VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SAMEKSHA(1BM23CS292)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019 Sep

2024-Jan 2025

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**OBJECT ORIENTED JAVA PROGRAMMING**" carried out by **SAMEKSHA(1BM23CS292),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

Dr. Nandhini Vineeth

Associate Professor,
Department of CSE,
BMSCE, Bengaluru

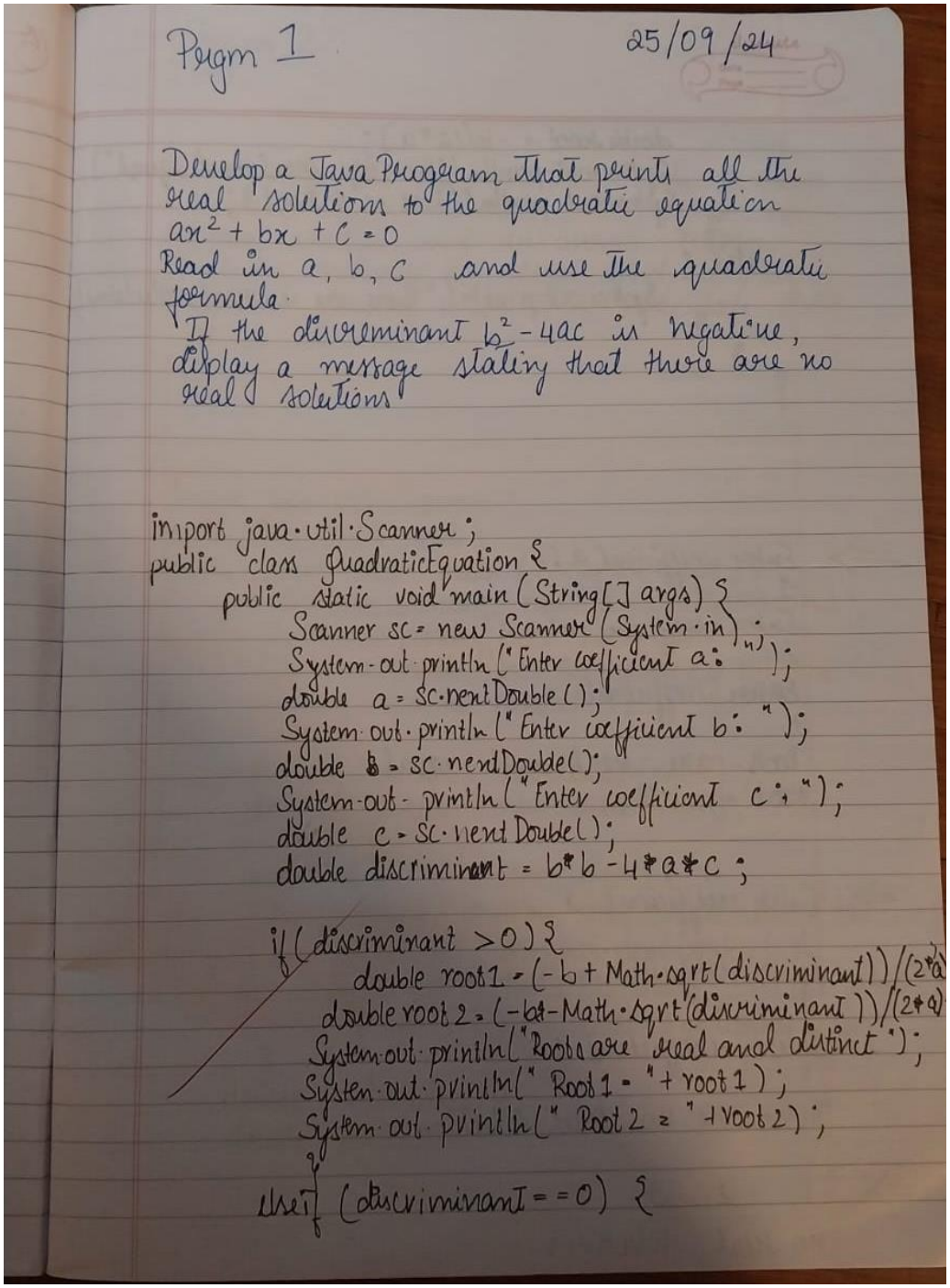
Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

WEEK 1:

Develop a Java program that prints all real solu ons to the quadra c equa on ax2+bx+c=0. Read in a, b, c and use the quadra c formula. If the discriminate b2-4ac is nega ve, display a message sta ng that there are no real solu ons.



Pgm 1 25/09/24

Develop a Java Program that prints all the real solutions to the quadratic equation $ax^2 + bx + c = 0$
Read in a, b, c and use the quadratic formula.
If the discriminant $b^2 - 4ac$ is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;
public class QuadraticEquation {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter coefficient a: ");
        double a = sc.nextDouble();
        System.out.println("Enter coefficient b: ");
        double b = sc.nextDouble();
        System.out.println("Enter coefficient c: ");
        double c = sc.nextDouble();
        double discriminant = b*b - 4*a*c;

        if(discriminant > 0) {
            double root1 = (-b + Math.sqrt(discriminant))/(2*a
            double root2 = (-b-Math.sqrt(discriminant))/(2*a
            System.out.println("Roots are real and distinct");
            System.out.println("Root 1 = " + root1);
            System.out.println("Root 2 = " + root2);
        }
        else if(discriminant == 0) {
```

```
            double root = -b/(2*a);
            System.out.println("Roots are real and equal");
            System.out.println("Root = " + root);
        }
    else {
            System.out.println("There are no real solutions");
        }
    }
}
```

⇒ Enter coefficient a:
1
Enter coefficient b:
2
Enter coefficient c:
1
Roots are real and equal
Root = -1.0

⇒ Enter coefficient a:
1
Enter coefficient b:
1
Enter coefficient c:
1
There are no real solutions.

⇒ 1, 2, 3.
no real solutions.

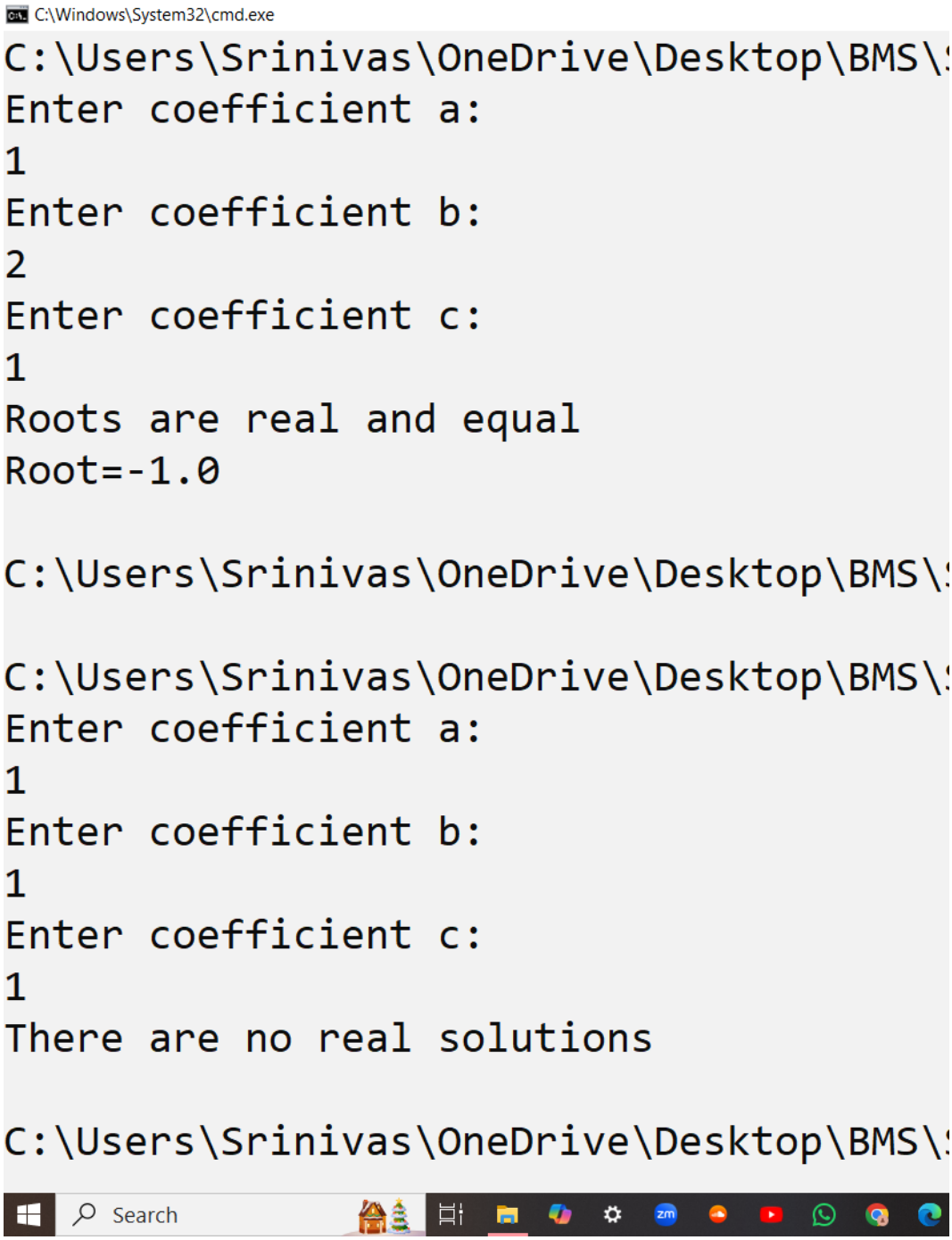
```
import java.util.Scanner;
public class QuadraticEquation{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter coefficient a:");
        double a=sc.nextDouble();
        System.out.println("Enter coefficient b:");
        double b=sc.nextDouble();
        System.out.println("Enter coefficient c:");
        double c=sc.nextDouble();
        double discriminant=b*b-4*a*c;

        if(discriminant>0){
            double root1=(-b+Math.sqrt(discriminant))/(2*a);
            double root2=(-b-Math.sqrt(discriminant))/(2*a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root 1="+root1);
            System.out.println("Root 2="+root2);
        }
        else if(discriminant==0){
            double root=-b/(2*a);
            System.out.println("Roots are real and equal");
            System.out.println("Root="+root);
        }
        else{
            System.out.println("There are no real solutions");
        }
```

```
    }
}
```

prgm 1 quadratic eq - Notepad

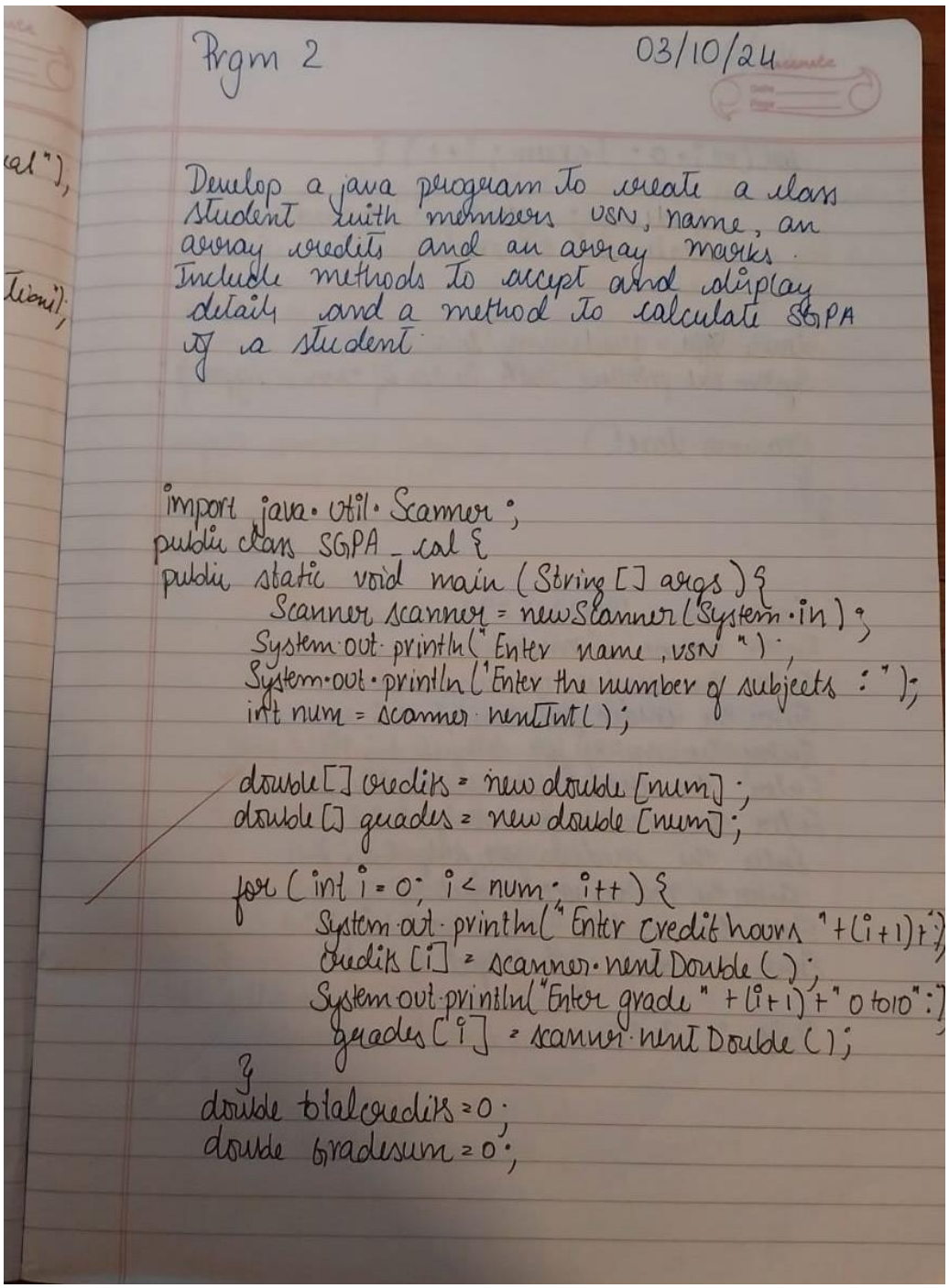
File Edit Format View Help

```
import java.util.Scanner;
public class QuadraticEquation{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter coefficient a:");
        double a=sc.nextDouble();
        System.out.println("Enter coefficient b:");
        double b=sc.nextDouble();
        System.out.println("Enter coefficient c:");
        double c=sc.nextDouble();
        double discriminant=b*b-4*a*c;

        if(discriminant>0){
            double root1=(-b+Math.sqrt(discriminant))/(2*a);
            double root2=(-b-Math.sqrt(discriminant))/(2*a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root 1="+root1);
            System.out.println("Root 2="+root2);
        }
        else if(discriminant==0){
            double root=-b/(2*a);
            System.out.println("Roots are real and equal");
            System.out.println("Root="+root);
        }
        else{
            System.out.println("There are no real solutions");
        }
    }
}
```


C:\Windows\System32\cmd.exe
C:\Users\Srinivas\OneDrive\Desktop\BMS\
Enter coefficient a:
1
Enter coefficient b:
2
Enter coefficient c:
1
Roots are real and equal
Root=-1.0
C:\Users\Srinivas\OneDrive\Desktop\BMS\
C:\Users\Srinivas\OneDrive\Desktop\BMS\
Enter coefficient a:
1
Enter coefficient b:
1
Enter coefficient c:
1
There are no real solutions
C:\Users\Srinivas\OneDrive\Desktop\BMS\
Search

WEEK 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.



Prgm 2 03/10/24

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.

```
import java.util.Scanner;
public class SGPA_cal {
public static void main(String[] args){
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter name, USN ");
    System.out.println("Enter the number of subjects :");
    int num = scanner.nextInt();

    double[] credits = new double[num];
    double[] grades = new double[num];

    for(int i=0; i<num; i++){
        System.out.println("Enter credit hours "+(i+1)+":");
        credits[i] = scanner.nextDouble();
        System.out.println("Enter grade "+(i+1)+" 0 to 10":");
        grades[i] = scanner.nextDouble();
    }
double totalcredits=0;
double gradesum=0;
```

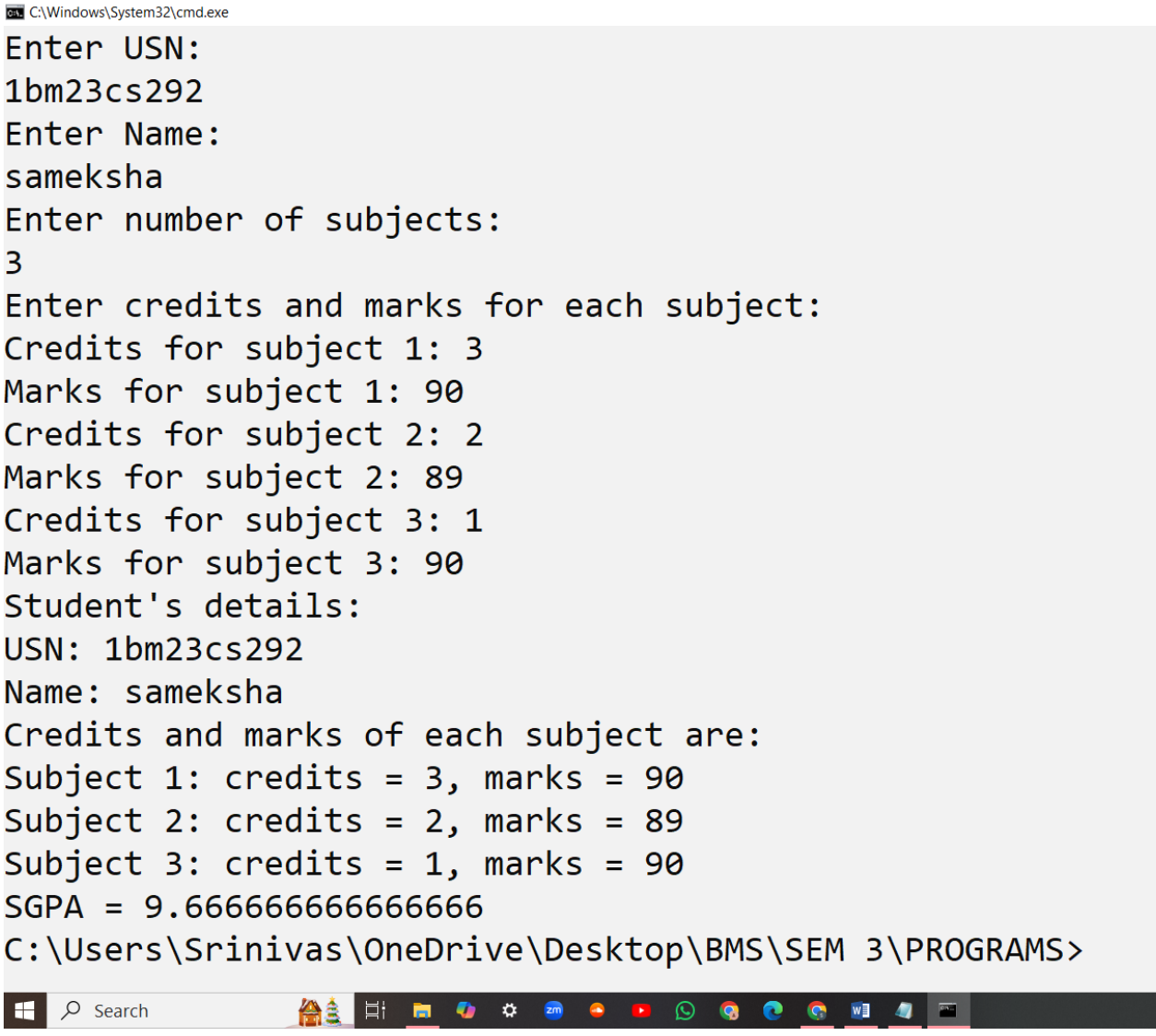
```
for (int i = 0; i < num; i++) {
        gradesum += grades[i] * credits[i];
        totalcredits += credits[i];
    }

double sgpa = gradesum / totalcredits;
System.out.println("SGPA is %.2f %n", sgpa);

scanner.close()
}
}
```

```
Enter USN: 1BM23CS292
Enter name: Sameksha
Enter the number of subjects: 4
Enter the credits for subject 1: 4
Enter the marks for subject 1: 10
Enter the credits for subject 2: 3
Enter the marks for subject 2: 9
Enter the credits for subject 3: 2
Enter the marks for subject 3: 9
Enter the credits for subject 4: 1
Enter the marks for subject 4: 10

SGPA is 9.50
```

```
import java.util.Scanner;
class Student_SGPA {
    String usn;
    String name;
    int n;
    int[] credits;
    int[] marks;

    public void acceptDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter USN:");
        usn = sc.next();
        System.out.println("Enter Name:");
        name = sc.next();
        System.out.println("Enter number of subjects:");
        n = sc.nextInt();
        credits = new int[n];
        marks = new int[n];
        System.out.println("Enter credits and marks for each subject:");
        for (int i = 0; i < n; i++) {
            System.out.print("Credits for subject " + (i + 1) + ": ");
            credits[i] = sc.nextInt();
            System.out.print("Marks for subject " + (i + 1) + ": ");
            marks[i] = sc.nextInt();
        }
    }

    public void display() {
        System.out.println("Student's details:");
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Credits and marks of each subject are:");
```

```
        for (int i = 0; i < n; i++) {
            System.out.println("Subject " + (i + 1) + ": credits = " + credits[i] + ", marks = "
+ marks[i]);
        }
    }

    private int getGradePoint(int mark) {
        if (mark >= 90) {
            return 10;
        }


 else if (mark >= 80) {
            return 9;
        } else if (mark >= 70) {
            return 8;
        } else if (mark >= 60) {
            return 7;
        } else if (mark >= 50) {
            return 6;
        } else if (mark >= 40) {
            return 5;
        } else {
            return 0;
        }
    }

    public double calculateSGPA() {
        int totalCredits = 0;
        int sum = 0;
        for (int i = 0; i < n; i++) {
            int gradePoint = getGradePoint(marks[i]);
            sum += gradePoint * credits[i];
            totalCredits += credits[i];
        }
        return (double) sum / totalCredits;
    }
}
```

