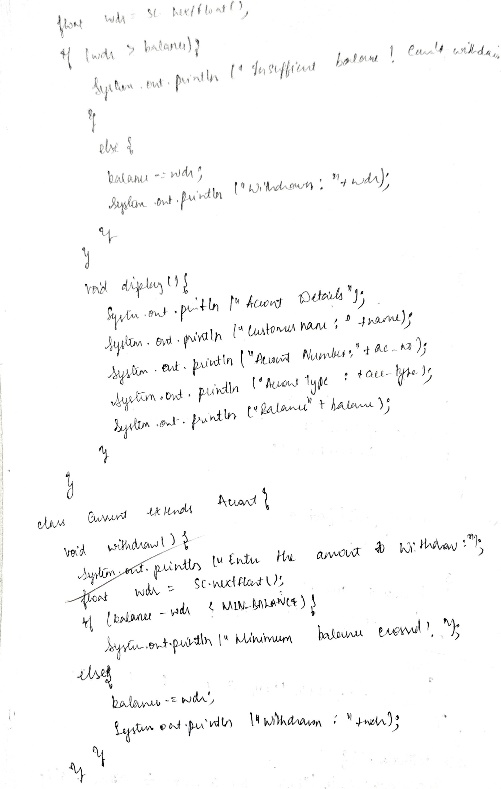
}

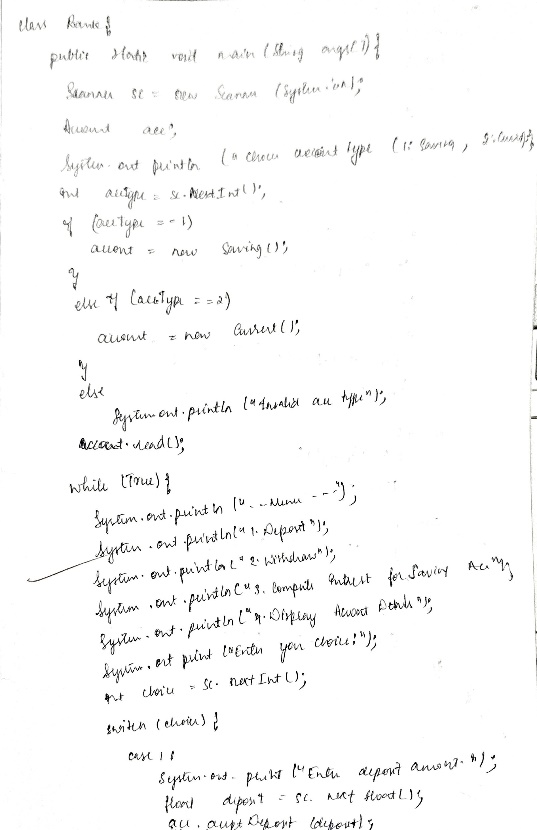
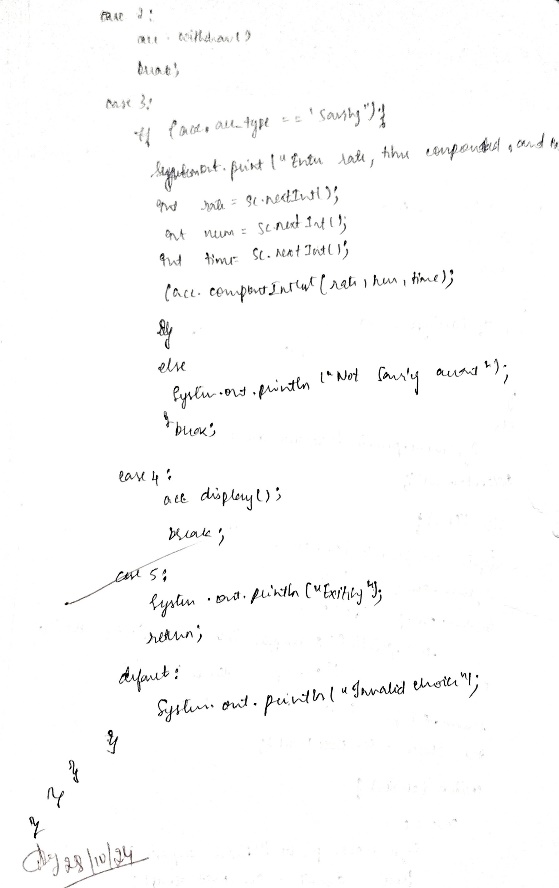


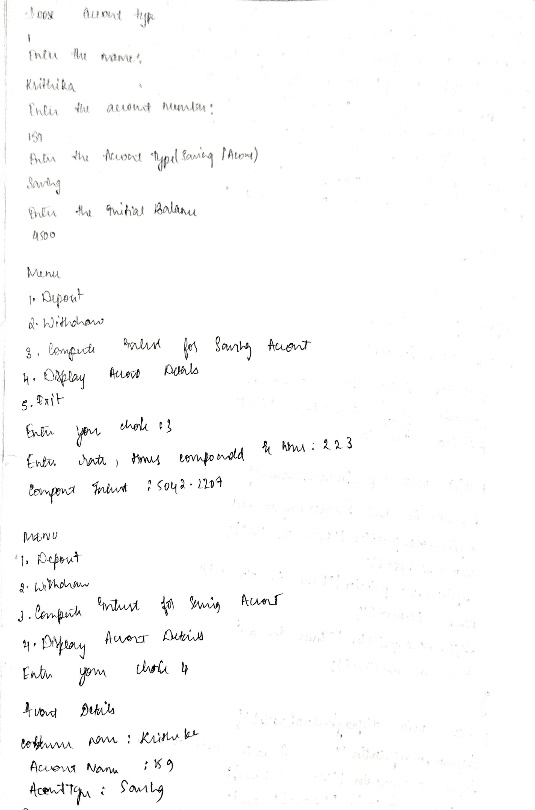
**Program 5**

**Implement Saving and Current Account**

**Algorithm:**

****

****

****

**Code:**

import java.util.Scanner;

class Account {

String name;

int acc\_no;

String acc\_type;

float balance;

Scanner sc = new Scanner(System.in);

void read() {

System.out.println("Enter the name:");

name = sc.next();

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

acc\_no = sc.nextInt();

System.out.println("Enter the Account Type (Saving/Current):");

acc\_type = sc.next();

System.out.println("Enter the initial balance:");

balance = sc.nextFloat();

}

void acceptDeposit(float amount) {

if (amount > 0) {

balance += amount;

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

} else {

System.out.println("Deposit amount must be positive.");

}

}

void withdraw() {

System.out.println("Enter the amount to Withdraw:");

float wdr = sc.nextFloat();

if (wdr > balance) {

System.out.println("Insufficient balance! Can't withdraw.");

} else {

balance -= wdr;

System.out.println("Withdrawn: " + wdr);

}

}

void display() {

System.out.println("Account Details");

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

System.out.println("Account Number: " + acc\_no);

System.out.println("Account Type: " + acc\_type);

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

}

}

class Saving extends Account {

void compoundInterest(int rate, int num, int time) {

float com\_int = balance \* (float) Math.pow(1 + (rate\*0.01 / (float) num), num \* time);

System.out.println("Compound Interest: " + com\_int);

balance += com\_int;

}

}

class Current extends Account {

private static final float MIN\_BALANCE = 500;

void withdraw() {

System.out.println("Enter the amount to Withdraw:");

float wdr = sc.nextFloat();

if (balance - wdr < MIN\_BALANCE) {

System.out.println("Minimum balance crossed! Can't withdraw.");

} else {

balance -= wdr;

System.out.println("Withdrawn: " + wdr);

}

}

void display() {

super.display();

if (balance < MIN\_BALANCE) {

System.out.println("Minimum balance crossed!");

balance -= 50;

}

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

}

}

class Bank {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

Account account;

System.out.println("Choose account type (1: Saving, 2: Current): ");

int accType = sc.nextInt();

if (accType == 1) {

account = new Saving();

} else if (accType == 2) {

account = new Current();

} else {

System.out.println("Invalid account type selected. Exiting...");

return;

}

account.read();

while (true) {

System.out.println("----Menu----");

System.out.println("1. Deposit");

System.out.println("2. Withdraw");

System.out.println("3. Compute interest for Saving Account");

System.out.println("4. Display Account Details");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

int choice = sc.nextInt();

switch (choice) {

case 1:

System.out.print("Enter deposit amount: ");

float deposit = sc.nextFloat();

account.acceptDeposit(deposit);

break;

case 2:

account.withdraw();

break;

case 3:

if (account instanceof Saving) {

System.out.print("Enter rate, times compounded, and time: ");

int rate = sc.nextInt();

int num = sc.nextInt();

int time = sc.nextInt();

((Saving) account).compoundInterest(rate, num, time);

} else {

System.out.println("Not a saving account.");

}

break;

case 4:

account.display();

break;

case 5:

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

return;

default:

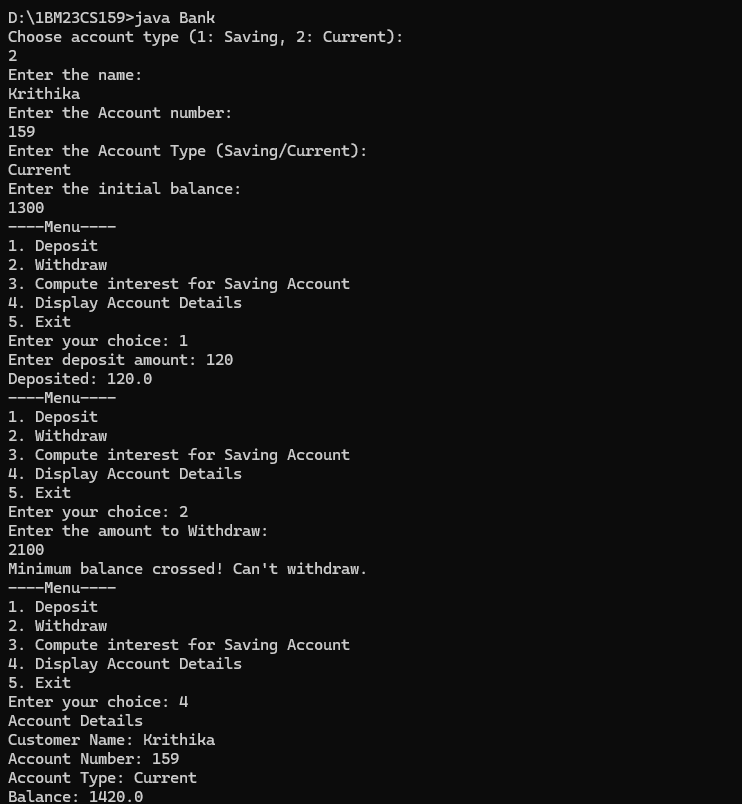
System.out.println("Invalid choice.");

