VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Nagaraja G (24BECS426)

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 **Nagaraja G (24BECS426),** 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.

Srushti C S Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

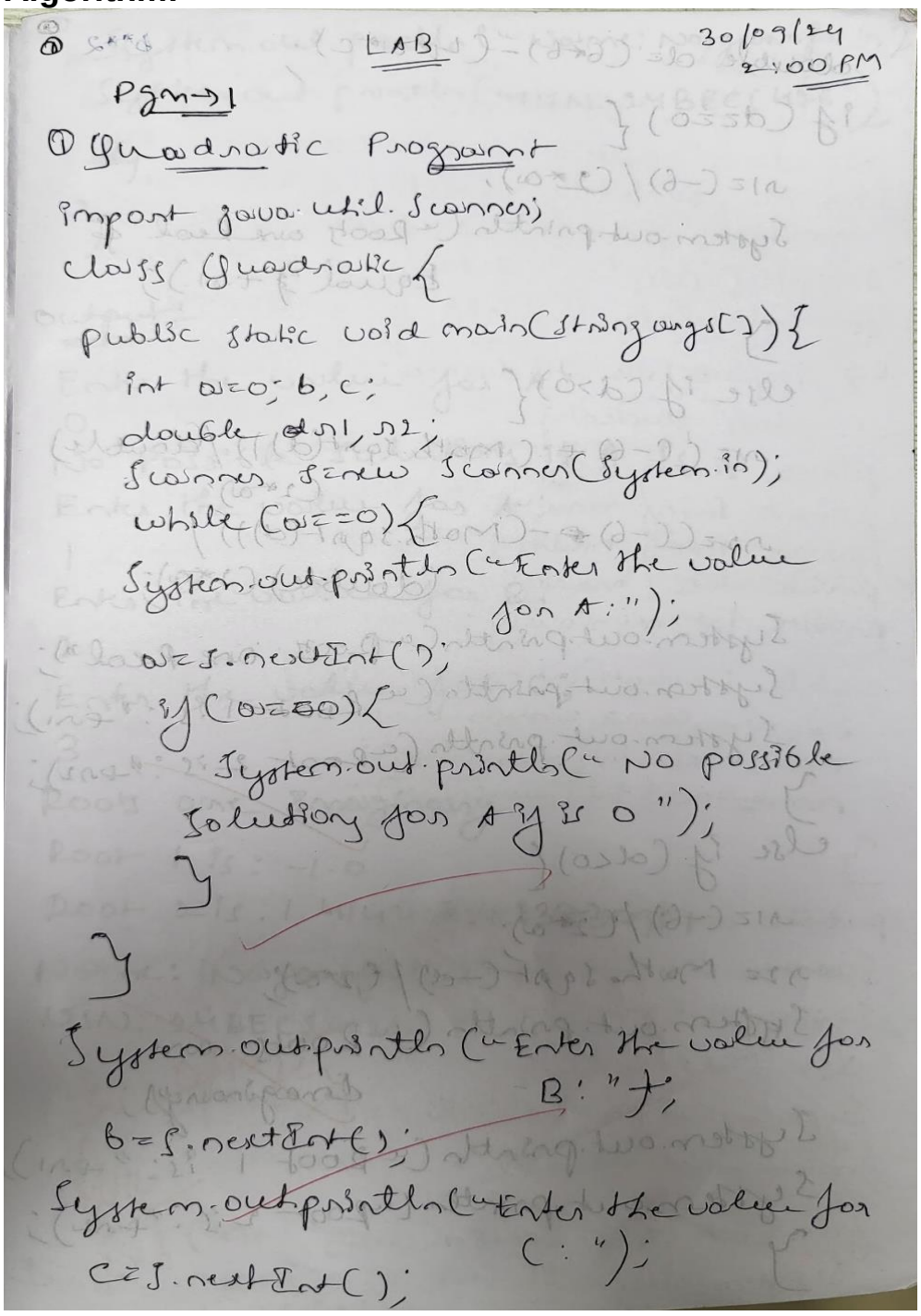
Index

Github Link: https://github.com/Naga1400/Java-Lab-Programs

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:



LAB 30/09/24 2:00 PM

Pgm→1

① Quadratic Program

```
import java.util.Scanner;
class Quadratic{
 public static void main(String args[]){
  int a=0; b, c;
  double r1, r2;
  Scanner s=new Scanner(System.in);
  while(a==0){
  System.out.println("Enter the value
                      for A:");
  a=s.nextInt();
   if(a==0){
    System.out.println("No possible
    Solution for A if is 0");
   }
}
System.out.println("Enter the value for
                    B:");
b=s.nextInt();
System.out.println("Enter the value for
                    C:");
c=s.nextInt();
```

```
double d= (b*b)-(4*a*c);          b**2

if (d==0) {
    r1=(-b)/(2*a);
    System.out.println("Roots are Real &
                        Equal"+r1);
}
else if (d>0) {
    r1=((-b)+(Math.sqrt(d)))/(double)
                        (2*a);
    r2=((-b)-(Math.sqrt(d)))/
                        (double)(2*a);
    System.out.println("Roots are Real");
    System.out.println("Root 1 is :"+r1);
    System.out.println("Root 2 is :"+r2);
}
else if (d<0) {
    r1=(-b)/(2*a);
    r2= Math.sqrt(-d)/(2*a);
    System.out.println("Roots are
                        Imaginary");
    System.out.println("Root 1 is :"+r1);
    System.out.println("Root 2 is :"+r2);
}
```

(13)

```
System.out.println("Name: Nalgarajo. G");
System.out.println("USN: 24BECS426");
    }
}
```

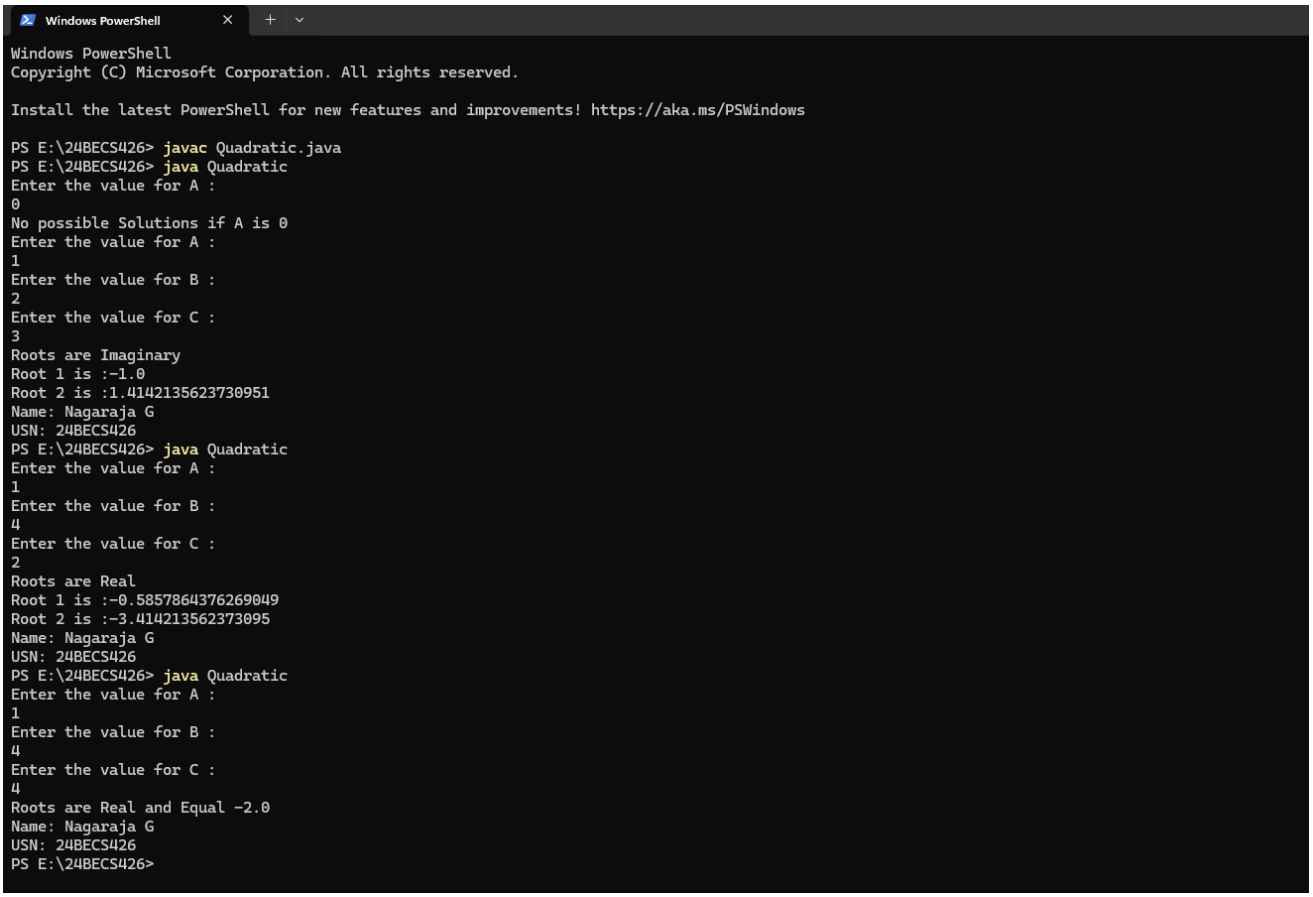
Output:

```
Enter the value for A:
0
No Possible solutions if A is 0
Enter the value for A:
1
Enter the value for B:
2
Enter the value for C:
3
Roots are Imaginary
Root 1 is: -1.0
Root 2 is: 1.4142135623730951
Name: Nalgarajo. G
USN: 24BECS426
```

Code:

```
import java.util.Scanner;
class Quadratic{
	public static void main(String args[]){

	int a=0,b,c;
	double r1,r2;
	Scanner s=new Scanner(System.in);
	while (a==0){
	System.out.println("Enter the value for A :");
	a=s.nextInt();
	if
(a==0){
			System.out.println("No possible Solutions if A is 0 ");
			}
	}
	System.out.println("Enter the value for B :");
	b=s.nextInt();
	System.out.println("Enter the value for C :");
	c=s.nextInt();
	double d=(b*b)-(4*a*c);
	if (d==0){
			r1=(-b)/(2*a);
			System.out.println("Roots are Real and Equal "+r1);
			}
			else if(d>0){
			r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
			r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a);
			System.out.println("Roots are Real");
			System.out.println("Root 1 is :"+r1);
			System.out.println("Root 2 is :"+r2);
			}
			else if(d<0){
			r1 = (-b)/(2*a);
			r2 = Math.sqrt(-d)/(2*a);
			System.out.println("Roots are Imaginary");
			System.out.println("Root 1 is :"+r1);
			System.out.println("Root 2 is :"+r2);
			}

			System.out.println("Name: Nagaraja G");
			System.out.println("USN: 24BECS426");
	}
}
```

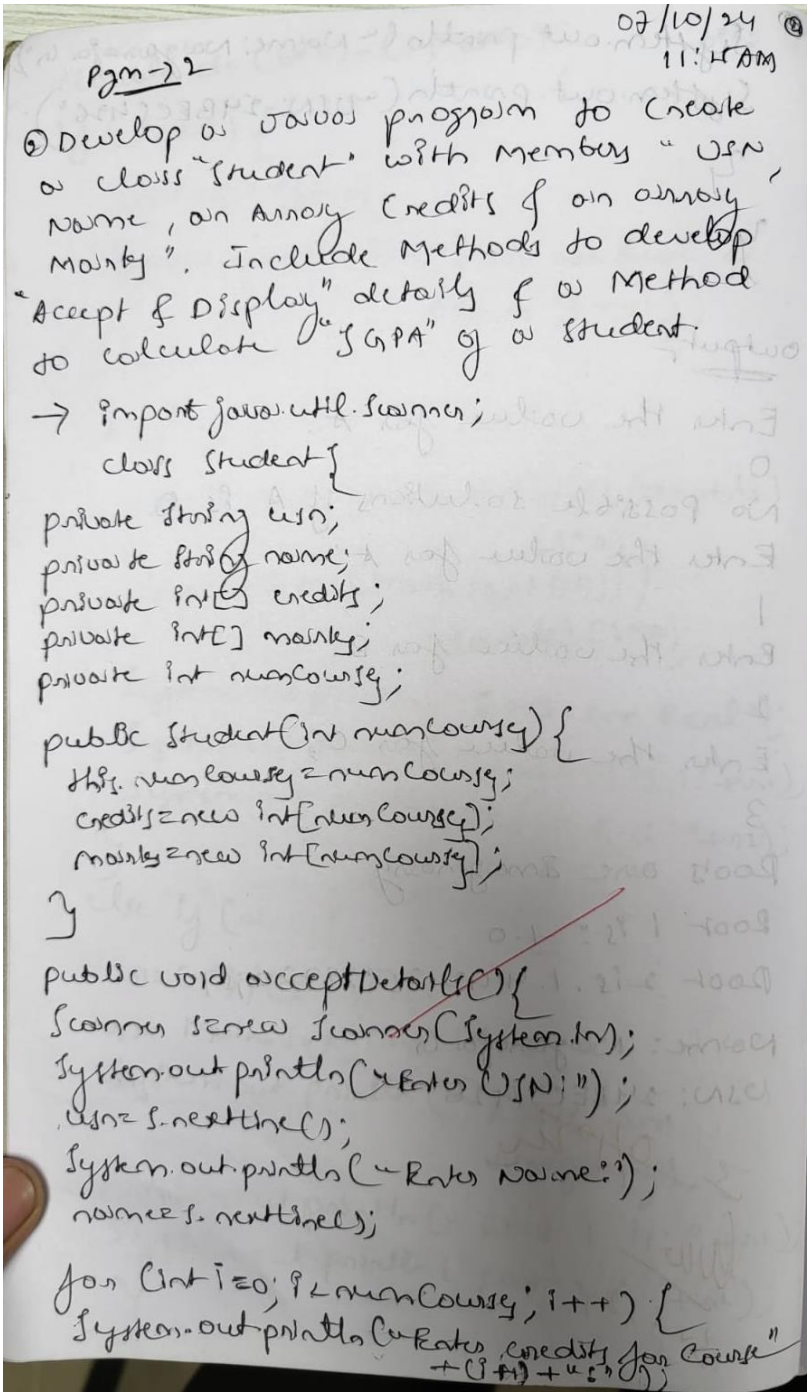
Output:


Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS E:\24BECS426> javac Quadratic.java
PS E:\24BECS426> java Quadratic
Enter the value for A :
0
No possible Solutions if A is 0
Enter the value for A :
1
Enter the value for B :
2
Enter the value for C :
3
Roots are Imaginary
Root 1 is :-1.0
Root 2 is :1.4142135623730951
Name: Nagaraja G
USN: 24BECS426
PS E:\24BECS426> java Quadratic
Enter the value for A :
1
Enter the value for B :
4
Enter the value for C :
2
Roots are Real
Root 1 is :-0.5857864376269049
Root 2 is :-3.414213562373095
Name: Nagaraja G
USN: 24BECS426
PS E:\24BECS426> java Quadratic
Enter the value for A :
1
Enter the value for B :
4
Enter the value for C :
4
Roots are Real and Equal -2.0
Name: Nagaraja G
USN: 24BECS426
PS E:\24BECS426>

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:

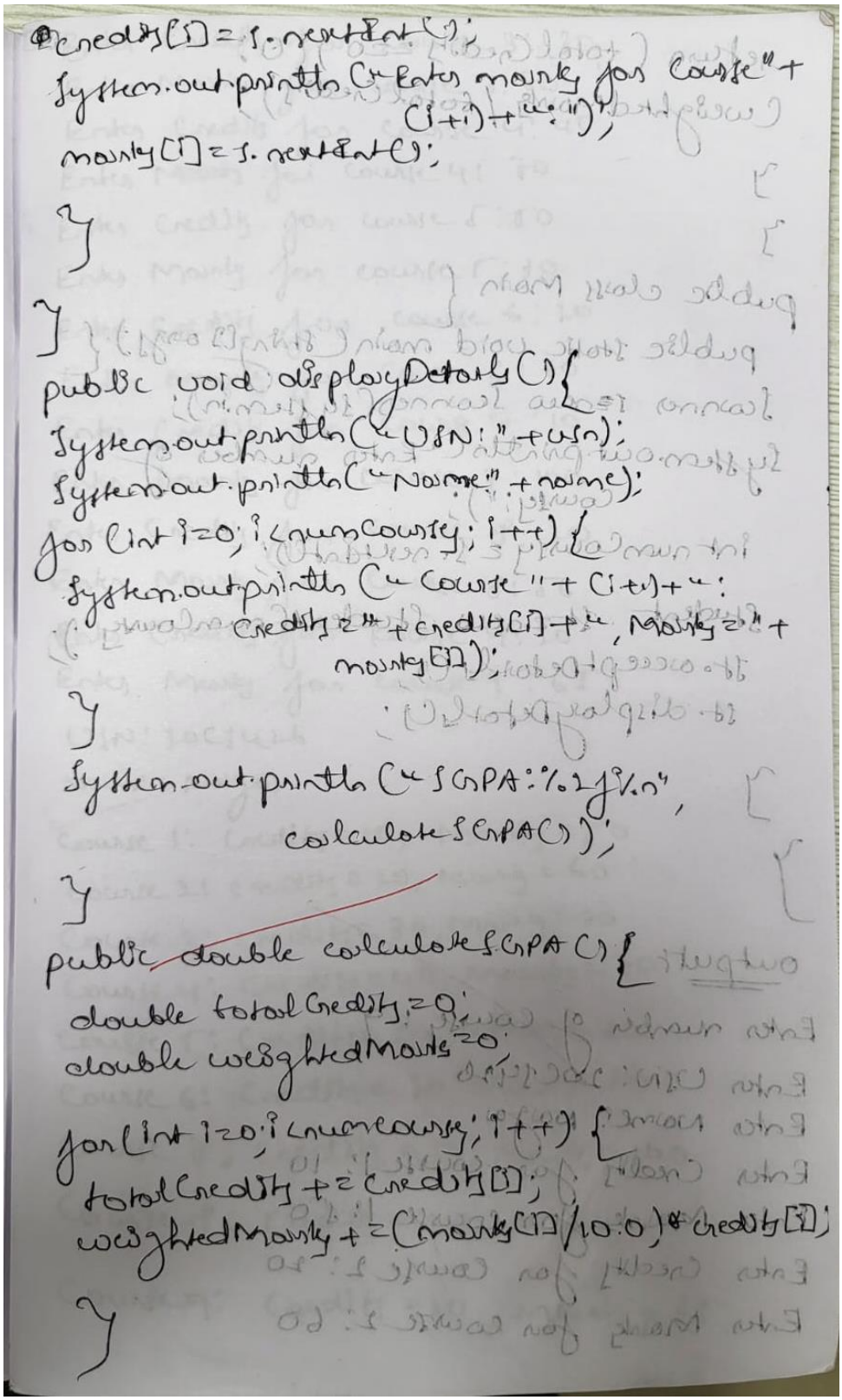


07/10/24
11:45AM

Pgm-12

① Develop a Java program to create a class "Student" with Members "USN, Name, an Array Credits & an array Marks". Include Methods to develop "Accept & Display" details & a Method to calculate "SGPA" of a student.

```
→ import java.util.Scanner;
  class Student{
private String usn;
private String name;
private int[] credits;
private int[] marks;
private int numCourse;
public Student(int numCourse){
  this.numCourse=numCourse;
  credits=new int[numCourse];
  marks=new int[numCourse];
}
public void acceptDetails(){
Scanner s=new Scanner(System.in);
System.out.println("Enter USN:");
usn=s.nextLine();
System.out.println("Enter Name:");
name=s.nextLine();
for(int i=0;i<numCourse;i++){
System.out.println("Enter credits for course"
+(i+1)+":");
```



```
credits[i] = s.nextInt();
System.out.println("Enter marks for course" +
                   (i+1) + ":");
marks[i] = s.nextInt();
}
}
public void displayDetails(){
System.out.println("USN:" + usn);
System.out.println("Name:" + name);
for(int i=0; i<numCourse; i++){
System.out.println("Course" + (i+1) + ":
  Credits=" + credits[i] + ", Marks=" +
  marks[i]);
}
System.out.println("SGPA: %.2f%n",
  calculateSGPA());
}
public double calculateSGPA(){
double totalCredits=0;
double weightedMarks=0;
for(int i=0; i<numCourse; i++){
totalCredits += credits[i];
weightedMarks += (marks[i]/10.0)*credits[i];
}
```