```
public class Sgpa {
   public static void main(String[] args) {
      Student_SGPA student = new Student_SGPA();
      student.acceptDetails();
      student.display();
      double SGPA = student.calculateSGPA();
      System.out.printf("SGPA = "+ SGPA);
   }
}
```



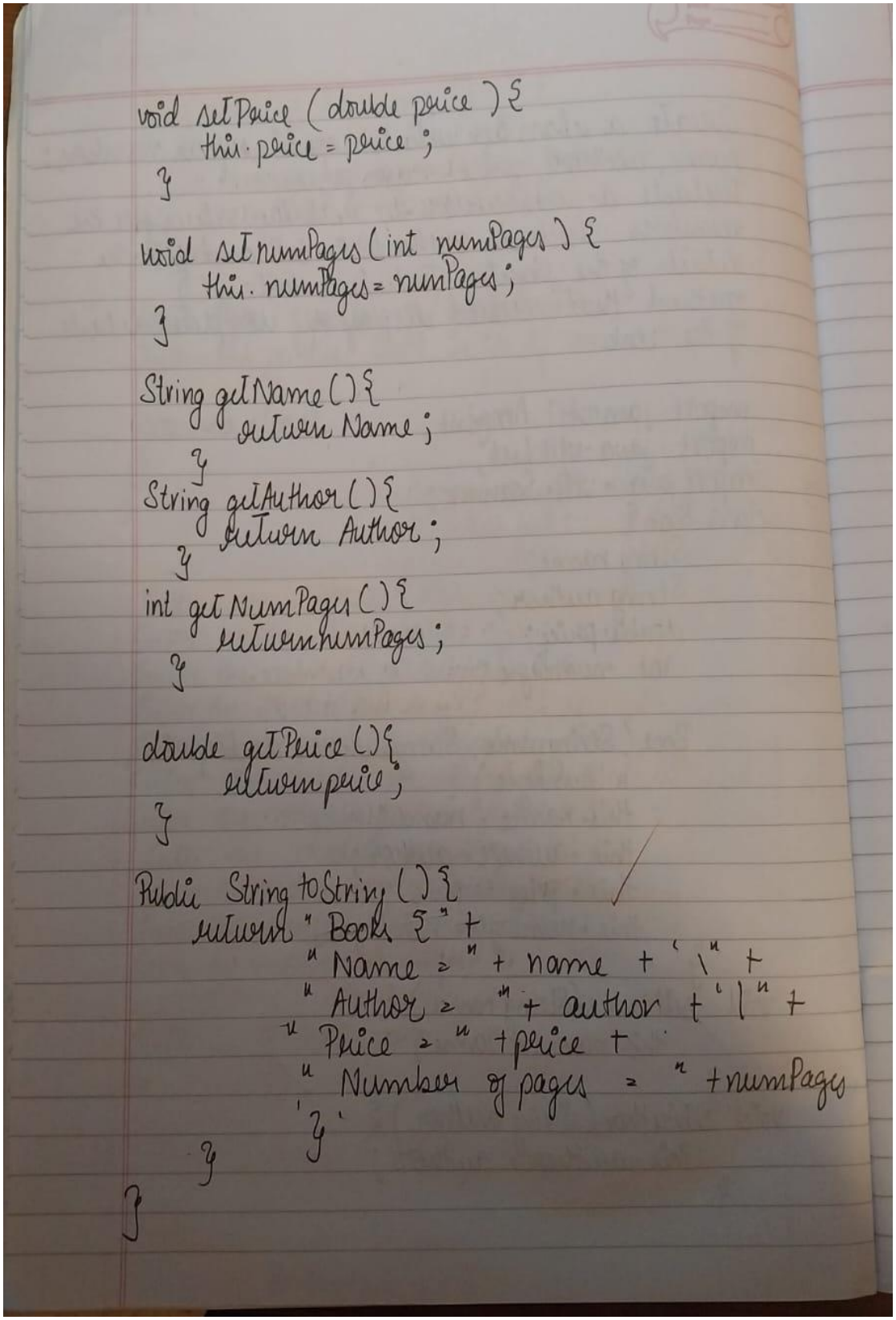
WEEK 3:

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.

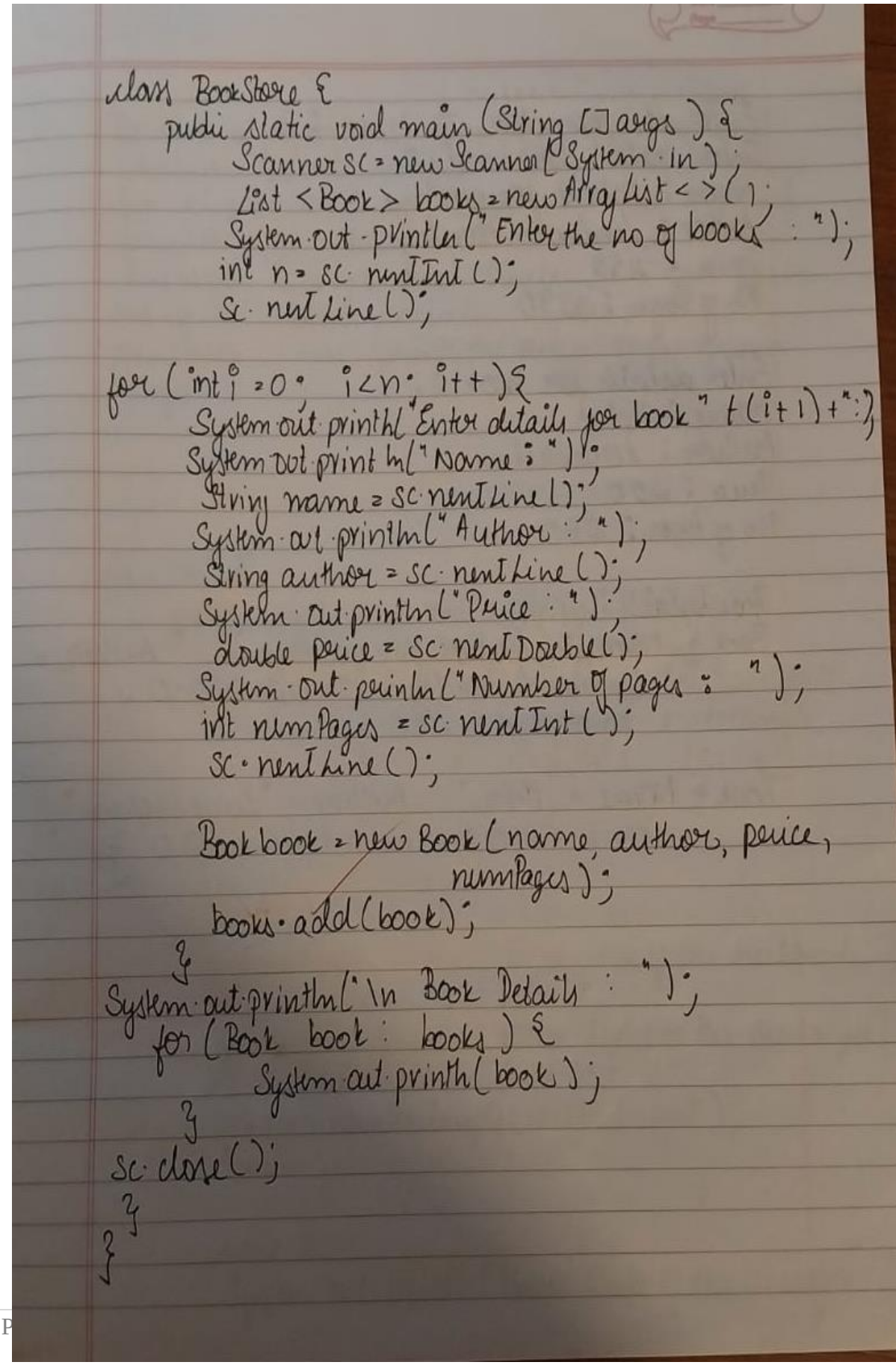
Prgm 3 19/10/24

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 would display the complete details of the book.

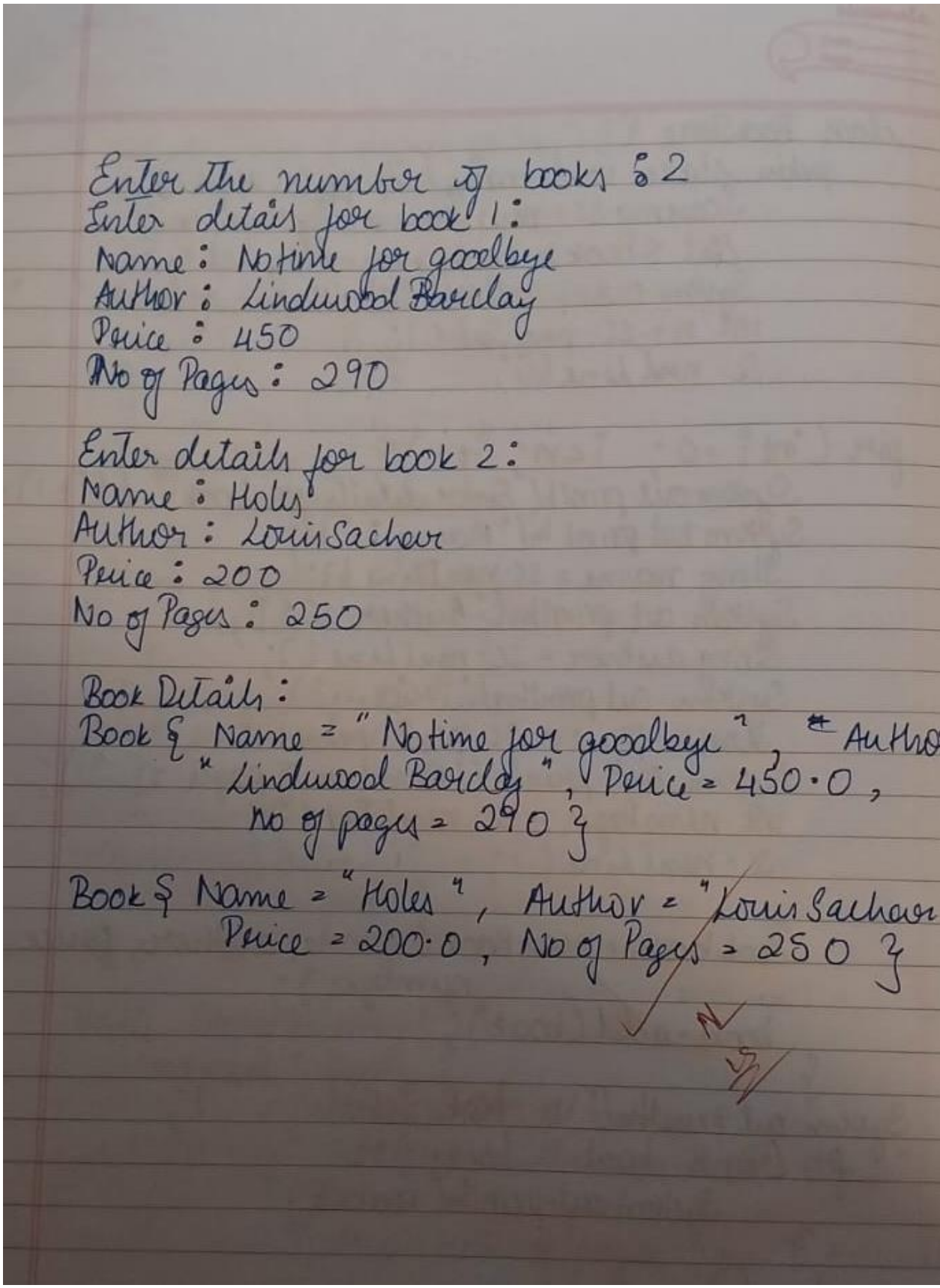
```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
class Book {
    String name;
    String author;
    double price;
    int numPages;

    Book(String name, String author, double price,
        int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
void setName(String name) {
    this.name = name;
}
void setAuthor(String author) {
    this.author = author;
}
```



```
void setPrice (double price ){
    this.price = price ;
}

void setNumPages (int numPages ) {
    this.numPages = numPages;
}

String getName (){
    return Name ;
}
String getAuthor (){
    return Author ;
}
int getNumPages (){
    return numPages ;
}

double getPrice (){
    return price;
}

Public String toString (){
    return "Book {" +
        "Name = " + name + '\'' +
        "Author = " + author + '\'' +
        "Price = " + price +
        "Number of pages = " + numPages
        '}';
}
}
```



```
class BookStore {
    publi static void main (String [] args ) {
        Scanner sc = new Scanner (System.in) ;
        List <Book> books = new ArrayList < > ( ) ;
        System.out.println (" Enter the no of books : ");
        int n = sc.nextInt ( ) ;
        Sc.nextLine ( ) ;

for (int i =0 ; i<n ; i++ ){
        System.out.println("Enter details for book " +(i+1)+":");
        System.out.println("Name : " ) ;
        String name = sc.nextLine ( ) ;
        System.out.println(" Author : " ) ;
        String author = sc.nextLine ( ) ;
        System.out.println(" Price : " ) ;
        double price = sc.nextDouble ( ) ;
        System.out.println(" Number of pages : " ) ;
        int numPages = sc.nextInt ( ) ;
        sc.nextLine ( ) ;

        Book book = new Book (name, author, price,
                              numPages ) ;
        books.add (book) ;
    }
System.out.println(" \n Book Details : " ) ;
    for (Book book : books ) {
            System.out.println (book ) ;
    }
sc.close ( ) ;
}
}
```


Enter the number of books : 2
Enter details for book 1:
Name : No time for goodbye
Author : Lindwood Barclay
Price : 450
No of Pages : 290

Enter details for book 2:
Name : Holes
Author : Louis Sachar
Price : 200
No of Pages : 250

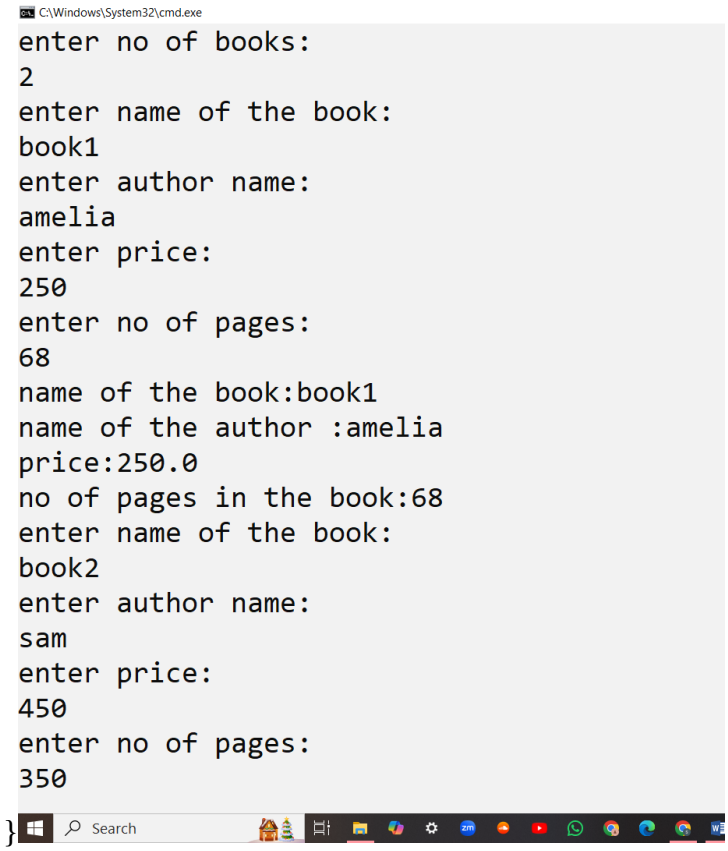
Book Details :
Book { Name = "No time for goodbye", Author = "Lindwood Barclay", Price = 450.0, No of pages = 290 }

Book { Name = "Holes", Author = "Louis Sachar", Price = 200.0, No of Pages = 250 }

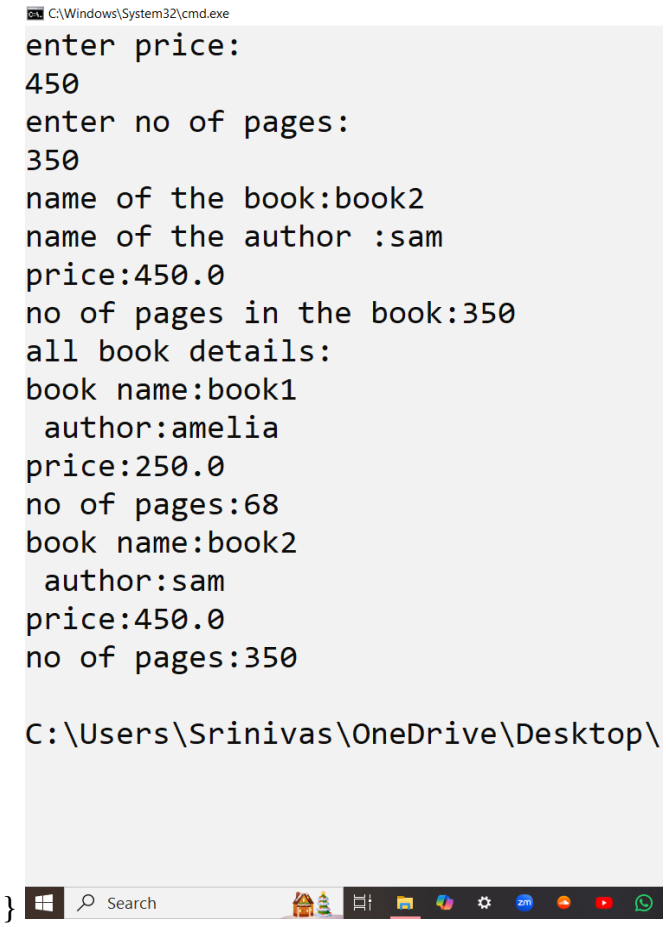
```
import java.util.Scanner;
class Book{
   String name, author;
   double price;
   int numpage;
   Book(String name , String author, double price, int numpage)
   {
      this.name =name;
      this.author =author;
      this.price =price;
      this.numpage =numpage;
   }
   Book(){
      this.name="";
      this.author ="";
      this.price =0.0;
      this.numpage =0;
   }
   void int_details(){
      Scanner sc = new Scanner(System.in);
      System.out.println("enter name of the book:");
      name = sc.nextLine();
      System.out.println("enter author name:");
```

```
        author = sc.nextLine();
        System.out.println("enter price:");
        price= sc.nextDouble();
        System.out.println("enter no of pages:");
        numpage =sc.nextInt();
    }


    void get_details(){
        System.out.println("name of the book:" +name);
        System.out.println("name of the author :" +author);
        System.out.println("price:"+price);
        System.out.println("no of pages in the book:"+numpage);
    }
    public String toString(){
        return"book name:" + name+ "\n "+ "author:" +author+ "\n"+ "price:" +price+ "\n"+
"no of pages:" +numpage;
    }
}
public class myBook{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("enter no of books:");
        int n= sc.nextInt();
        Book[] books = new Book[n];
        for (int i=0;i<n;i++)
        {
```

```
    books[i] = new Book();
    books[i].int_details();
    books[i].get_details();
  }
  System.out.println("all book details:" );
  for(Book book: books)
  {
    System.out.println(book);
  }
```



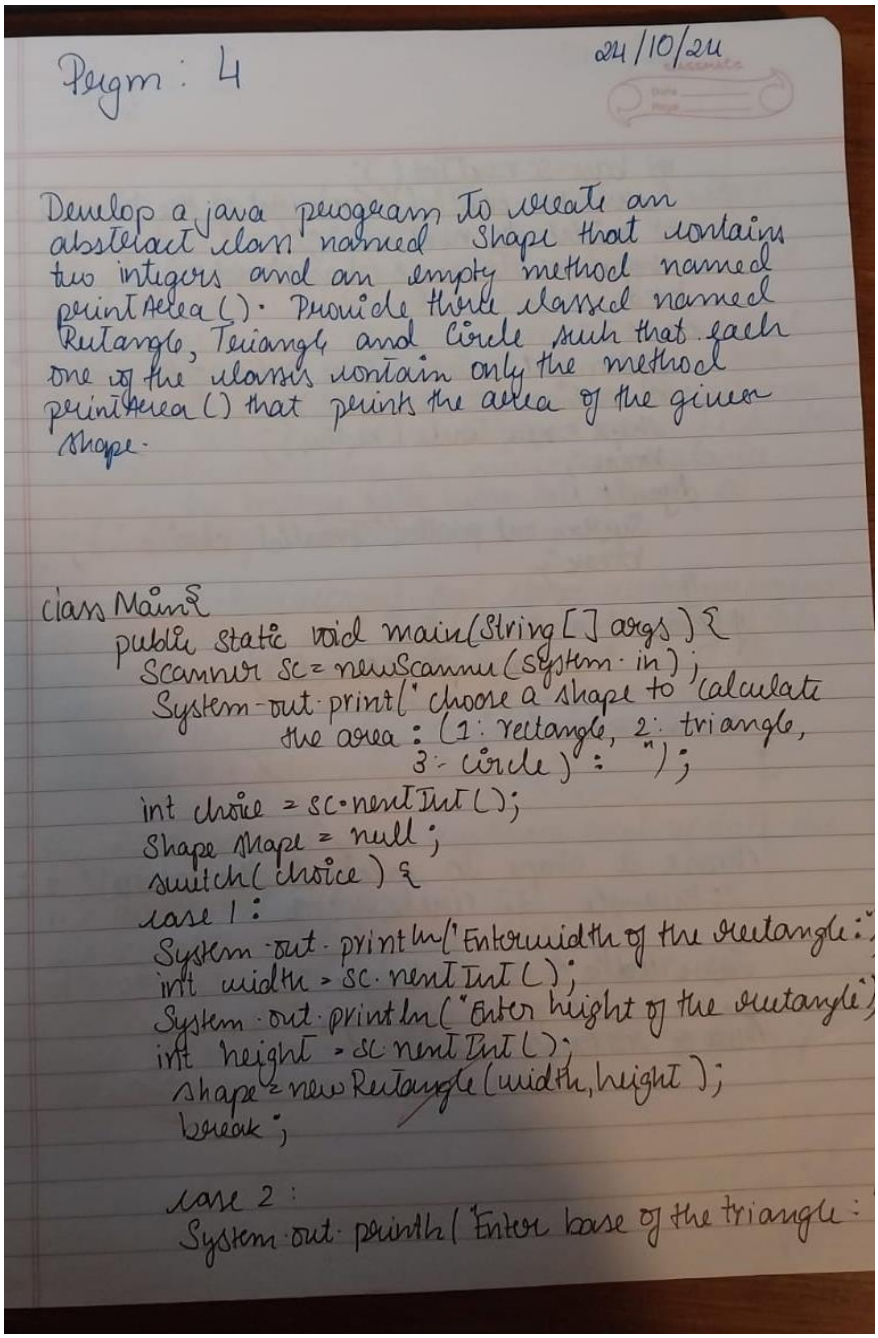
}


C:\Windows\System32\cmd.exe
enter price:
450
enter no of pages:
350
name of the book:book2
name of the author :sam
price:450.0
no of pages in the book:350
all book details:
book name:book1
author:amelia
price:250.0
no of pages:68
book name:book2
author:sam
price:450.0
no of pages:350
C:\Users\Srinivas\OneDrive\Desktop\
Search

}

WEEK 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.



Pegm : 4

24/10/24

Develop a java peogeam to ceeate an abstelact clam named Shape that contains two integers and an empty method named peintAeea( ). Peouide thoue claseed named Retangle, Teiangle and Cicle such that each one of the clamsis contain only the method peintAeea( ) that peints the aeea of the giuen shape.