}

}

}

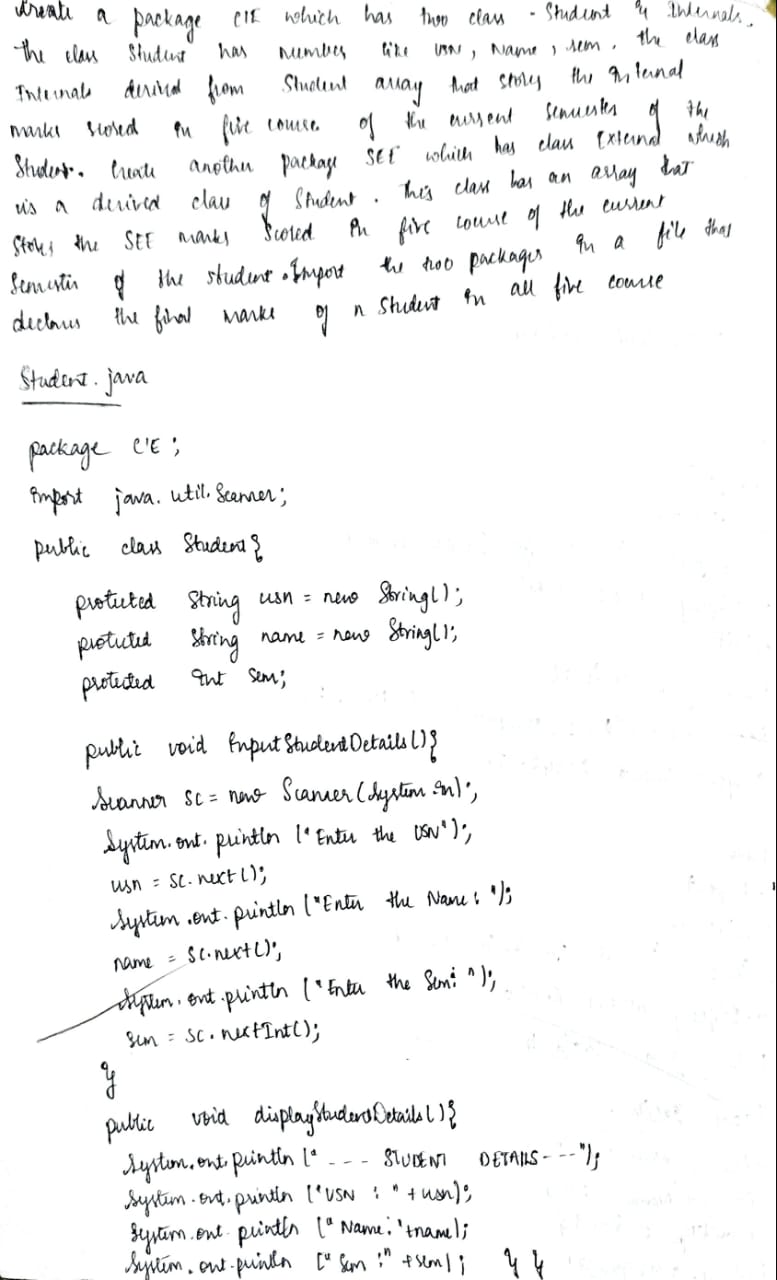
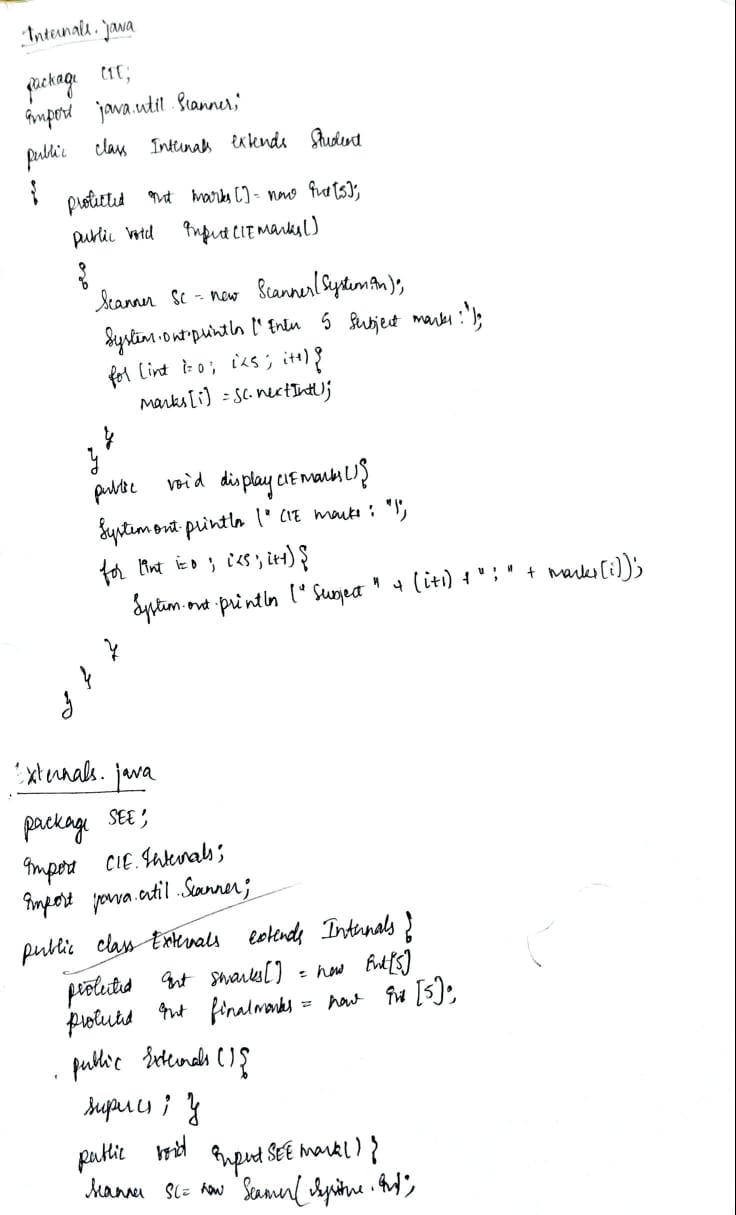
**}**

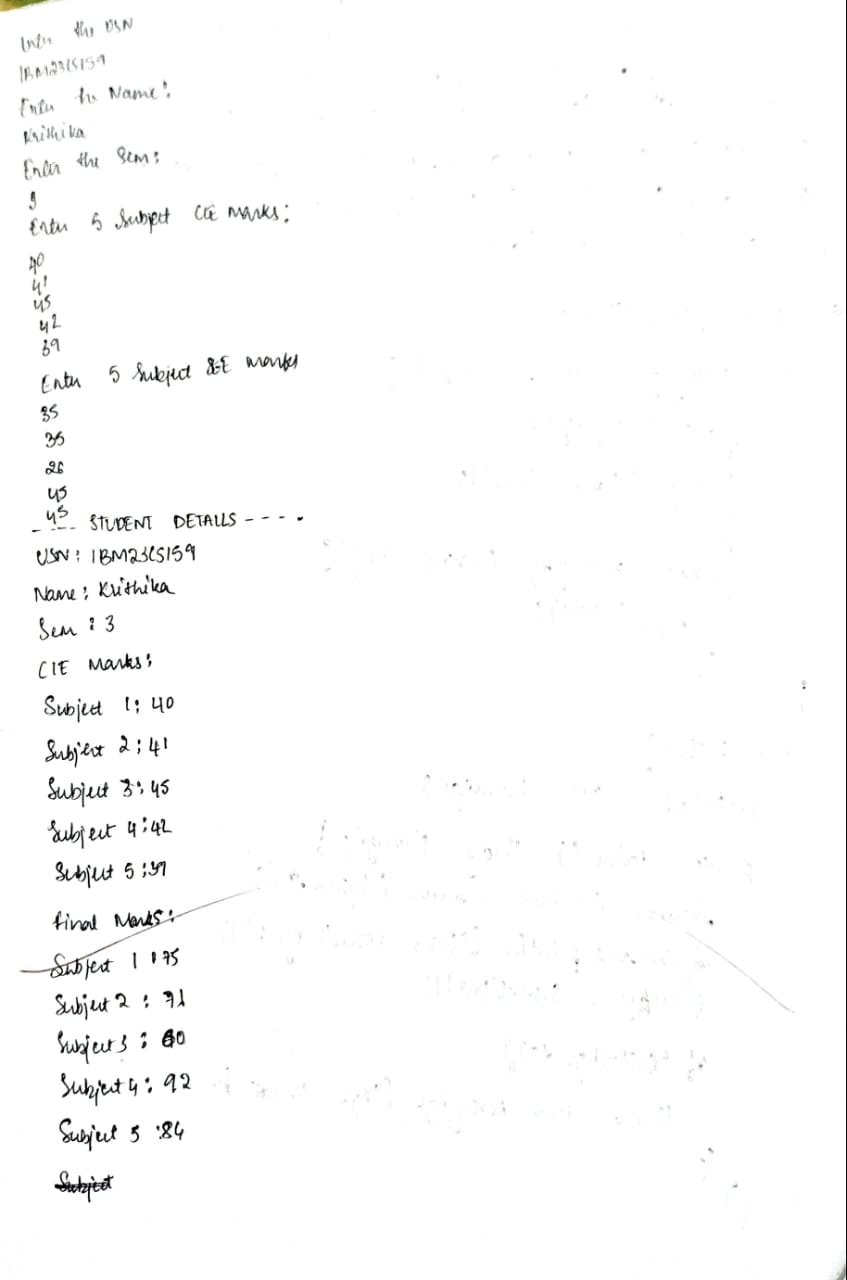
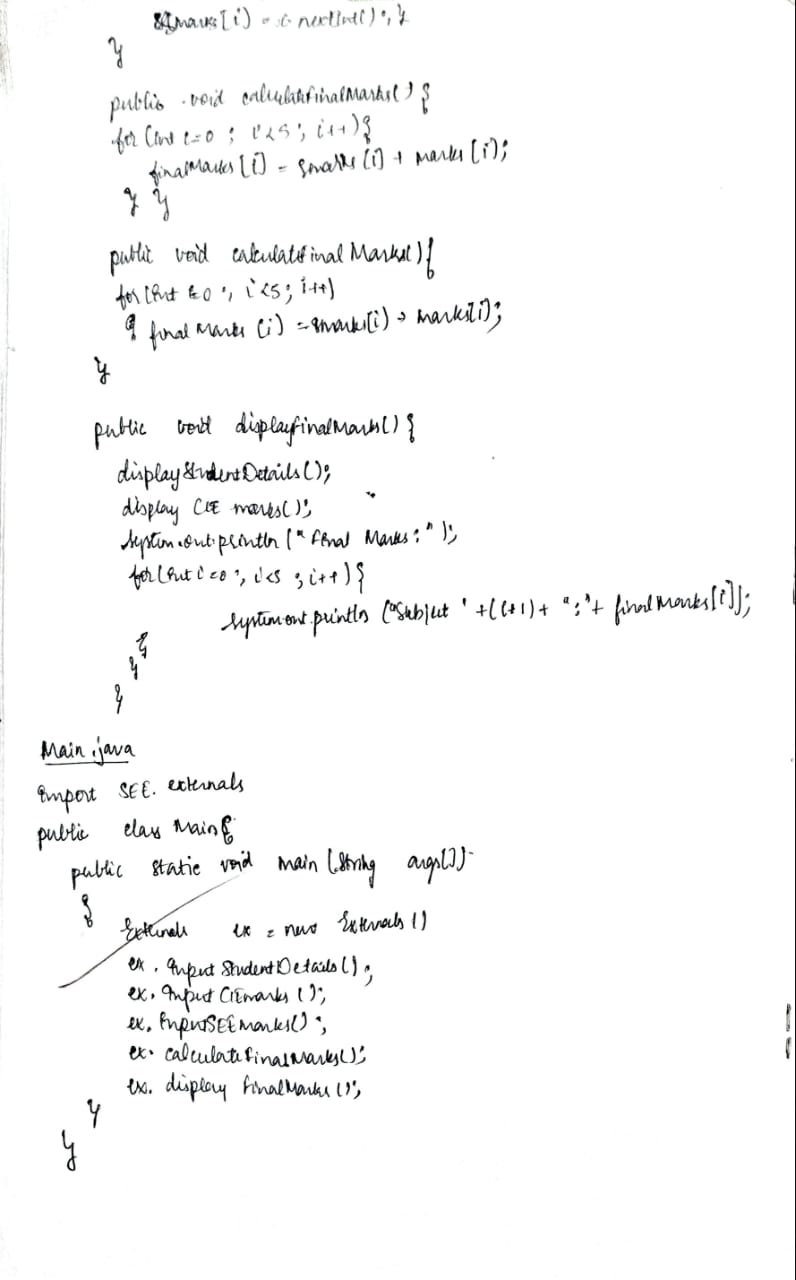