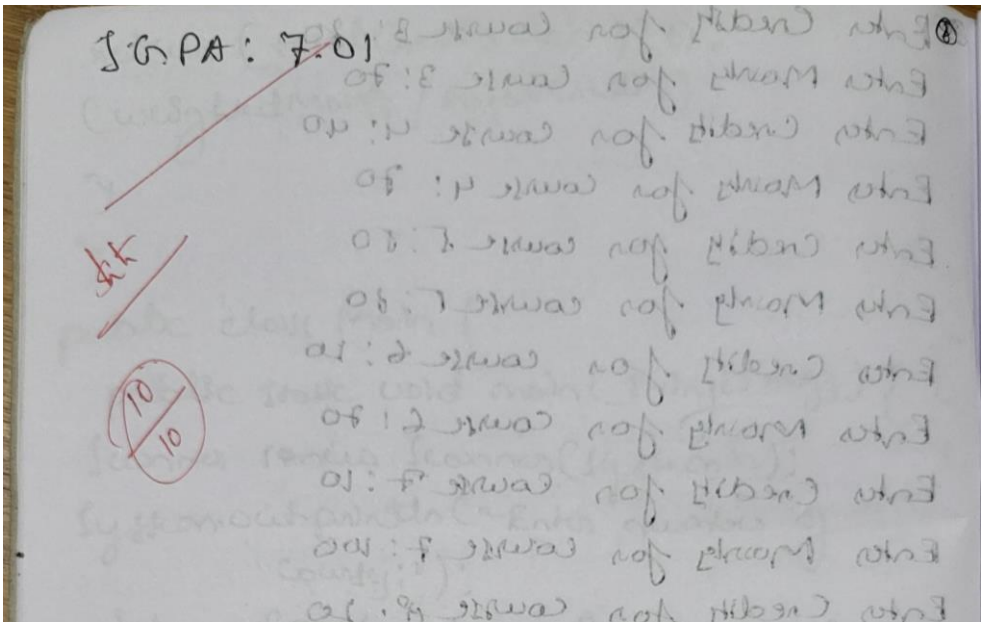
```
return (totalCredits == 0) ? 0 :
(weightedMarks / totalCredits);
}
}

public class Main {
  public static void main(String[] args) {
  Scanner sc = new Scanner(System.in);
  System.out.println("Enter number of
          Courses:");
  int numCourses = sc.nextInt();
  Student st = new Student(numCourses);
  st.acceptDetails();
  st.displayDetails();
  }
}
```

output:

```
Enter number of courses: 2
Enter USN: 2BC19IS20
Enter Name: Navya
Enter Credits for course 1: 10
Enter Marks for course 1: 50
Enter Credits for course 2: 20
Enter Marks for course 2: 60
```

Enter Credits for course 3: 30
Enter Marks for course 3: 70
Enter Credits for course 4: 40
Enter Marks for course 4: 70
Enter Credits for course 5: 50
Enter Marks for course 5: 80
Enter Credits for course 6: 20
Enter Marks for course 6: 70
Enter Credits for course 7: 10
Enter Marks for course 7: 100
Enter Credits for course 8: 20
Enter Marks for course 8: 56
Enter Credits for course 9: 20
Enter Marks for course 9: 65
USN: 20CS426
Name: Navya

Course 1: Credits = 10, Marks = 50
Course 2: Credits = 20, Marks = 60
Course 3: Credits = 30, Marks = 70
Course 4: Credits = 40, Marks = 70
Course 5: Credits = 50, Marks = 80
Course 6: Credits = 20, Marks = 70
Course 7: Credits = 10, Marks = 100
Course 8: Credits = 20, Marks = 56
Course 9: Credits = 20, Marks = 65



Code:

```
import java.util.Scanner;

class Student {
    private String usn;
    private String name;
    private int[] credits;
    private int[] marks;
    private int numCourses;


    public Student(int numCourses) {
        this.numCourses = numCourses;
        credits = new int[numCourses];
        marks = new int[numCourses];
    }


    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 < numCourses; i++) {
```

```
            System.out.print("Enter credits for course " + (i + 1) + ": ");
            credits[i] = scanner.nextInt();

            System.out.print("Enter marks for course " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }


    public void displayDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        for (int i = 0; i < numCourses; i++) {
            System.out.println("Course " + (i + 1) + ": Credits = " + credits[i] + ", Marks = " + marks[i]);
        }
        System.out.printf("SGPA: %.2f%n", calculateSGPA());
    }


    public double calculateSGPA() {
        double totalCredits = 0;
        double weightedMarks = 0;

        for (int i = 0; i < numCourses; i++) {
            totalCredits += credits[i];
            weightedMarks += (marks[i] / 10.0) * credits[i];
        }

        return (totalCredits == 0) ? 0 : (weightedMarks / totalCredits);
    }
}

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

        System.out.print("Enter number of courses: ");
        int numCourses = scanner.nextInt();

        Student student = new Student(numCourses);

        student.acceptDetails();

        student.displayDetails();
    }
}
```

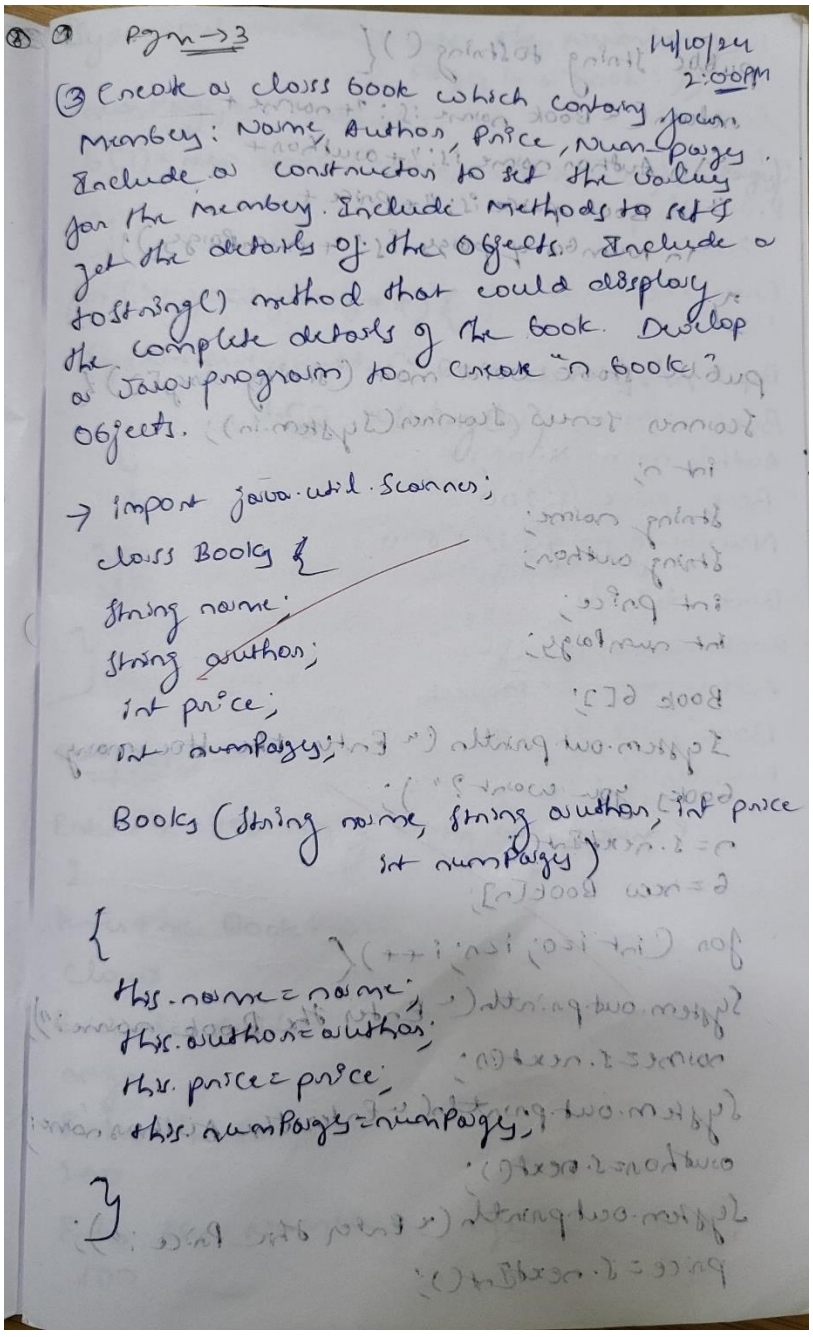
Output:

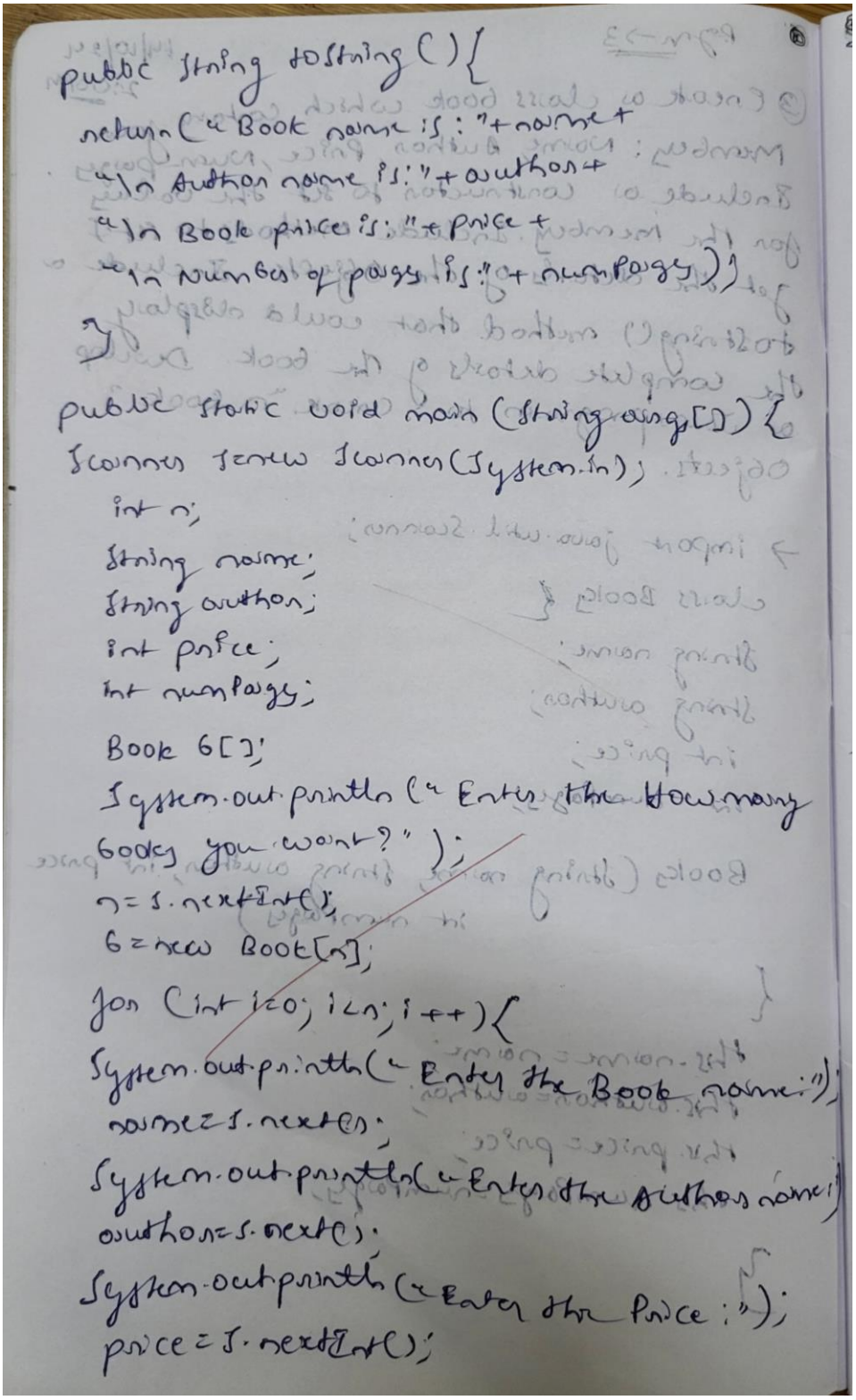
```
PS D:\24CSBE426> javac Main.java
PS D:\24CSBE426> java Main
Enter number of courses: 9
Enter USN: 20cs426
Enter Name: Naga
Enter credits for course 1: 10
Enter marks for course 1: 50
Enter credits for course 2: 20
Enter marks for course 2: 60
Enter credits for course 3: 30
Enter marks for course 3: 70
Enter credits for course 4: 40
Enter marks for course 4: 70
Enter credits for course 5: 50
Enter marks for course 5: 80
Enter credits for course 6: 20
Enter marks for course 6: 70
Enter credits for course 7: 10
Enter marks for course 7: 100
Enter credits for course 8: 20
Enter marks for course 8: 56
Enter credits for course 9: 20
Enter marks for course 9: 65
USN: 20cs426
Name: Naga
Course 1: Credits = 10, Marks = 50
Course 2: Credits = 20, Marks = 60
Course 3: Credits = 30, Marks = 70
Course 4: Credits = 40, Marks = 70
Course 5: Credits = 50, Marks = 80
Course 6: Credits = 20, Marks = 70
Course 7: Credits = 10, Marks = 100
Course 8: Credits = 20, Marks = 56
Course 9: Credits = 20, Marks = 65
SGPA: 7.01
PS D:\24CSBE426>
```

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:





```
public String toString(){
return("Book name is: "+name+
"\n Author name is: "+author+
"\n Book price is: "+price+
"\n Number of pages is: "+numPages);
}
public static void main(String args[]){
Scanner sc=new Scanner(System.in);
int n;
String name;
String author;
int price;
int numPages;
Book b[];
System.out.println("Enter the How many
books you want?");
n=s.nextInt();
b=new Book[n];
for(int i=0;i<n;i++){
System.out.println("Enter the Book name:");
name=s.next();
System.out.println("Enter the Author name:");
author=s.next();
System.out.println("Enter the Price:");
price=s.nextInt();
```

```
System.out.println("Enter the number of
                    pages in a Book: ");
numPages = s.nextInt();
b[i] = new Books(name, author, price, numPages);
}
for(int j=0; j<n; j++){
System.out.println("Book details:\n" +
        b[j].toString());
}
}
}
```

output:

Enter the how many books you want?
2
Enter the Book name:
Cloud
Enter the Author name:
naga
Enter the Price:
200
Enter the Number of pages in a Book:
500

Enter the Book name:
Java
Enter the Author name:
Mahesh
Enter the price:
300
Enter the number of pages in Book:
1000

Book details:
Book name is: cloud
Author name is: naga
Book price is: 200
Number of pages is: 500

Book details
Book name is: Java
Author name is: Mahesh
Book price is: 300
Number of pages is: 1000

16/10/24

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 is :"+name+
"\nAuthor name is :"+author+
"\nBook price is :"+price+
"\nNumber of pages is :"+numPages);
}

public static void main(String args[]){
Scanner s=new Scanner(System.in);
int n;
String name;
String author;
int price;
int numPages;

Books b[];

System.out.println("Enter the How many books you want ?");
n=s.nextInt();
b=new Books[n];

for(int i=0;i<n;i++){
System.out.println("Enter the book name :");
name=s.next();
```

```
System.out.println("Enter the author name :");
author=s.next();
System.out.println("Enter the Price :");
price=s.nextInt();
System.out.println("Enter the Number of Pages in Book :");
numPages=s.nextInt();
b[i] = new Books(name,author,price,numPages);
}

for(int j=0;j<n;j++){
System.out.println("Book details \n"+b[j].toString());
}
}
}
```

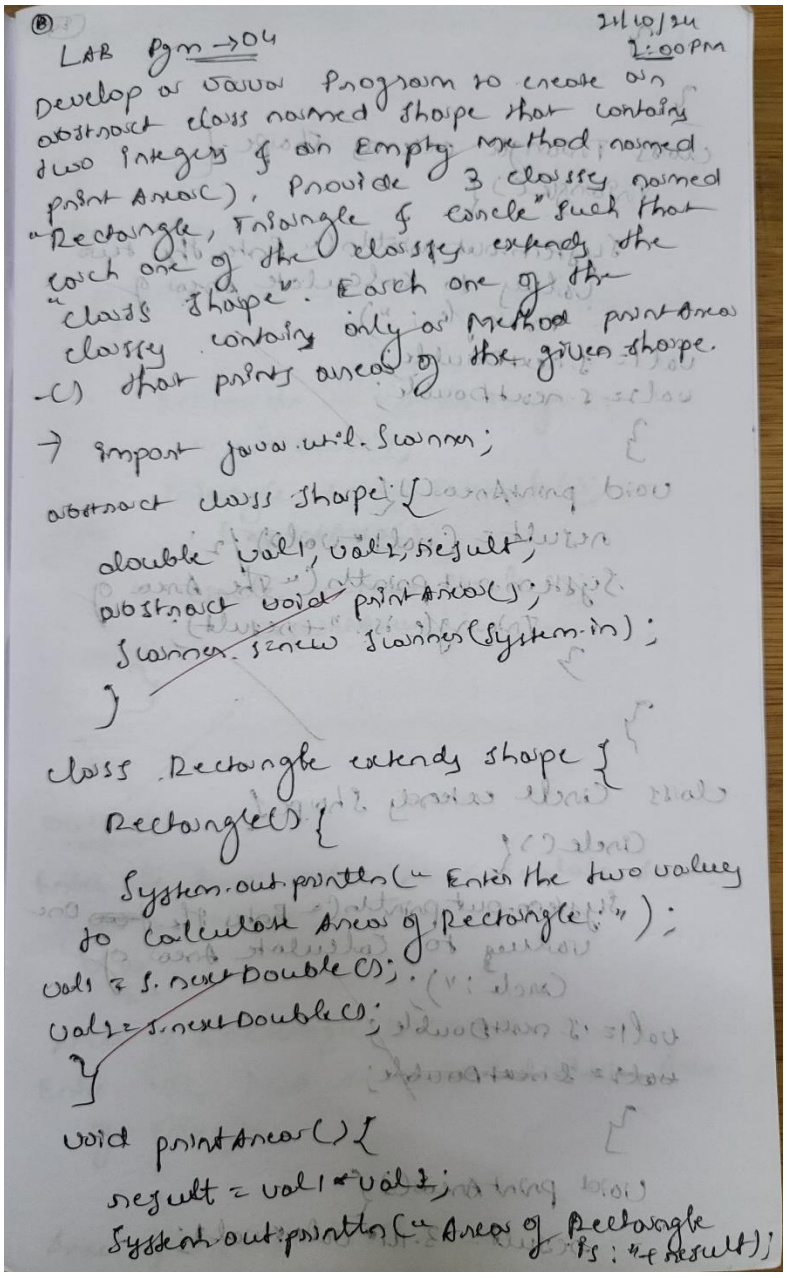
Output:

```
PS D:\24CSBE426> java Books
Enter the How many books you want ?
2
Enter the book name :
cloud
Enter the author name :
naga
Enter the Price :
200
Enter the Number of Pages in Book :
1000
Enter the book name :
Java
Enter the author name :
Mahesh
Enter the Price :
300
Enter the Number of Pages in Book :
500
Book details
Book name is :cloud
Author name is :naga
Book price is :200
Number of pages is :1000
Book details
Book name is :Java
Author name is :Mahesh
Book price is :300
Number of pages is :500
```

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.

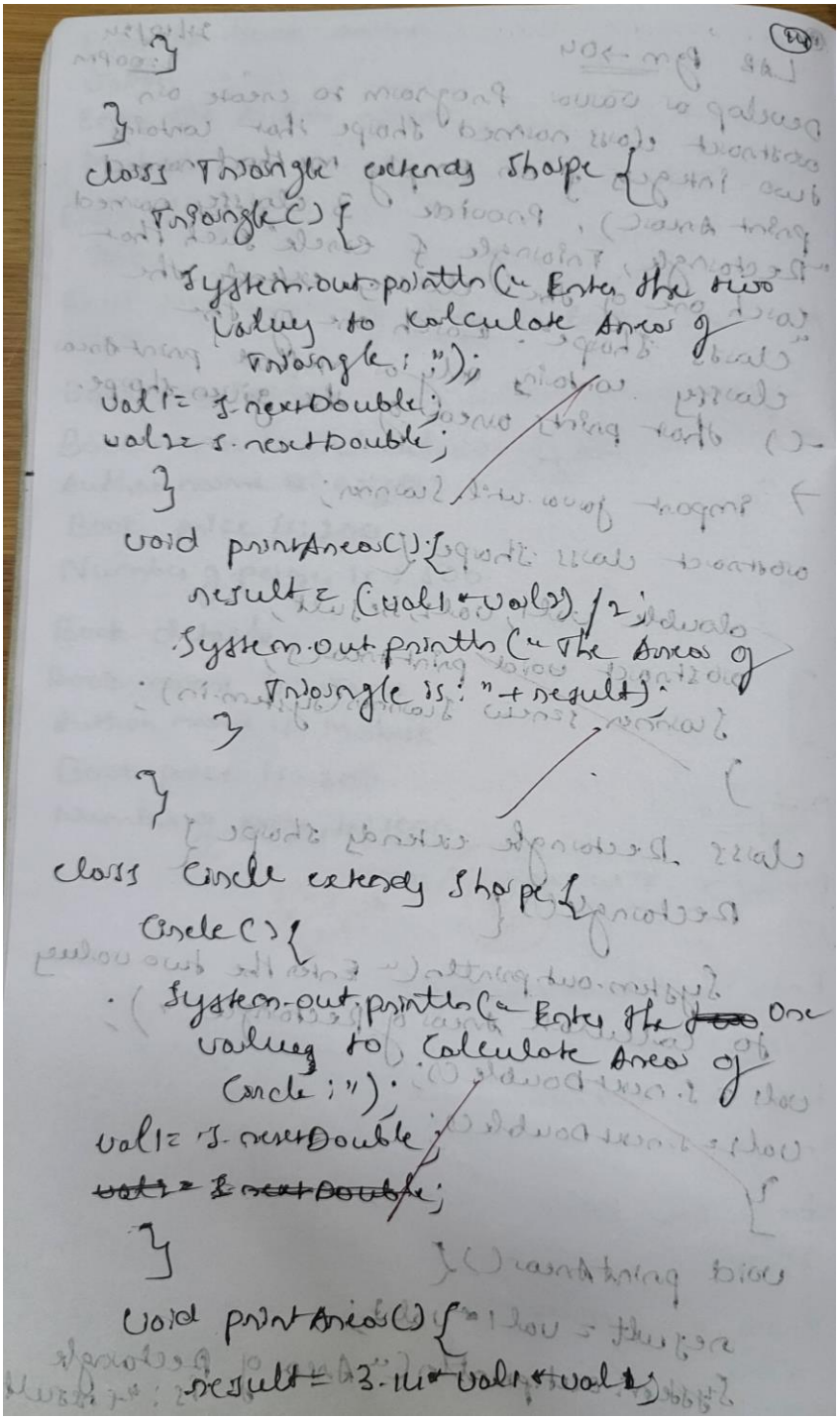
Algorithm:



LAB Pgm → 04 — 21/10/24 — 2:00 PM

Develop a Java Program to create an abstract class named Shape that contains two integers & an Empty method named printArea(). Provide 3 classes named "Rectangle, Triangle & Circle" such that each one of the classes extends the class "Shape". Each one of the classes contain only a Method printArea() that prints area of the given shape.

```
→ import java.util.Scanner;
abstract class Shape{
    double val1, val2, result;
    abstract void printArea();
    Scanner s=new Scanner(System.in);
}
class Rectangle extends Shape {
    Rectangle(){
    System.out.println("Enter the two values
    to calculate Area of Rectangle:");
val1 = s.nextDouble();
val2=s.nextDouble();
}
void printArea(){
    result = val1*val2;
System.out.println("Area of Rectangle
                    is:"+result);
```

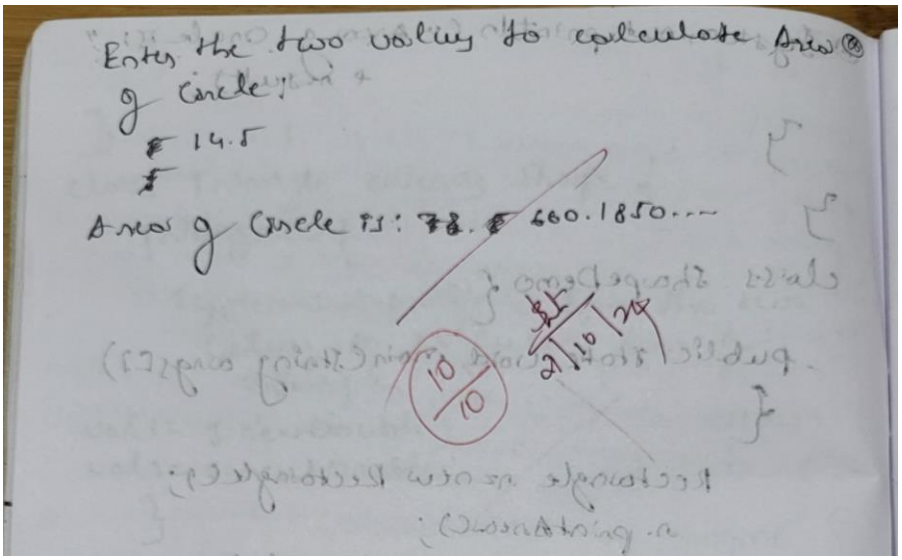


```
}
}
class Triangle extends Shape{
    Triangle(){
        System.out.println("Enter the two
        values to calculate Area of
        Triangle:");
    val1= s.nextDouble;
    val2= s.nextDouble;
    }
    void printArea(){
        result= (val1*val2)/2;
        System.out.println("The Area of
        Triangle is:"+result);
    }
}
class Circle extends Shape{
    Circle(){
        System.out.println("Enter the ~~two~~ One
        values to Calculate Area of
        Circle:");
    val1= s.nextDouble;
    ~~val2= s.nextDouble;~~
    }
    Void printArea(){
        result= 3.14*val1*val2;
```

```
    System.out.println("Area of Circle is: " + result);
  }
}
class ShapeDemo {
  public static void main(String args[])
  {
    Rectangle r = new Rectangle();
    r.printArea();
    Triangle t = new Triangle();
    t.printArea();
    Circle c = new Circle();
    c.printArea();
  }
}
```

Output:

```
Enter the two values to calculate Area
of Rectangle:
10
20
Area of Rectangle is: 200.0
Enter the two values to calculate Area
of Triangle:
1
2
Area of Triangle is: 1.0
```



Code:

```
import java.util.Scanner;

abstract class Shape{
 double val1,val2,result;
abstract void printArea();
Scanner s=new Scanner(System.in);
}

class Rectangle extends Shape{
Rectangle(){
System.out.println("Enter the Two values to Calculate Area of Rectangle :");
val1=s.nextDouble();
val2=s.nextDouble();
}
void printArea(){
result=val1*val2;
System.out.println("Area of Rectangle is :"+result);
}
}

class Triangle extends Shape{
Triangle(){
System.out.println("Enter the Two values to Calculate Area of Triangle :");
val1=s.nextDouble();
```

```
val2=s.nextDouble();
}
void printArea(){
result=(val1*val2)/2;
System.out.println("Area of Triangle is :"+result);
}
}

class Circle extends Shape{
Circle(){
System.out.println("Enter the One value to Calculate Area of Circle :");
val1=s.nextDouble();
}
void printArea(){
result=3.14*val1*val1;
System.out.println("Area of Circle is :"+result);
}
}

class shapeDemo{
public static void main(String args[]){
Rectangle r=new Rectangle();
r.printArea();
Triangle t=new Triangle();
t.printArea();
Circle c=new Circle();
c.printArea();
System.out.println("Name: Nagaraja G \n USN: 24BECS426");
 }
}
```

output:

```
PS D:\1bm23cs156> javac shapeDemo.java
PS D:\1bm23cs156> java shapeDemo
Enter the Two values to Calculate Area of Rectangle :
10
20
Area of Rectangle is :200.0
Enter the Two values to Calculate Area of Triangle :
1
2
Area of Triangle is :1.0
Enter the One value to Calculate Area of Circle :
14.5
Area of Circle is :660.1850000000001
Name: Nagaraja G
 USN: 24BECS426
PS D:\1bm23cs156>
```

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.

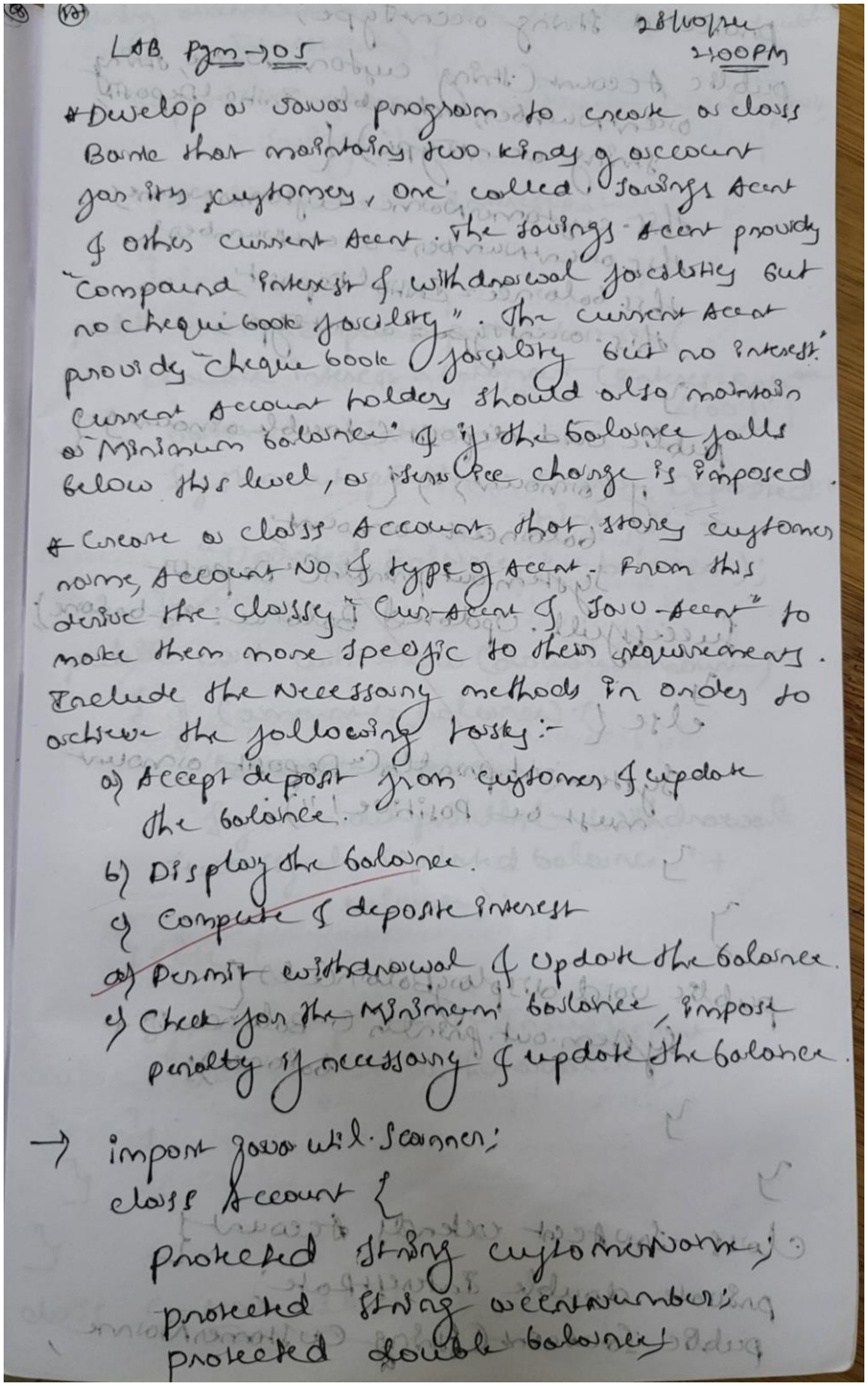
b)Display the balance.

c)Compute and deposit interest

d)Permit withdrawal and update the balance

e) Check for the minimum balance, impose penalty if necessary and update the balance.

Algorithm:



LAB Pgm → 05 28/10/22 2:00PM

* Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called 'savings Acct' & other 'current Acct'. The savings Acct provide "compound interest & withdrawal facility but no cheque book facility". The current Acct provide "cheque book facility but no interest". Current Account holder should also maintain a "Minimum balance" if the balance falls below this level, a service charge is imposed.

* Create a class Account that stores customer name, Account No. & type of Acct. From this derive the classes 'Cur-Acct' & '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 & update the balance.

b) Display the balance.

c) Compute & deposite interest

d) Permit withdrawal & update the balance.

e) Check for the Minimum balance, impose penalty if necessary & update the balance.

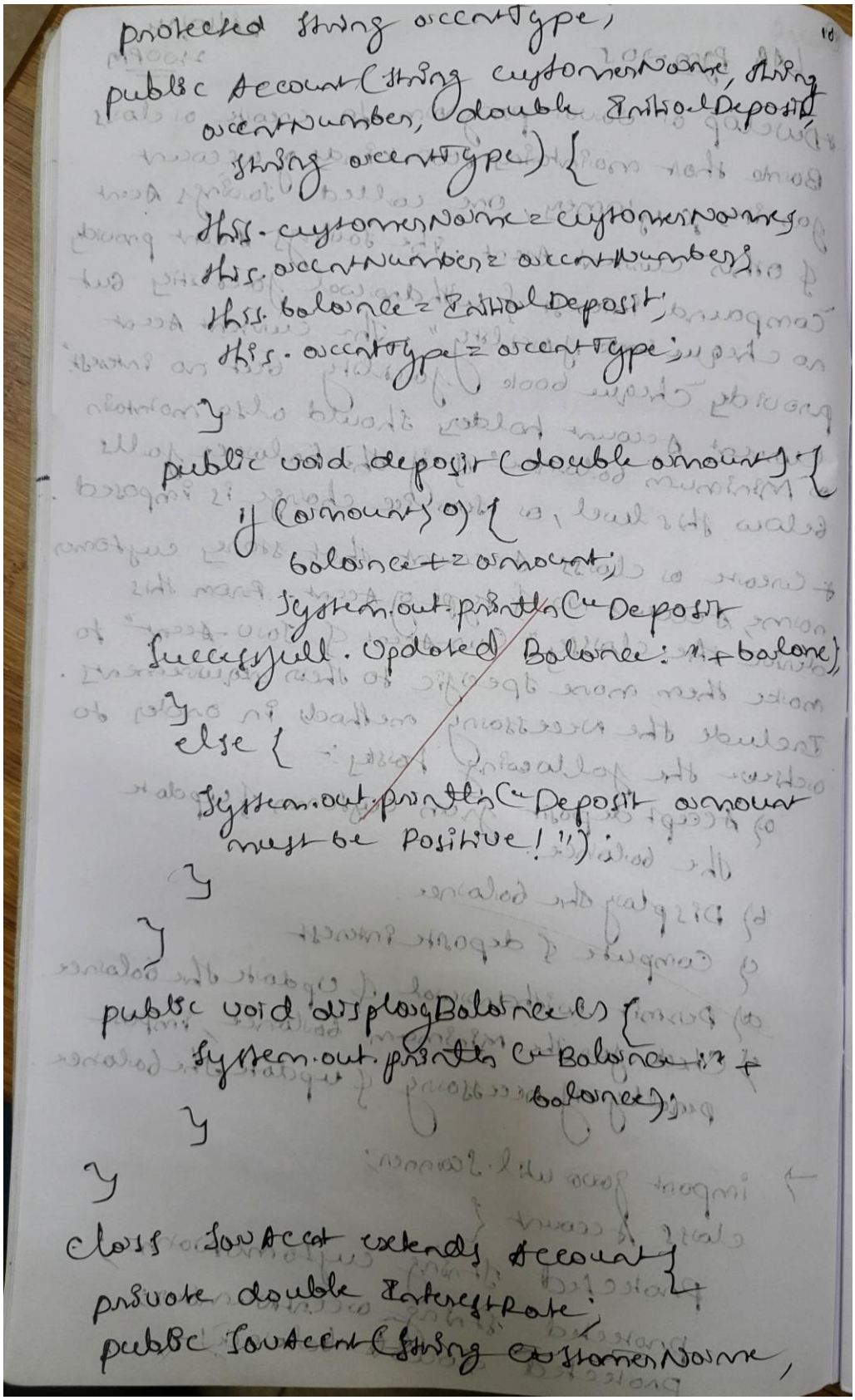
→ import java.util.Scanner;

class Account {

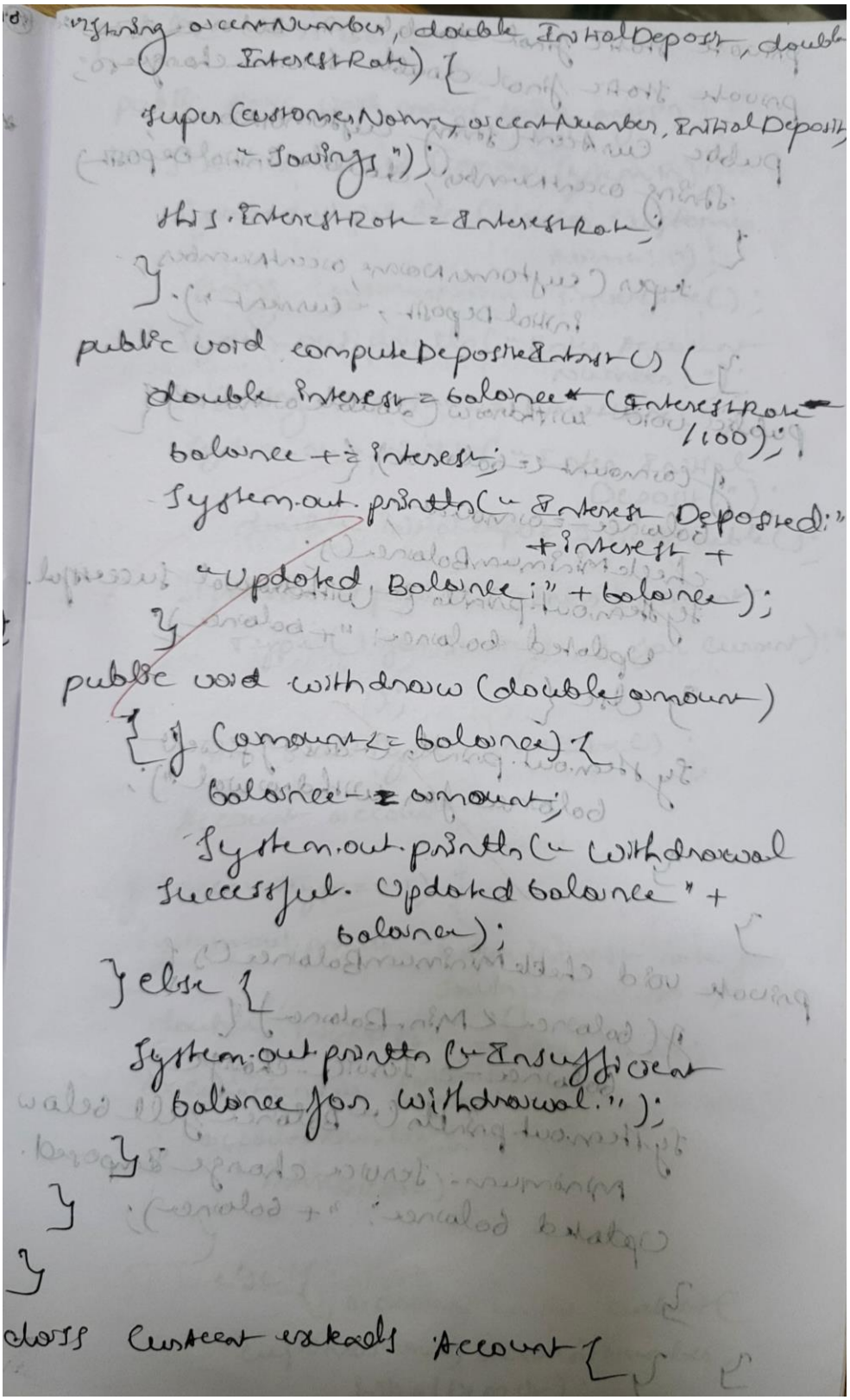
protected String customerName;

protected String accountNumber;

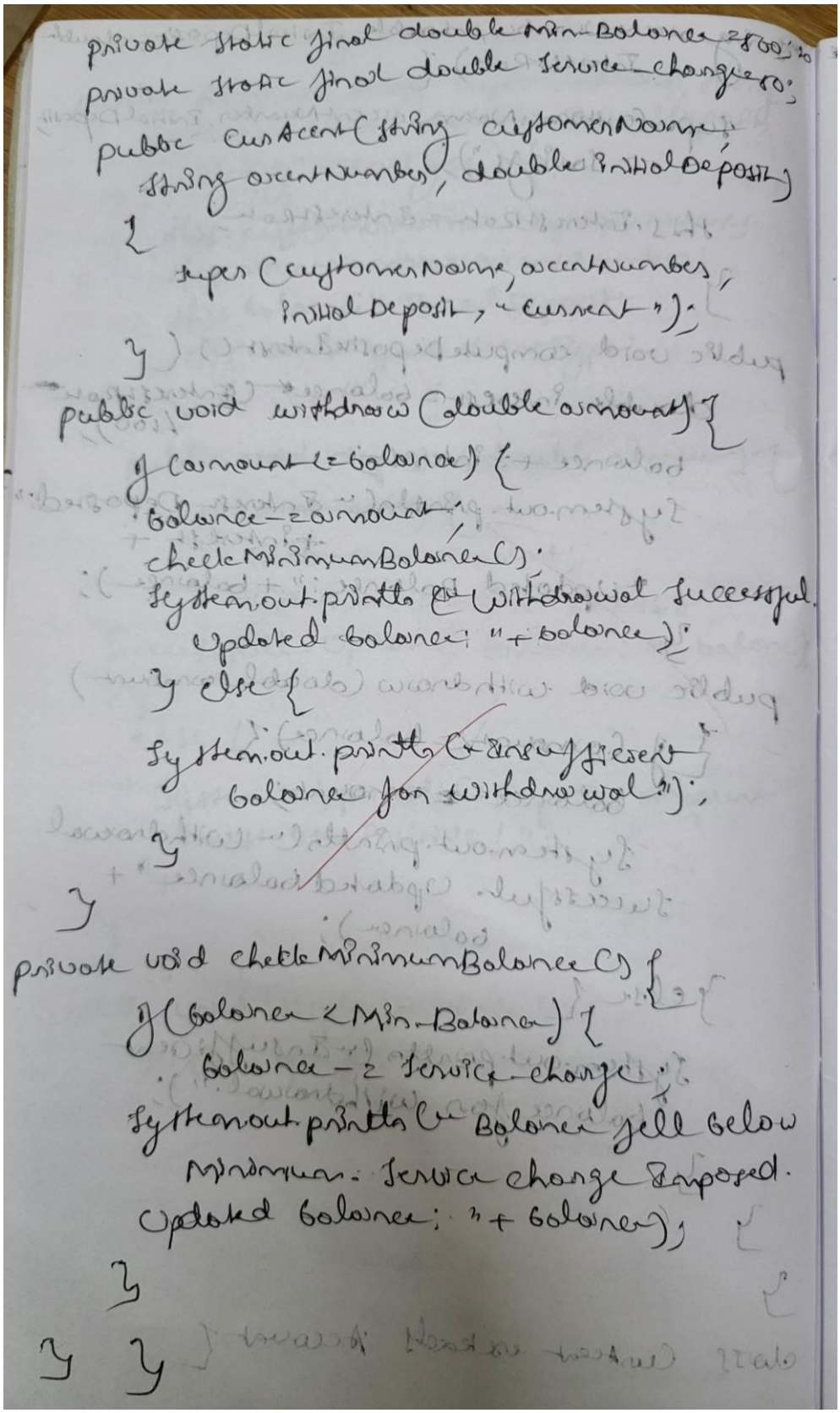
protected double balance;



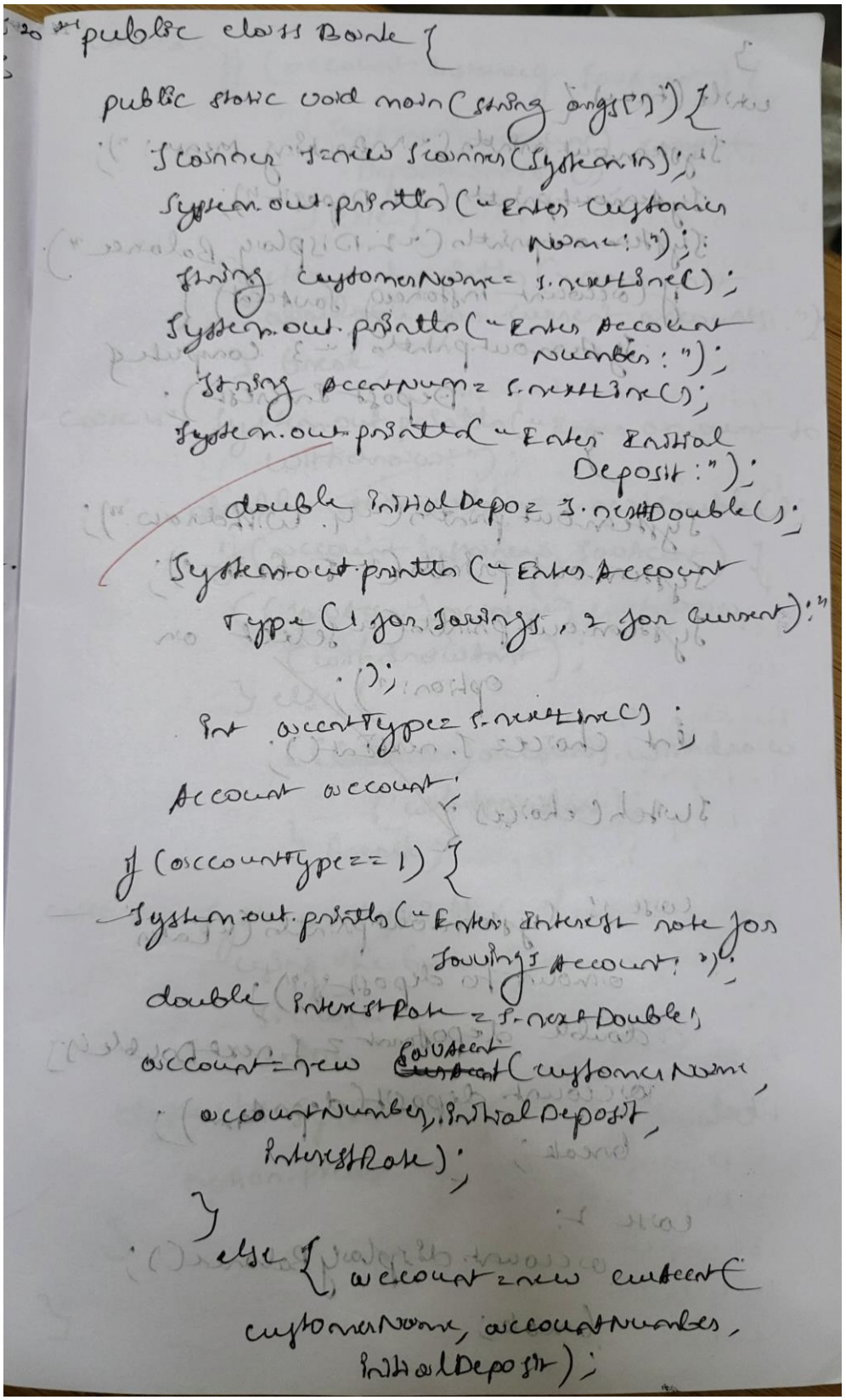
```
protected String accountType;
public Account(String customerName, String
    accountNumber, double initialDeposit,
    String accountType) {
    this.customerName = customerName;
    this.accountNumber = accountNumber;
    this.balance = initialDeposit;
    this.accountType = accountType;
}
public void deposit(double amount) {
    if (amount > 0) {
        balance += amount;
        System.out.println("Deposit
successful. Updated Balance: " + balance);
    } else {
        System.out.println("Deposit amount
must be positive!");
    }
}
public void displayBalance() {
    System.out.println("Balance: " +
        balance);
}
}
class SavAcct extends Account {
private double interestRate;
public SavAcct(String customerName,
```