```
class Main{
    public static void main(String[] args ){
    Scanner sc = new Scannu(System.in);
     System.out.print(" choose a shape to calculate
                the aeea : (1: rectangle, 2: triangle,
                        3: circle ) : ");
    int choice = sc.nextInt();
    Shape shape = null;
    switch( choice ) {
    case 1:
     System.out.println(" Enteruidth of the eectangle:");
     int uidth = sc.nextInt();
     System.out.println(" Enter height of the eectangle");
     int height = sc.nextInt();
      shape = new Rectangle(uidth, height);
      break;

    case 2:
     System.out.println( Enter base of the triangle:
```

```
        int base=sc.nextInt();
        System.out.println("Enter height of the Triangle:");
        int height=sc.nextInt();
        shape=new Triangle(base, height);
        break;
      case 3:
        System.out.println("Enter radius of the circle:");
        int radius=sc.nextInt();
        shape=new Circle(radius);
        break;
      default
        System.out.println("Invalid choice");
        break;
    }
    if(shape!=null){
        shape.printArea();
    }
    scanner.close();
}
```

Choose a shape to calculate the area (1: Rectangle, 2: Triangle, 3: Circle): 1

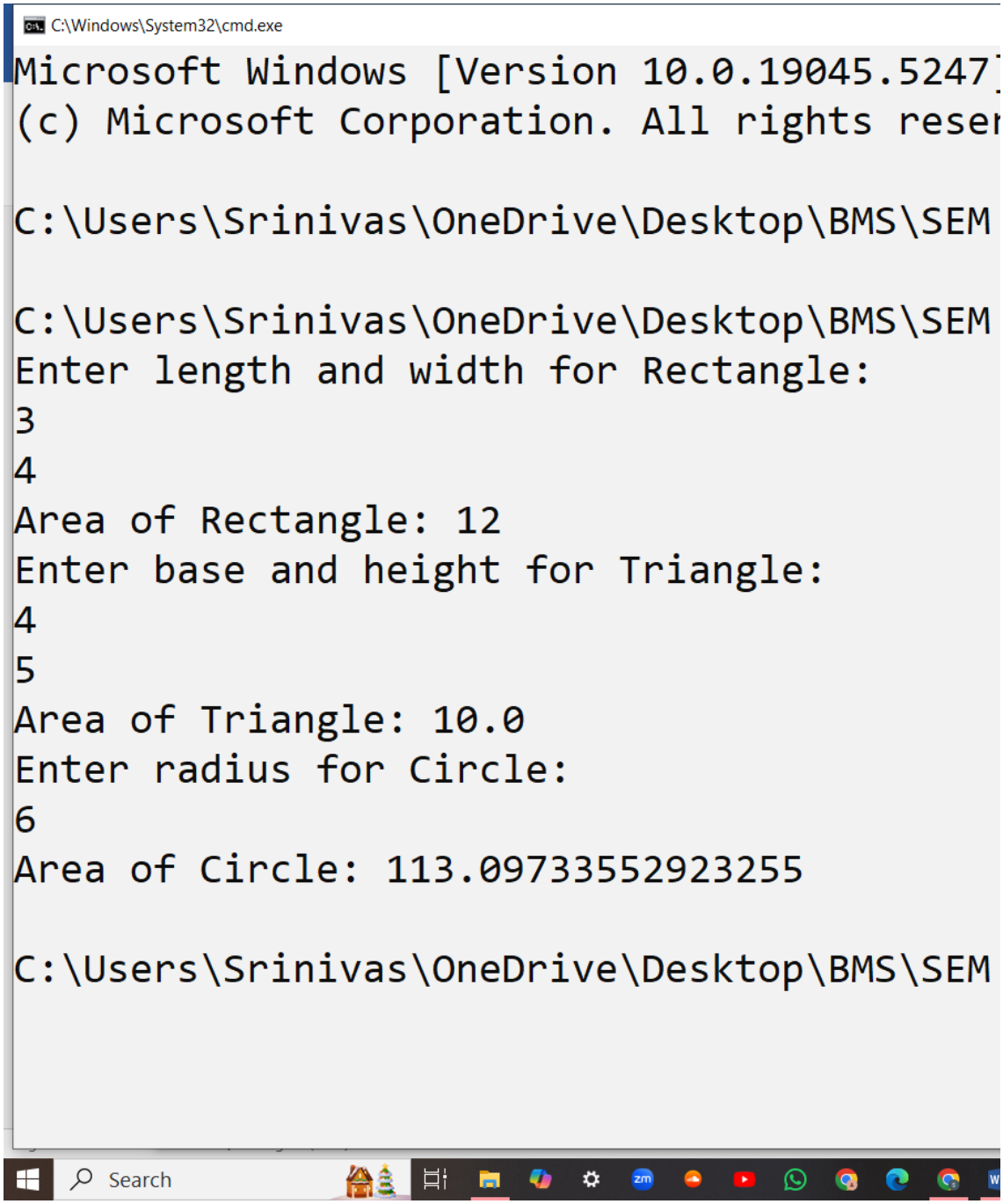
Enter width of the rectangle: 12
Enter height of the rectangle: 44
Area of rectangle: 528

```
import java.util.Scanner;
abstract class Shape {
    int dim1;
    int dim2;
    Shape(int dim1, int dim2) {
        this.dim1 = dim1;
        this.dim2 = dim2;}
    abstract void printArea();}
class Rectangle extends Shape {
    Rectangle(int length, int width) {
        super(length, width);
    }


    @Override
    void printArea() {
        int area = dim1 * dim2;
        System.out.println("Area of Rectangle: " + area);
```

```
    }
}class Triangle extends Shape {
    Triangle(int base, int height) {
        super(base, height);
    }    @Override
    void printArea() {
        double area = 0.5 * dim1 * dim2;
        System.out.println("Area of Triangle: " + area);
    }
}class Circle extends Shape {
    Circle(int radius) {
        super(radius, 0);
    }@Override
    void printArea() {
        double area = Math.PI * dim1 * dim1;
        System.out.println("Area of Circle: " + area);
    }
```

```
}

public class Shapes1 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter length and width for Rectangle:");

        int length = sc.nextInt();

        int width = sc.nextInt();

        Rectangle a1 = new Rectangle(length, width);

        a1.printArea();

        System.out.println("Enter base and height for Triangle:");

        int base = sc.nextInt();

        int height = sc.nextInt();

        Triangle a2 = new Triangle(base, height);

        a2.printArea();

       System.out.println("Enter radius for Circle:");

        int radius = sc.nextInt();

        Circle a3 = new Circle(radius);
```

```
        a3.printArea();

    }

}
```



WEEK 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 facili es but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

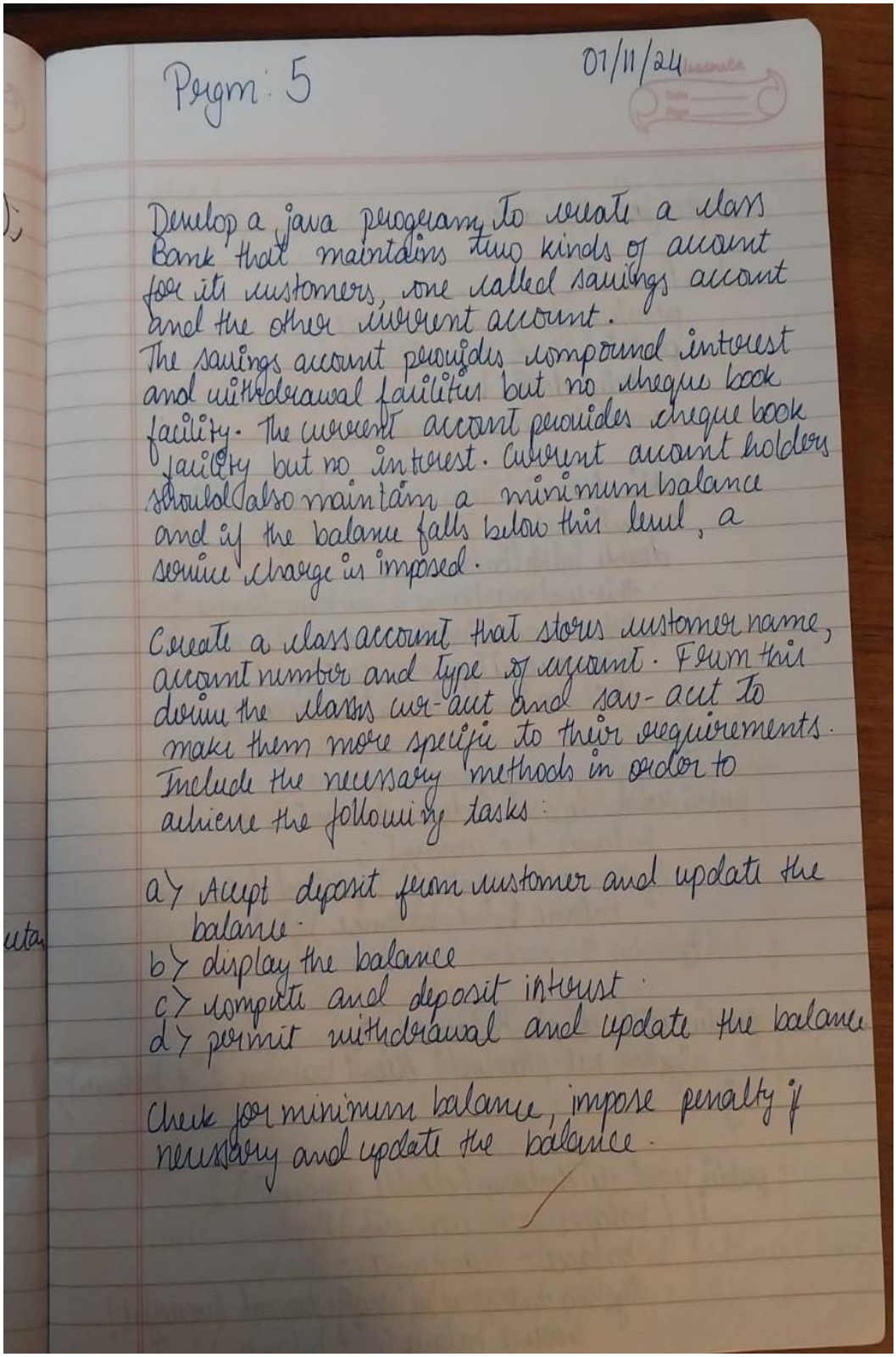
a) Accept deposit from customer and update the balance.

b) Display the balance.

c) Compute and deposit interest

d) Permit withdrawal and update the balance

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


Pgm: 5

01/11/24

Develop a java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account.
The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

Create a class account that stores customer name, account number and type of account. From this derive the classes cur-acct and sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

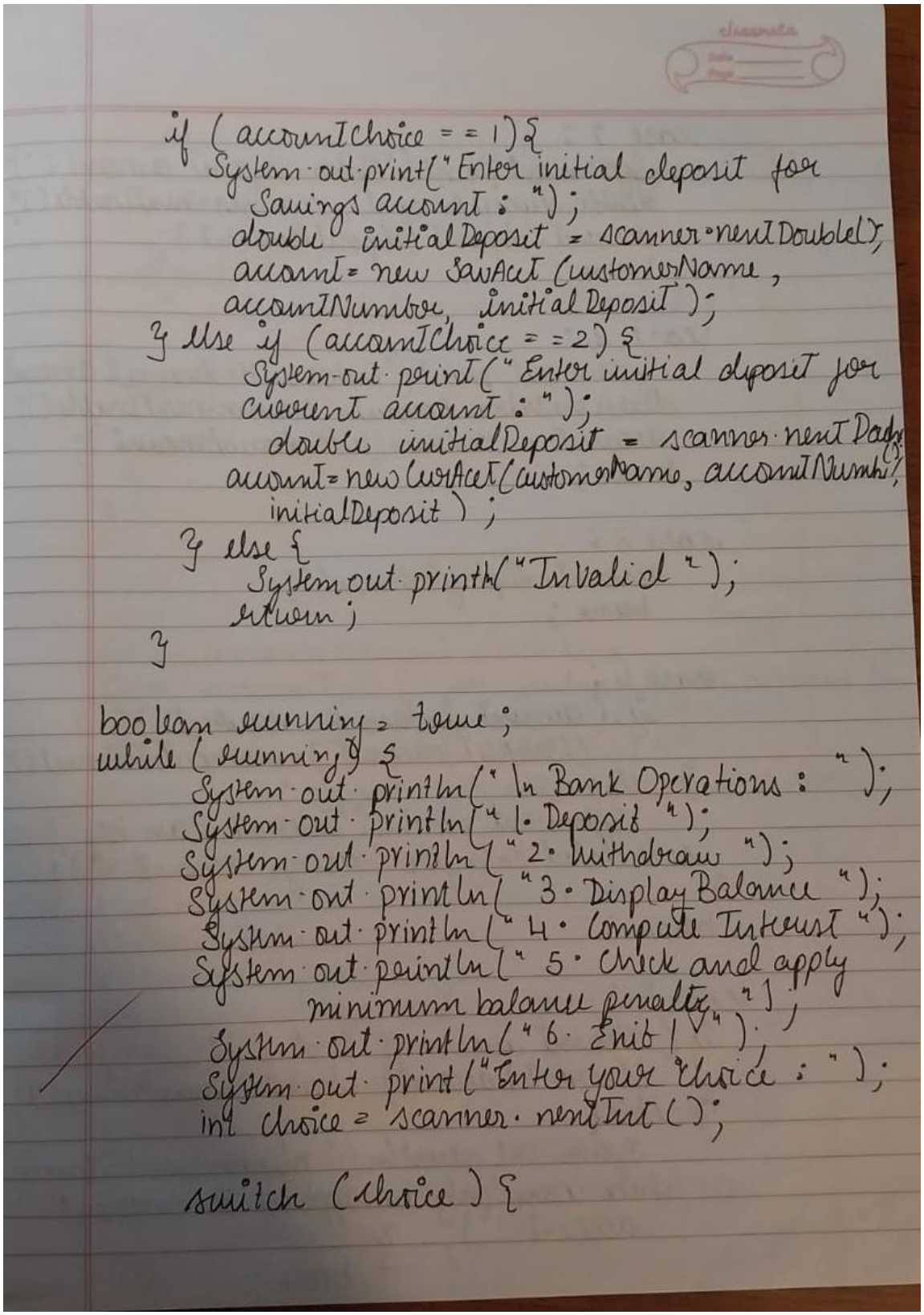
a) Accept deposit from customer and update the balance.
b) display the balance
c) compute and deposit interest
d) permit withdrawal and update the balance

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

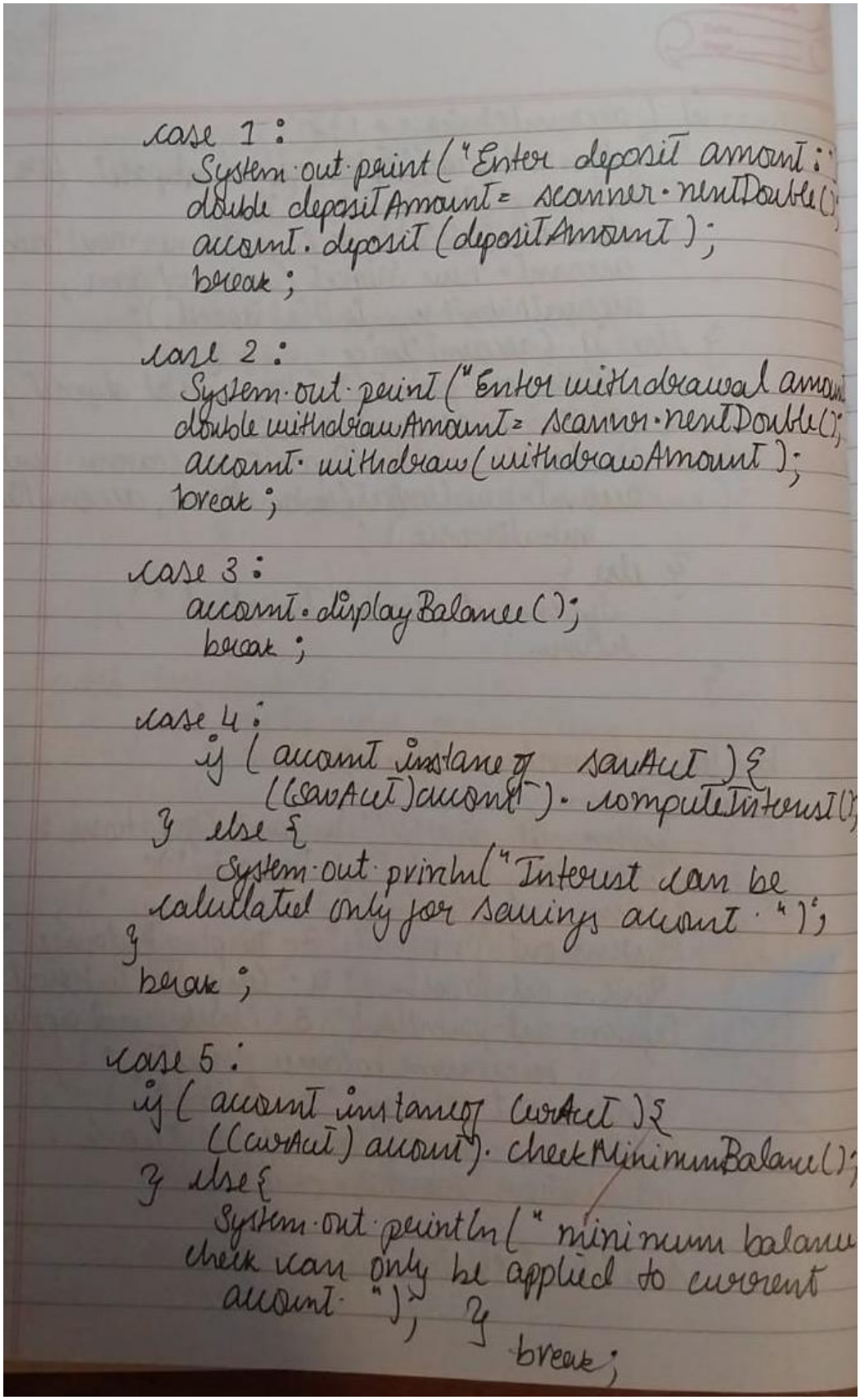
```
import java.util.Scanner;

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

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

public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited! Current
            balance: " + balance);
    }

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

public void withdraw(double amount) {
    if (balance >= amount) {
        balance -= amount;
        System.out.println("withdrawal Successful!
            current balance: " + balance); }
```