**Program 6**

**Implement Package**

**Algorithm:**





**Code:**

**Internals.java**

package CIE;

import java.util.Scanner;

public class Internals extends Student {

protected int marks[] = new int[5];

public void inputCIEmarks() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter 5 Subject CIE marks: ");

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

marks[i] = sc.nextInt();

}

}

public void displayCIEmarks() {

System.out.println("CIE Marks: ");

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

System.out.println("Subject " + (i + 1) + ": " + marks[i]);

}

}

}

**Externals.java**

package SEE;

import CIE.Internals;

import java.util.Scanner;

public class Externals extends Internals {

protected int smarks[] = new int[5]; // SEE marks

protected int finalMarks[] = new int[5]; // Final marks (CIE + SEE)

public Externals() {

super();

}

public void inputSEEmarks() {

Scanner sc = new Scanner(System.in);

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

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

smarks[i] = sc.nextInt(); // Store each subject's SEE marks

}

}

public void calculateFinalMarks() {

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

finalMarks[i] = smarks[i] + marks[i]; // Final marks = CIE marks + SEE marks

}

}

public void displayFinalMarks() {

displayStudentDetails();

displayCIEmarks();

System.out.println("Final Marks: ");

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

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

}

}

}

**Student.java**

package CIE;

import java.util.Scanner;

public class Student {

protected String usn = new String();

protected String name = new String();

protected int sem;

public void inputStudentDetails() {

Scanner sc = new Scanner(System.in);

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

usn = sc.next(); // Read the USN

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

name = sc.next(); // Read the Name

System.out.println("Enter the Sem: ");

sem = sc.nextInt(); // Read the Semester (as an integer)

}

public void displayStudentDetails() {

System.out.println("----STUDENT DETAILS----");

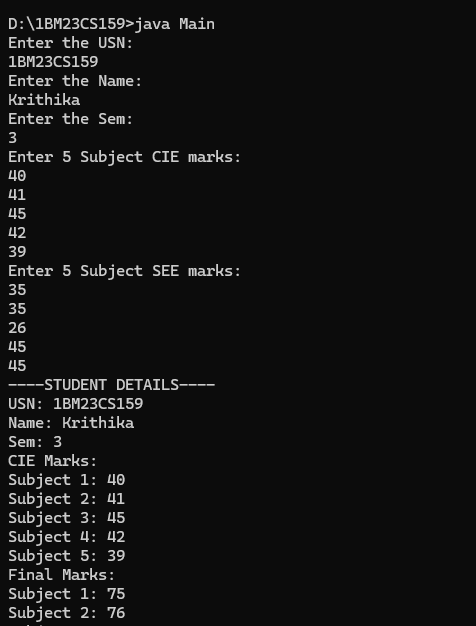
System.out.println("USN: " + usn);

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

System.out.println("Sem: " + sem);

}

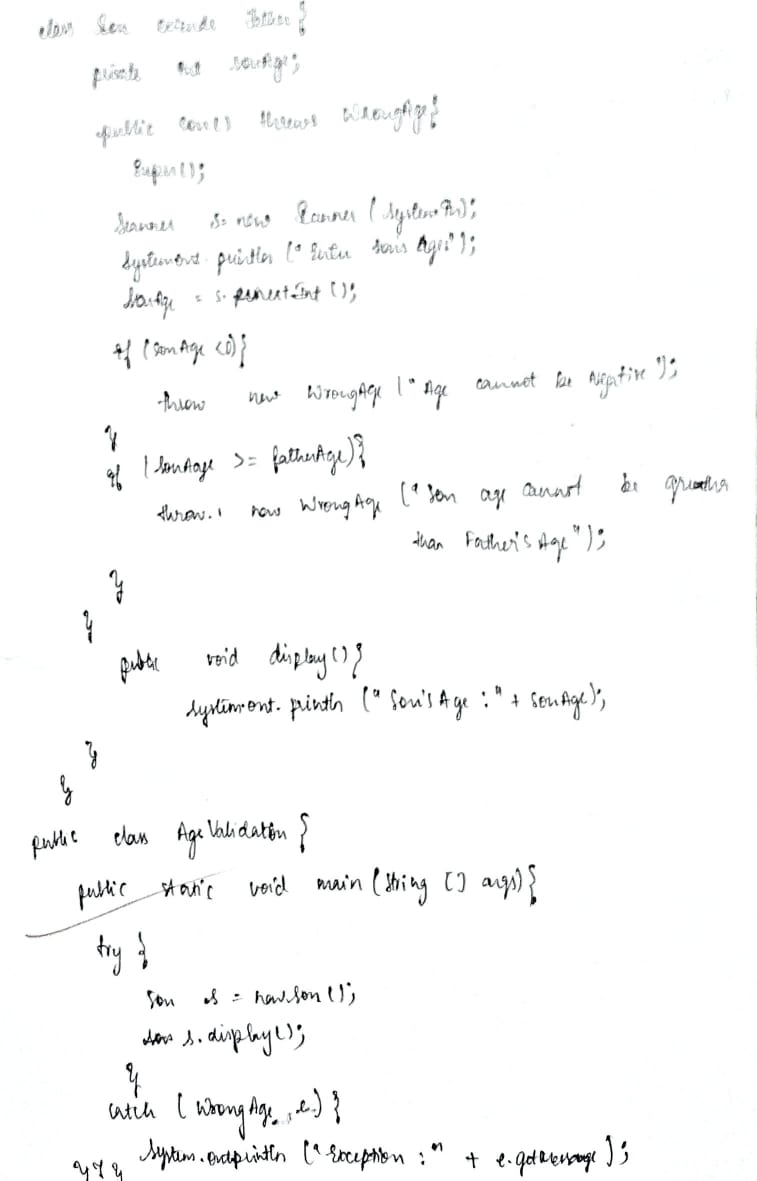
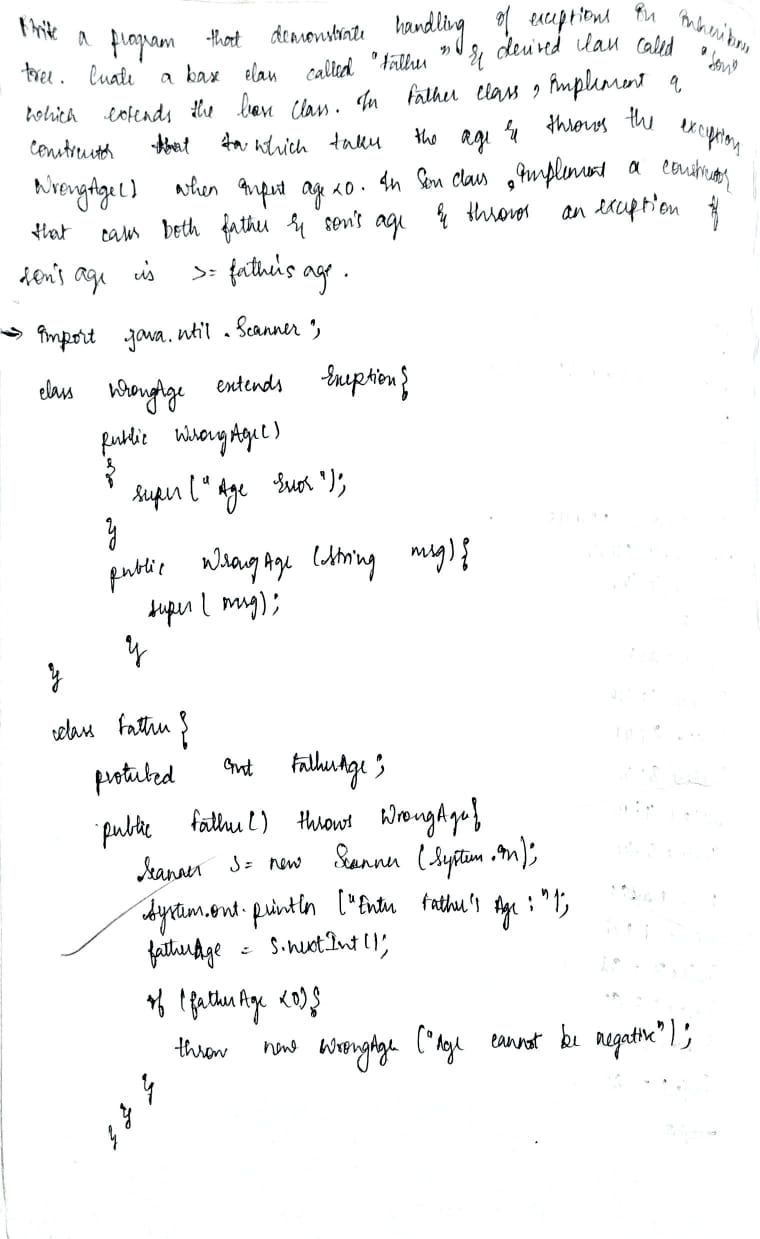
}