```
String accountNumber, double InitialDeposit, double
    InterestRate) {
    super(CustomerName, accountNumber, InitialDeposit
        + "Savings");
    this.InterestRate = InterestRate;
}
public void computeDepositInterest() {
    double interest = balance * (InterestRate
                                 /100);
    balance += interest;
    System.out.println("Interest Deposited:"
                       + interest +
    "Updated Balance:" + balance);
}
public void withdraw(double amount)
{ if (amount <= balance) {
    balance -= amount;
    System.out.println("Withdrawal
    Successful. Updated balance" +
                       balance);
} else {
    System.out.println("Insufficient
    balance for withdrawal.");
}
}
}
class CurrentAccount extends Account {
```

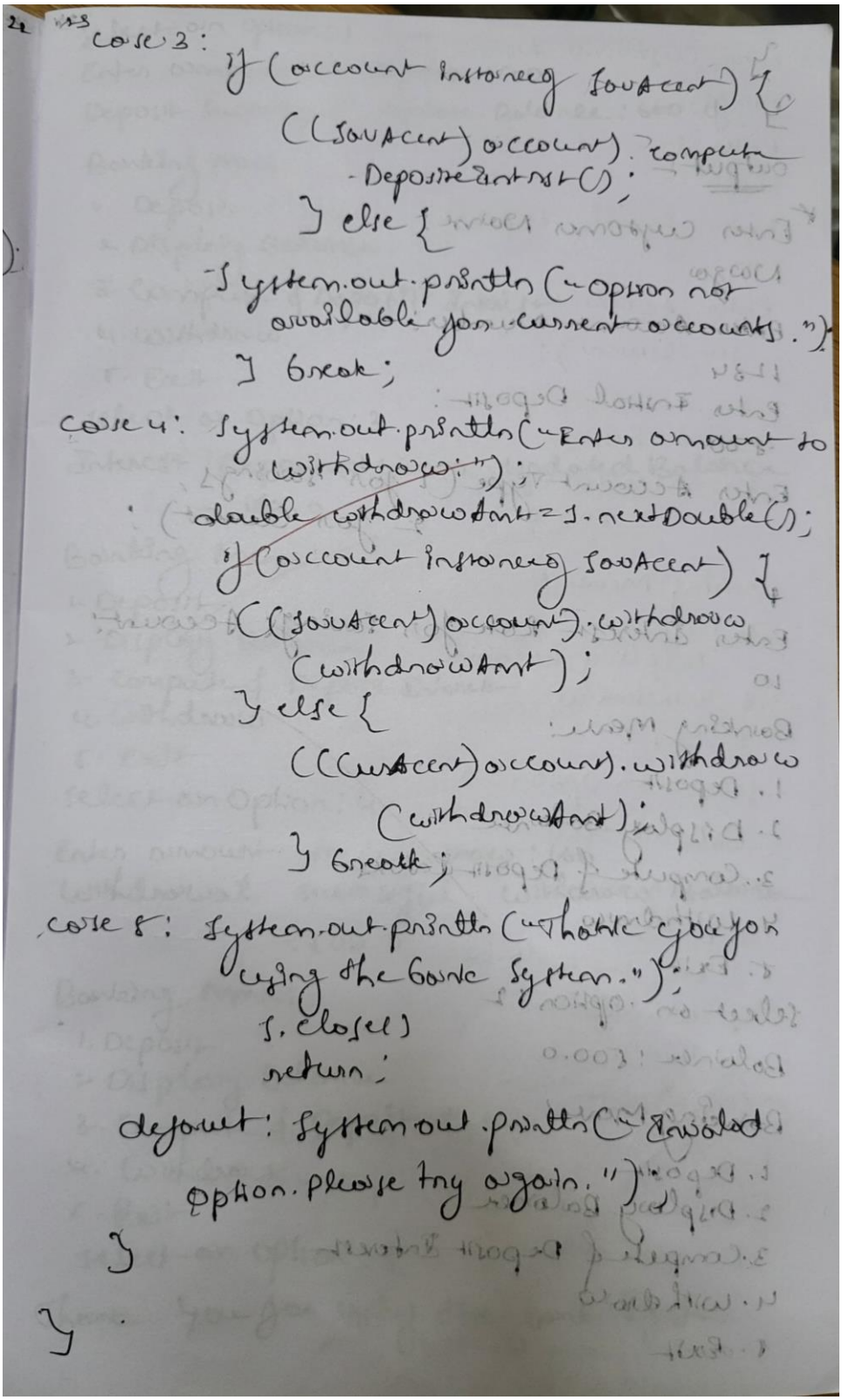


```
private static final double Min_Balance = 500.0;
private static final double Service_charge = 50;
public CurAcct(String customerName,
    String accntNumber, double initialDeposit)
{
    super(customerName, accntNumber,
        initialDeposit, "current");
}
public void withdraw(double amount){
    if(amount <= balance){
        balance -= amount;
        checkMinimumBalance();
        System.out.println("Withdrawal Successful.
            Updated balance: " + balance);
    } else {
        System.out.println("Insufficient
            balance for withdrawal");
    }
}
private void checkMinimumBalance(){
    if(balance < Min_Balance){
        balance -= Service_charge;
        System.out.println("Balance fell below
            Minimum. Service charge imposed.
            Updated balance: " + balance);
    }
}
}
```



```
public class Bank {
    public static void main(String args[]) {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Customer Name:");
        String customerName = s.nextLine();
        System.out.println("Enter Account Number:");
        String accNum = s.nextLine();
        System.out.println("Enter Initial Deposit:");
        double initialDepo = s.nextDouble();
        System.out.println("Enter Account Type (1 for Savings, 2 for Current):");
        int accountType = s.nextLine();
        Account account;
        if (accountType == 1) {
            System.out.println("Enter Interest rate for Savings Account:");
            double interestRate = s.nextDouble();
            account = new SavAcct(customerName, accountNumber, initialDeposit, interestRate);
        } else {
            account = new CurAcct(customerName, accountNumber, initialDeposit);
```

```
}
while (true) {
    System.out.println("\n Banking Menu: ");
    System.out.println("1. Deposit");
    System.out.println("2. Display Balance");
    if (account instanceof SavingsAccount) {
        System.out.println("3. Compute
                Deposit Interest");
    }
    System.out.println("4. Withdraw");
    System.out.println("5. Exit");
    System.out.println("Select an
                option: ");
    int choice = S.nextInt();
    switch (choice) {

    case 1: System.out.println("Enter
        amount to deposit: ");
        double depositAmount = S.nextDouble();
        account.deposit(depositAmount);
        break;

    case 2:
        account.displayBalance();
        break;
```



```
case 3:
    if (account instanceof SavAccount) {
        ((SavAccount) account).computeDepositInt();
    } else {
        System.out.println("Option not available for current accounts.");
    }
    break;
case 4: System.out.println("Enter amount to withdraw:");
    double withdrawAmt = s.nextDouble();
    if (account instanceof SavAccount) {
        ((SavAccount) account).withdraw(withdrawAmt);
    } else {
        ((CurAccount) account).withdraw(withdrawAmt);
    } break;
case 5: System.out.println("Thanks for using the Bank System.");
    s.close();
    return;
default: System.out.println("Invalid option. Please try again.");
    }
}
```

```
}
}
```

output :

```
Enter Customer Name:
Nwgo
Enter Account Number:
1234
Enter Initial Deposit:
500
Enter Account Type (1 for savings,
                    2 for current):
1
Enter Interest Rate for savings Account:
10

Banking Menu:
1. Deposit
2. Display Balance
3. Compute & Deposit Interest
4. Withdraw
5. Exit
Select an Option: 2
Balance: 500.0

Banking Menu:
1. Deposit
2. Display Balance
3. Compute & Deposit Interest
4. Withdraw
5. Exit
```

Select an option: 1
Enter amount to deposit: 100
Deposit successful. Update Balance: 600.0

Banking Menu:
1. Deposit
2. Display Balance
3. Compute & Deposit Interest
4. Withdraw
5. Exit
Select an option: 3
Interest Deposited: 60.0. Updated Balance
-: 660.0.

Banking Menu:
1. Deposit
2. Display Balance
3. Compute & Deposit Interest
4. Withdraw
5. Exit
Select an Option: 4
Enter amount to Withdraw: 160
Withdrawal successful. Withdraw Balance
-: 500.0

10/10

Banking Menu:
1. Deposit
2. Display Balance
3. Compute & Deposit Interest
4. Withdraw
5. Exit
Select an option: 5
Thank you for using the bank system.

29/10/24

Code:

```
import java.util.Scanner;

class Account {
    protected String customerName;
    protected String accountNumber;
    protected double balance;
    protected String accountType;

    public Account(String customerName, String accountNumber, double initialDeposit, String
accountType) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.balance = initialDeposit;
        this.accountType = accountType;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposit Successful. Updated Balance: " + balance);
        } else {
            System.out.println("Deposit amount must be positive.");
        }
    }

    public void displayBalance() {
        System.out.println("Balance: " + balance);
    }
}

class SavAcct extends Account {
    private double interestRate;

    public SavAcct(String customerName, String accountNumber, double initialDeposit, double
interestRate) {
        super(customerName, accountNumber, initialDeposit, "Savings");
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest() {
```

```
        double interest = balance * (interestRate / 100);
        balance += interest;
        System.out.println("Interest Deposited: " + interest + ", Updated Balance: " + balance);
    }

    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawal Successful. Updated Balance: " + balance);
        } else {
            System.out.println("Insufficient balance for withdrawal.");
        }
    }
}

class CurAcct extends Account {
    private static final double MINIMUM_BALANCE = 500;
    private static final double SERVICE_CHARGE = 50;

    public CurAcct(String customerName, String accountNumber, double initialDeposit) {
        super(customerName, accountNumber, initialDeposit, "Current");

        if (initialDeposit < MINIMUM_BALANCE) {
            System.out.println("Error: Initial deposit must be at least " + MINIMUM_BALANCE);
            this.balance = 0;         }
    }

    public void deposit(double amount) {
        super.deposit(amount);
        if (balance >= MINIMUM_BALANCE) {
            System.out.println("Your account meets the minimum balance requirement.");
        }
    }

    public void withdraw(double amount) {
        if (balance - amount < MINIMUM_BALANCE) {
            System.out.println("Warning: Withdrawing this amount will bring your balance below the
minimum. A service charge of "
                    + SERVICE_CHARGE + " will be applied.");
        }
```

```
        if (amount <= balance) {
            balance -= amount;
            checkMinimumBalance();
            System.out.println("Withdrawal Successful. Updated Balance: " + balance);
        } else {
            System.out.println("Insufficient balance for withdrawal.");
        }
    }

    private void checkMinimumBalance() {
        if (balance < MINIMUM_BALANCE) {
            balance -= SERVICE_CHARGE;
            System.out.println("Balance fell below minimum. Service charge imposed. Updated Balance:
" + balance);
        }
    }
}


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

        System.out.println("Enter Customer Name:");
        String customerName = scanner.nextLine();
        System.out.println("Enter Account Number:");
        String accountNumber = scanner.nextLine();
        System.out.println("Enter Initial Deposit:");
        double initialDeposit = scanner.nextDouble();
        System.out.println("Enter Account Type (1 for Savings, 2 for Current):");
        int accountType = scanner.nextInt();

        Account account;

        if (accountType == 1) {
            System.out.println("Enter Interest Rate for Savings Account:");
            double interestRate = scanner.nextDouble();
            account = new SavAcct(customerName, accountNumber, initialDeposit, interestRate);
        } else {
            account = new CurAcct(customerName, accountNumber, initialDeposit);
        }
```

```
while (true) {
    System.out.println("\nBanking Menu:");
    System.out.println("1. Deposit");
    System.out.println("2. Display Balance");
    if (account instanceof SavAcct) {
        System.out.println("3. Compute and Deposit Interest");
    }
    System.out.println("4. Withdraw");
    System.out.println("5. Exit");
    System.out.print("Select an option: ");
    int choice = scanner.nextInt();

    switch (choice) {
        case 1:
            System.out.print("Enter amount to deposit: ");
            double depositAmount = scanner.nextDouble();
            account.deposit(depositAmount);
            break;
        case 2:
            account.displayBalance();
            break;
        case 3:
            if (account instanceof SavAcct) {
                ((SavAcct) account).computeAndDepositInterest();
            } else {
                System.out.println("Option not available for current accounts.");
            }
            break;
        case 4:
            System.out.print("Enter amount to withdraw: ");
            double withdrawAmount = scanner.nextDouble();
            if (account instanceof SavAcct) {
                ((SavAcct) account).withdraw(withdrawAmount);
            } else {
                ((CurAcct) account).withdraw(withdrawAmount);
            }
            break;
        case 5:
            System.out.println("Thank you for using the bank system.");
            scanner.close();
```

```
                return;
            default:
                System.out.println("Invalid option. Please try again.");
        }
    }
  }
}
```

output:

```
D:\MyPack>java Bank
Enter Customer Name:
Naga
Enter Account Number:
1234
Enter Initial Deposit:
100
Enter Account Type (1 for Savings, 2 for Current):
1
Enter Interest Rate for Savings Account:
10

Banking Menu:
1. Deposit
2. Display Balance
3. Compute and Deposit Interest
4. Withdraw
5. Exit
Select an option: 1
Enter amount to deposit: 400
Deposit Successful. Updated Balance: 500.0

Banking Menu:
1. Deposit
2. Display Balance
3. Compute and Deposit Interest
4. Withdraw
5. Exit
Select an option: 3
Interest Deposited: 50.0, Updated Balance: 550.0

Banking Menu:
1. Deposit
2. Display Balance
3. Compute and Deposit Interest
4. Withdraw
5. Exit
Select an option: 4
Enter amount to withdraw: 200
Withdrawal Successful. Updated Balance: 350.0
```

```
Banking Menu:
1. Deposit
2. Display Balance
3. Compute and Deposit Interest
4. Withdraw
5. Exit
Select an option: 5
Thank you for using the bank system.