```
        else {
            System.out.println("Insufficient balance!");
        }
    }

    public String getAccountType() {
        return accountType;
    }
}

class SavAcct extends Account {
    private static final double interestRate = 0.04;

    public SavAcct(String customerName, String
    accountNumber, double initialBalance) {
        super(customerName, accountNumber,
            "Savings", initialBalance);
    }

    public void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest of " + interest + "
            has been added. New Balance: " + balance);
    }
}

class curAcct extends Account {
    private static final double min_balance = 500;
    private static final double penalty = 50;
```

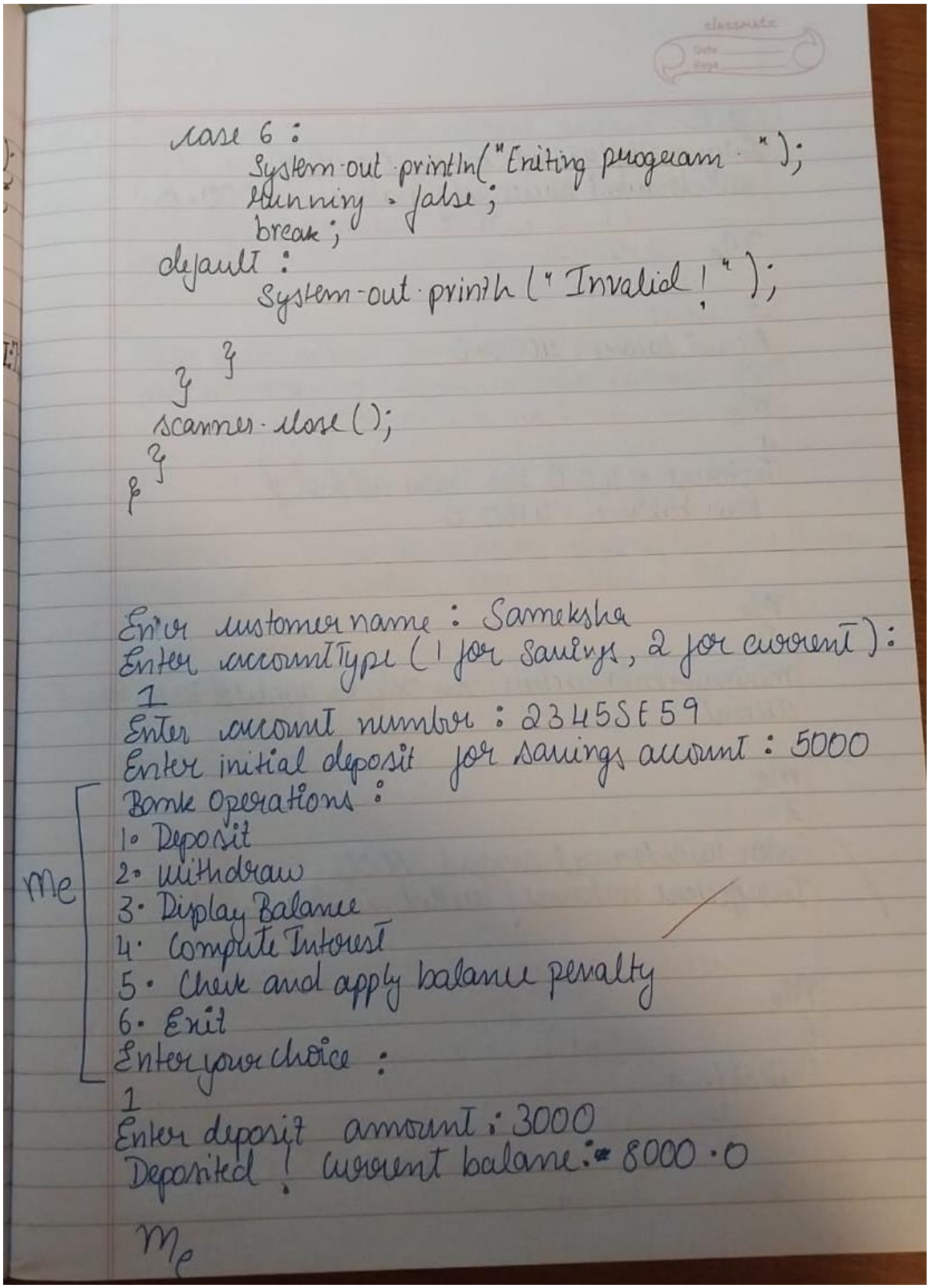
```
publi curAct (String customerNam, String accountNumb
    double initialBalance) {
        super(customerName, accountNumber,
        "current", initialBalance);
    }
publi void checkminimumBalance() {
    if (balance < min_balance) {
        balance -= penalty;
        System.out.println("Balance is below
            minimum. Penalty added/deducte
            new balance: " + balance);
    }
}
}

publi class Bank1 {
    publi static void main(String[] args) {
        Scanner scanner = new scanner(System.in);

        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter account type (1 for
            Savings, 2 for Current): ");
        int accountChoice = scanner.nextInt();

        scanner.nextLine();
        System.out.print("Enter account number: ");
        String accountNumber = scanner.nextLine();

        Account account = null;
```



```
if (accountChoice == 1) {
    System.out.print("Enter initial deposit for
        Savings account : ");
    double initialDeposit = scanner.nextDouble();
    account = new SavAcct(customerName,
    accountNumber, initialDeposit);
} else if (accountChoice == 2) {
        System.out.print("Enter initial deposit for
        current account : ");
        double initialDeposit = scanner.nextDou
    account = new CurrAcct(customerName, accountNumb
        initialDeposit);
} else {
    System.out.println("Invalid");
    return;
}

boolean running = true;
while (running) {
    System.out.println("\n Bank Operations : ");
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Display Balance");
    System.out.println("4. Compute Interest");
    System.out.println("5. Check and apply
        minimum balance penalty");
    System.out.println("6. Exit");
    System.out.print("Enter your choice : ");
    int choice = scanner.nextInt();

switch (choice) {
```



```
case 1:
    System.out.print("Enter deposit amount:");
    double depositAmount = scanner.nextDouble();
    account.deposit(depositAmount);
    break;

case 2:
    System.out.print("Enter withdrawal amount:");
    double withdrawAmount = scanner.nextDouble();
    account.withdraw(withdrawAmount);
    break;

case 3:
    account.displayBalance();
    break;

case 4:
    if (account instanceof SavAct) {
        ((SavAct)account).computeInterest();
    } else {
        System.out.println("Interest can be
    calculated only for savings account.");
}
break;

case 5:
    if (account instanceof CurAct) {
        ((CurAct) account).checkMinimumBalance();
    } else {
        System.out.println("minimum balance
    check can only be applied to current
    account."); }
                    break;
```

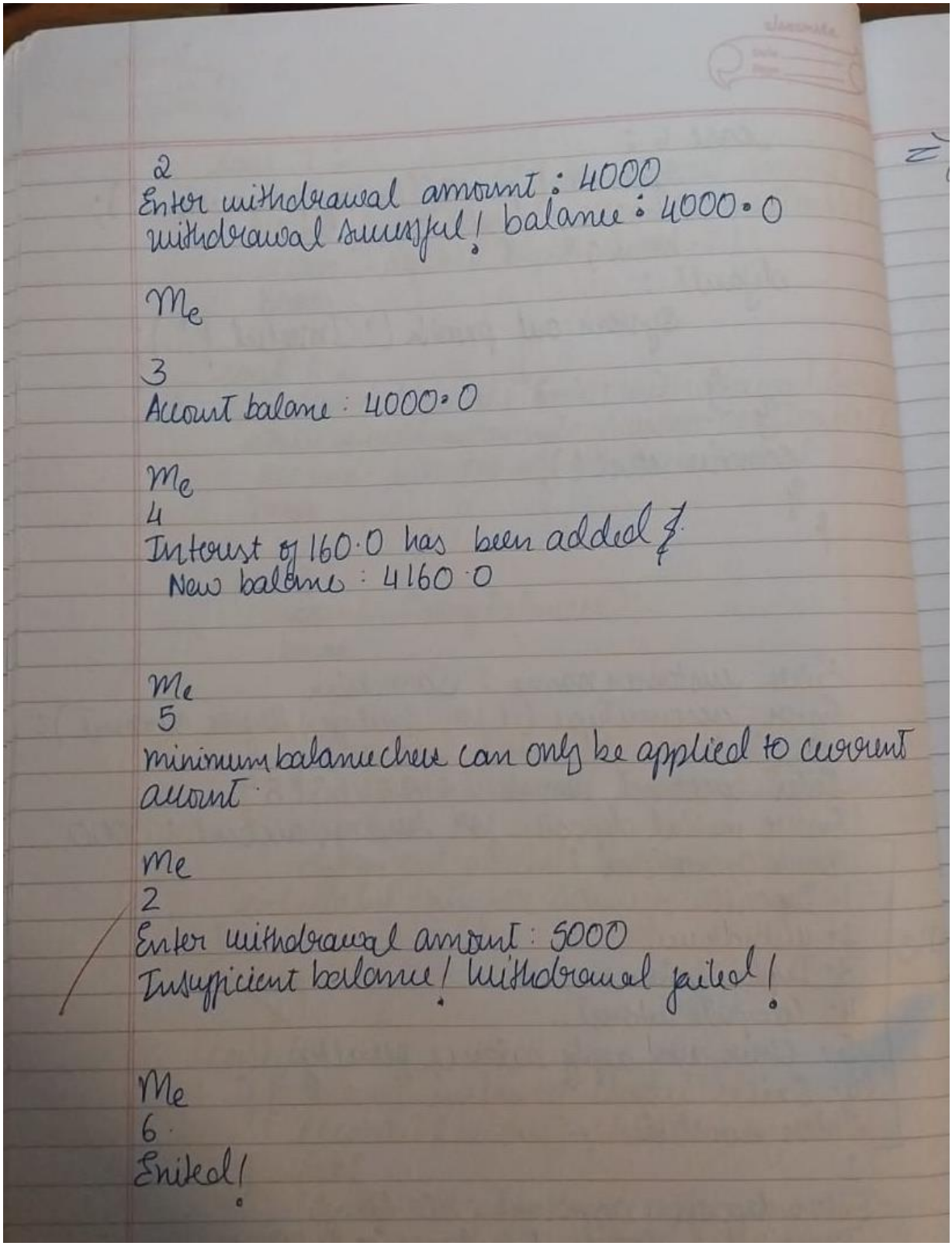


```
            case 6 :
                System.out.println("Exiting program ");
                running = false;
                break;
            default :
                System.out.println("Invalid!");

        }
    }
    scanner.close();
}
```

```
Enter customer name : Sameksha
Enter accountType (1 for Savings, 2 for current):
1
Enter account number : 2345SE59
Enter initial deposit for savings account : 5000
Bank Operations :
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest
5. Check and apply balance penalty
6. Exit
Enter your choice :
1
Enter deposit amount : 3000
Deposited ! current balance : 8000.0
```

me

me


2
Enter withdrawal amount : 4000
withdrawal successful! balance : 4000.0

Me

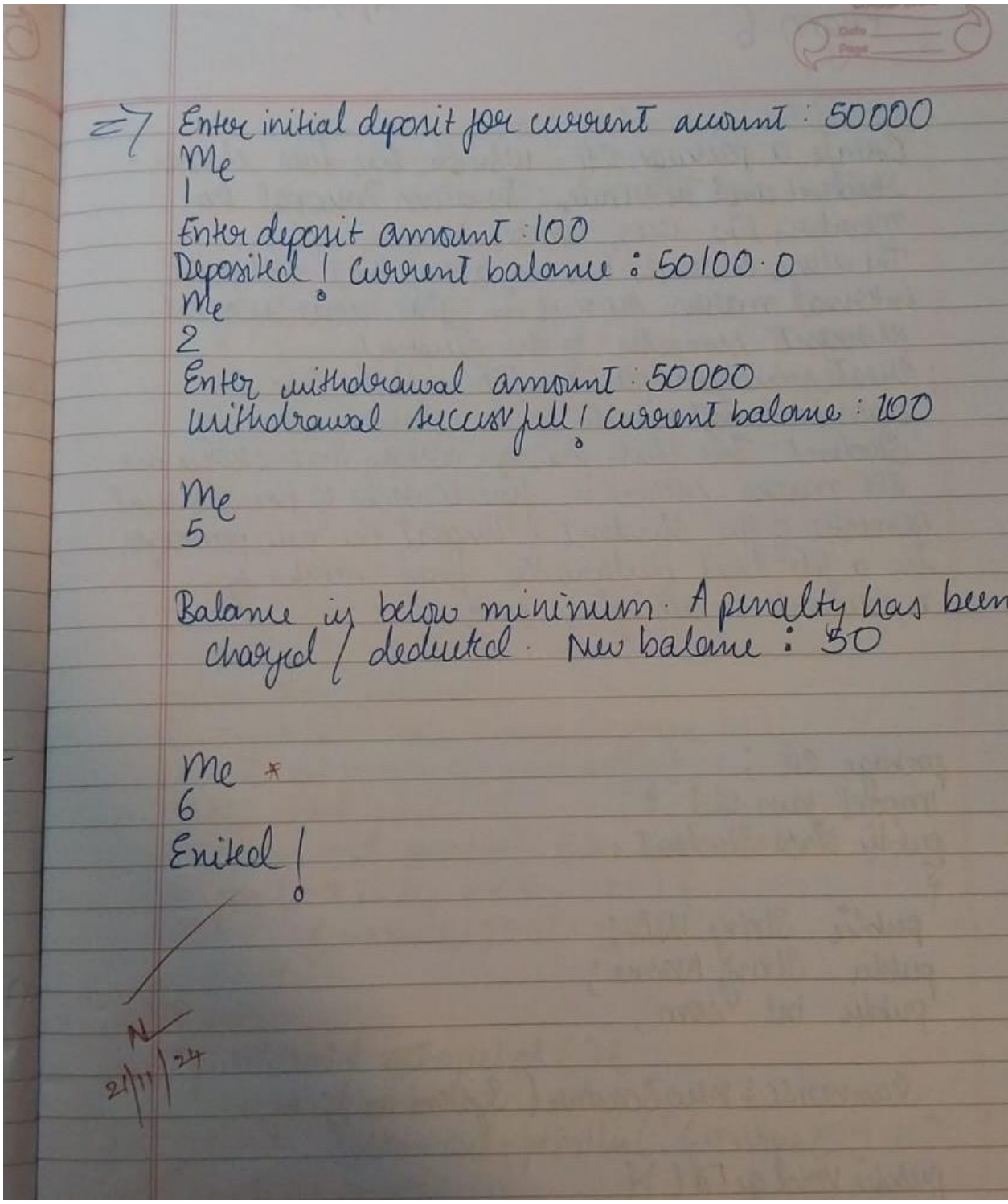
3
Account balance : 4000.0

Me
4
Interest of 160.0 has been added!
New balance : 4160.0

Me
5
minimum balance check can only be applied to current account.

Me
2
Enter withdrawal amount : 5000
Insufficient balance! withdrawal failed!

Me
6.
Exited!


=7 Enter initial deposit for current account : 50000
Me
1
Enter deposit amount : 100
Deposited ! current balance : 50100.0
Me
2
Enter withdrawal amount : 50000
withdrawal successfull ! current balance : 100

Me
5

Balance is below minimum. A penalty has been
charged / deducted. New balance : 50

Me *
6
Exited !

21/11/24

```
import java.util.Scanner;

class Account {

    protected String customerName;

    protected String accountNumber;

    protected double balance;

    protected String accountType;

  public Account(String customerName, String accountNumber, String accountType, double initialBalance) {

        this.customerName = customerName;

        this.accountNumber = accountNumber;

        this.accountType = accountType;

        this.balance = initialBalance;

    }

    public void deposit(double amount) {

        balance += amount;

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

    }

    public void displayBalance() {

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

    }

  public void withdraw(double amount) {

        if (balance >= amount) {
```

```
            balance -= amount;
            System.out.println("Withdrawal successful! Current balance: " + balance);
        } else {
            System.out.println("Insufficient balance! Withdrawal failed.");
        }
    }
    public String getAccountType() {
        return accountType;
    }
}
class SavAcct extends Account {
    private static final double interestRate = 0.04; // 4% interest rate for savings account
    public SavAcct(String customerName, String accountNumber, double initialBalance) {
        super(customerName, accountNumber, "Savings", initialBalance);
    }
public void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest of " + interest + " has been added to your account. New balance: " + balance);
    }
}
class CurAcct extends Account {
    private static final double MIN_BALANCE = 500; // Minimum balance required for Current Account
```

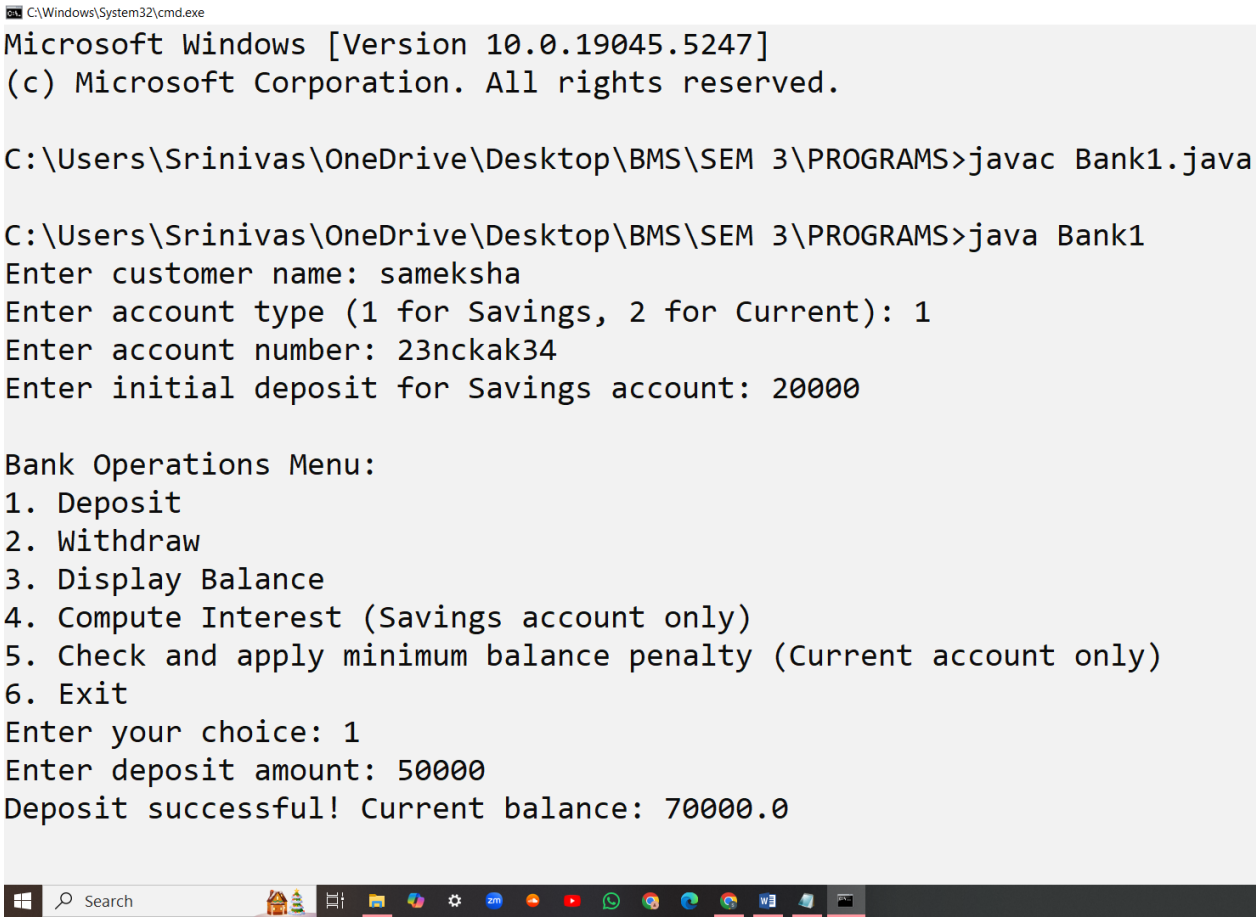
```
    private static final double PENALTY = 50; // Penalty fee for falling below minimum balance