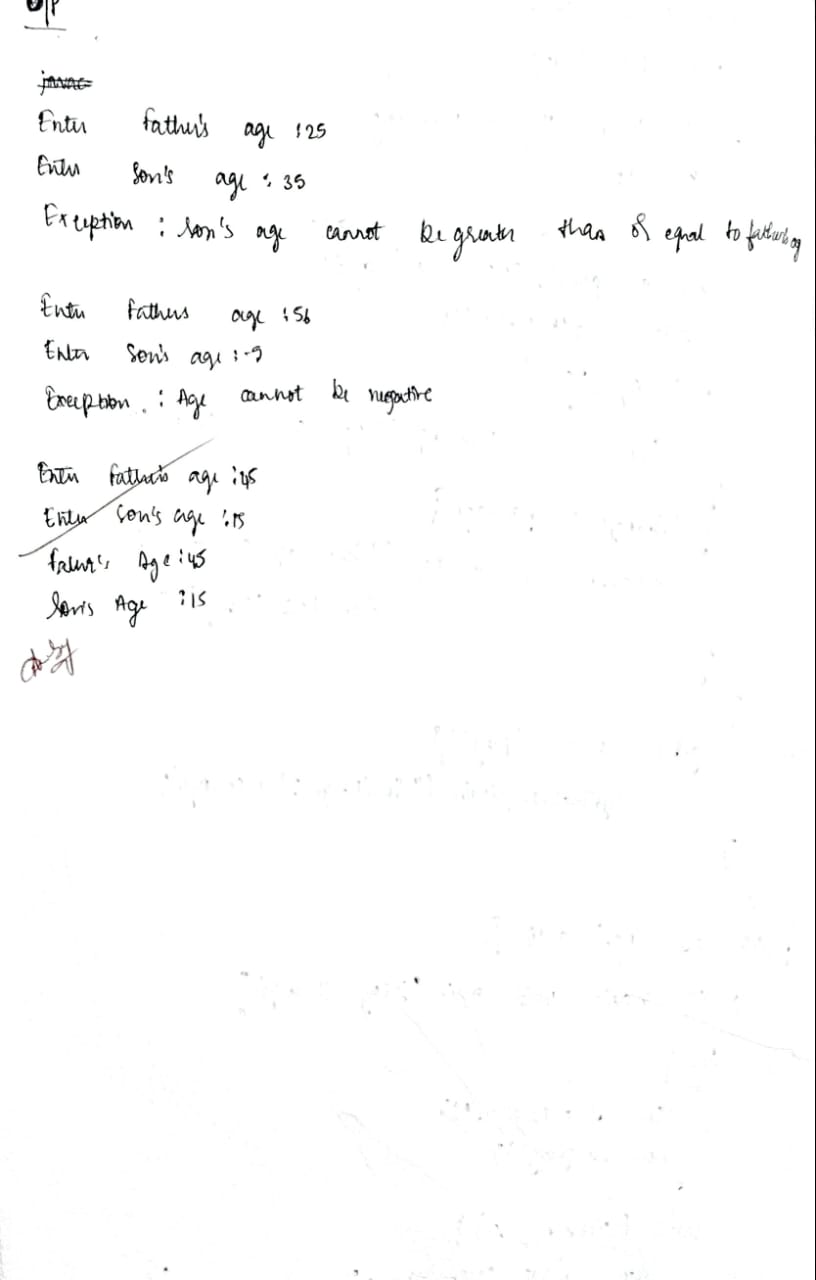


**Program 7**

**Implement Age Exception**

**Algorithm:**





**Code:**

import java.util.Scanner;

class WrongAge extends Exception {

public WrongAge() {

super("Age Error");

}

public WrongAge(String message) {

super(message);

}

}

class Father {

protected int fatherAge;

public Father() throws WrongAge {

Scanner s = new Scanner(System.in);

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

fatherAge = s.nextInt();

if (fatherAge < 0) {

throw new WrongAge("Age cannot be negative");

}

}

public void display() {

System.out.println("Father's Age: " + fatherAge);

}

}

class Son extends Father {

private int sonAge;

public Son() throws WrongAge {

super();

Scanner s = new Scanner(System.in);

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

sonAge = s.nextInt();

if (sonAge < 0) {

throw new WrongAge("Age cannot be negative");

} else if (sonAge >= fatherAge) {

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

}

}

public void display() {

super.display();

System.out.println("Son's Age: " + sonAge);

}

}

public class AgeValid {

public static void main(String[] args) {

try {

System.Out.println("Krithika H Kotian\n1BM23CS159");

Son son = new Son();

son.display();