D:\MyPack>java Bank
Enter Customer Name:
Naga
Enter Account Number:
12345
Enter Initial Deposit:
100
Enter Account Type (1 for Savings, 2 for Current):
2
Error: Initial deposit must be at least 500.0

Banking Menu:
1. Deposit
2. Display Balance
4. Withdraw
5. Exit
Select an option: 2
Balance: 0.0

Banking Menu:
1. Deposit
2. Display Balance
4. Withdraw
5. Exit
Select an option: 1
Enter amount to deposit: 400
Deposit Successful. Updated Balance: 400.0
```

```
Banking Menu:
1. Deposit
2. Display Balance
4. Withdraw
5. Exit
Select an option: 4
Enter amount to withdraw: 200
Warning: Withdrawing this amount will bring your balance below the minimum. A service charge of 50.0 will be applied.
Balance fell below minimum. Service charge imposed. Updated Balance: 150.0
Withdrawal Successful. Updated Balance: 150.0

Banking Menu:
1. Deposit
2. Display Balance
4. Withdraw
5. Exit
Select an option: 2
Balance: 150.0

Banking Menu:
1. Deposit
2. Display Balance
4. Withdraw
5. Exit
Select an option: 5
Thank you for using the bank system.

D:\MyPack>
```

Program 6

Create a package CIE which has two classes- Student and Internals. The class Student has members like usn, name, sem. The class Internals derived from Student 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:

Week-06

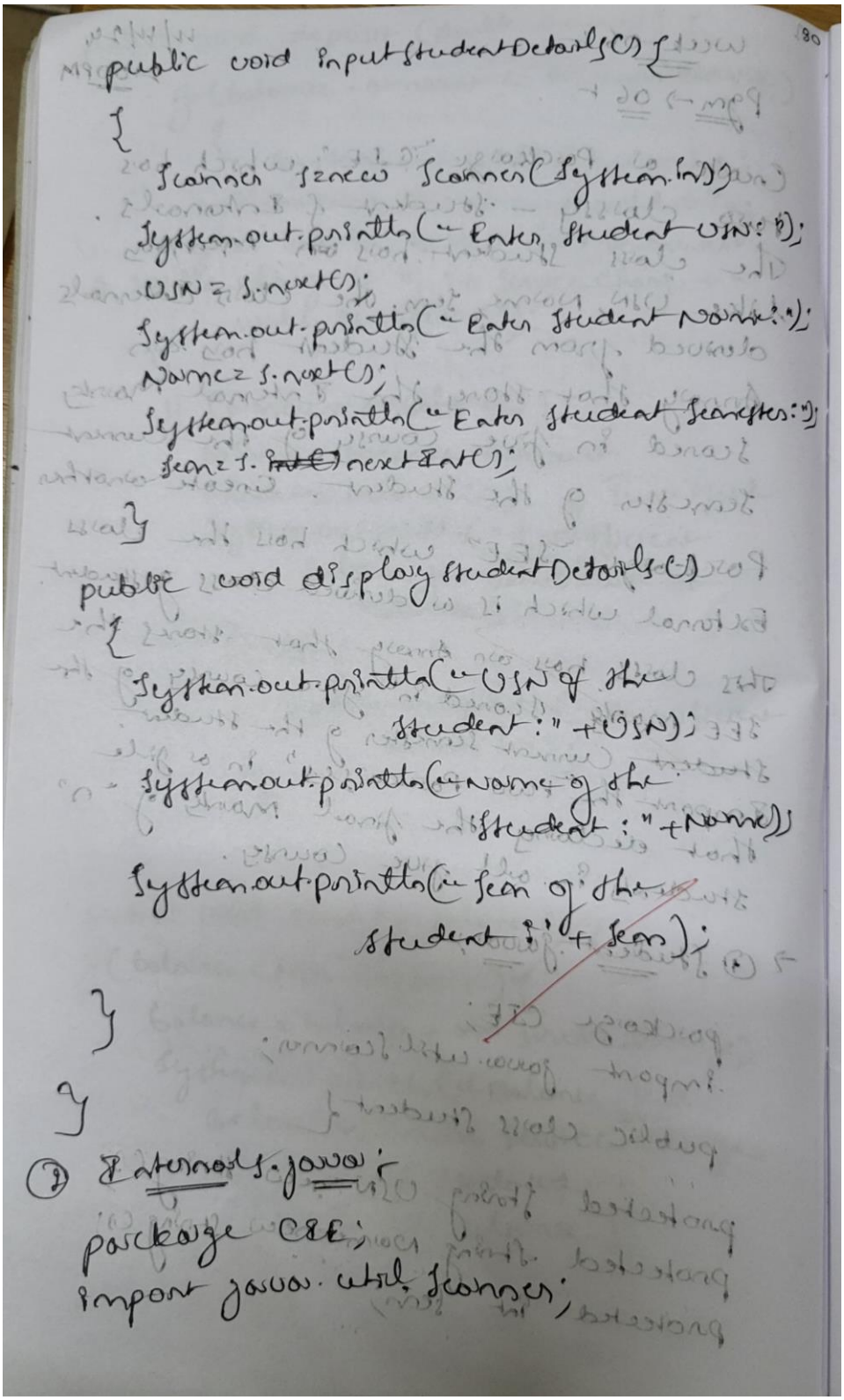
4/11/24
2:00PM

Pgm → 06

Create a Package "CIE" which has two classes - Student & Externals. The class Student has Members like USN, Name, Sem. The class Externals derived from the Student has an Array that stores the External 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 Student Current semester of the Student. Import the two Packages in a file that declares the final Marks of n Students in all five courses.

→ ① Student.java

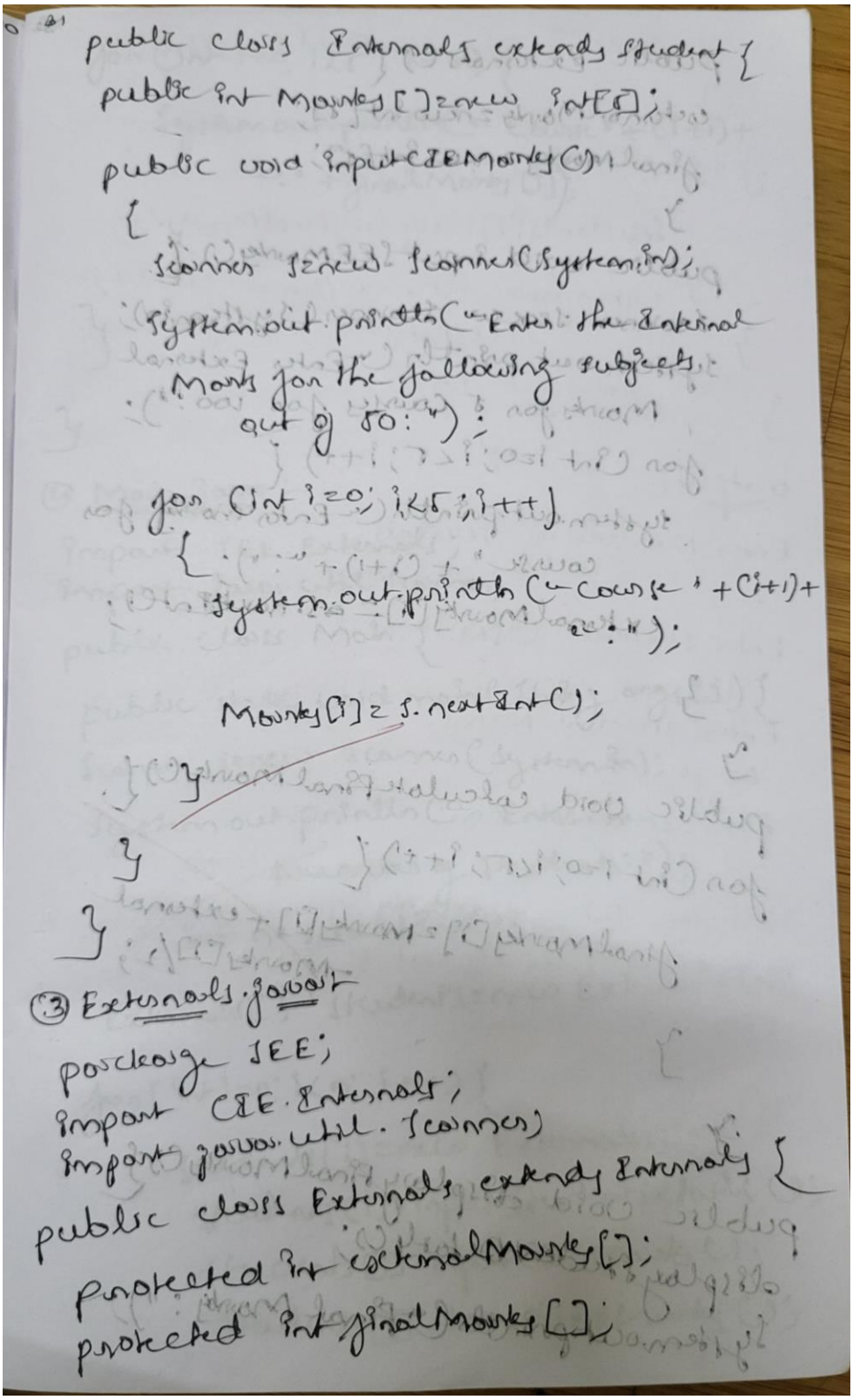
```
package CIE;
import java.util.Scanner;
public class Student {
protected String USN = new String();
protected String Name = new String();
protected int Sem;
```



```
public void inputStudentDetails(){
  {
    Scanner sc=new Scanner(System.in);
    System.out.println("Enter Student USN:");
    USN=s.next();
    System.out.println("Enter Student Name:");
    Name=s.next();
    System.out.println("Enter Student Semester:");
    Sem=s.nextInt();
  }
public void displayStudentDetails()
  {
    System.out.println("USN of the
                Student:"+USN);
    System.out.println("Name of the
                Student:"+Name);
    System.out.println("Sem of the
                Student:"+Sem);
  }
}
```

(2) External.java;

```
package CSE;
import java.util.Scanner;
```



```
public class Internals extends Student {
public int Marks[] = new int[5];

public void inputCIEMarks() {
  Scanner s = new Scanner(System.in);
  System.out.println("Enter the Internal
    Marks for the following subjects,
      out of 50:");
  for (int i=0; i<5; i++) {
    System.out.println("Course " + (i+1) +
                       ":");
    Marks[i] = s.nextInt();
  }
}
}
```

(3) Externals.java

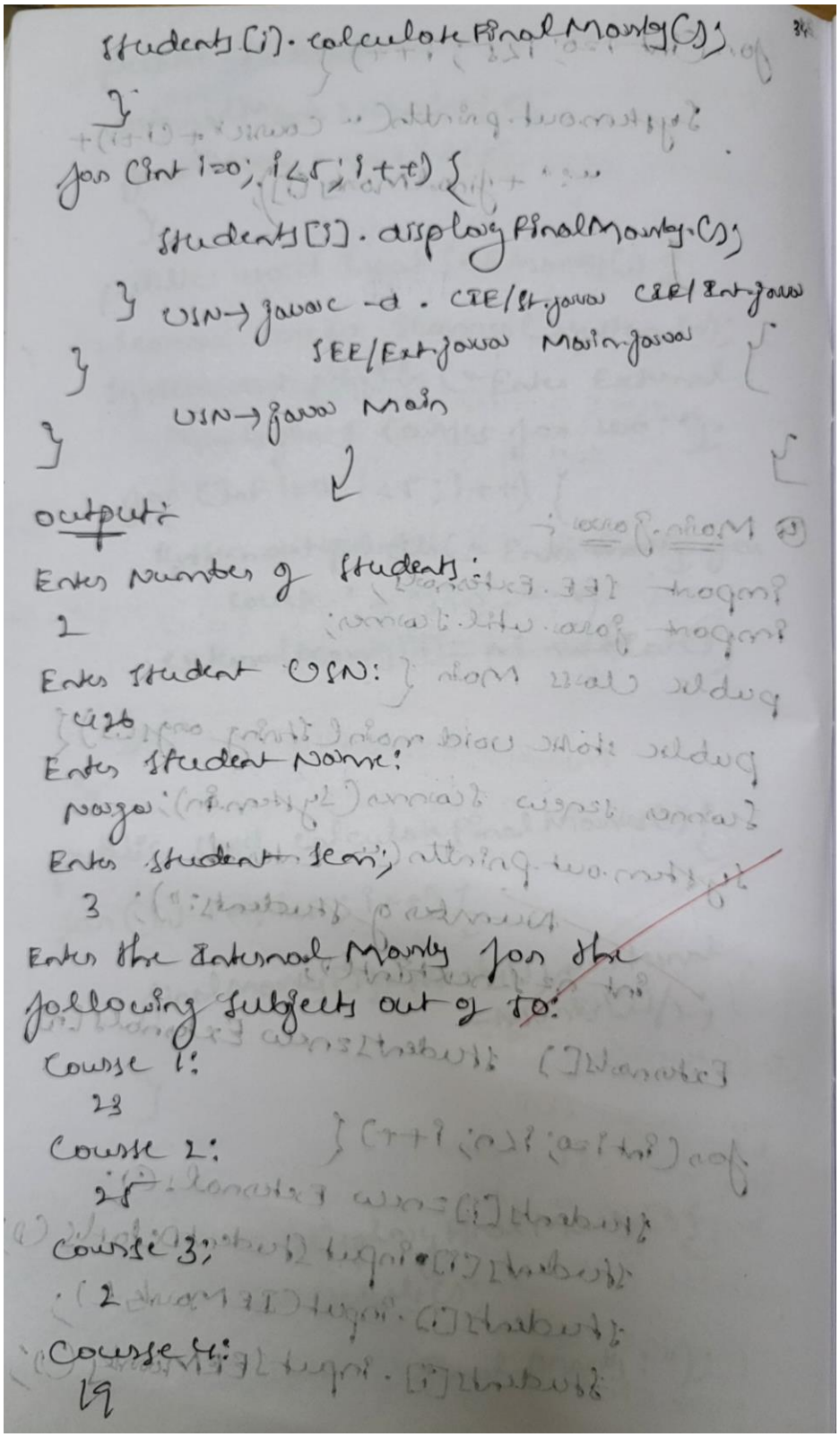
```
package CIE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals {
  protected int externalMarks[];
  protected int finalMarks[];
```

```
public External() {
    externalMarks = new int[5];
    finalMarks = new int[5];
}
public void inputSEEMarks() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter External
        Marks for 5 courses for 100:");
    for (int i=0; i<5; i++) {
        System.out.println("Enter marks for
            course" + (i+1) + ":");
        externalMarks[i] = sc.nextInt();
    }
}
public void calculateFinalMarks() {
    for (int i=0; i<5; i++) {
        finalMarks[i] = marks[i] + external
            Marks[i]/2;
    }
}
public void displayFinalMarks() {
    displayStudentDetails();
    System.out.println("Final Marks:");
```

```
for(int i=0; i<5; i++) {
    System.out.println("Course" + (i+1) +
        ": " + finalMarks[i]);
  }
}
}
```

④ Main.java :-

```
import IEE.Externals;
import java.util.Scanner;
public class Main {
  public static void main(String args[]){
  Scanner sc=new Scanner(System.in);
  System.out.println("Enter the
        Number of students:");
    int n=sc.nextInt();
  Externals[] students=new Externals[n];
for(int i=0; i<n; i++) {
    students[i]=new Externals();
    students[i].inputStudentDetails();
    students[i].inputIEEMarks();
    students[i].inputIEEMarks();
```



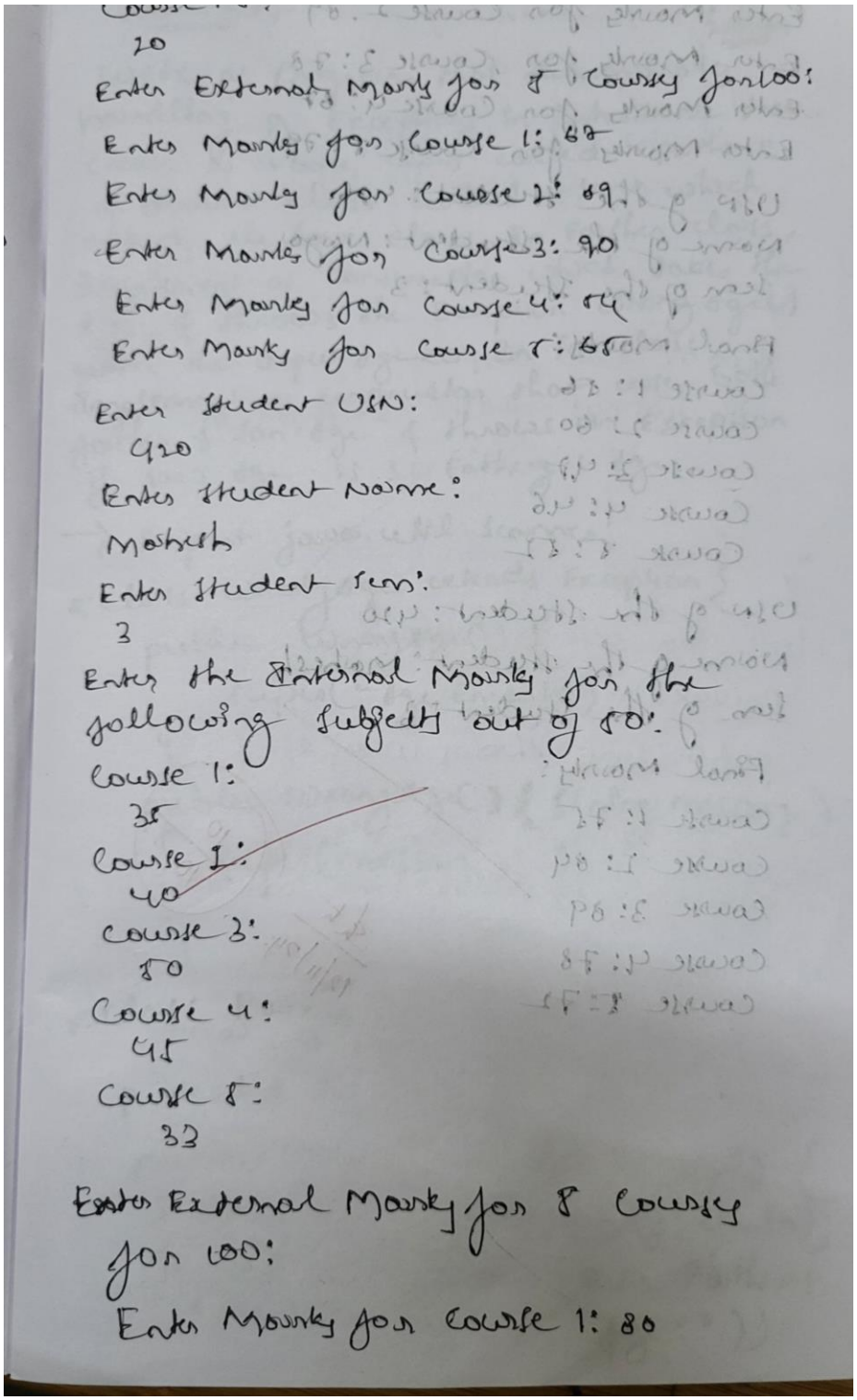
```
        students[i].calculateFinalMarks();
    }
    for(int i=0; i<5; i++){
        students[i].displayFinalMarks();
    }
}
```

USN → javac -d . CIE/St.java CIE/Int.java SEE/Ext.java Main.java

USN → java Main

output:

```
Enter Number of students:
2
Enter Student USN:
4126
Enter Student Name:
Naga
Enter student sem:
3
Enter the Internal Marks for the following subjects out of 50:
Course 1:
23
Course 2:
25
Course 3:
Course 4:
19
```



20
Enter External Marks for 5 Courses for 100:
Enter Marks for Course 1: 82
Enter Marks for Course 2: 89
Enter Marks for Course 3: 90
Enter Marks for Course 4: 84
Enter Marks for Course 5: 85
Enter Student USN:
4120
Enter Student Name:
Mahesh
Enter Student Sem:
3
Enter the Internal Marks for the following Subjects out of 50:
Course 1:
35
Course 2:
40
Course 3:
50
Course 4:
45
Course 5:
33
Enter External Marks for 5 Courses for 100:
Enter Marks for Course 1: 80

Enter Marks for Course 2: 89
Enter Marks for Course 3: 78
Enter Marks for Course 4: 67
Enter Marks for Course 5: 79
USN of the Student: 426
Name of the Student: Nagaj
Sem of the Student: 3
Final Marks:
Course 1: 56
Course 2: 60
Course 3: 47
Course 4: 46
Course 5: 52
USN of the Student: 420
Name of the Student: Mahesh
Sem of the Student: 3
Final Marks:
Course 1: 75
Course 2: 84
Course 3: 89
Course 4: 78
Course 5: 72

12/11/24

Code:

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 s=new Scanner(System.in);
System.out.println("Enter Student USN :");
usn=s.next();
System.out.println("Enter Student Name :");
name=s.next();
System.out.println("Enter Student Semester :");
sem=s.nextInt();
}

public void displayStudentDetails()
{
System.out.println("USN of the Student : "+ usn);
System.out.println("Name of the Student : "+ name);
System.out.println("Semester of the Student : "+ sem);
}
}:
```

Internals.java

```
package CIE;
import java.util.Scanner;

public class Internals extends Student{
public int marks[] = new int[5];

public void inputCIEmarks()
{
Scanner s=new Scanner(System.in);
System.out.println("Enter the Internal Marks for the following Subjects out of 50 :");
```

```
for (int i=0;i<5;i++)
{
System.out.println("Course "+(i+1)+":");
marks[i]=s.nextInt();
}

}
}
```

Externals.java

```
package SEE;
import CIE.Internals;
import java.util.Scanner;

public class Externals extends Internals {
protected int externalmarks[];
protected int finalMarks[];

public Externals(){
externalmarks=new int[5];
finalMarks = new int[5];
}

public void inputSEEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter External marks for 5 courses for 100: ");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for course " + (i + 1) + ": ");
            externalmarks[i] = s.nextInt();
        }
    }

public void calculateFinalMarks(){
for (int i = 0; i < 5; i++) {
            finalMarks[i] = marks[i] + externalmarks[i]/2;
}
    }

public void displayFinalMarks(){
displayStudentDetails();
System.out.println("Final Marks: ");
```

```
        for (int i = 0; i < 5; i++) {
            System.out.println("Course "+(i+1)+":"+finalMarks[i]);
        }
}
}
```

main.java

```
import SEE.Externals;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
Scanner s = new Scanner(System.in);
System.out.println("Enter number of students: ");
        int n = s.nextInt();
Externals[] students = new Externals[n];

for (int i = 0; i < n; i++) {
            students[i] = new Externals();
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
            students[i].calculateFinalMarks();
        }

for (int i = 0; i < n; i++) {
            students[i].displayFinalMarks();
        }
}
}
```

Output:

```
D:\MyPack>java Main.java
Enter number of students:

2
Enter Student USN :
426
Enter Student Name :
naga
Enter Student Semester :
3
Enter the Internal Marks for the following Subjects out of 50 :
Course 1:
23
Course 2:
25
Course 3:
2
Course 4:
19
Course 5:
20
Enter External marks for 5 courses for 100:
Enter marks for course 1: 67
Enter marks for course 2: 89
Enter marks for course 3: 90
Enter marks for course 4: 54
Enter marks for course 5: 65
Enter Student USN :
420
Enter Student Name :
mahesh
Enter Student Semester :
3
Enter the Internal Marks for the following Subjects out of 50 :
Course 1:
35
Course 2:
40
Course 3:
50
Course 4:
45
Course 5:
33
Enter External marks for 5 courses for 100:
Enter marks for course 1: 80
Enter marks for course 2: 89
Enter marks for course 3: 78
Enter marks for course 4: 67
Enter marks for course 5: 79
USN of the Student : 426
Name of the Student : naga
Semester of the Student : 3
Final Marks:
Course 1:56
Course 2:69
Course 3:47
Course 4:46
Course 5:52
USN of the Student : 420
Name of the Student : mahesh
Semester of the Student : 3
Final Marks:
Course 1:75
Course 2:84
Course 3:89
Course 4:78
Course 5:72
```

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<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is >=father’s age

Algorithm:

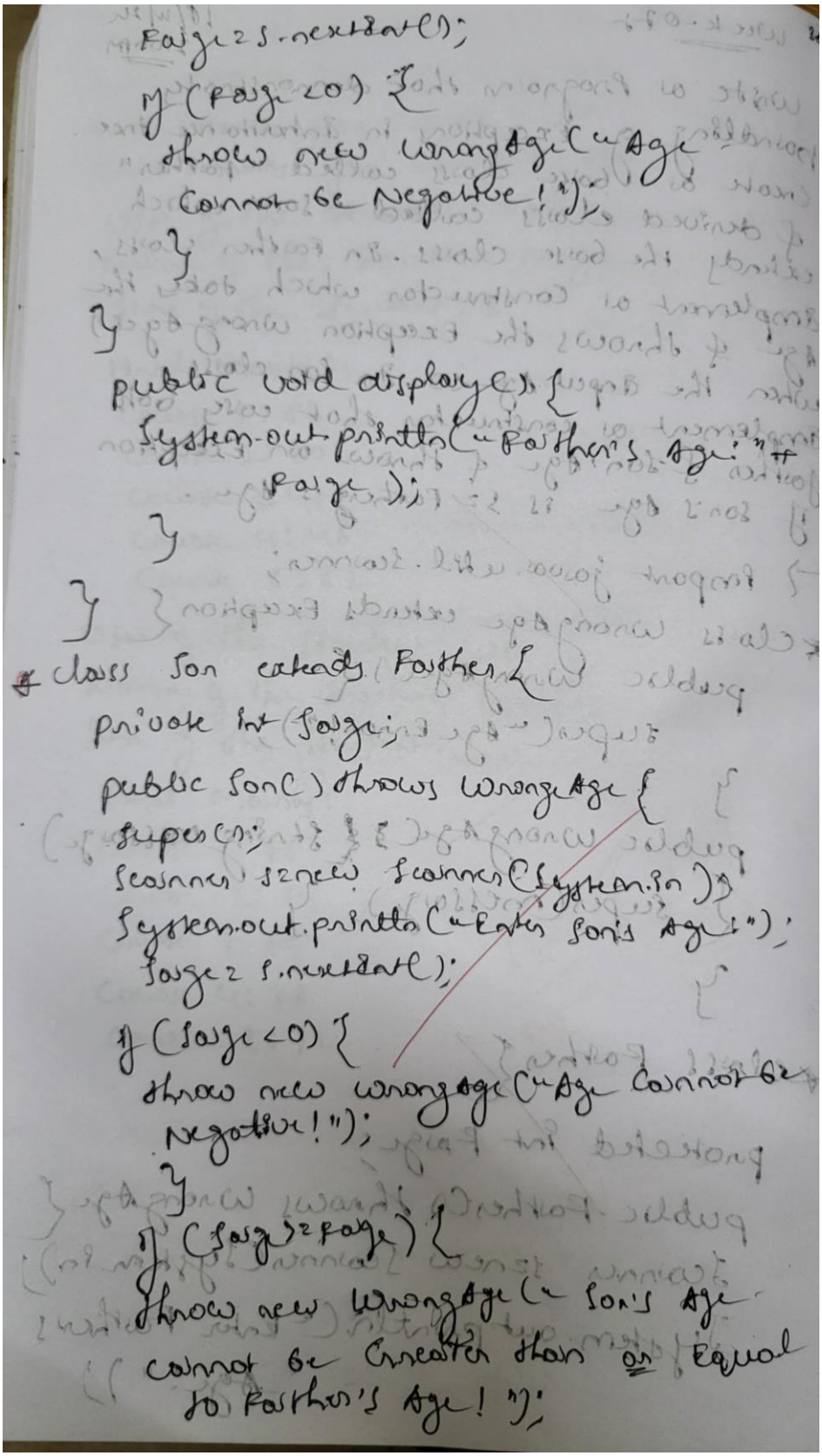
26 39 Week-07 10/11/24
2:00pm

Write a Program that demonstrate handling of Exception in Inheritance tree. Create a base class called "Father" & derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age & throws the Exception WrongAge() when the input age<0. In Son class, implement a constructor that takes both father & son's age & throws an Exception if son's age is >= Father's age.

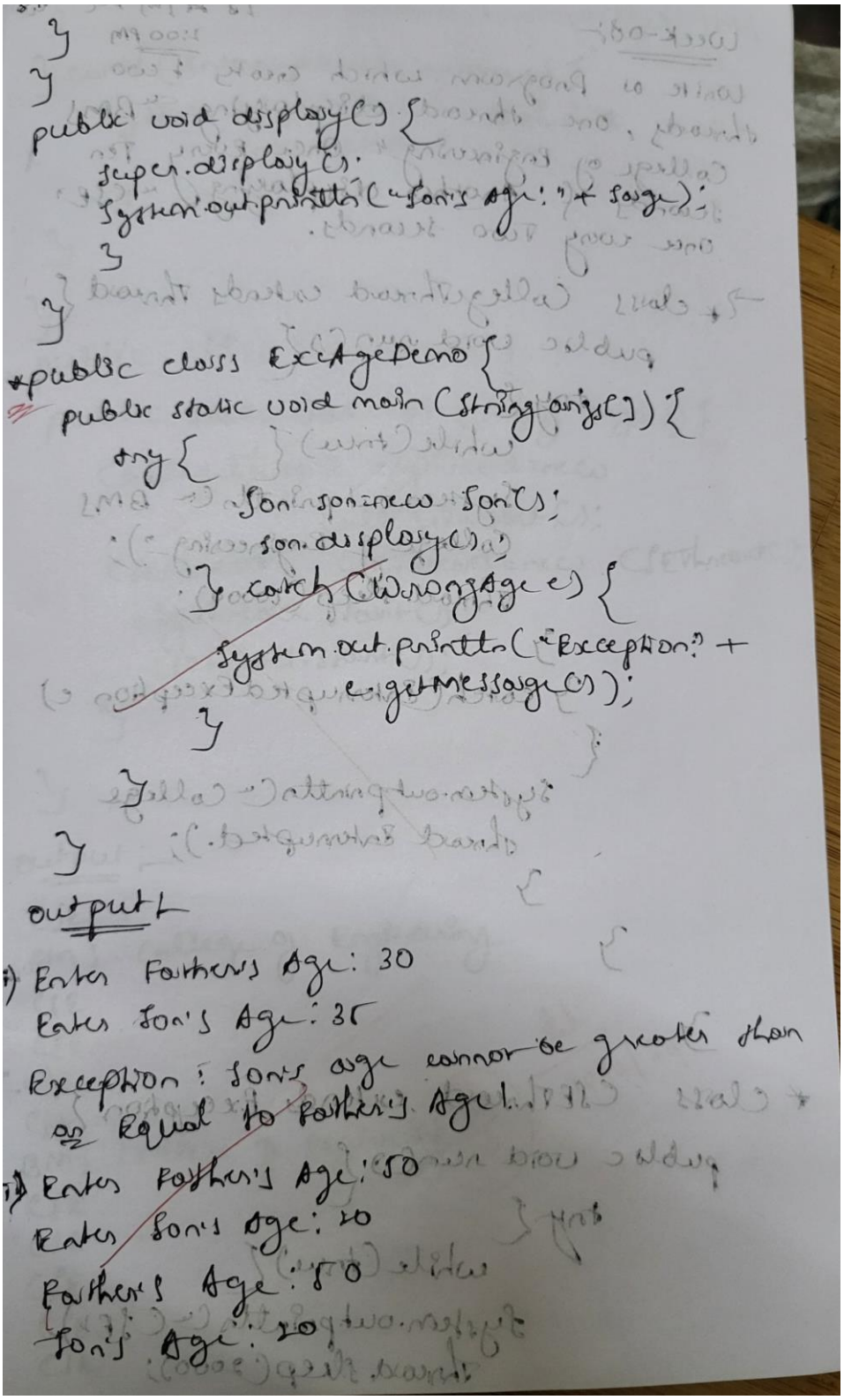
→ import java.util.Scanner;

class WrongAge extends Exception {
    public WrongAge() {
        super("Age Error!");
    }
    public WrongAge(String message) {
        super(message);
    }
}

class Father {
    protected int Fage;
    public Father() throws WrongAge {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Father's Age:");



```
Fage2 s.nextInt();
if (Fage<0) {
  throw new WrongAge("Age
  Cannot be Negative!");
}
}
public void display() {
System.out.println("Father's Age:"+
        Fage);
}
}
class Son extends Father{
private int Sage;
public Son() throws WrongAge {
super();
Scanner s2=new Scanner(System.in);
System.out.println("Enter Son's Age:");
Sage2 s.nextInt();
if (Sage<0) {
throw new WrongAge("Age Cannot be
  Negative!");
}
if (Sage>=Fage) {
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: " + sage);
    }
}
public class ExcAgeDemo {
    public static void main (String args[]) {
        try {
            Son son=new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Exception: " +
                e.getMessage());
        }
    }
}
```

output:

i) Enter Father's Age: 30
Enter Son's Age: 35
Exception: Son's age cannot be greater than or Equal to Father's Age!

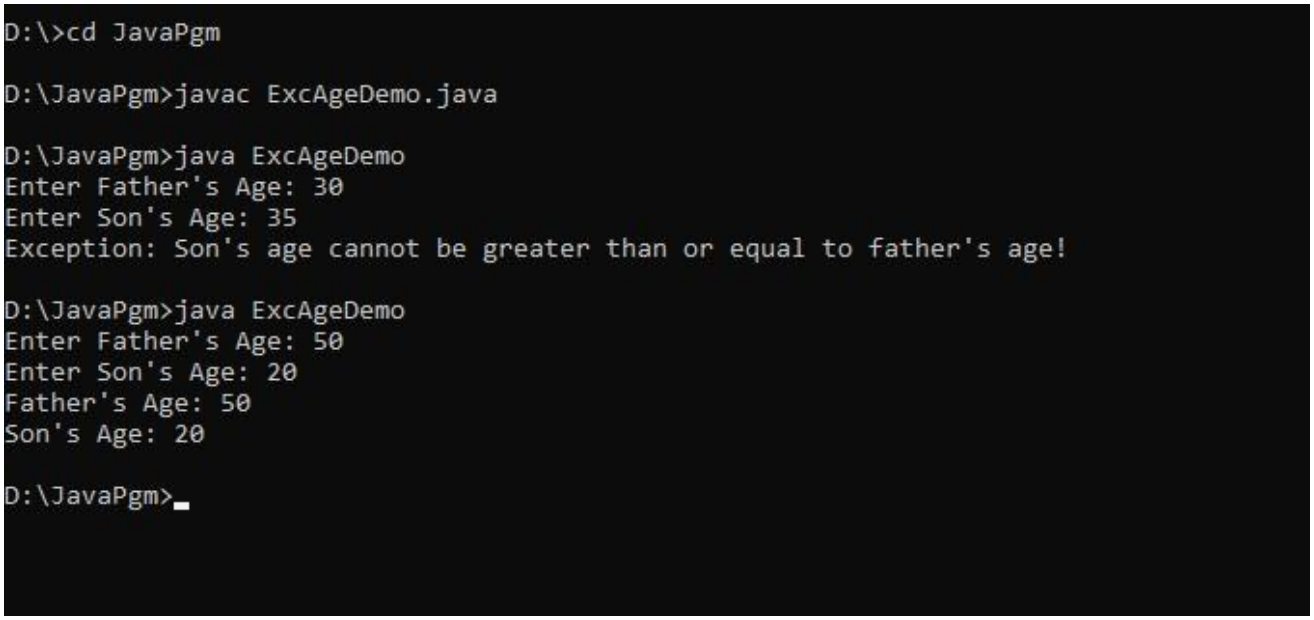
ii) Enter Father's Age: 50
Enter Son's Age: 20
Father's Age: 50
Son's Age: 20

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!");
        }
        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 ExcAgeDemo{
    public static void main(String[] args) {
        try {
            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

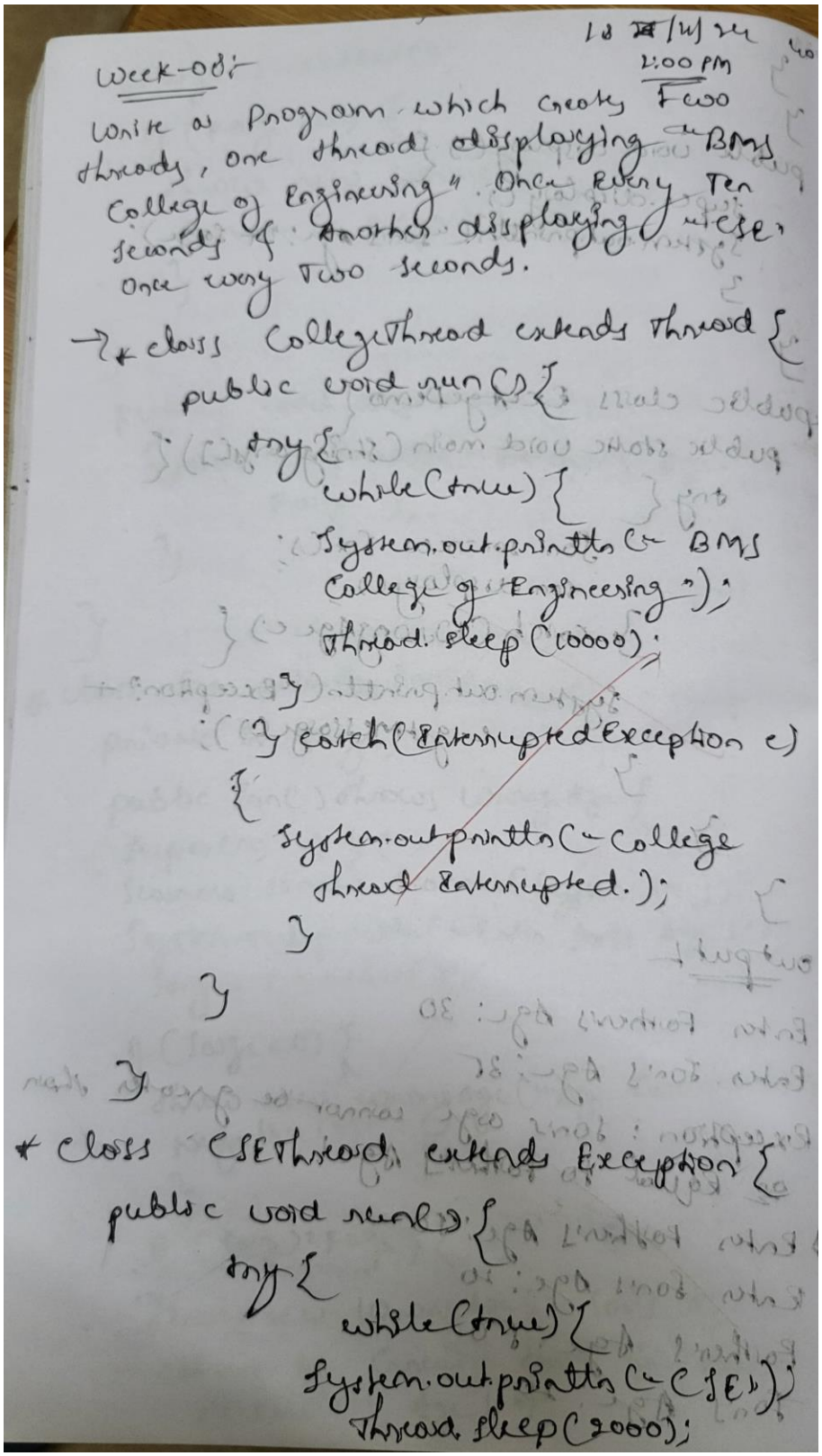
Output:


D:\>cd JavaPgm
D:\JavaPgm>javac ExcAgeDemo.java
D:\JavaPgm>java ExcAgeDemo
Enter Father's Age: 30
Enter Son's Age: 35
Exception: Son's age cannot be greater than or equal to father's age!
D:\JavaPgm>java ExcAgeDemo
Enter Father's Age: 50
Enter Son's Age: 20
Father's Age: 50
Son's Age: 20
D:\JavaPgm>

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.