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

    public void checkMinimumBalance() {
        if (balance < MIN_BALANCE) {
            balance -= PENALTY;
            System.out.println("Balance is below minimum. A penalty of " + PENALTY + " has been charged. New balance: " + balance);
        }
    }
}
```

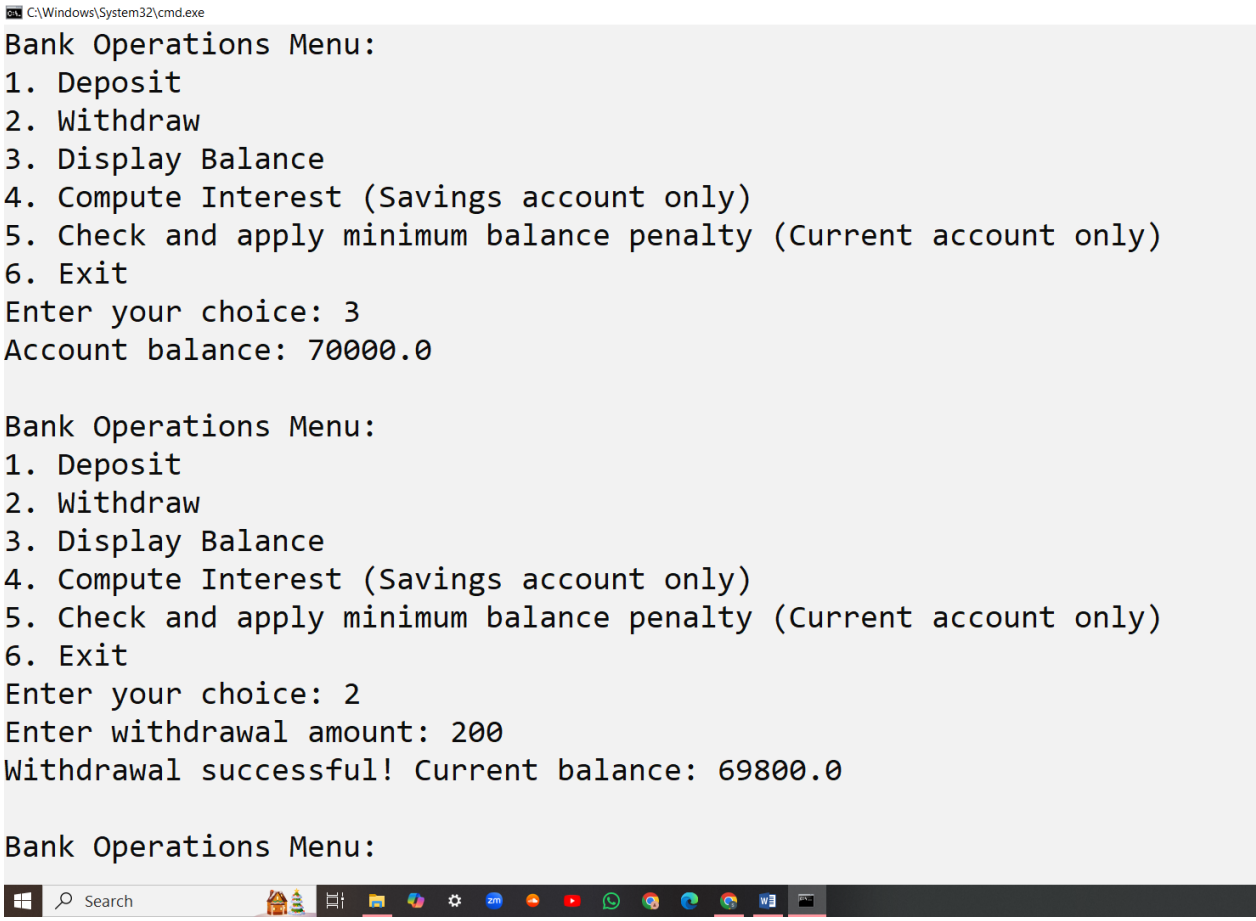
```
public class Bank1 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();
        System.out.print("Enter account type (1 for Savings, 2 for Current): ");
        int accountChoice = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Enter account number: ");
```

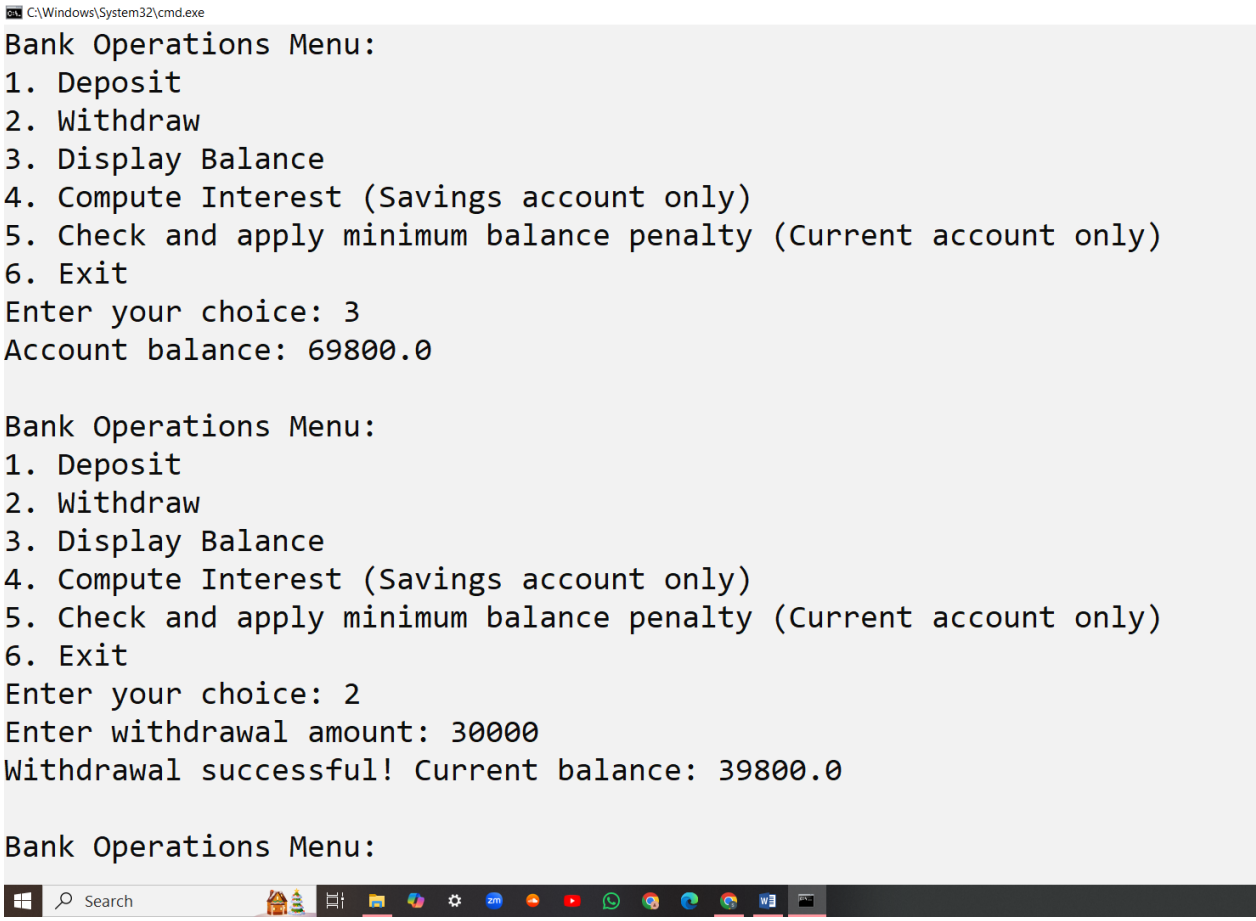
```
        String accountNumber = scanner.nextLine();
        Account account = null;
        if (accountChoice == 1) {
            System.out.print("Enter initial deposit for Savings account: ");
            double initialDeposit = scanner.nextDouble();
            account = new SavAcct(customerName, accountNumber, initialDeposit);
        } else if (accountChoice == 2) {
            System.out.print("Enter initial deposit for Current account: ");
            double initialDeposit = scanner.nextDouble();
            account = new CurAcct(customerName, accountNumber, initialDeposit);
        } else {
            System.out.println("Invalid choice! Exiting program.");
            return;
        }
        boolean running = true;
        while (running) {
            System.out.println("\nBank Operations Menu:");
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Display Balance");
            System.out.println("4. Compute Interest (Savings account only)");
            System.out.println("5. Check and apply minimum balance penalty (Current account only)");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();
```

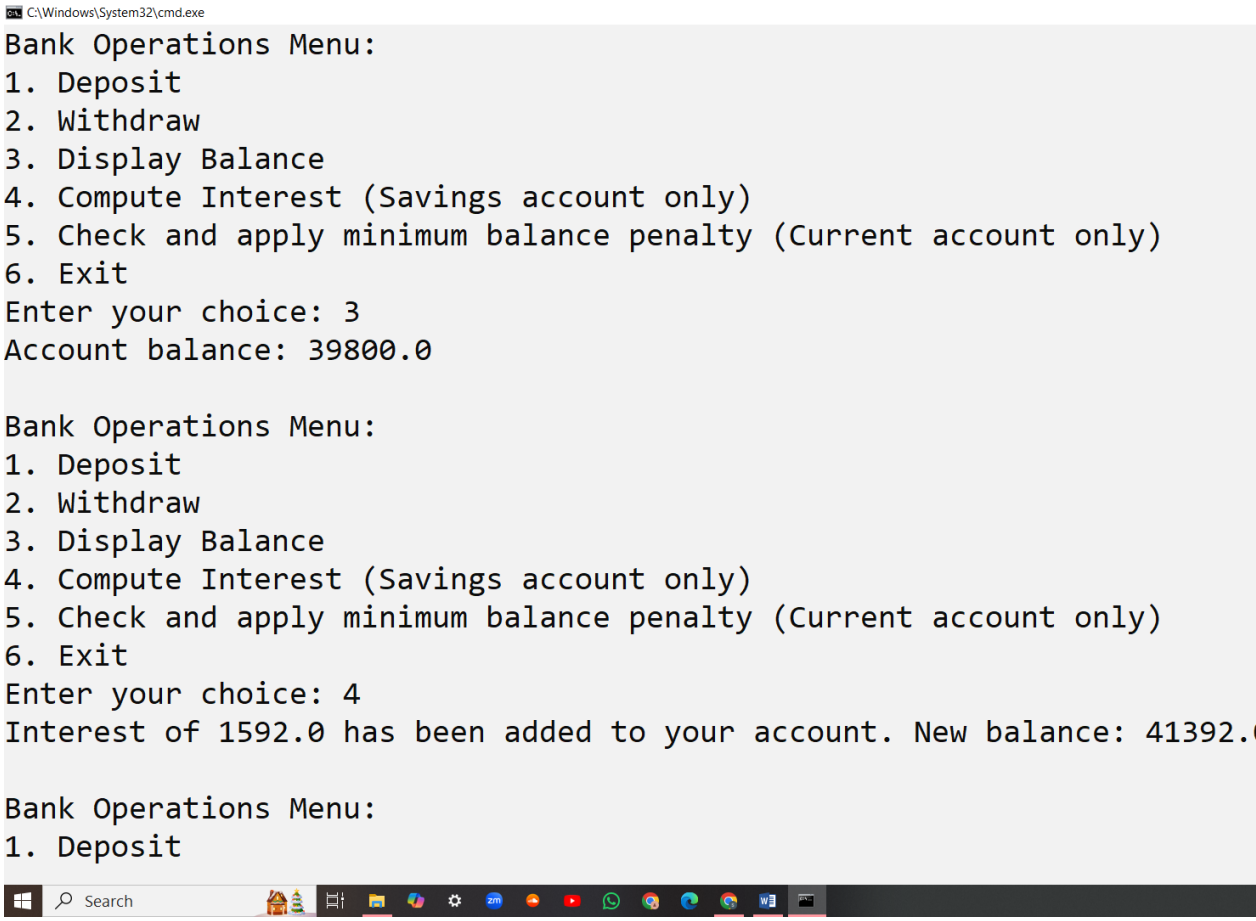
```
        switch (choice) {
            case 1:
                System.out.print("Enter deposit amount: ");
                double depositAmount = scanner.nextDouble();
                account.deposit(depositAmount);
                break;
            case 2:
                System.out.print("Enter withdrawal amount: ");
                double withdrawAmount = scanner.nextDouble();
                account.withdraw(withdrawAmount);
                break;
            case 3:
                account.displayBalance();
                break;
            case 4:
                if (account instanceof SavAcct) {
                    ((SavAcct) account).computeInterest();
                } else {
                    System.out.println("Interest can only be computed for Savings account.");
                }
                break;
            case 5:
                if (account instanceof CurAcct) {
                    ((CurAcct) account).checkMinimumBalance();
                } else {
```

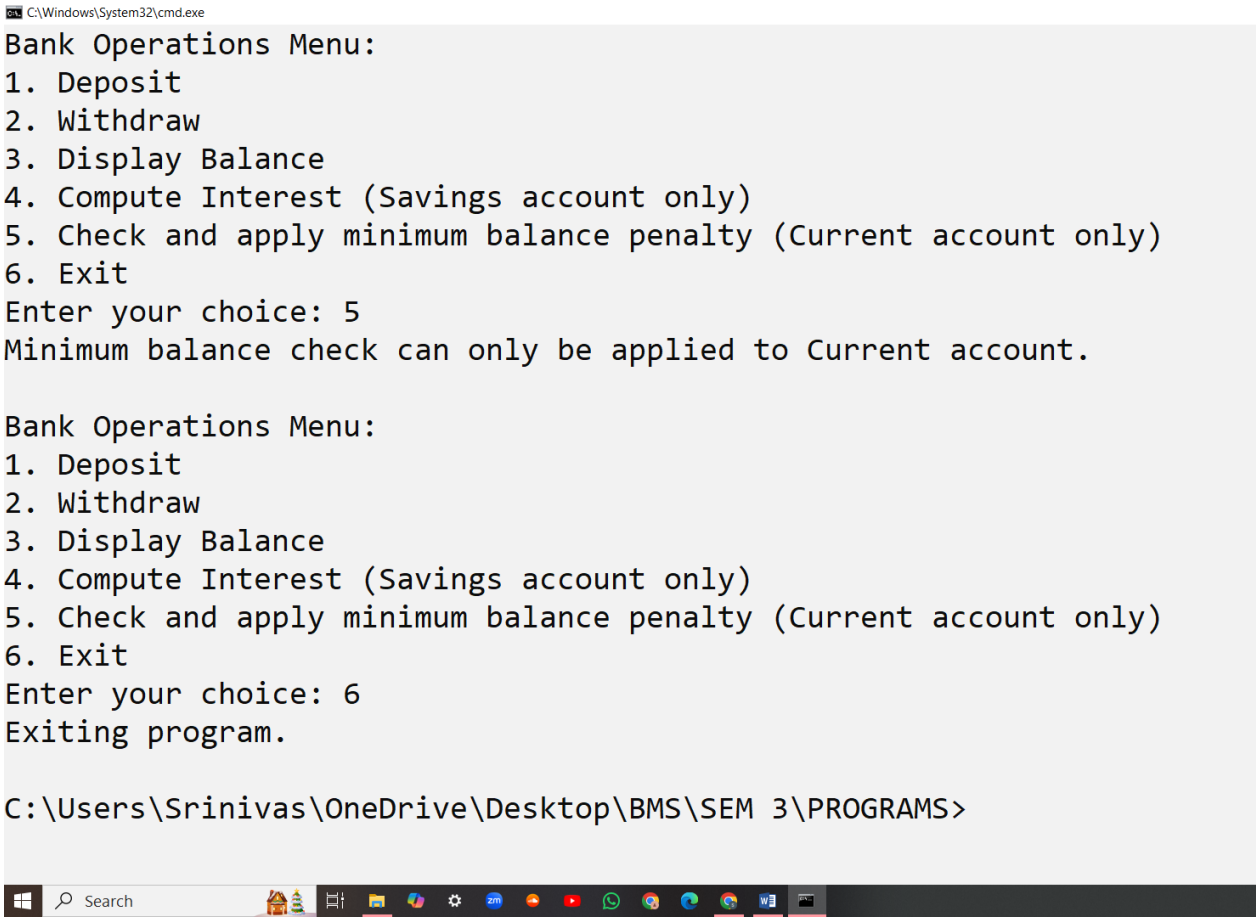
```
                    System.out.println("Minimum balance check can only be applied to Current account.");
                }
                break;
            case 6:
                System.out.println("Exiting program.");
                running = false;
                break;
            default:
                System.out.println("Invalid choice! Please select a valid option.");
        }
    }

    scanner.close();
  }
}
```


C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.
C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3\PROGRAMS>javac Bank1.java
C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3\PROGRAMS>java Bank1
Enter customer name: sameksha
Enter account type (1 for Savings, 2 for Current): 1
Enter account number: 23nckak34
Enter initial deposit for Savings account: 20000
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 1
Enter deposit amount: 50000
Deposit successful! Current balance: 70000.0
Search


C:\Windows\System32\cmd.exe
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 3
Account balance: 70000.0
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 2
Enter withdrawal amount: 200
Withdrawal successful! Current balance: 69800.0
Bank Operations Menu:
Search

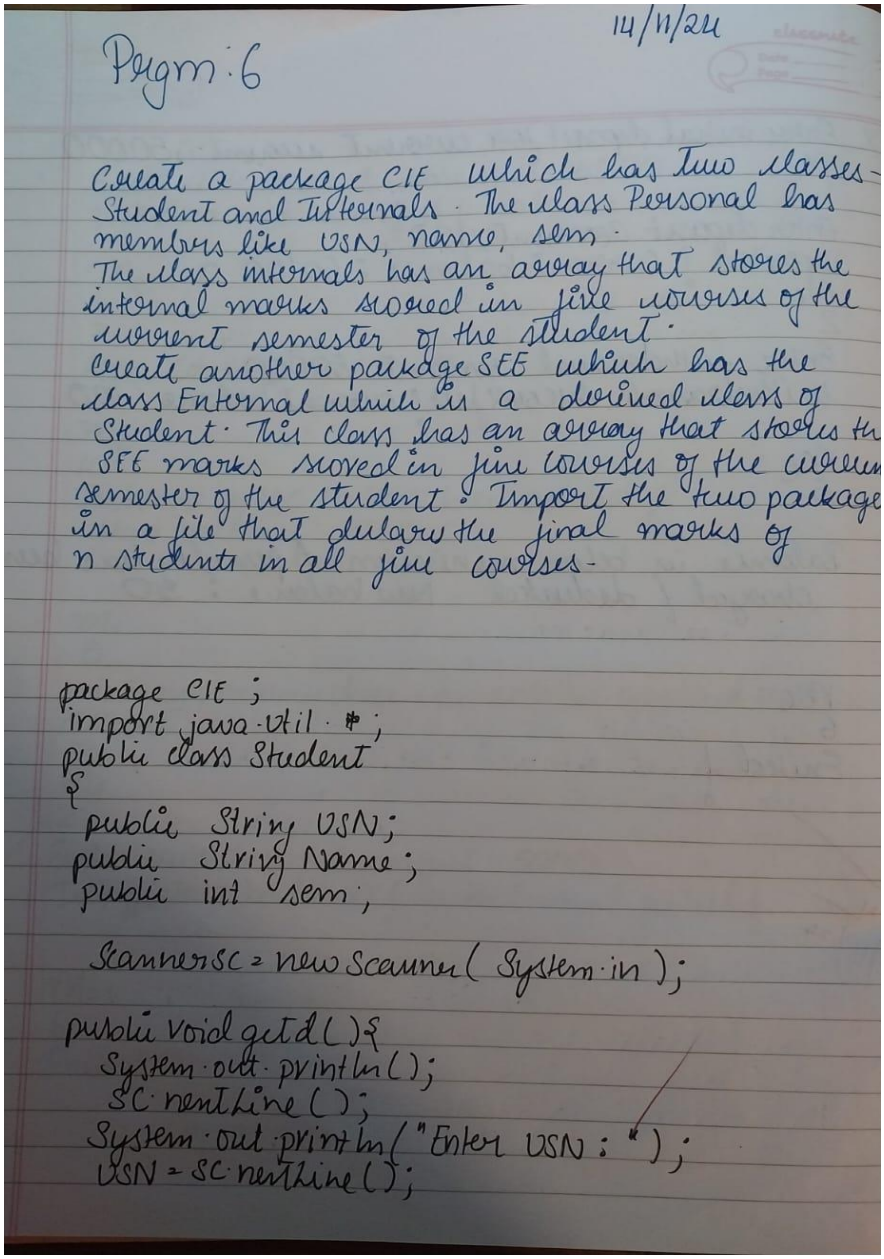

C:\Windows\System32\cmd.exe
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 3
Account balance: 69800.0
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 2
Enter withdrawal amount: 30000
Withdrawal successful! Current balance: 39800.0
Bank Operations Menu:
Search


C:\Windows\System32\cmd.exe
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 3
Account balance: 39800.0
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 4
Interest of 1592.0 has been added to your account. New balance: 41392.
Bank Operations Menu:
1. Deposit
Search


C:\Windows\System32\cmd.exe
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 5
Minimum balance check can only be applied to Current account.
Bank Operations Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings account only)
5. Check and apply minimum balance penalty (Current account only)
6. Exit
Enter your choice: 6
Exiting program.
C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3\PROGRAMS>
Search

WEEK 6 :

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

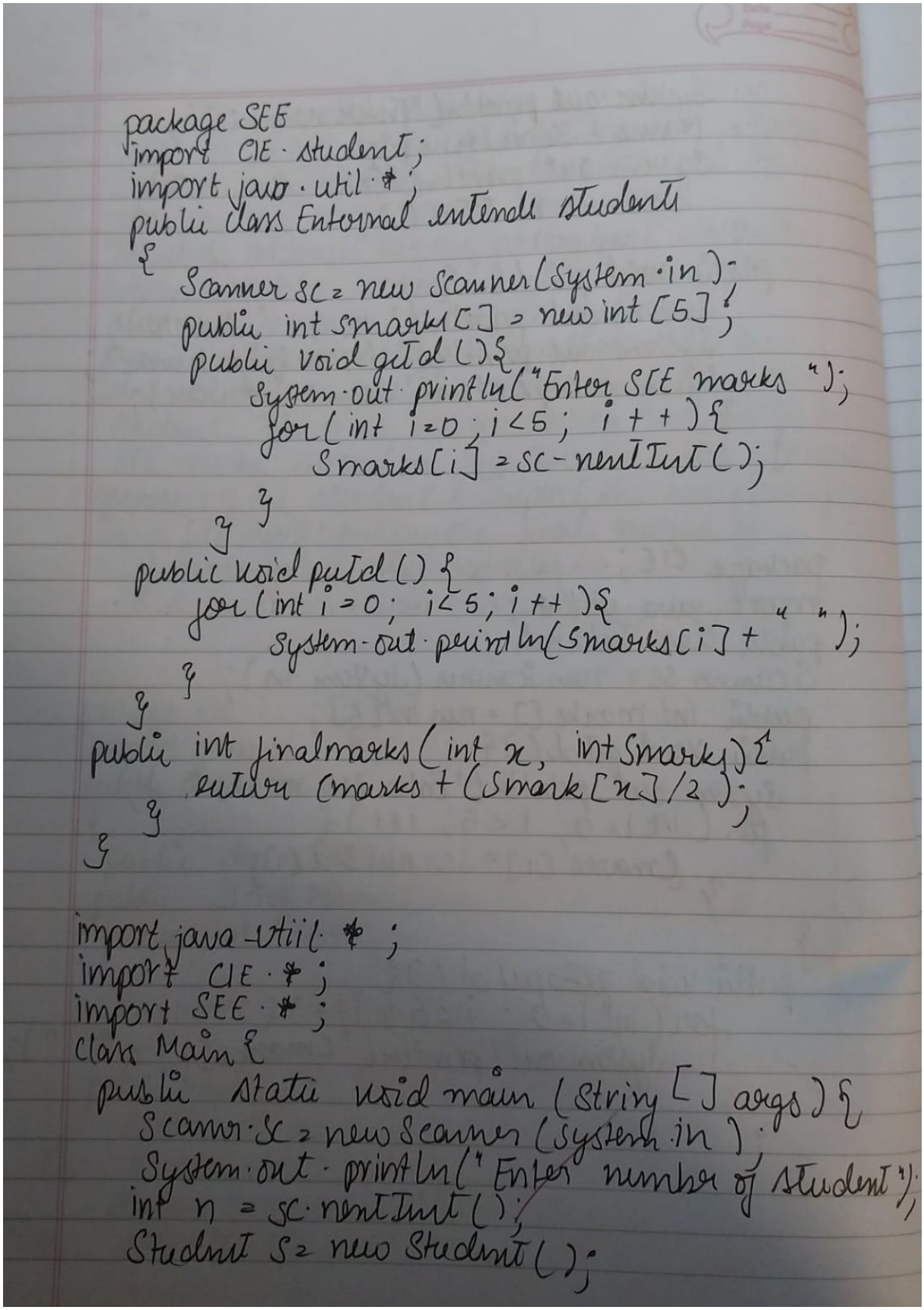


14/11/24

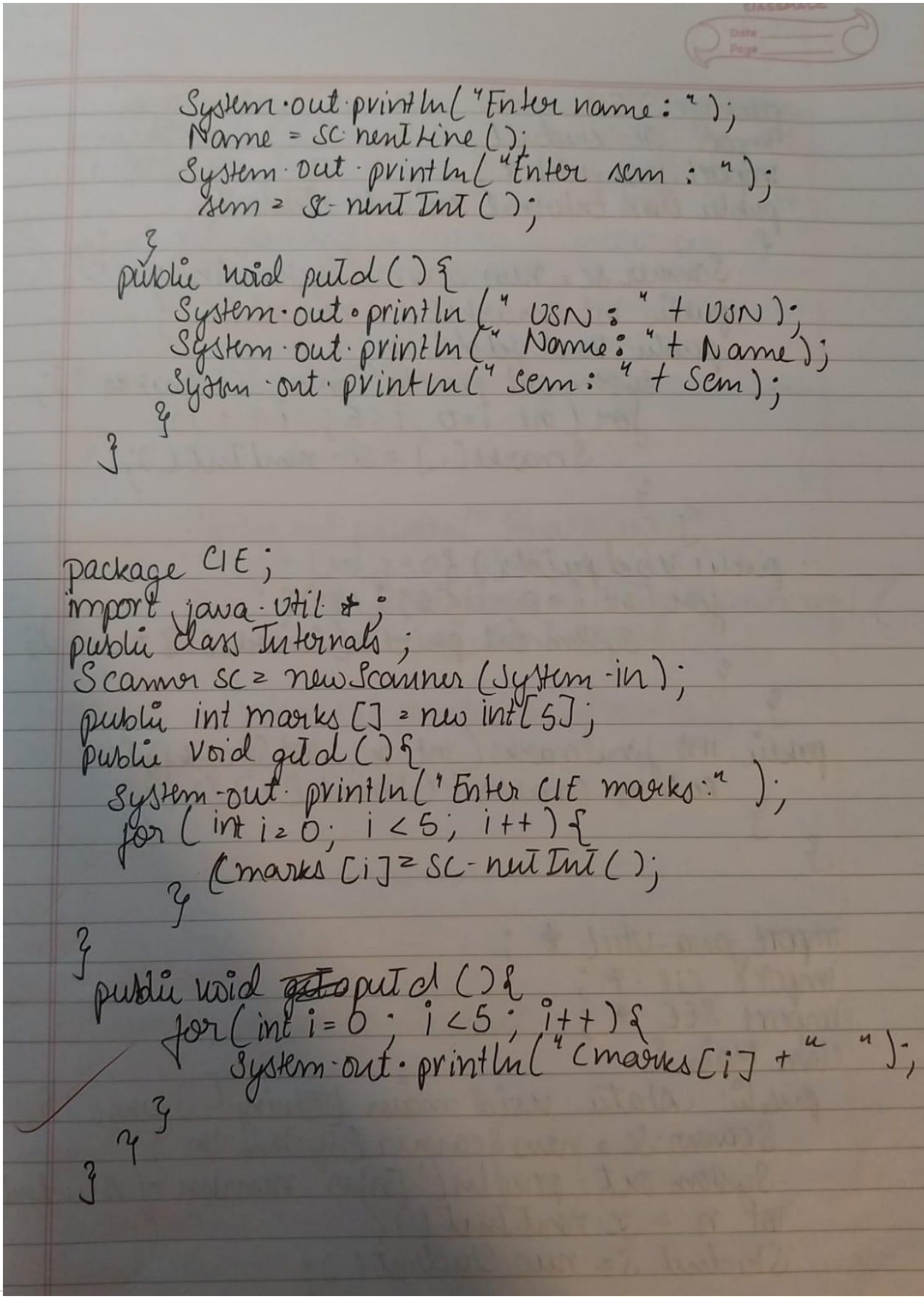
Pgm: 6

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

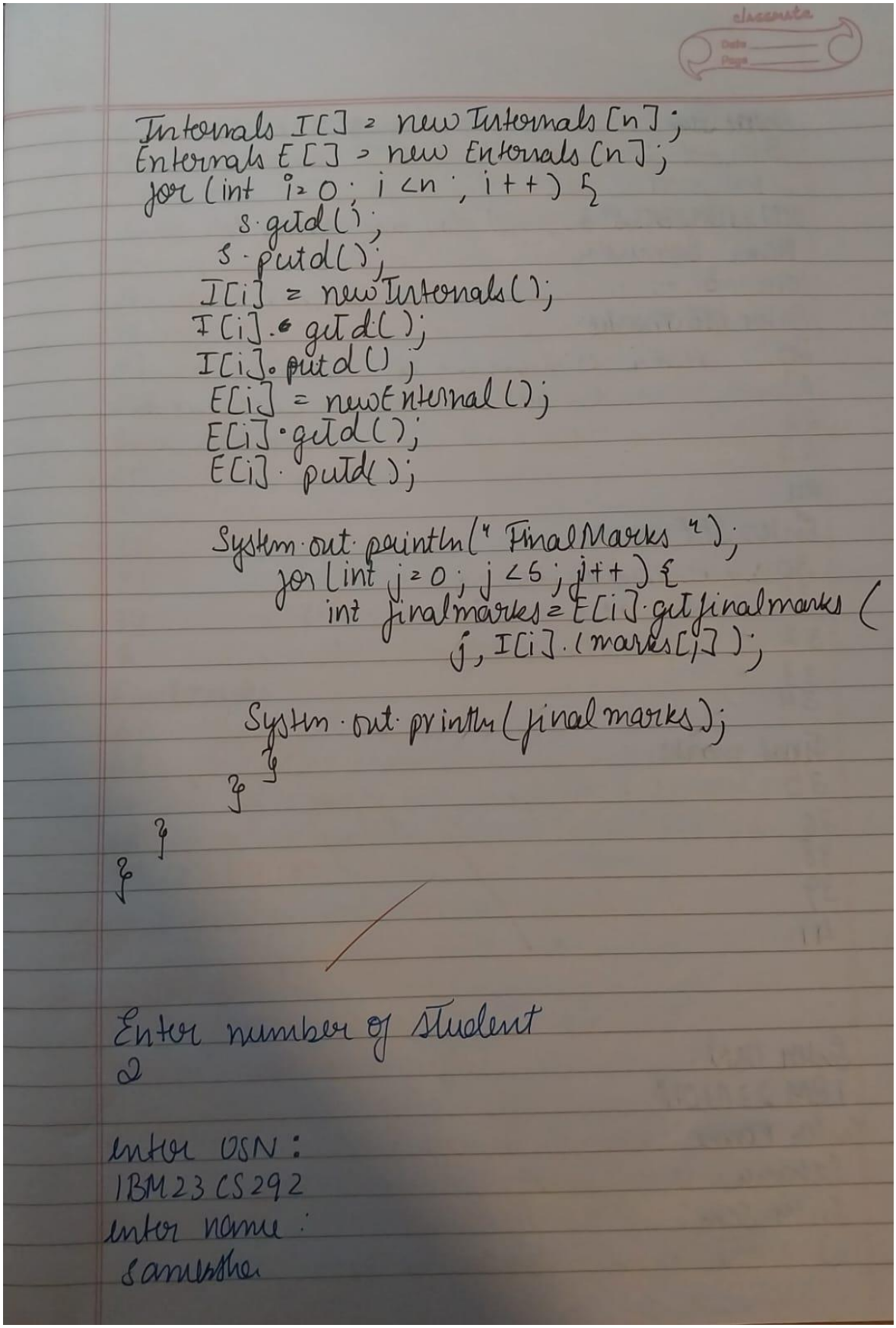
```
package CIE ;
import java.util.*;
public class Student
{
 public String USN;
 public String Name;
 public int sem;