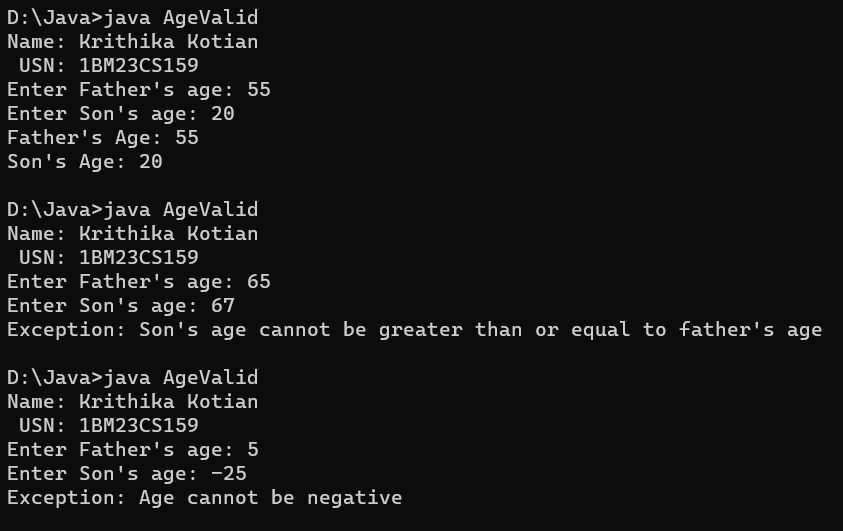
} catch (WrongAge e) {

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

}

}

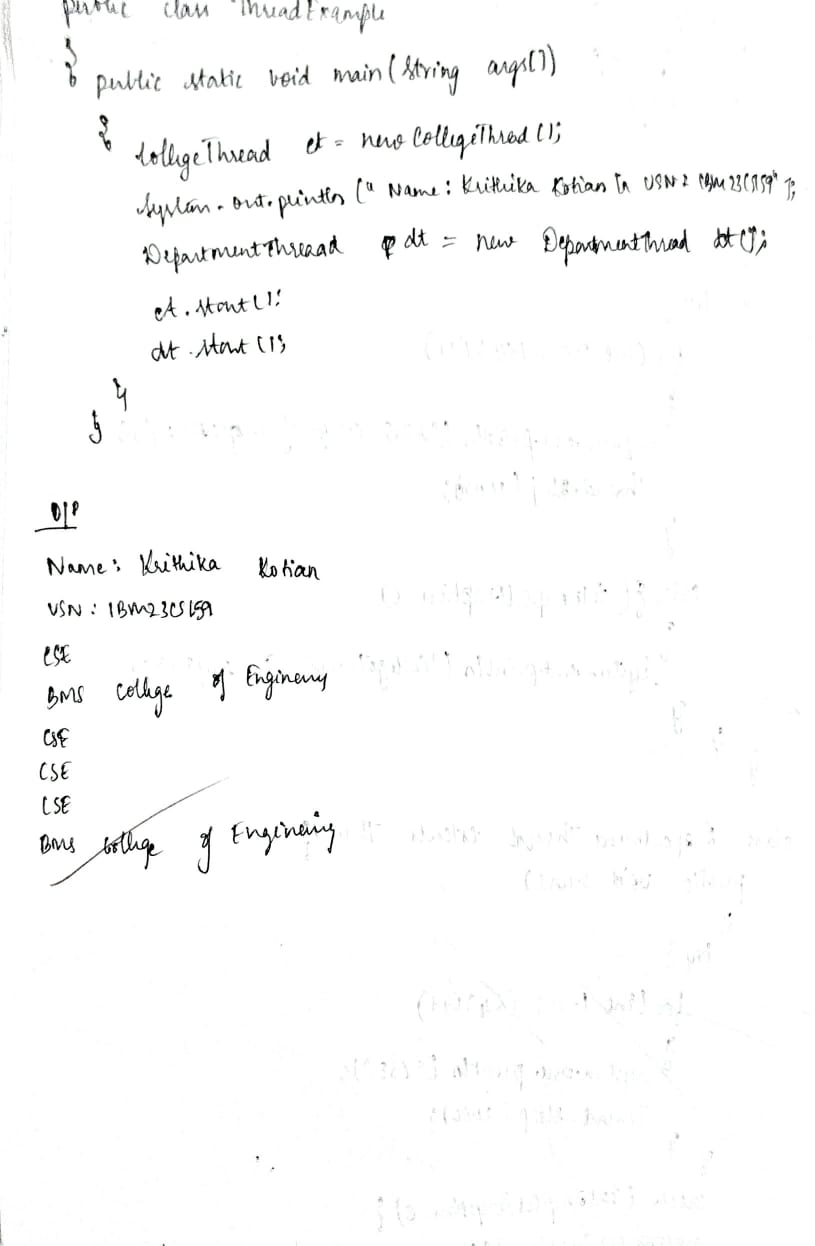
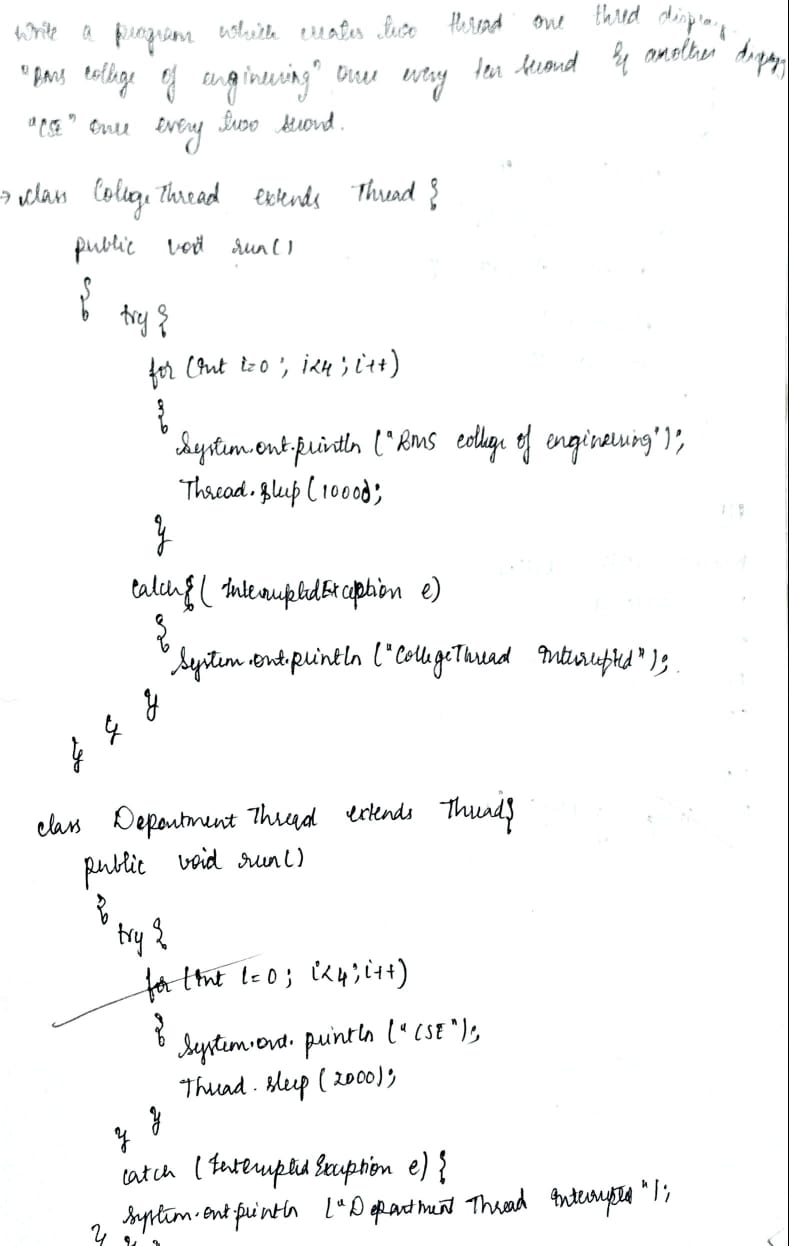
}



**Program 8**

**Implement Threads**

**Algorithm:**



**Code:**

class CollegeThread extends Thread {

public void run() {

try {

for(int i=0;i<4;i++ ){

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

Thread.sleep(10000); // Pause for 10 seconds

}

} catch (InterruptedException e) {

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

}

}

}

class DepartmentThread extends Thread {

public void run() {

try {

for(int i=0;i<4;i++){

System.out.println("CSE");

Thread.sleep(2000); // Pause for 2 seconds

}

} catch (InterruptedException e) {

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

}

}

}

public class ThreadExample {

public static void main(String[] args) {

CollegeThread collegeThread = new CollegeThread();

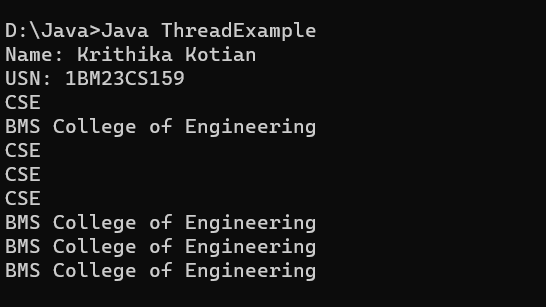
System.out.println("Name: Krithika Kotian\nUSN: 1BM23CS159");

DepartmentThread departmentThread = new DepartmentThread();

collegeThread.start();

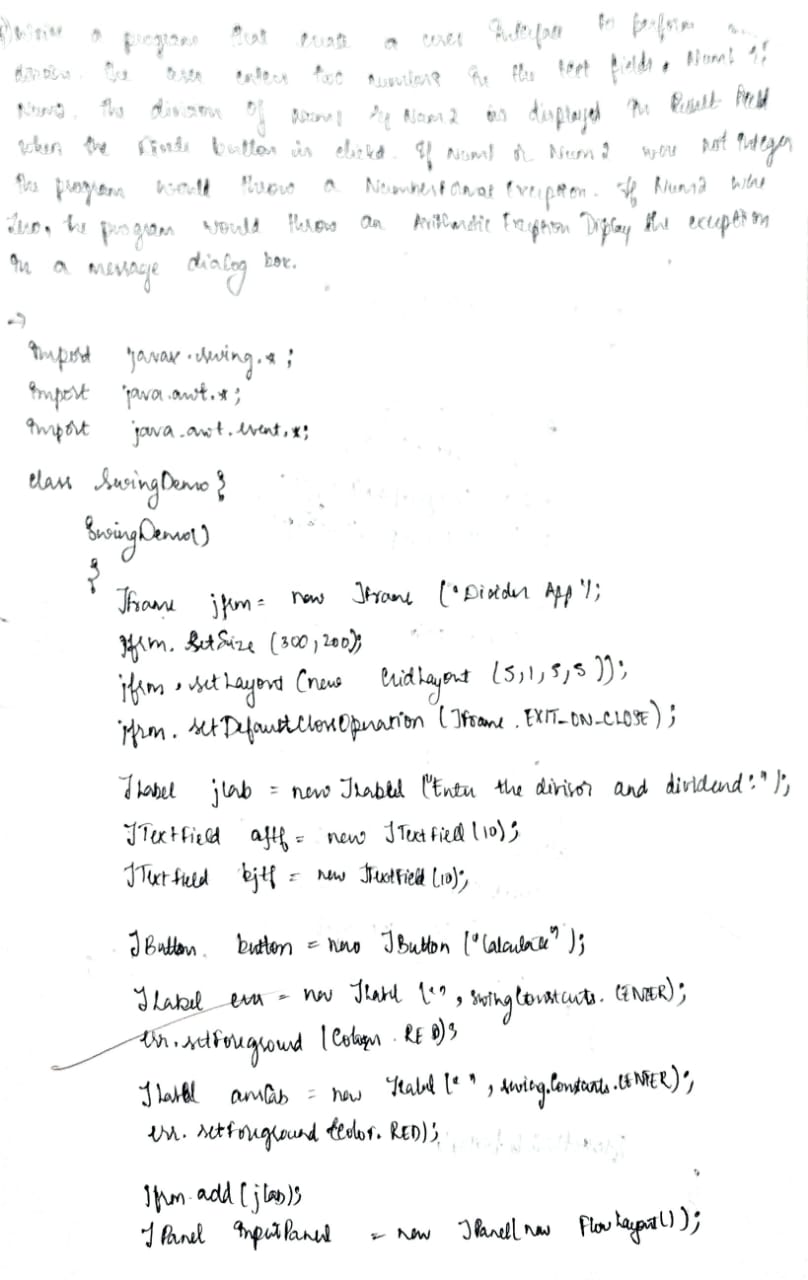
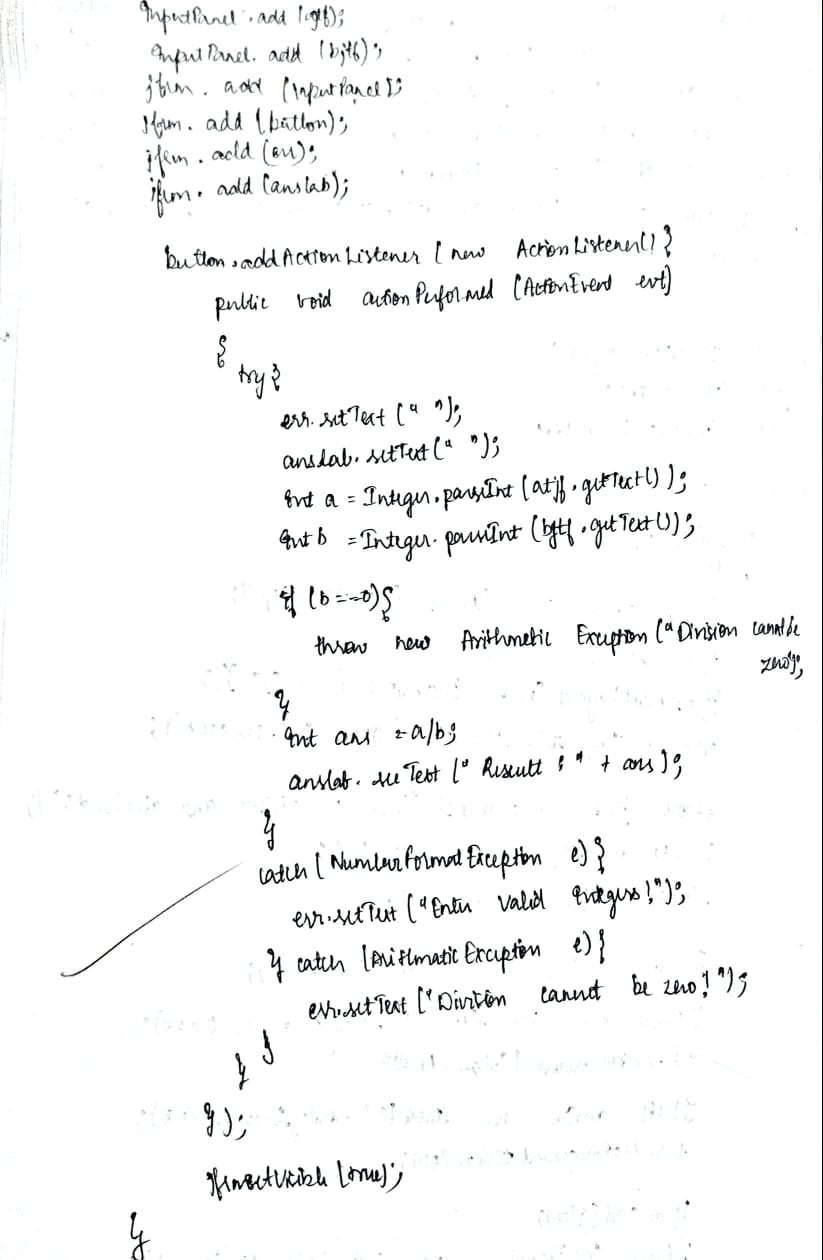
departmentThread.start();