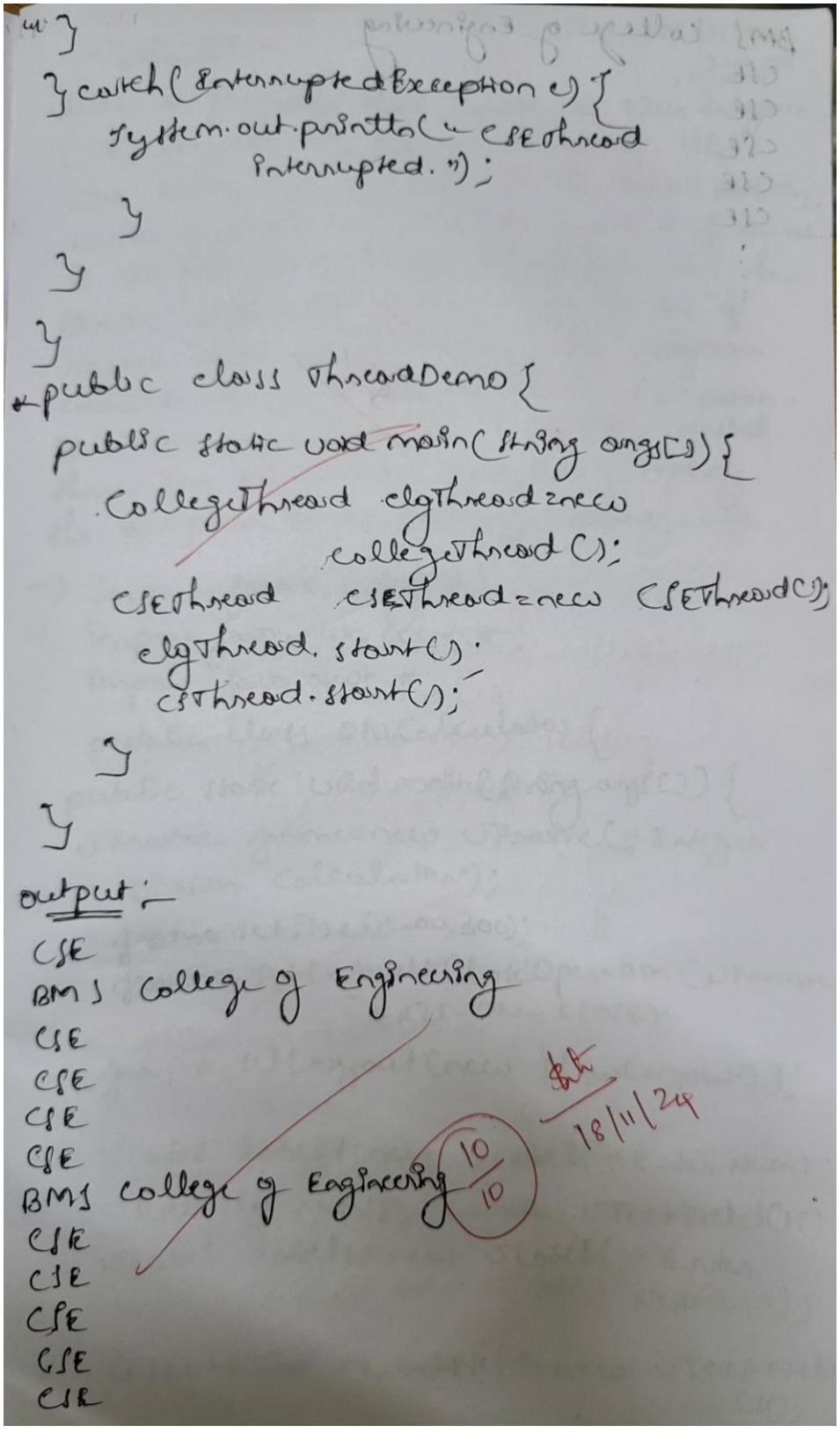
Algorithm:



Week-08:

Write a Program which creates two threads, one thread displaying "BMS College of Engineering" once every Ten seconds & another displaying "CSE" once every Two seconds.

```
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("College Thread Interrupted.");
        }
    }
}
class CSEThread extends Exception {
    public void run() {
        try {
            while(true) {
                System.out.println("CSE");
                Thread.sleep(2000);
```



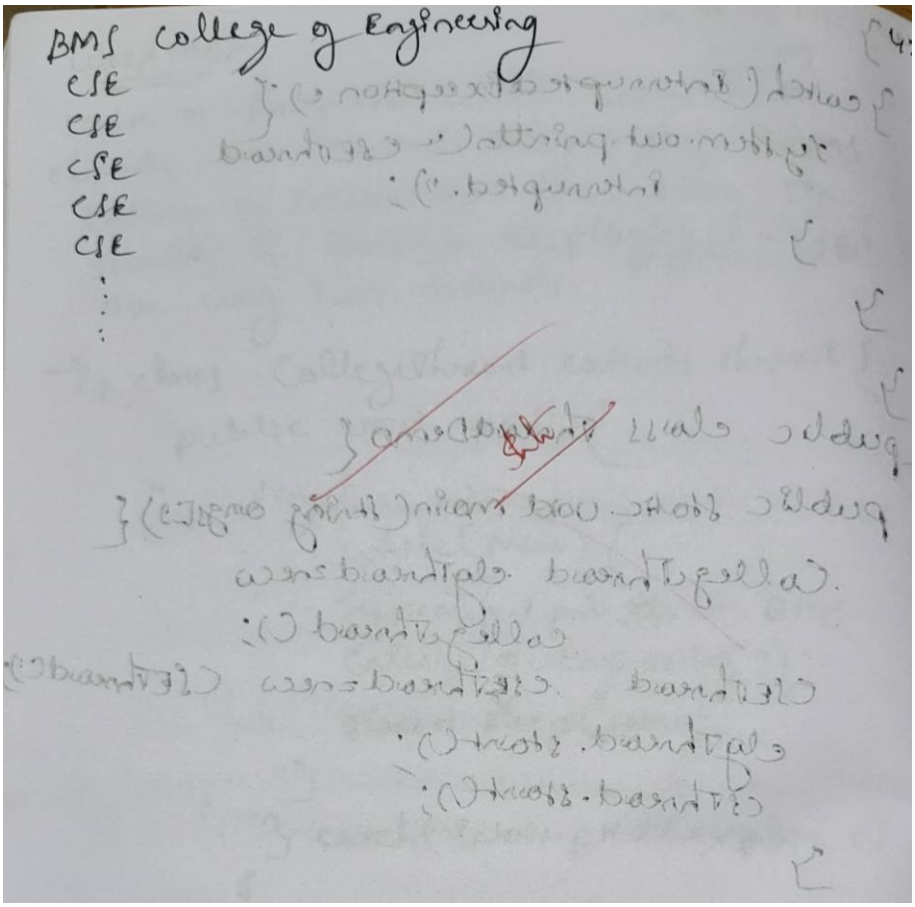
```
    }
  }catch(InterruptedException e){
      System.out.println("CSEThread
              interrupted.");
    }
  }
}
public class ThreadDemo{
  public static void main(String args[]){
    CollegeThread clgThread=new
                  collegeThread();
    CSEThread cseThread=new CSEThread();
     clgThread.start();
     cseThread.start();
  }
}
```

output:-

```
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
```

10/10

18/11/24



.

Code:

```
class CollegeThread extends Thread {
   public void run() {
      try {
         while (true) {
            System.out.println("BMSCE");
            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 ThreadDemo{
    public static void main(String[] args) {
       CollegeThread collegeThread = new CollegeThread();
       CSEThread cseThread = new CSEThread();
       collegeThread.start();
       cseThread.start();
    }
}
```

Output:

```
D:\JavaPgm>javac ThreadDemo.java

D:\JavaPgm>java ThreadDemo
CSE
BMSCE
CSE
CSE
CSE
CSE
BMSCE
CSE
CSE
CSE
CSE
CSE
BMSCE
CSE
CSE
CSE
CSE
CSE
```

Program 9

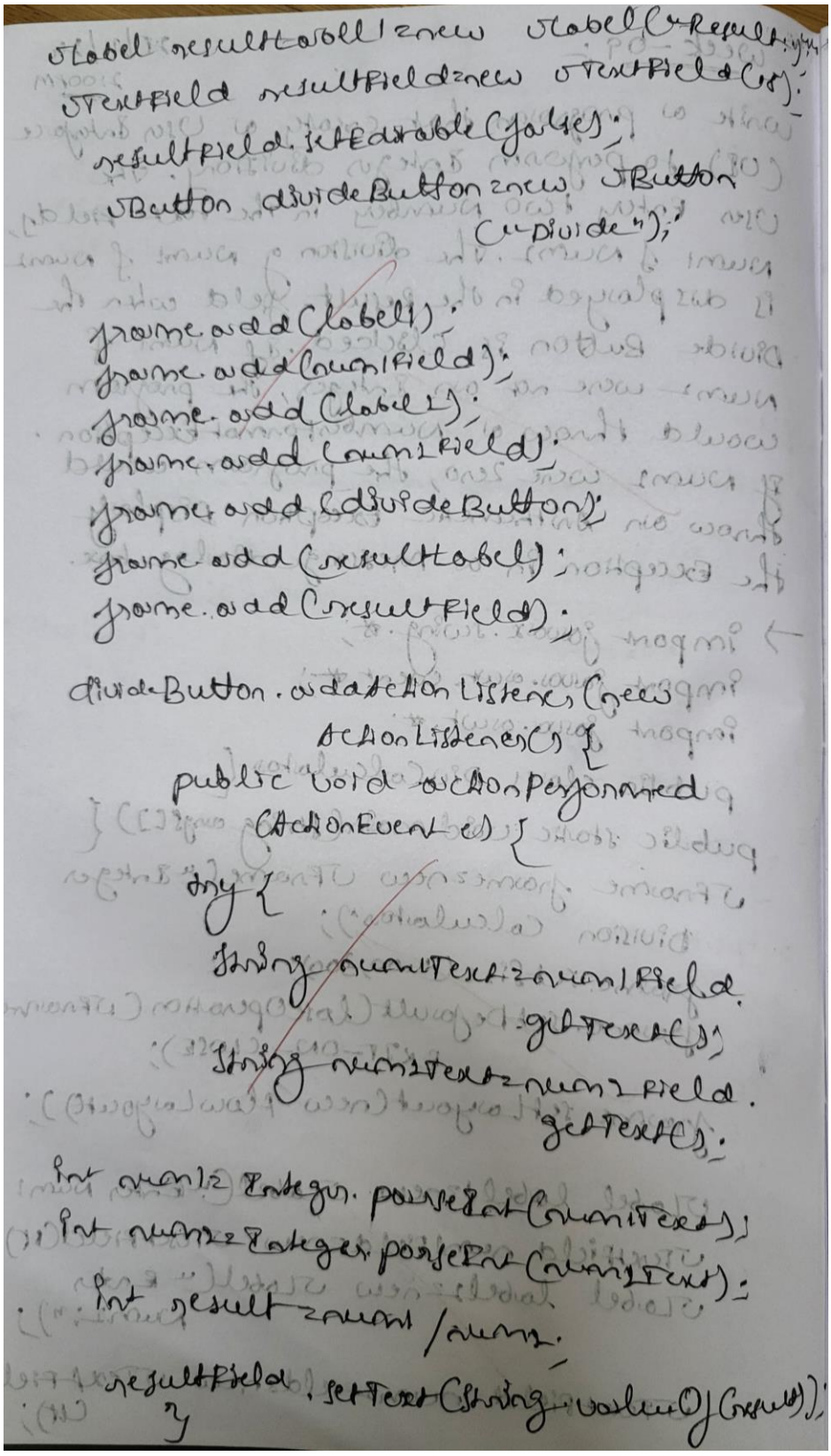
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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

Algorithm:

Week-09 :-

Write a program that creates a User Interface (UI) to perform integer division. The User Enters two Numbers in the text fields, Num1 & Num2. The division of Num1 & Num2 is displayed in the Result field when the Divide Button is clicked. If Num1 & Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were zero, the program would throw an ArithmeticException Display the Exception in a MessageDialog box.

→
```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;
public class DivCalculator {
public static void main(String args[]) {
JFrame frame=new JFrame("Integer
  Division Calculator");
frame.setSize(400,300);
frame.setDefaultCloseOperation(JFrame.
                EXIT_ON_CLOSE);
frame.setLayout(new FlowLayout());

JLabel label1=new JLabel("Enter Num1 :");
JTextField num1Field=new JTextField(10);
JLabel label2=new JLabel("Enter
                Num2:");
JTextField num2Field=new JTextField
                (10);
```



```
JLabel resultLabel = new JLabel("Result:");
JTextField resultField = new JTextField(10);
resultField.setEditable(false);
JButton divideButton = new JButton
                        ("Divide");

frame.add(label1);
frame.add(num1Field);
frame.add(label2);
frame.add(num2Field);
frame.add(divideButton);
frame.add(resultLabel);
frame.add(resultField);

divideButton.addActionListener(new
                ActionListener() {
    public void actionPerformed
        (ActionEvent e) {
        try {
            String num1Text = num1Field.
                                getText();
            String num2Text = num2Field.
                                getText();

int num1 = Integer.parseInt(num1Text);
int num2 = Integer.parseInt(num2Text);
int result = num1 / num2;
resultField.setText(String.valueOf(result));
        }
```

```
} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(frame,
    "please enter valid integers.", "Input
    Error", JOptionPane.ERROR_MESSAGE);
} catch (ArithmeticException ex) {
    JOptionPane.showMessageDialog(frame,
    "Cannot divide by zero.", "Arithmetic
    Error", JOptionPane.ERROR_MESSAGE);
}
});
frame.setVisible(true);
```

output:-

case 1:-

Enter Num 1: 10

Enter Num 2: 2

Divide

Result: 5

case 2:-

Enter Num 1: 10

Enter Num 2: 0

Divide

Result:

⊗ Cannot divide by zero

case 3:-

Enter Num 1: dc

Enter Num 2: 2

Divide

Result

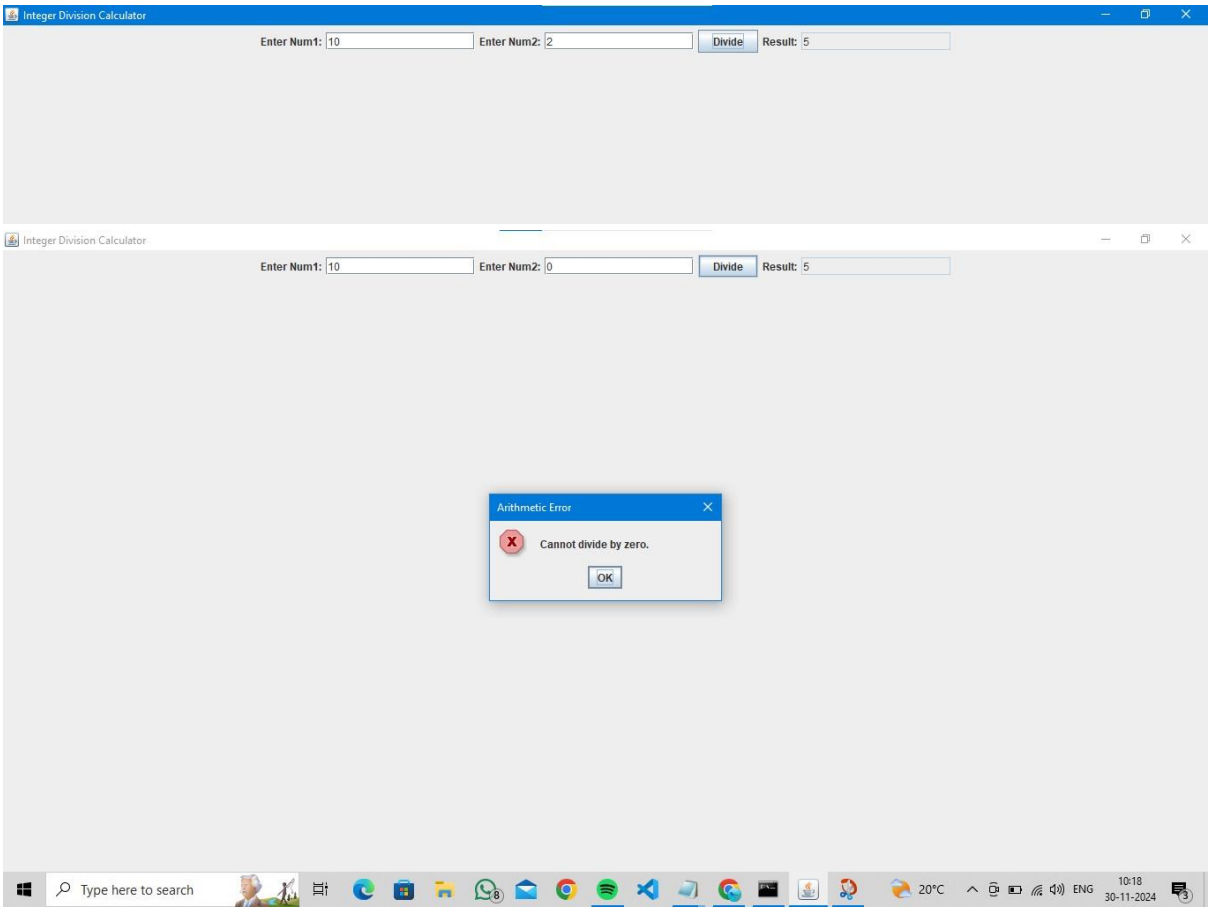
⊗ Please enter valid Integers

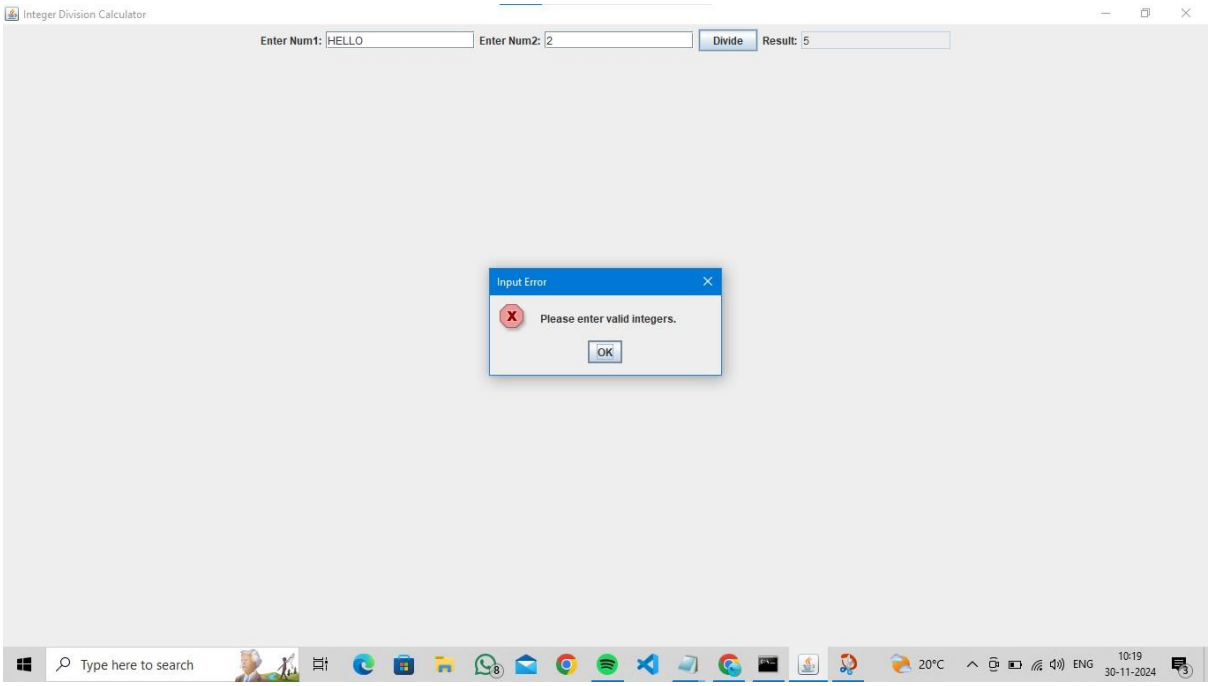
Code:

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class DivCalculator {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Integer Division Calculator");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());

        JLabel label1 = new JLabel("Enter Num1:");
        JTextField num1Field = new JTextField(15);
        JLabel label2 = new JLabel("Enter Num2:");
        JTextField num2Field = new JTextField(15);
        JLabel resultLabel = new JLabel("Result:");
        JTextField resultField = new JTextField(15);
        resultField.setEditable(false); // Result field is non-editable
        JButton divideButton = new JButton("Divide");

        frame.add(label1);
        frame.add(num1Field);
        frame.add(label2);
        frame.add(num2Field);
        frame.add(divideButton);
        frame.add(resultLabel);
        frame.add(resultField);

        divideButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    String num1Text = num1Field.getText();
                    String num2Text = num2Field.getText();

                    int num1 = Integer.parseInt(num1Text);
                    int num2 = Integer.parseInt(num2Text);

                    int result = num1 / num2;
```

```
                resultField.setText(String.valueOf(result));
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(frame, "Please enter valid integers.",
"Input Error", JOptionPane.ERROR_MESSAGE);
            } catch (ArithmeticException ex) {
                JOptionPane.showMessageDialog(frame, "Cannot divide by zero.",
"Arithmetic Error", JOptionPane.ERROR_MESSAGE);
            }
        }
    });

    frame.setVisible(true);
  }
}
```

Output:




Integer Division Calculator
Enter Num1: HELLO
Enter Num2: 2
Divide
Result: 5
Input Error
Please enter valid integers.
OK
Type here to search
20°C
ENG
10:19
30-11-2024

Program 10

Open Ended Exercise

Demonstrate Inter process Communication and deadlock

Algorithm:

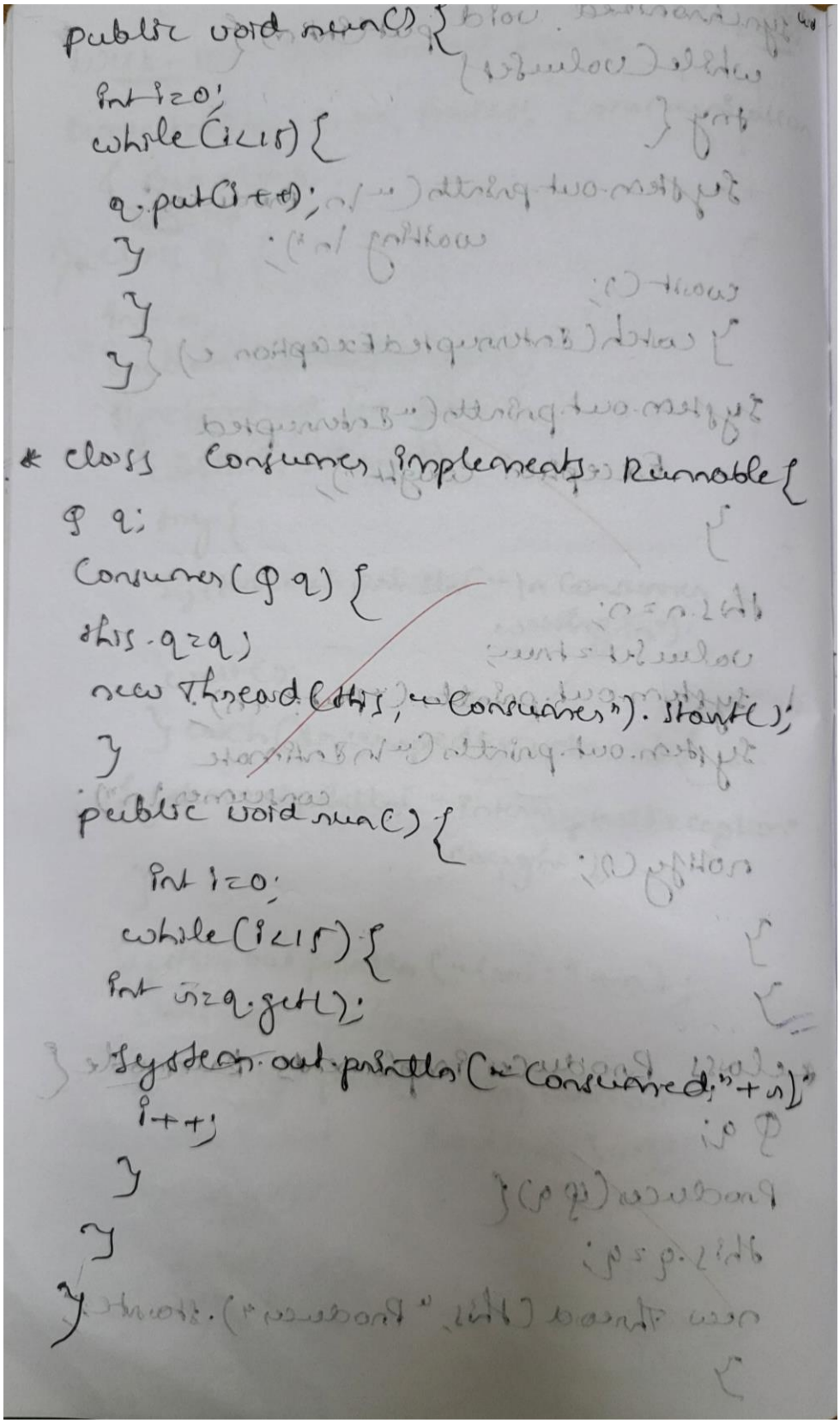
Week-10 :- Open Ended Exercise.

Demonstrating Inter Process Communication & Deadlock.

IPC →

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get() {
        while(!valueSet)
        try {
            System.out.println("\n Consumer
                                waiting\n");
            wait();
        } catch(InterruptedException e) {
            System.out.println("InterruptedException
                                caught");
        }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\n Informing
                            Producer\n");
        notify();
        return n;
    }
```

```
synchronized void put(int n) {
  while(valueSet)
   try {
   System.out.println("\n Producer
                  waiting \n");
   wait();
   } catch(InterruptedException e) {
   System.out.println("Interrupted
       Exception caught");
   }
   this.n = n;
   valueSet = true;
   System.out.println("Put: "+n);
   System.out.println("\n Intimate
                        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 {
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:

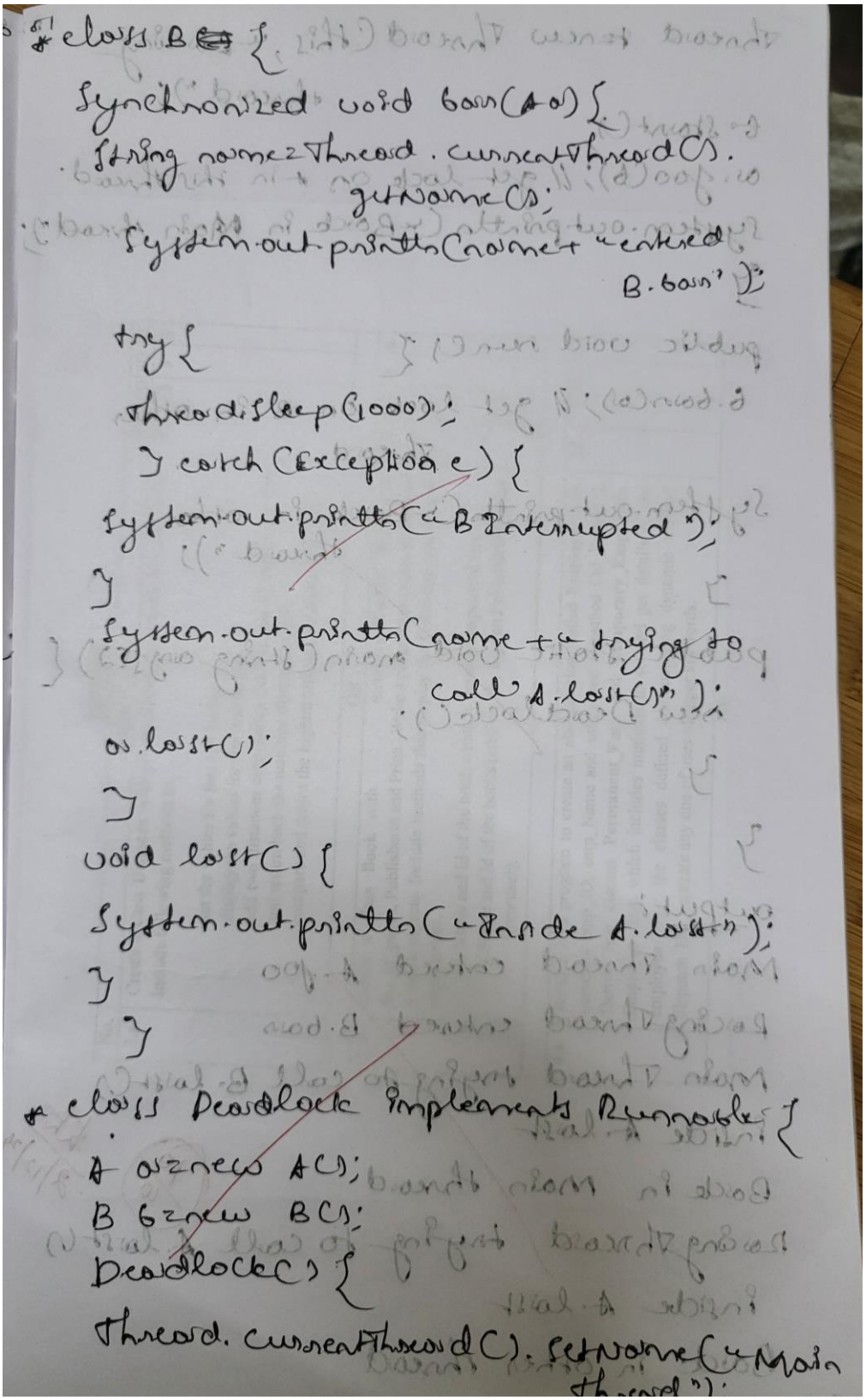
```
Press Control-C to stop
Put:0
Notify Consumer
Producer Waiting
Got:0
Notify Producer
Put:1
Consumed:0
Notify Consumer
Producer Waiting
Got:1
Notify Producer
Put:2
Consumed 1
```

Notify Continues
Producer waiting
Cnt: 2

Notify Producer
Consumed: 2
Put: 3 ... etc

11) Deadlock Pgm:

```
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 A.last");
    }
}
class Deadlock implements Runnable {
    A a = new A();
    B b = new B();
    Deadlock() {
        Thread.currentThread().setName("Main
                thread");
```

```
Thread t=new Thread(this, "Racing thread");
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 :-

Main Thread entered A.foo
Racing Thread entered B.bar
Main Thread trying to call B.last()
Inside A.last
Back in Main thread
Racing Thread trying to call A.last()
Inside A.last
Back in other thread

10/10
8/12/24

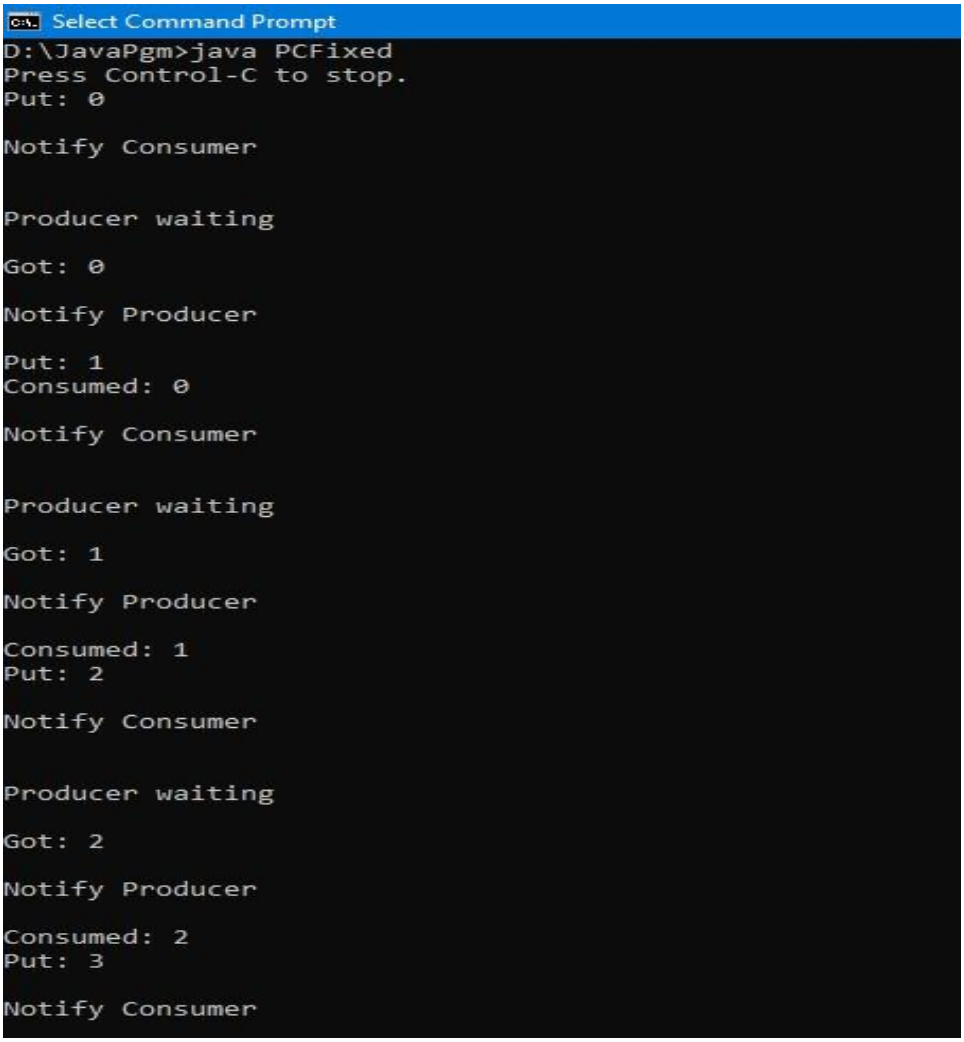
Code 10A:

```
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("\nNotify 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("\nNotify Consumer\n");
      notify();
```

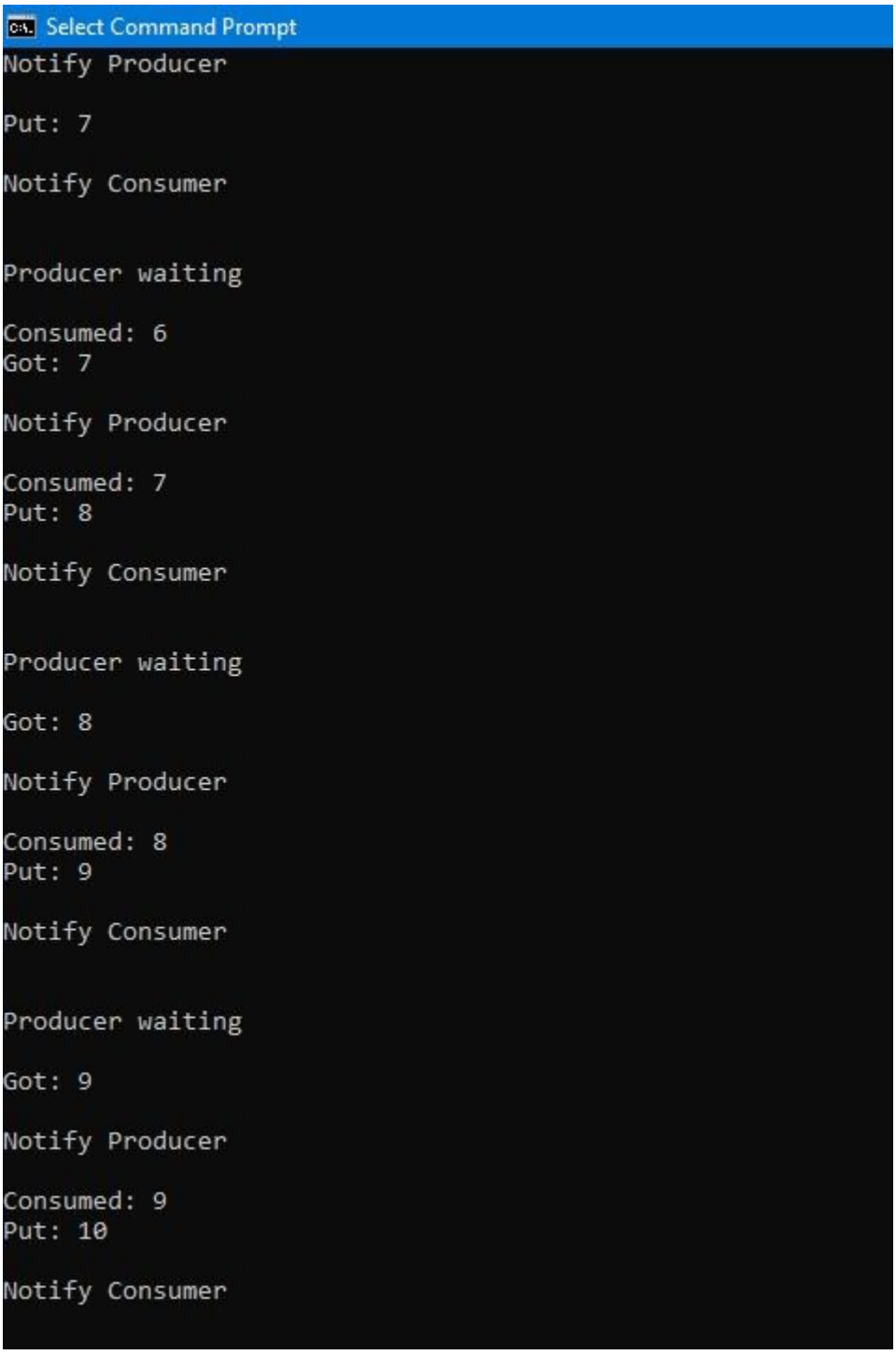
```
    }
}

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 {
   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:



```
Producer waiting

Got: 3

Notify Producer

Consumed: 3
Put: 4

Notify Consumer

Got: 4

Notify Producer

Consumed: 4
Put: 5

Notify Consumer


Producer waiting

Got: 5

Notify Producer

Put: 6

Notify Consumer


Producer waiting

Consumed: 5
Got: 6

Notify Producer

Put: 7
```


Select Command Prompt
Notify Producer
Put: 7
Notify Consumer
Producer waiting
Consumed: 6
Got: 7
Notify Producer
Consumed: 7
Put: 8
Notify Consumer
Producer waiting
Got: 8
Notify Producer
Consumed: 8
Put: 9
Notify Consumer
Producer waiting
Got: 9
Notify Producer
Consumed: 9
Put: 10
Notify Consumer

```
Producer waiting

Got: 10

Notify Producer

Consumed: 10
Put: 11

Notify Consumer


Producer waiting

Got: 11

Notify Producer

Consumed: 11
Put: 12

Notify Consumer

Got: 12

Notify Producer

Consumed: 12
Put: 13

Notify Consumer


Producer waiting

Got: 13

Notify Producer

Consumed: 13
Put: 14

Notify Consumer
```

```
Notify Consumer

Got: 14

Notify Producer

Consumed: 14
```

Code 10B:

```
class A {
synchronized void foo(B b) {
String name =Thread.currentThread().getName();
System.out.println(name + " enteredA.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 A.last");
}
}

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:

```
D:\JavaPgm>javac Deadlock.java

D:\JavaPgm>java Deadlock
MainThread enteredA.foo
RacingThread entered B.bar
MainThread trying to call B.last()
RacingThread trying to call A.last()
Inside A.last
Inside A.last
Back in other thread
Back in main thread
```