 Scanner SC = new Scanner( System.in );

public void getd(){
  System.out.println();
  SC.nextLine();
  System.out.println("Enter USN : ");
  USN = SC.nextLine();
```



```
package SEE
import CIE.student;
import java.util.*;
public class External extends student
{
    Scanner sc = new Scanner(System.in);
    public int smarks[] = new int[5];
    public void getd(){
        System.out.println("Enter SEE marks ");
        for(int i=0; i<5; i++){
            smarks[i] = sc.nextInt();
        }
    }
public void putd(){
    for(int i=0; i<5; i++){
        System.out.println(smarks[i] + " ");
    }
}
public int finalmarks(int x, int smark){
    return (marks + (smark[x]/2));
}
}

import java.util.*;
import CIE.*;
import SEE.*;
class Main{
 public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter number of student");
    int n = sc.nextInt();
    Student s = new Student();
```

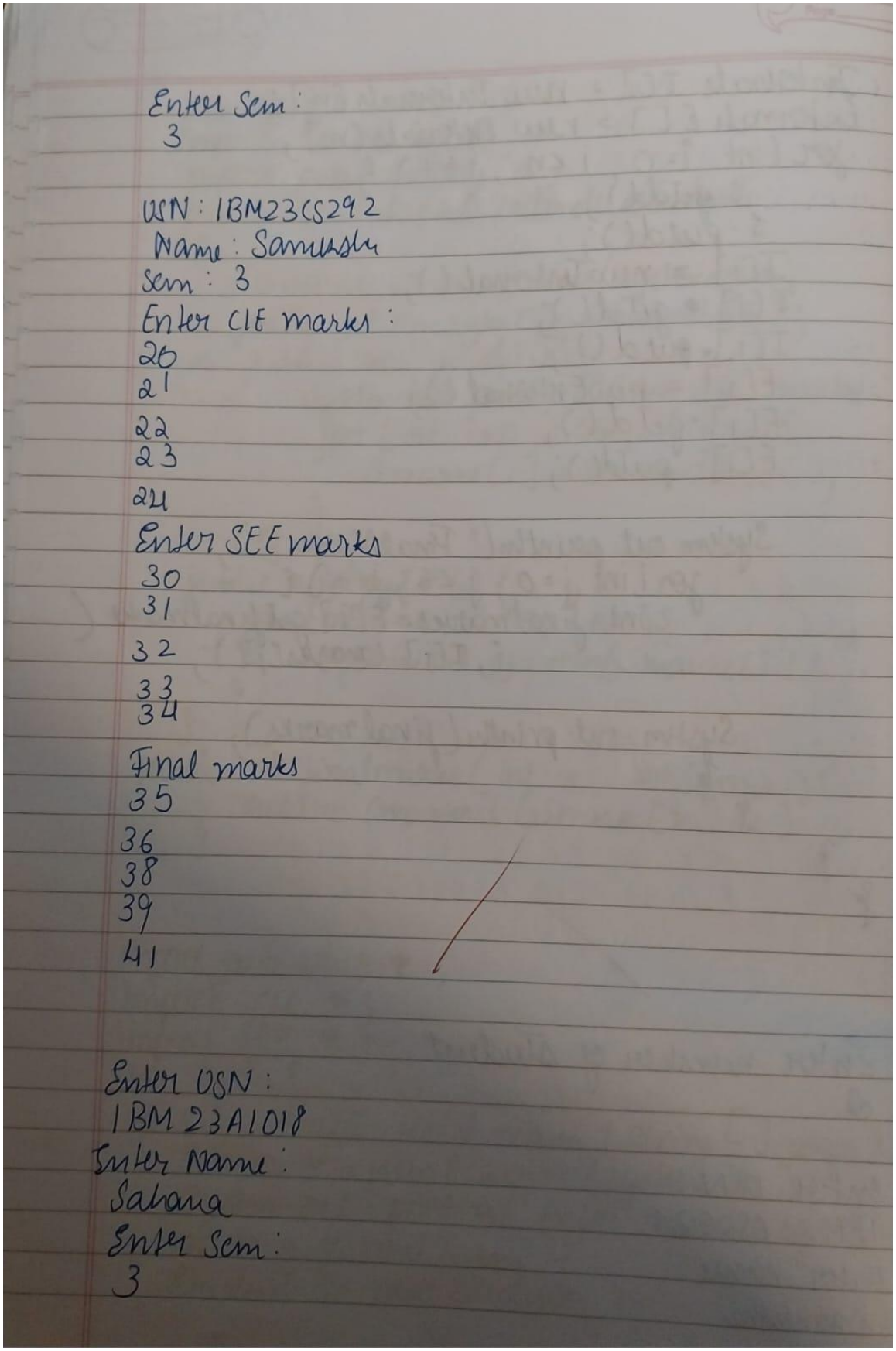


```
        System.out.println("Enter name: ");
        Name = sc.nextLine();
        System.out.println("Enter sem: ");
        Sem = sc.nextInt();
    }
    public void putd(){
        System.out.println("USN: " + USN);
        System.out.println("Name: " + Name);
        System.out.println("Sem: " + Sem);
    }
}

package CIE;
import java.util.*;
public class Internals;
Scanner sc = new Scanner(System.in);
public int marks[] = new int[5];
public void getd(){
    System.out.println("Enter CIE marks:");
    for(int i=0; i<5; i++){
        marks[i] = sc.nextInt();
    }
}
public void putd(){
    for(int i=0; i<5; i++){
        System.out.println(marks[i] + " ");
    }
}
}
```



```
Internals I[] = new Internals[n];
External E[] = new External[n];
for(int i=0; i<n; i++) {
    s.getId();
    s.putId();
    I[i] = new Internals();
    I[i].getId();
    I[i].putId();
    E[i] = new External();
    E[i].getId();
    E[i].putId();

    System.out.println("FinalMarks");
    for(int j=0; j<5; j++) {
        int finalmarks = E[i].getfinalmarks(
            j, I[i].(marks[j]));

        System.out.println(finalmarks);
    }
}
}
```

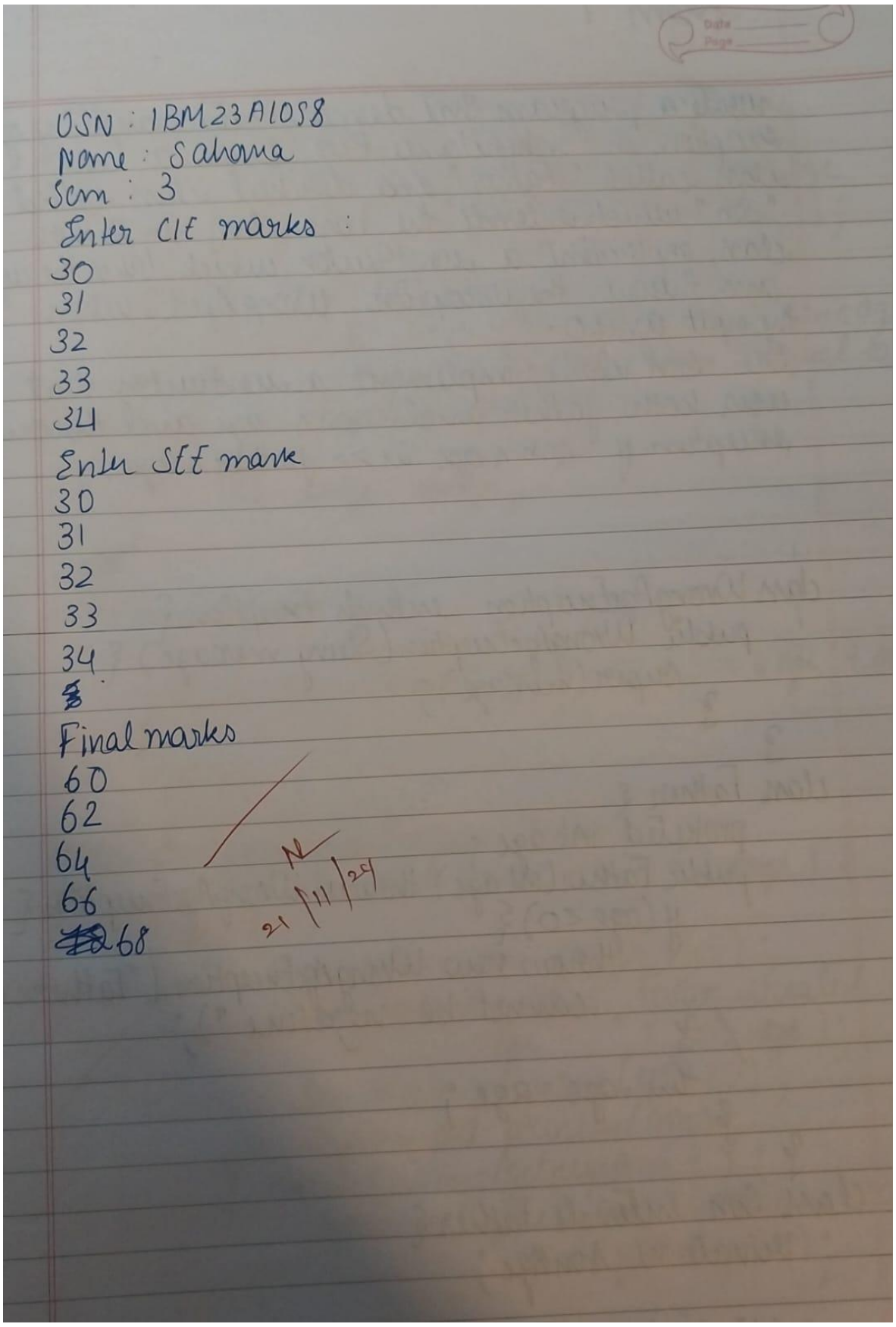
```
Enter number of Student
2

Enter USN:
1BM23CS292
Enter name:
Samiksha
```


Enter Sem:
3

USN: 1BM23CS292
Name: Samudra
Sem: 3
Enter CIE marks:
20
21
22
23
24
Enter SEE marks
30
31
32
33
34
Final marks
35
36
38
39
41

Enter USN:
1BM 23A1018
Enter Name:
Sahana
Enter Sem:
3

USN : 1BM23AI058
Name : Sahana
Sem : 3
Enter CIE marks :
30
31
32
33
34
Enter SEE marks
30
31
32
33
34
~~35~~
Final marks
60
62
64
66
~~70~~ 68



```
package SEE;
import CIE.Student;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Student {
	int marksSee[] = new int[5];
	public void getMarks() {
		for(int i=0;i<5;i++) {
			Scanner sc = new Scanner(System.in);
			System.out.println("Enter SEE marks in subject "+(i+1));
			marksSee[i]=sc.nextInt();
		}
	}
	public void calcTotalMarks(Internals i1) {
		for(int i=0;i<5;i++) {
			System.out.println("Subject "+(i+1)+":
"+(i1.returnCieMarks(i)+(marksSee[i]/2)));
		}
		System.out.println();

	}
}


package CIE;
import java.util.Scanner;
public class Internals {
	public int marksCie[] = new int[5];
	public void getMarks() {
		for(int i=0;i<5;i++) {
			Scanner sc = new Scanner(System.in);
			System.out.println("Enter CIE marks in subject "+(i+1));
			marksCie[i]=sc.nextInt();
		}
	}
	public int returnCieMarks(int i) {
```

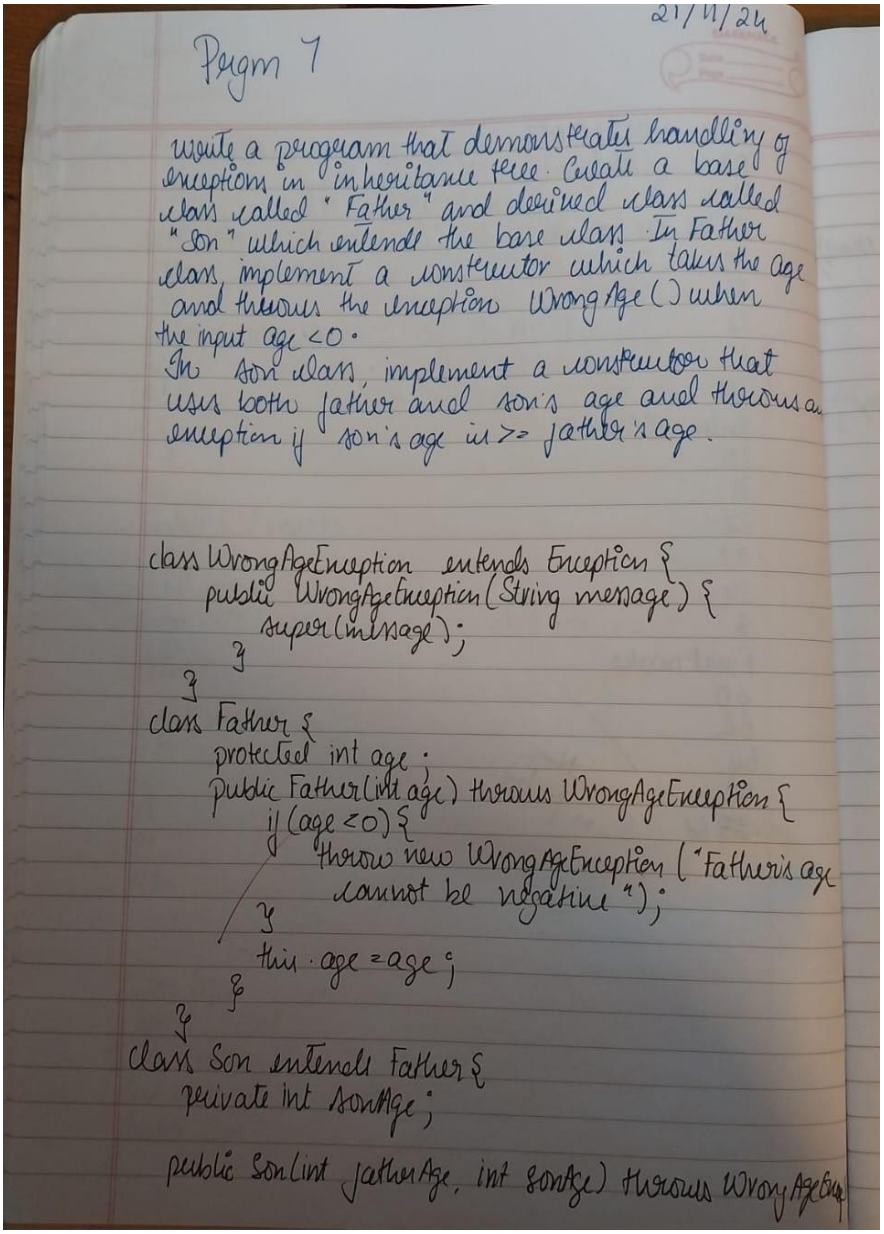
```
                return marksCie[i];
        }
}
package CIE;
import java.util.Scanner;
public class Student {
        String usn;
        String name;
        int sem;
        public void getd() {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter student USN");
                usn = sc.nextLine();
                System.out.println("Enter student name");
                name = sc.nextLine();
                System.out.println("Enter semester");
                sem = sc.nextInt();
        }
        public void display() {
                System.out.println();
                System.out.println("Student USN: "+usn);
                System.out.println("Student name: "+name);
                System.out.println("Semester: "+sem);
                System.out.println();
        }
}


import CIE.Student;
import CIE.Internals;
import SEE.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 whose details you want
to enter");
                int n = sc.nextInt();
                Internals[] i1 = new Internals[n];
                Externals[] e1 = new Externals[n];
                for(int i=0;i<n;i++) {
                        System.out.println("Student "+(i+1)+" details:");
                        e1[i] = new Externals();
```