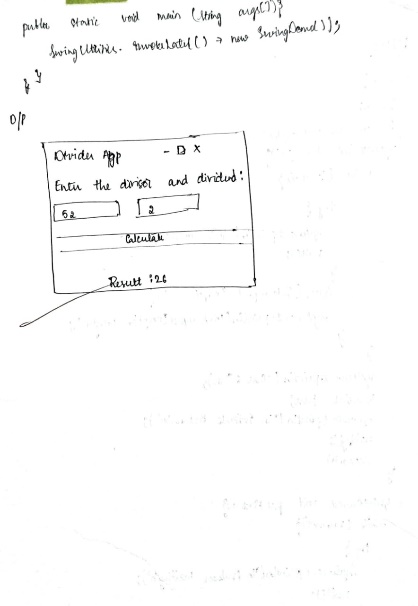
}

}

**Program 9**

**Implement Swing demo**

Algorithm



**Code:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

class SwingDemo {

SwingDemo() {

// Create JFrame container

JFrame jfrm = new JFrame("Divider App");

jfrm.setSize(300, 200);

jfrm.setLayout(new GridLayout(5, 1, 5, 5)); // GridLayout with spacing

jfrm.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// Create labels and text fields

JLabel jlab = new JLabel("Enter the divisor and dividend:");

JTextField ajtf = new JTextField(10);

JTextField bjtf = new JTextField(10);

JButton button = new JButton("Calculate");

JLabel err = new JLabel("", SwingConstants.CENTER);

err.setForeground(Color.RED); // Highlight error messages in red

JLabel anslab = new JLabel("", SwingConstants.CENTER);

// Add components to the frame

jfrm.add(jlab);

JPanel inputPanel = new JPanel(new FlowLayout());

inputPanel.add(ajtf);

inputPanel.add(bjtf);

jfrm.add(inputPanel);

jfrm.add(button);

jfrm.add(err);

jfrm.add(anslab);

// Add ActionListener to the button

button.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent evt) {

try {

err.setText(""); // Clear previous errors

anslab.setText(""); // Clear previous results

int a = Integer.parseInt(ajtf.getText());

int b = Integer.parseInt(bjtf.getText());

if (b == 0) {

throw new ArithmeticException("Divisor cannot be zero");

}

int ans = a / b;

anslab.setText("Result: " + ans);

} catch (NumberFormatException e) {

err.setText("Enter valid integers!");

} catch (ArithmeticException e) {

err.setText("Divisor cannot be zero!");

}

}

});

// Display frame

jfrm.setVisible(true);

}

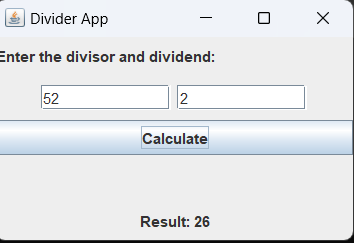
public static void main(String[] args) {

// Create frame on event dispatching thread

SwingUtilities.invokeLater(() -> new SwingDemo());

}

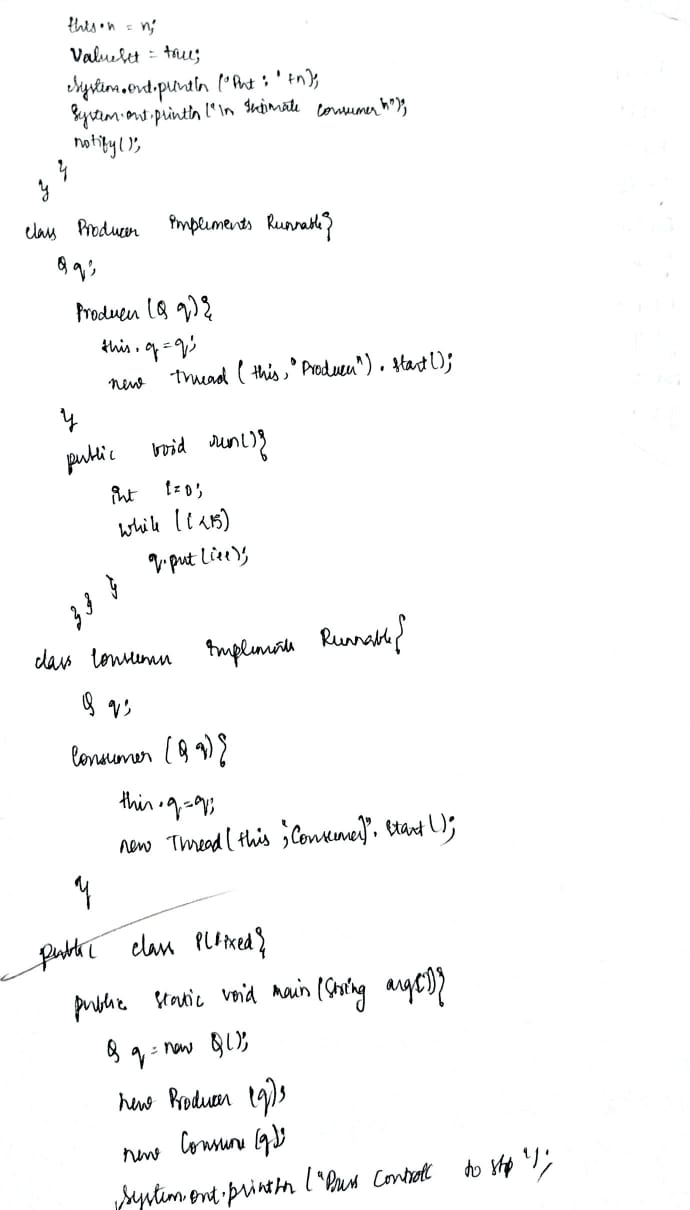
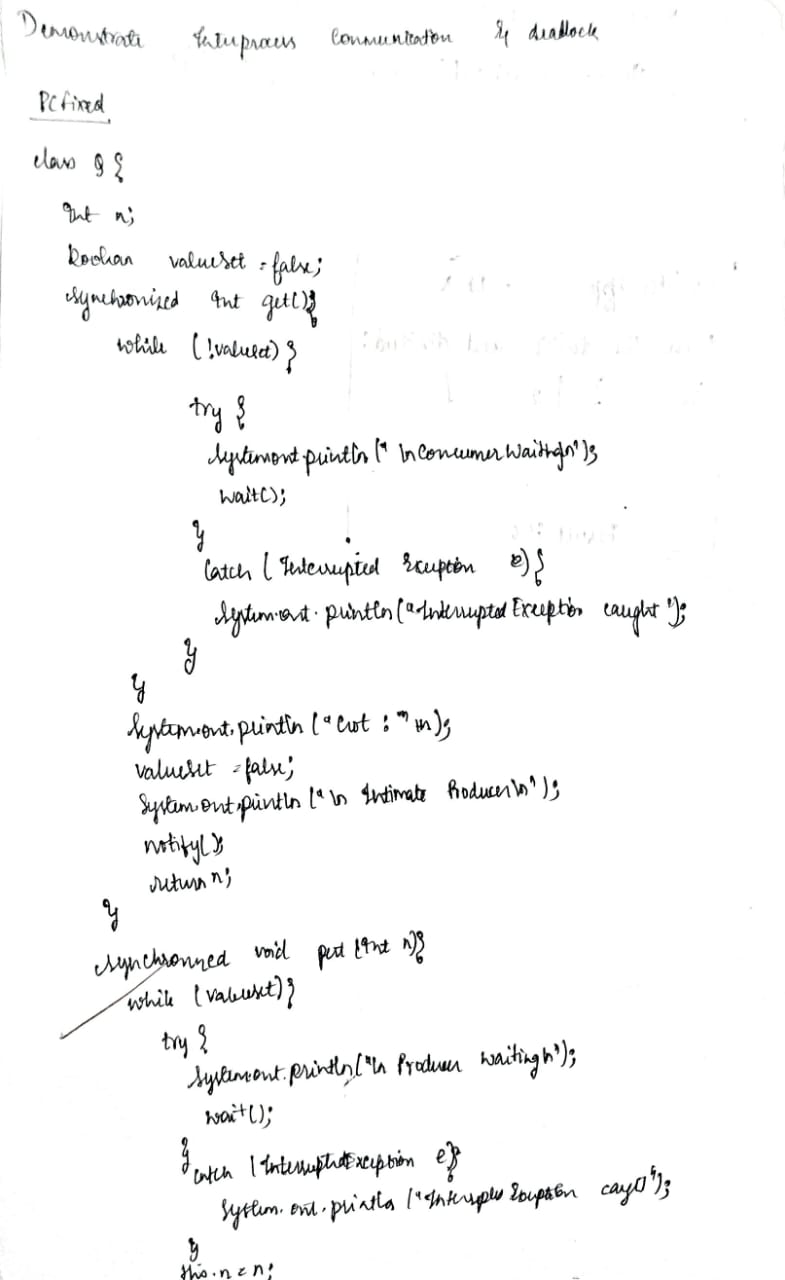
}