```
                i1[i] = new Internals();
                e1[i].getd();
                i1[i].getMarks();
                e1[i].getMarks();
            }
            for(int i=0;i<n;i++) {
                e1[i].display();
                e1[i].calcTotalMarks(i1[i]);
            }

        }
    }
```

```
Enter number of students: 2
Enter details for CIE Student 1:
USN: 1
Name: sagar
Semester: 2
Enter internal marks for 5 courses:
38 40 41 45 46
Enter details for SEE Student 1:
USN: 1
Name: sagar
Semester: 2
Enter external marks for 5 courses:
39 42 45 50 48
Enter details for CIE Student 2:
USN: 2
Name: chetan
Semester: 3
Enter internal marks for 5 courses:
40 44 46 47 50
Enter details for SEE Student 2:
USN: 2
Name: chetan
Semester: 3
Enter external marks for 5 courses:
40 44 46 47 50

Final Marks for all students:
USN: 1, Name: sagar, Semester: 2
Internal Marks: 38 40 41 45 46
USN: 1, Name: sagar, Semester: 2
External Marks: 39 42 45 50 48
Final Marks: 77 82 86 95 94

USN: 2, Name: chetan, Semester: 3
Internal Marks: 40 44 46 47 50
USN: 2, Name: chetan, Semester: 3
External Marks: 40 44 46 47 50
Final Marks: 80 88 92 94 100
```

WEEK 7 :

Write a program that demonstrates handling of excep ons 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 excep on WrongAge( ) when the input age<0. In Son class, implement a constructor that uses both father and son’s age and throws an excep on if son’s age is >=father’s age.

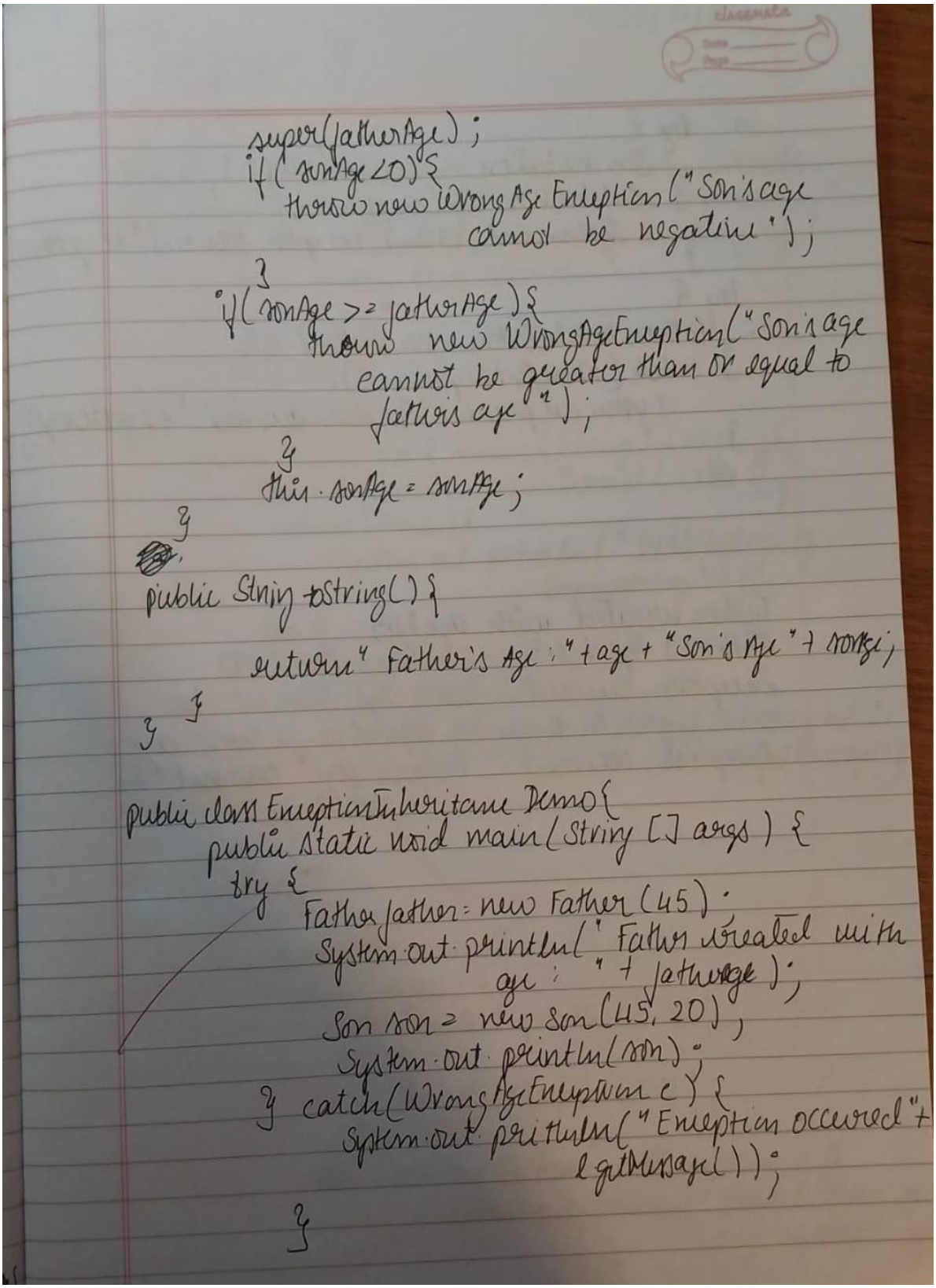


Pgm 7

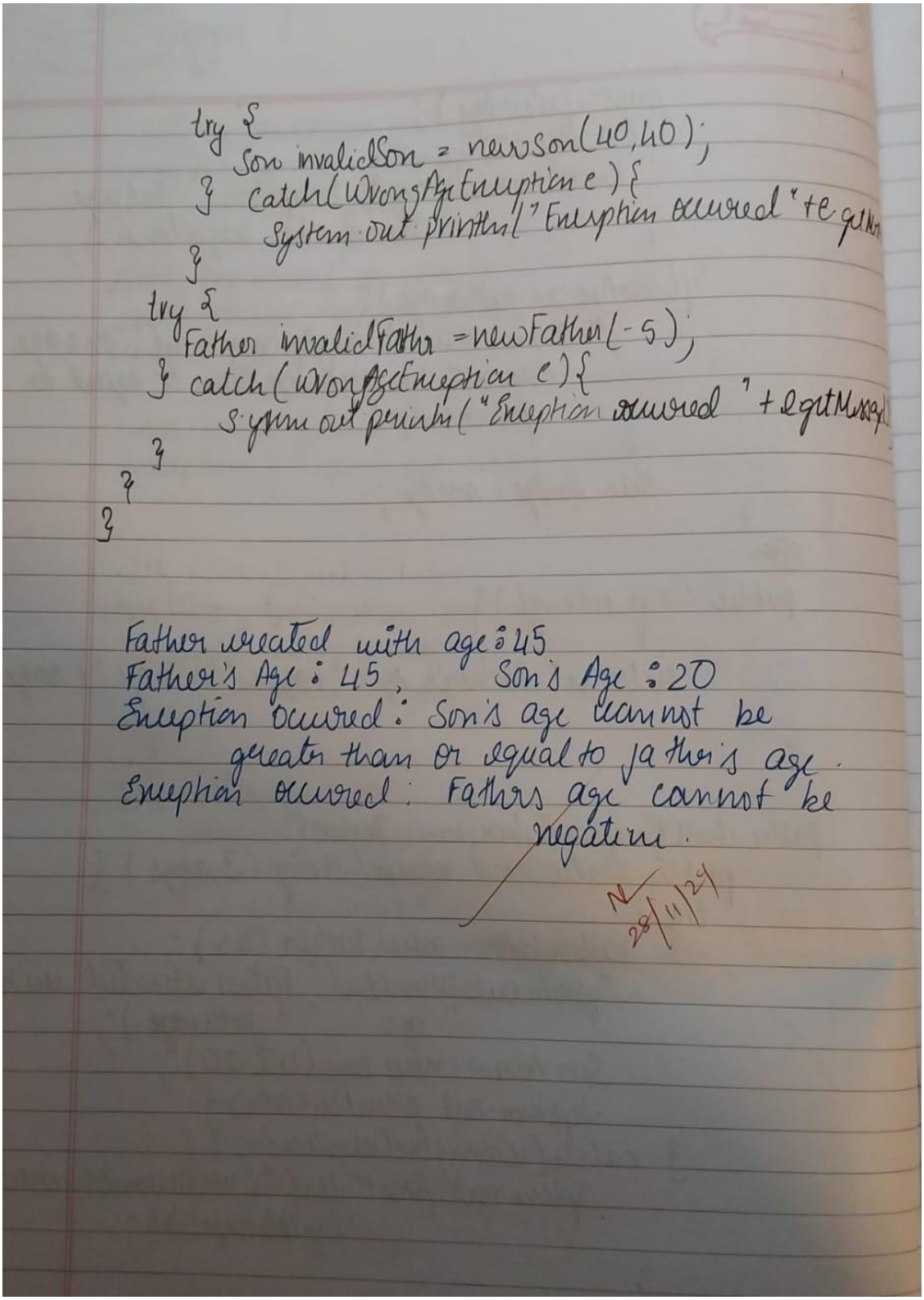
21/11/24

write a program that demonstrates handling of exception 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 uses both father and son's age and throws an exception if son's age is >= father's age.

```
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}
class Father {
    protected int age;
    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age
                cannot be negative");
        }
        this.age = age;
    }
}
class Son extends Father {
    private int sonAge;

    public Son(int fatherAge, int sonAge) throws WrongAgeEx
```



```
        super(fatherAge);
        if(sonAge<0){
            throw new WrongAgeException("Son's age
                                cannot be negative");
        }
        if(sonAge >= fatherAge){
            throw new WrongAgeException("Son's age
                cannot be greater than or equal to
                father's age");
        }
        this.sonAge = sonAge;
    }
    public String toString(){
        return "Father's Age: "+age+"Son's Age"+sonAge;
    }
}

public class ExceptionInheritanceDemo{
    public static void main(String[] args){
        try{
            Father father = new Father(45);
            System.out.println("Father created with
                                age: "+fatherAge);
            Son son = new Son(45, 20);
            System.out.println(son);
        }catch(WrongAgeException e){
            System.out.println("Exception occured"+
                                e.getMessage());
        }
```



```
try {
    Son invalidSon = new Son(40,40);
} catch(WrongAgeException e) {
    System.out.println("Exception occured "+e.getMessage());
}
try {
    Father invalidFather = new Father(-5);
} catch(WrongAgeException e) {
    System.out.println("Exception occured "+e.getMessage());
}
}
}
```

Father created with age : 45
Father's Age : 45, Son's Age : 20
Exception occured : Son's age cannot be greater than or equal to father's age.
Exception occured : Fathers age cannot be negative.

28/11/24

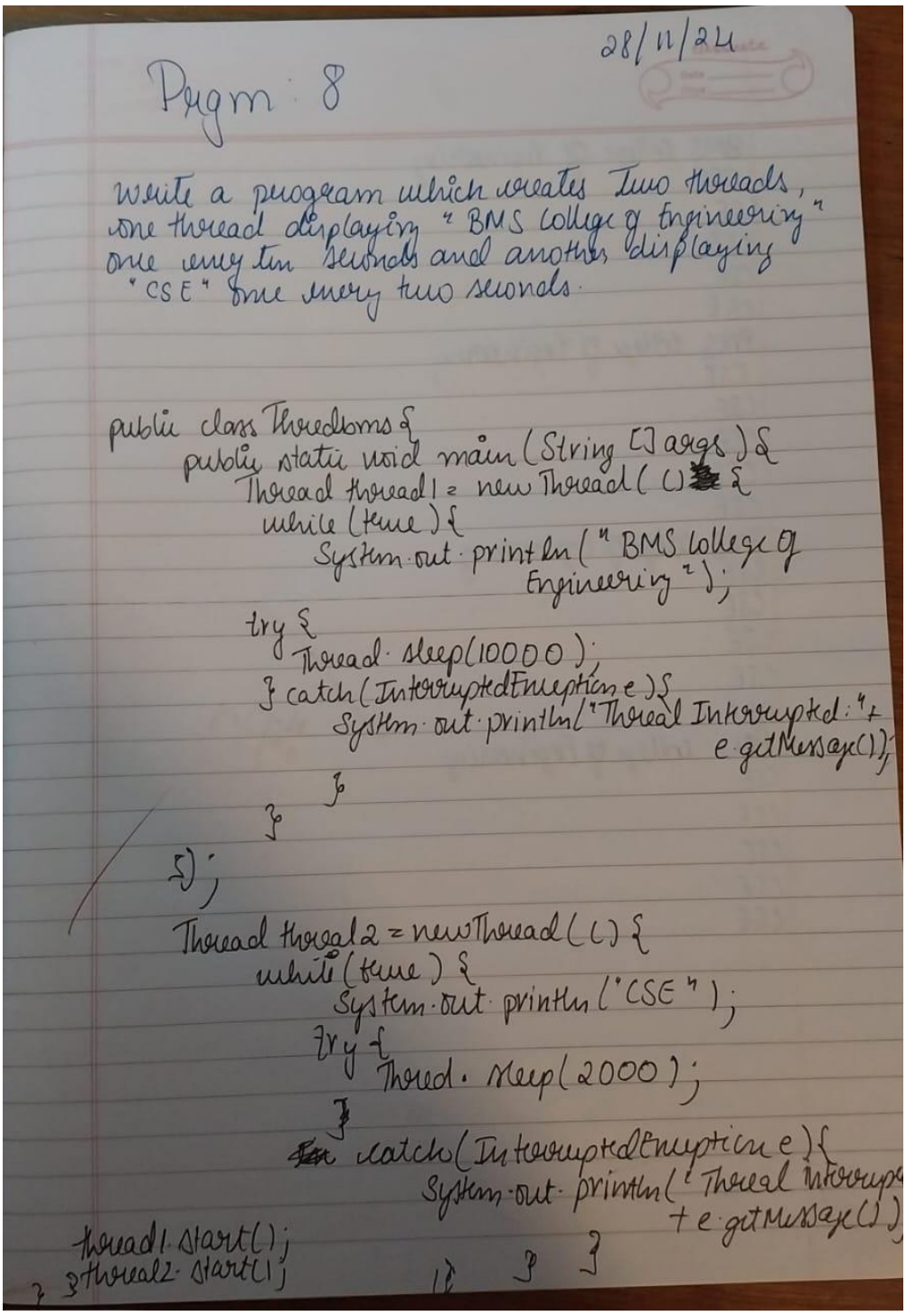
```
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}

// Base class Father
class Father {
    int age;

    // Constructor for Father class
    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age cannot be negative!");
        }
        this.age = age;
    }
}

// Derived class Son
class Son extends Father {
    int sonAge;

    // Constructor for Son class
    public Son(int fatherAge, int sonAge) throws WrongAgeException {
        super(fatherAge); // Call the Father constructor
        if (sonAge < 0) {
            throw new WrongAgeException("Son's age cannot be negative!");
        }
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age cannot be greater than or equal to father's age!");
        }
        this.sonAge = sonAge;
    }
}
public class ExceptionHandlingInheritance {
    public static void main(String[] args) {
        try {
            // Create a Father object
            Father father = new Father(40);

            // Create a Son object
            Son son = new Son(40, 20);
```

```
            System.out.println("Father's age: " + father.age);
            System.out.println("Son's age: " + son.sonAge);
        } catch (WrongAgeException e) {
            System.out.println("Exception occurred: " + e.getMessage());
        }
        try {
            Father invalidFather = new Father(-5);
        } catch (WrongAgeException e) {
            System.out.println("Exception occurred: " + e.getMessage());
        }

        try {
            Son invalidSon = new Son(30, 35);
        } catch (WrongAgeException e) {
            System.out.println("Exception occurred: " + e.getMessage());
        }
    }
}
```

```
Father's age: 40
Son's age: 20
Exception occurred: Father's age cannot be negative!
Exception occurred: Son's age cannot be greater than or equal to father's age!
```

WEEK 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.



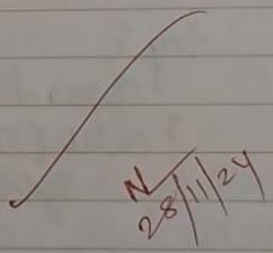
Pgm 8

28/11/24

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.

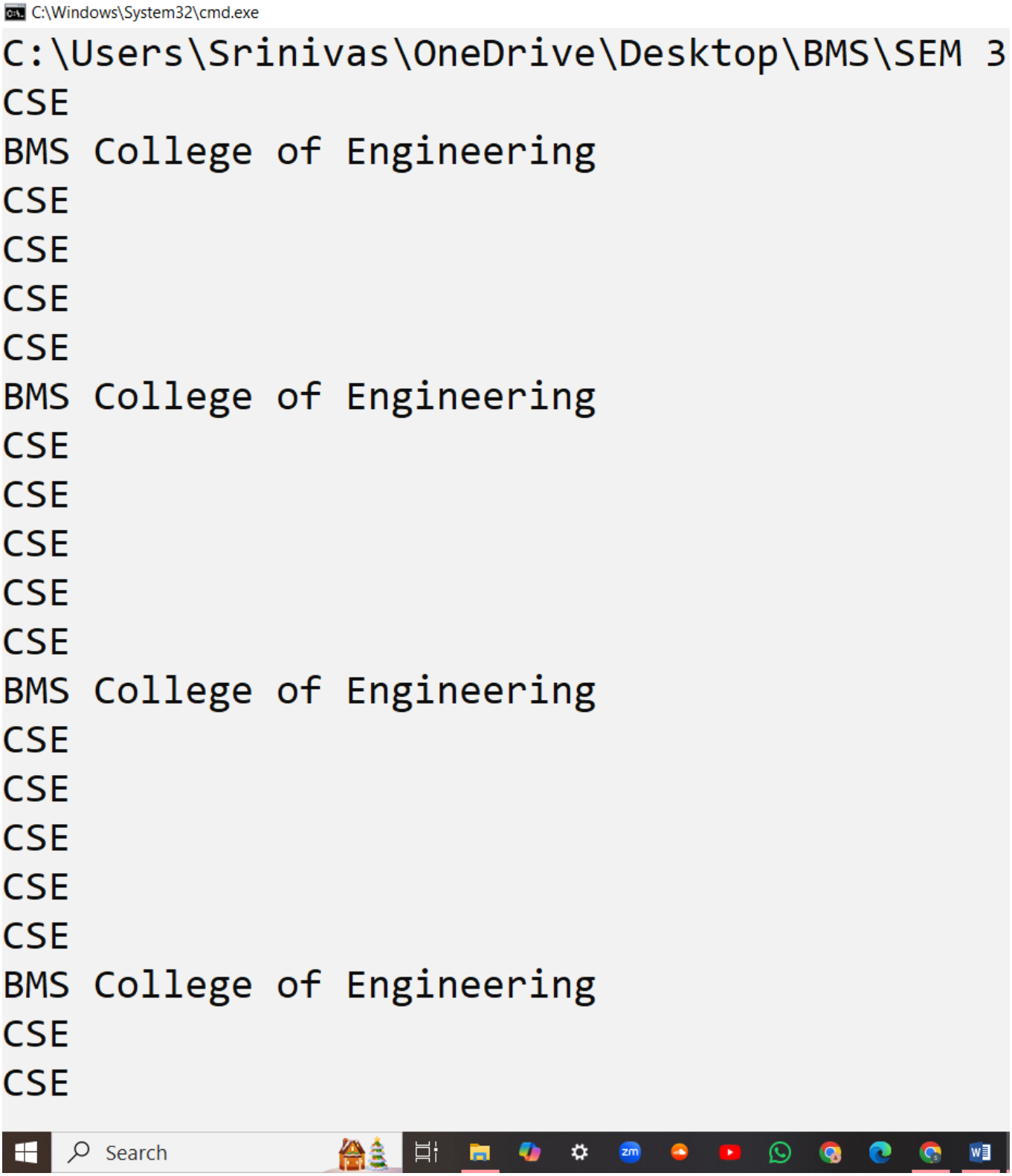
```
public class Threadbms {
    public static void main(String [] args) {
        Thread thread1 = new Thread( () -> {
            while (true) {
                System.out.println("BMS College of
                                    Engineering");
                try {
                    Thread.sleep(10000);
                } catch (InterruptedException e) {
                    System.out.println("Thread Interrupted:" +
                                        e.getMessage());
                }
            }
        });
        Thread thread2 = new Thread( () -> {
            while (true) {
                System.out.println("CSE");
                try {
                    Thread.sleep(2000);
                } catch (InterruptedException e) {
                    System.out.println("Thread Interrupt
                                        + e.getMessage());
                }
            }
        });
        thread1.start();
        thread2.start();
    }
}
```

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



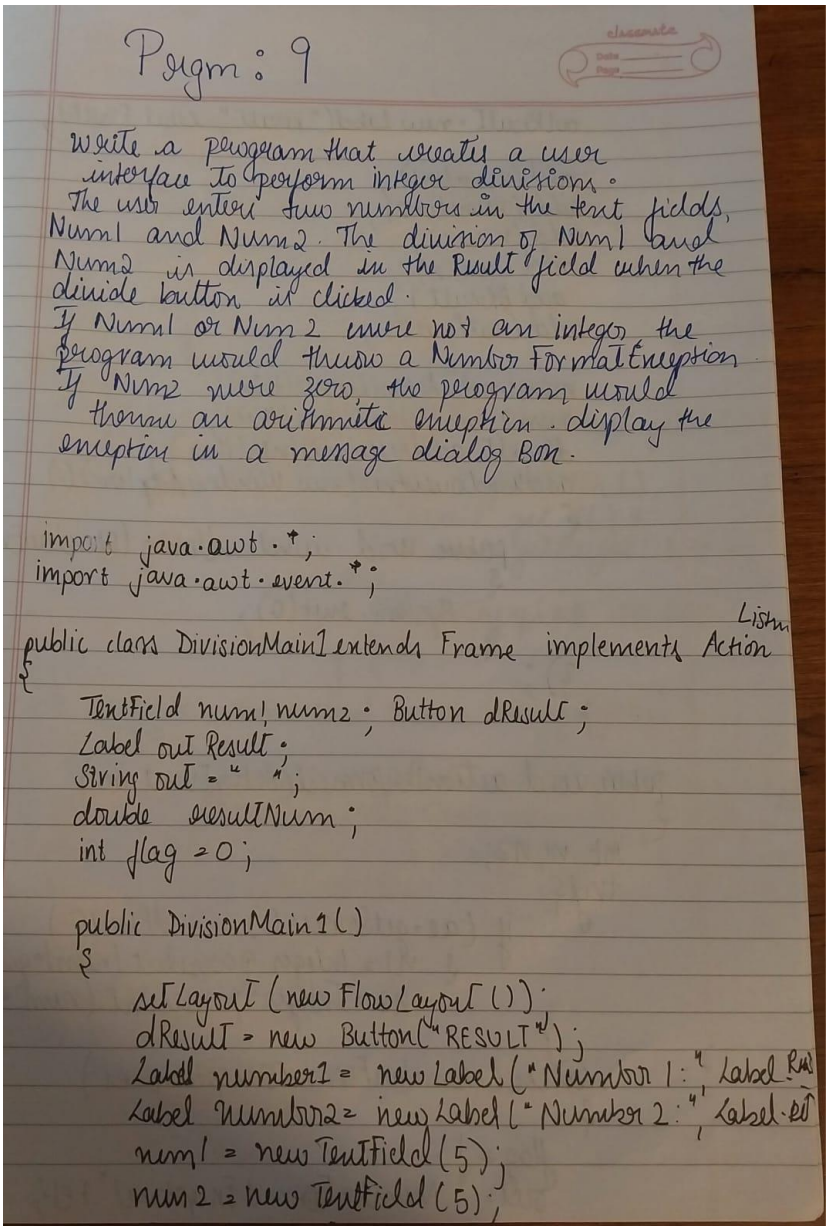
N
28/11/24