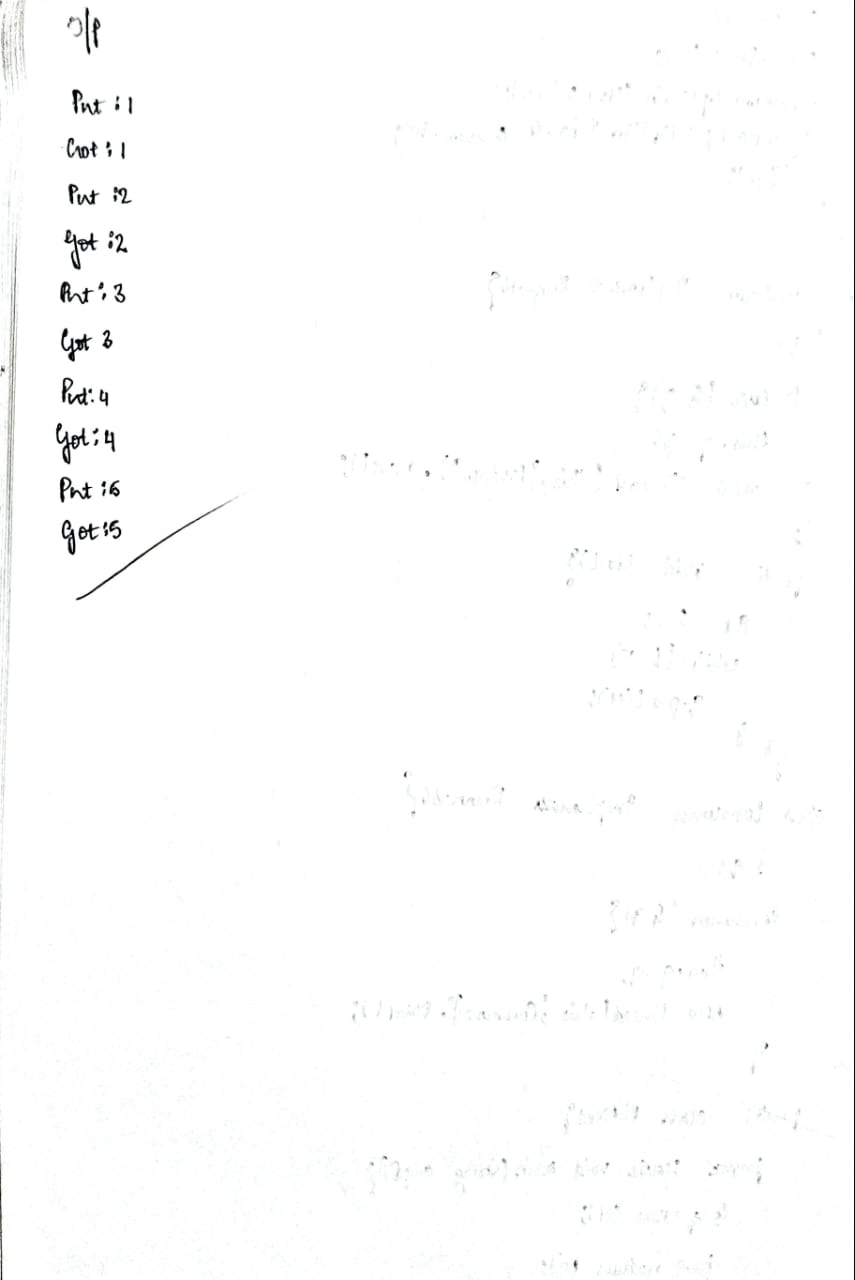


**Program 10**

1. **Implement PCFixed**

**Algorithm:**





**Code:**

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++;

}

}

}

public class PCFixed {

public static void main(String args[]) {

Q q = new Q();

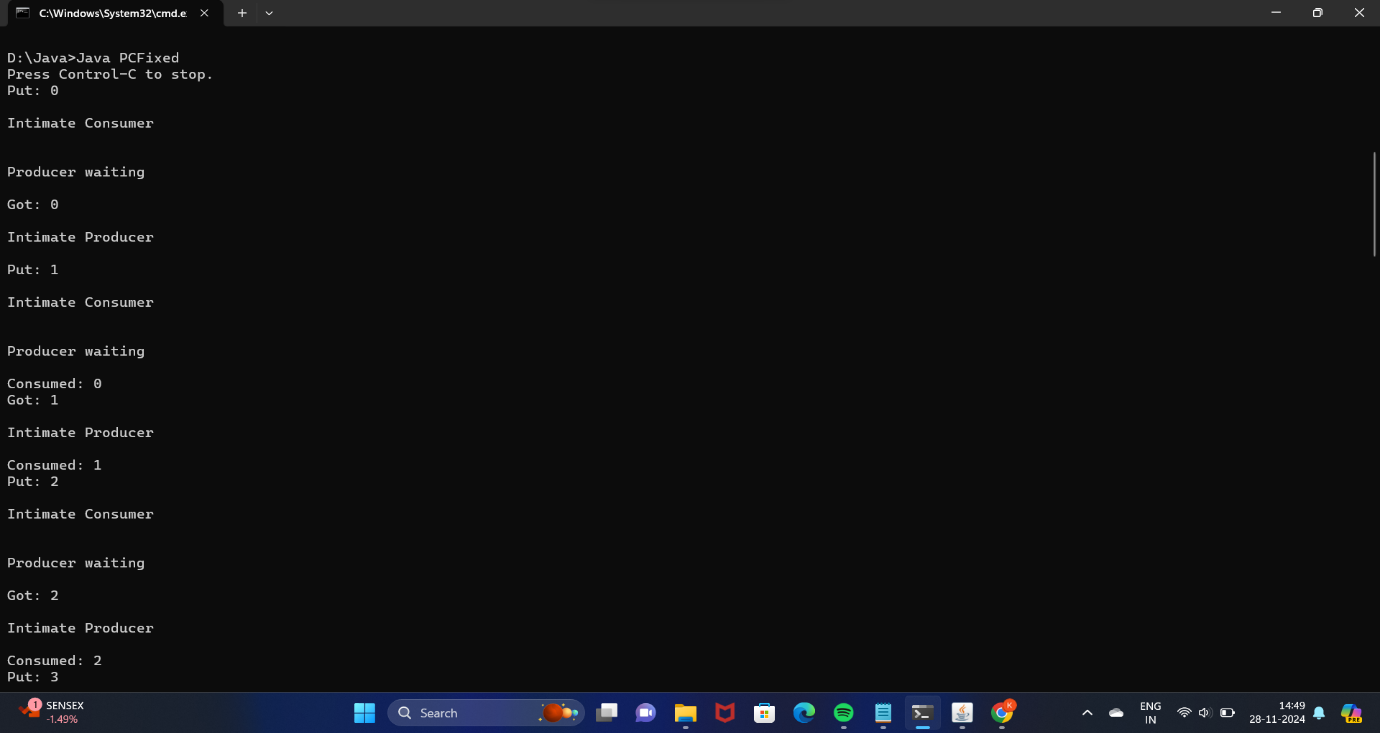
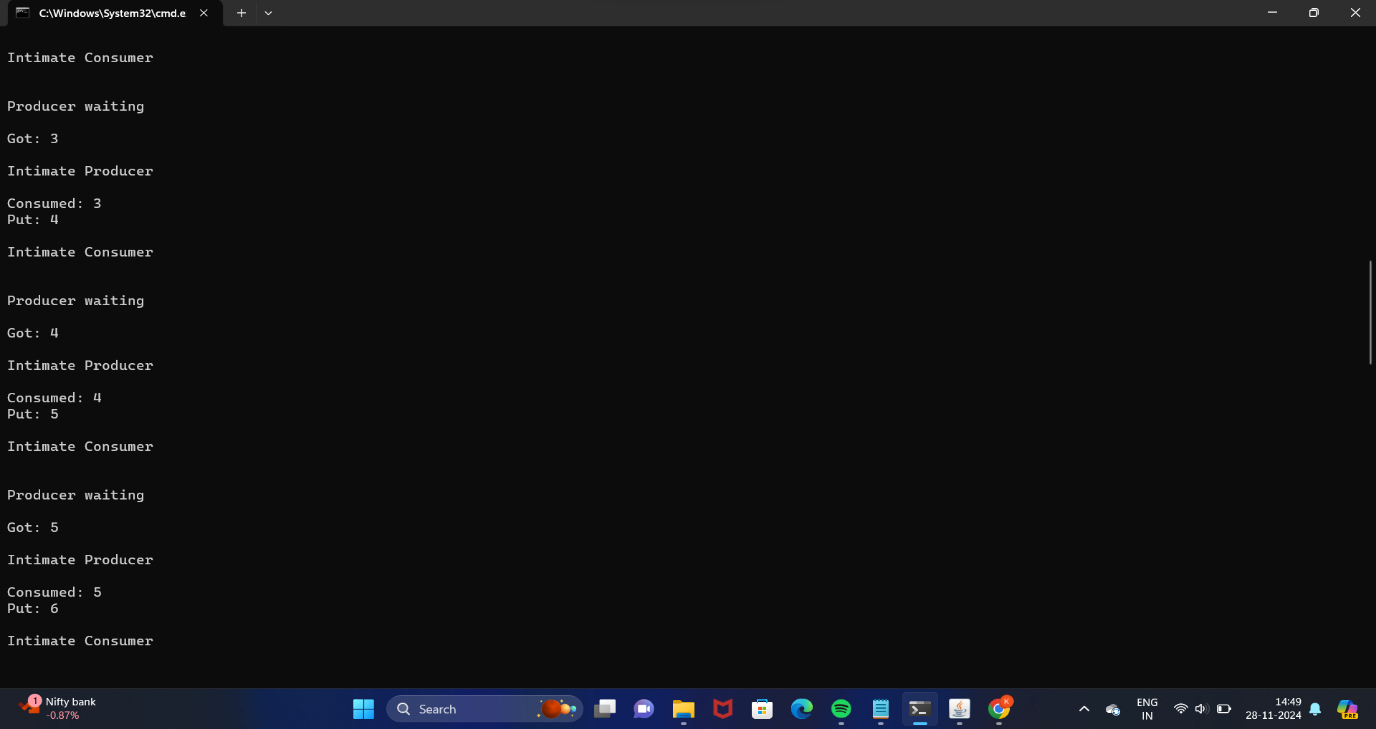
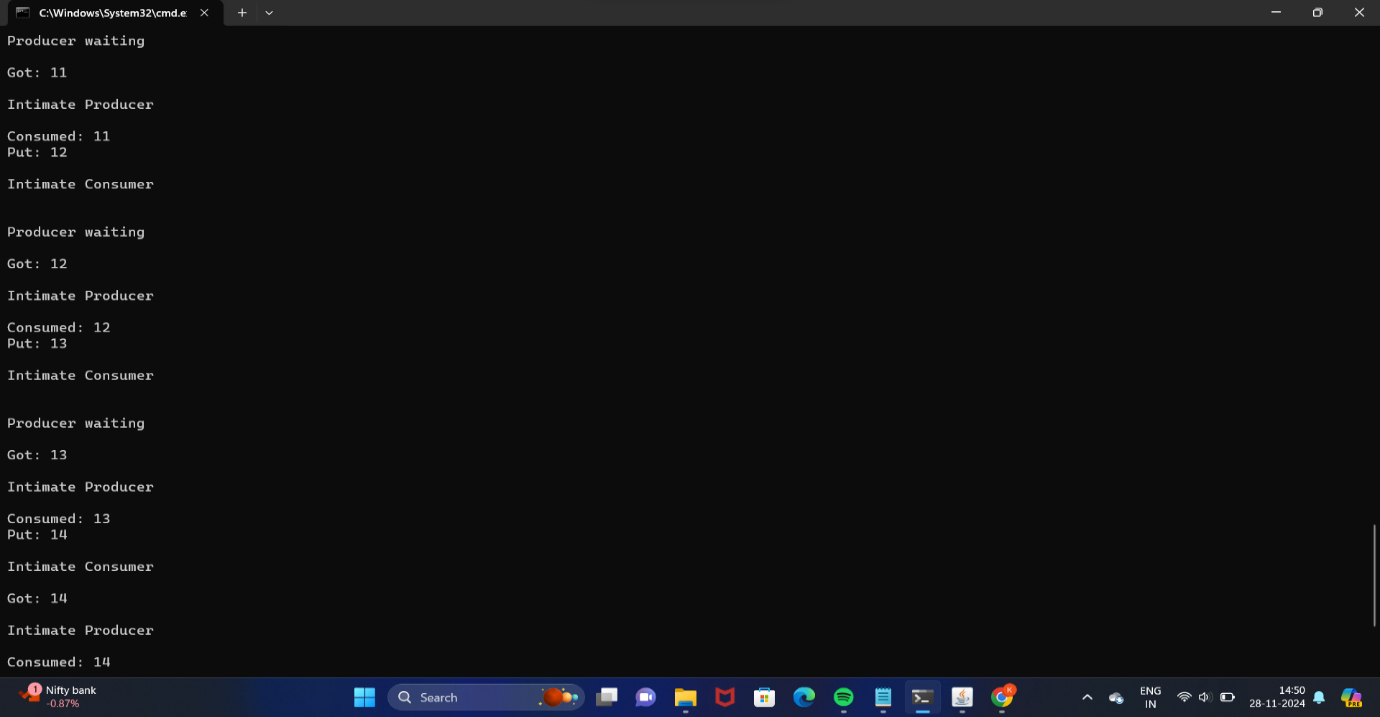
new Producer(q);

new Consumer(q);

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

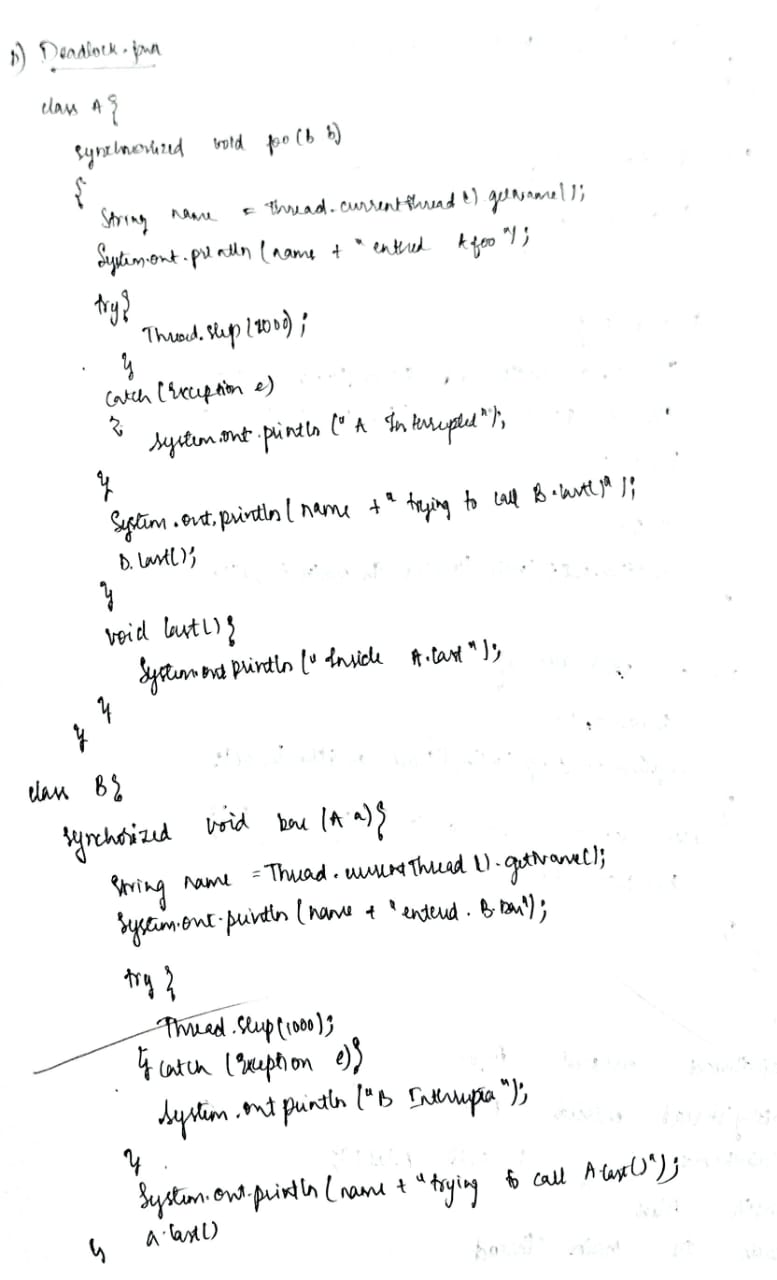
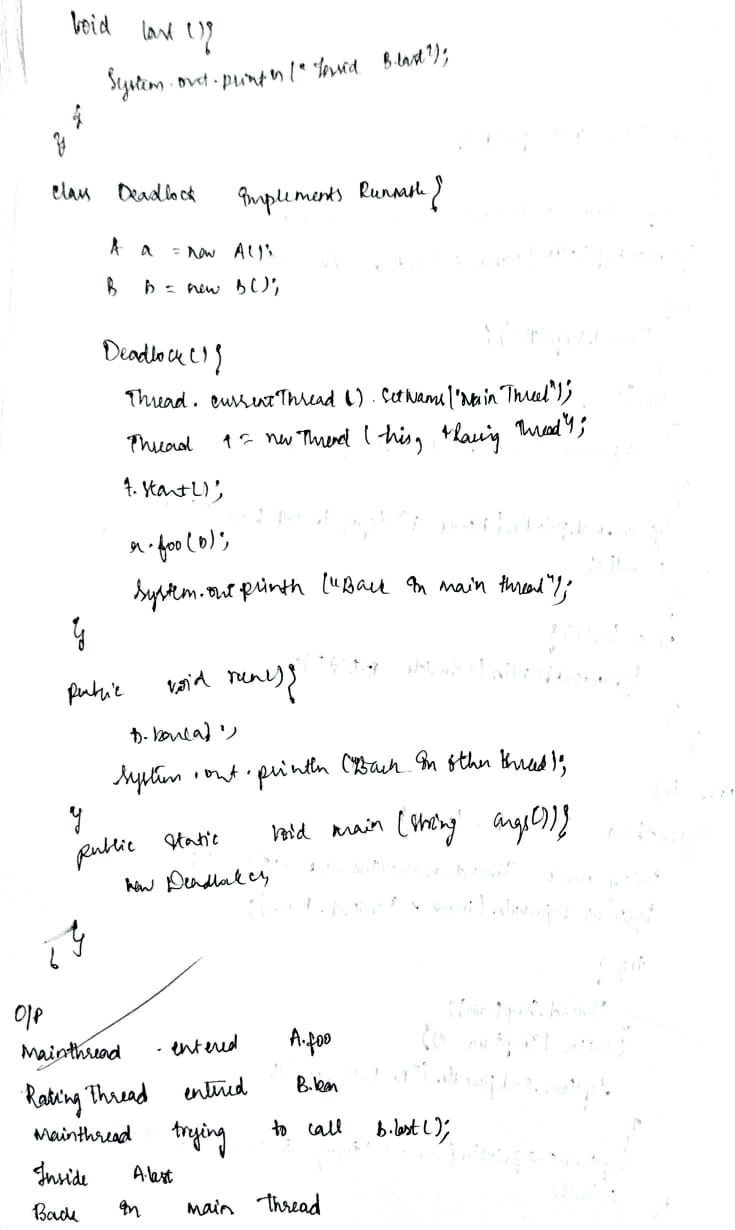
}

}



1. **ImplementDeadLock**

**Algorithm:**





**Code:**

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();

}

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();

}

void last() {

System.out.println("Inside B.last");

}

}

class Deadlock implements Runnable {

A a = new A();

B b = new B();

Deadlock() {

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

// Create a new thread and start it

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

t.start();

// Main thread locks object A

a.foo(b);

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

}

public void run() {

// Other thread locks object B

b.bar(a);

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

}

public static void main(String args[]) {

new Deadlock();

}

}