```
class CollegeThread extends Thread {
    public void run() {
        try {
            while (true) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            }
        } catch (InterruptedException e) {
            System.out.println("CollegeThread interrupted");
        }
    }
}
class DepartmentThread extends Thread {
    public void run() {
        try {
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000);            }
        } catch (InterruptedException e) {
            System.out.println("DepartmentThread interrupted");
        }
    }
}
public class MultiThreadExample {
    public static void main(String[] args) {
        CollegeThread collegeThread = new CollegeThread();
        DepartmentThread departmentThread = new DepartmentThread();
        collegeThread.start();
        departmentThread.start();
    }
}
```


C:\Windows\System32\cmd.exe
C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
Search

Week 9

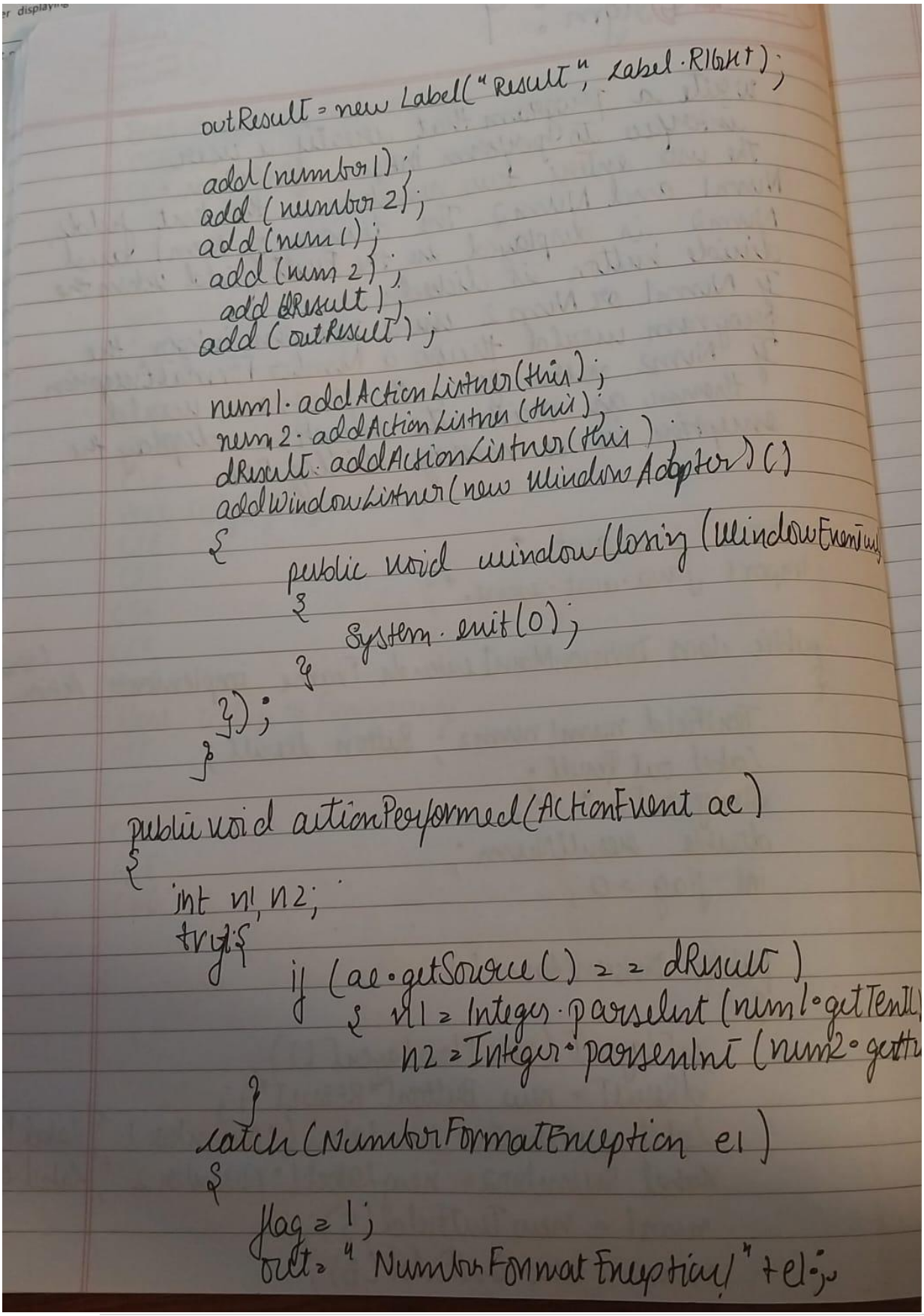
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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



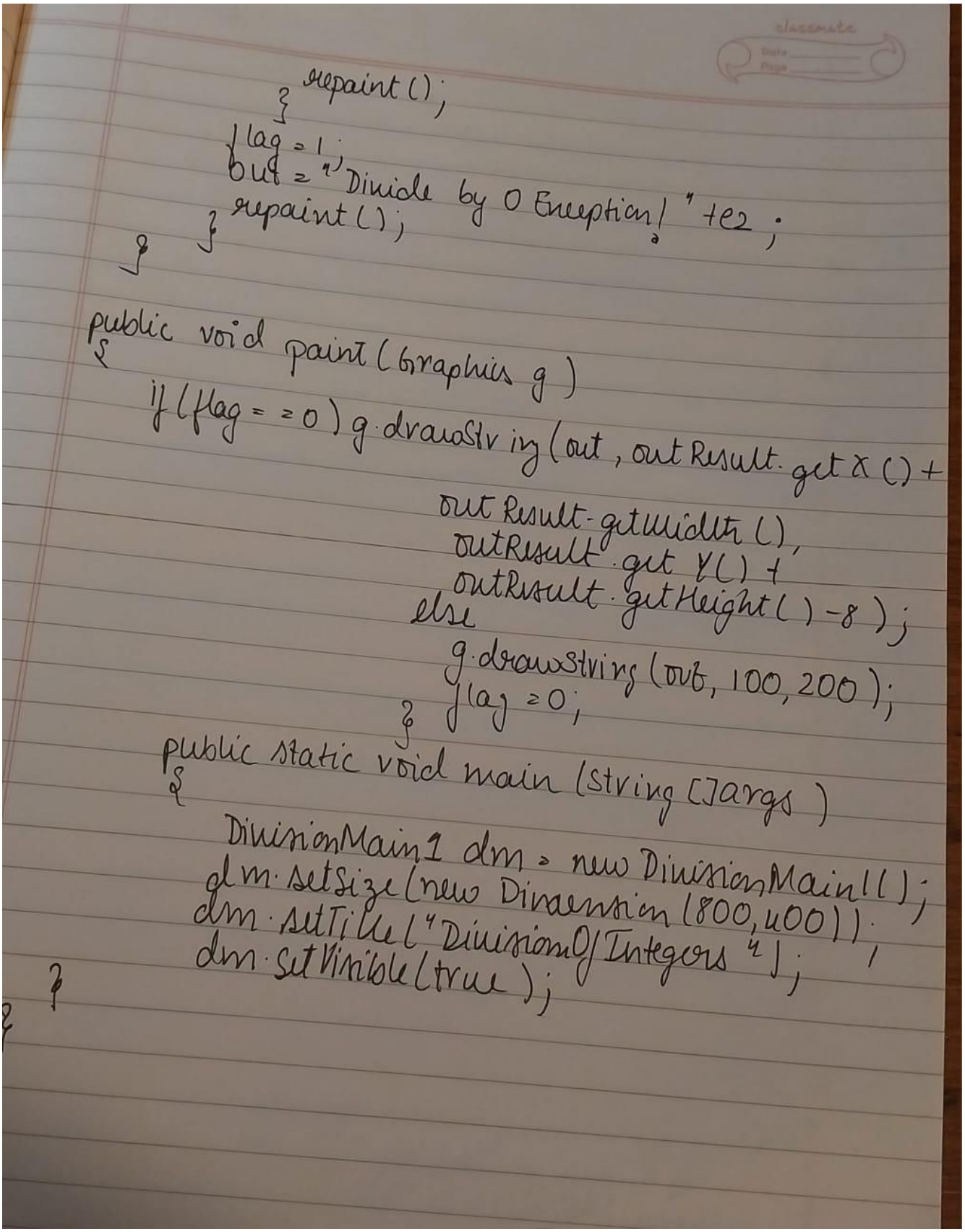
Pgm : 9

Write a program that creats 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 NumberFormatEnception If Num2 were zero, the program would throw an arithmetic exception. display the exception in a message dialog Box.

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

public class DivisionMain1 extends Frame implements ActionListn
{
    TextField num1, num2; Button dResult;
    Label out Result;
    String out = " ";
    double resultNum;
    int flag = 0;

    public DivisionMain1()
    {
        setLayout(new FlowLayout());
        dResult = new Button("RESULT");
        Label number1 = new Label("Number 1:", Label.RIG
        Label number2 = new Label("Number 2:", Label.RI
        num1 = new TextField(5);
        num2 = new TextField(5);
```



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

add(number1);
add(number2);
add(num1);
add(num2);
add(dResult);
add(outResult);

num1.addActionListner(this);
num2.addActionListner(this);
dResult.addActionListner(this);
addWindowListner(new WindowAdapter()
{
    public void windowClosing(WindowEvent
    {
        System.exit(0);
    }
});
}

public void actionPerformed(ActionEvent ae)
{
int n1, n2;
try{
    if (ae.getSource() == dResult)
    { n1 = Integer.parseInt(num1.getTeni
      n2 = Integer.parseInt(num2.getTe

}
catch(NumberFormatException e1)
{
    flag = 1;
    out = "NumberFormatException" + e1.
```

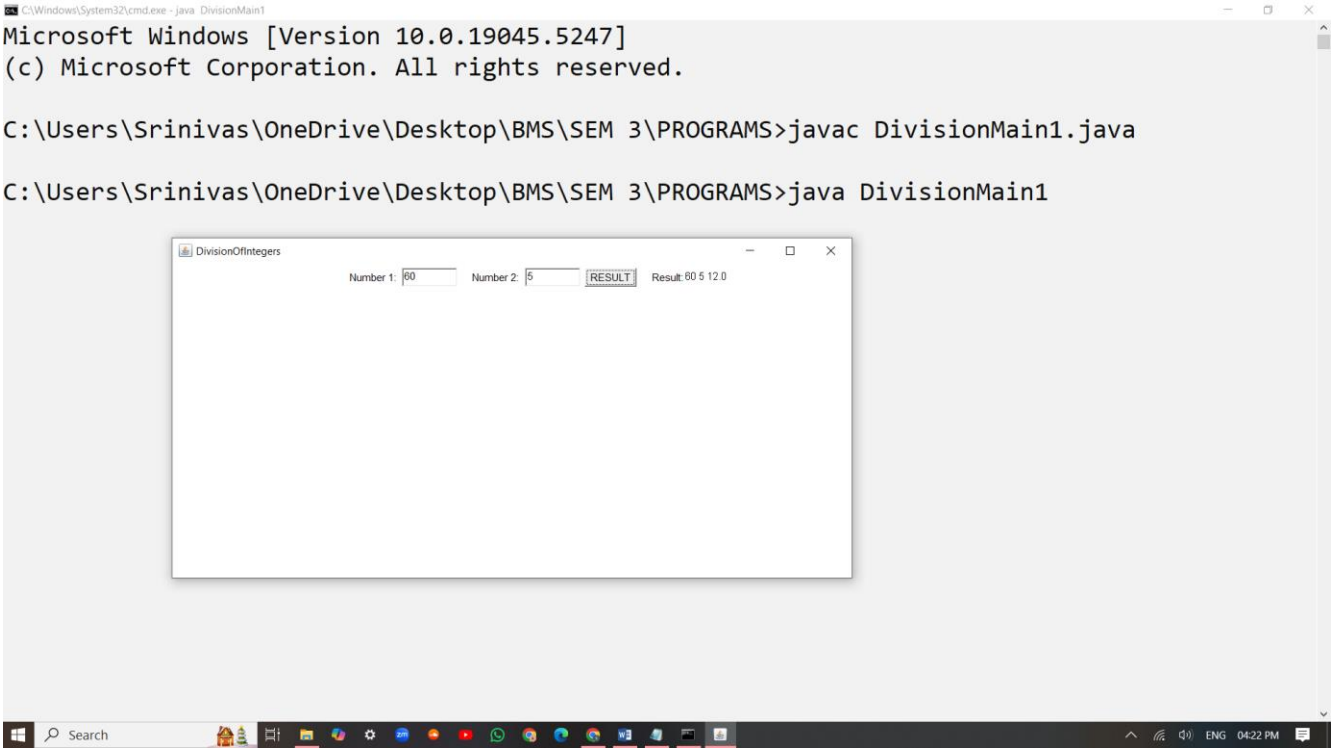


```
        repaint();
    }
    flag = 1;
    out = "Divide by 0 Exception!" + e2;
    repaint();
  }
}

public void paint(Graphics g)
{
  if(flag == 0) g.drawString(out, outResult.getX() +
                    outResult.getWidth(),
                    outResult.getY() +
                    outResult.getHeight() - 8);
  else
    g.drawString(out, 100, 200);
    flag = 0;
  }
public static void main(String []args)
{
  DivisionMain1 dm = new DivisionMain1();
  dm.setSize(new Dimension(800,400));
  dm.setTitle("Division Of Integers");
  dm.setVisible(true);
}
}
```

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

public class DivisionMain1 extends Frame implements ActionListener
{
        TextField
        num1,num2; Button
        dResult;
        Label
        outResult;
        String out="";
        double
        resultNum; int
        flag=0;

        public DivisionMain1()
        {
                setLayout(new FlowLayout());

                dResult = new Button("RESULT");
                Label number1 = new Label("Number
                1:",Label.RIGHT); Label number2 = new
                Label("Number 2:",Label.RIGHT);
                num1=new TextField(5);
                num2=new TextField(5);
                outResult = new Label("Result:",Label.RIGHT);

                add(number1);
                add(num1);
                add(number2);
                add(num2);
                add(dResult);
                add(outResult)
                ;

                num1.addActionListener(this);
                num2.addActionListener(this);
                dResult.addActionListener(this);
                addWindowListener(new
                WindowAdapter()
```

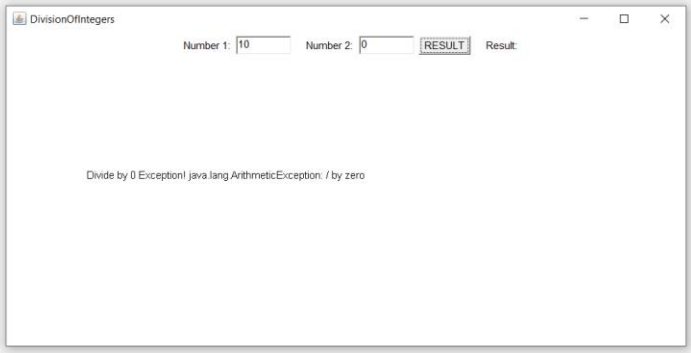
```
        {
                public void windowClosing(WindowEvent we)
                {
                        System.exit(0);
                }
        });
}
public void actionPerformed(ActionEvent ae)
{
        int n1,n2;
        try
        {
                if (ae.getSource() == dResult)
                {
                        n1=Integer.parseInt(num1.getText());
                        n2=Integer.parseInt(num2.getText());

                        /*if(n2==0)
                                throw new ArithmeticException();*/
                        out=n1+" "+n2+" ";
                        resultNum=n1/n2;
                        out+=String.valueOf(resultNu
                        m); repaint();

                }
        }
        catch(NumberFormatException e1)
        {
                flag=1;
                out="Number Format Exception!
                "+e1; repaint();
        }
        catch(ArithmeticException e2)
        {
                flag=1;
                out="Divide by 0 Exception!
                "+e2; repaint();
        }

}
public void paint(Graphics g)
```

```
{
        if(flag==0)
        g.drawString(out,outResult.getX()+outResult.getWidth(),outResult.
        getY()+outResult. getHeight()-8);
        else
        g.drawString(out,100,
        200); flag=0;
}
```



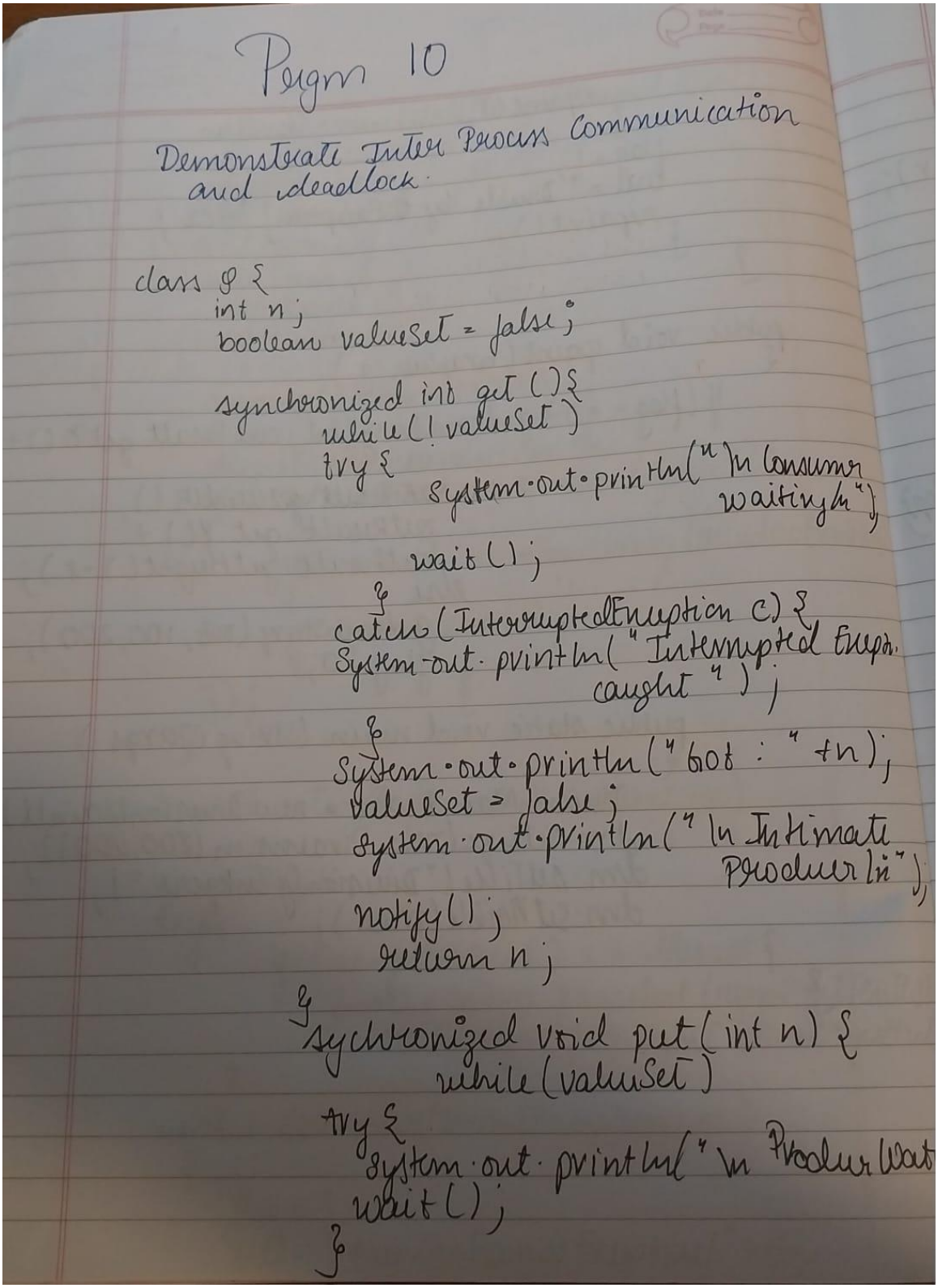
```
C:\Windows\System32\cmd.exe - java DivisionMain1
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3\PROGRAMS>javac DivisionMain1.java

C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3\PROGRAMS>java DivisionMain1
```


```
C:\Windows\System32\cmd.exe - java DivisionMain1
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3\PROGRAMS>javac DivisionMain1.java

C:\Users\Srinivas\OneDrive\Desktop\BMS\SEM 3\PROGRAMS>java DivisionMain1
```



WEEK 10

Demonstrate Inter process Communication and deadlock

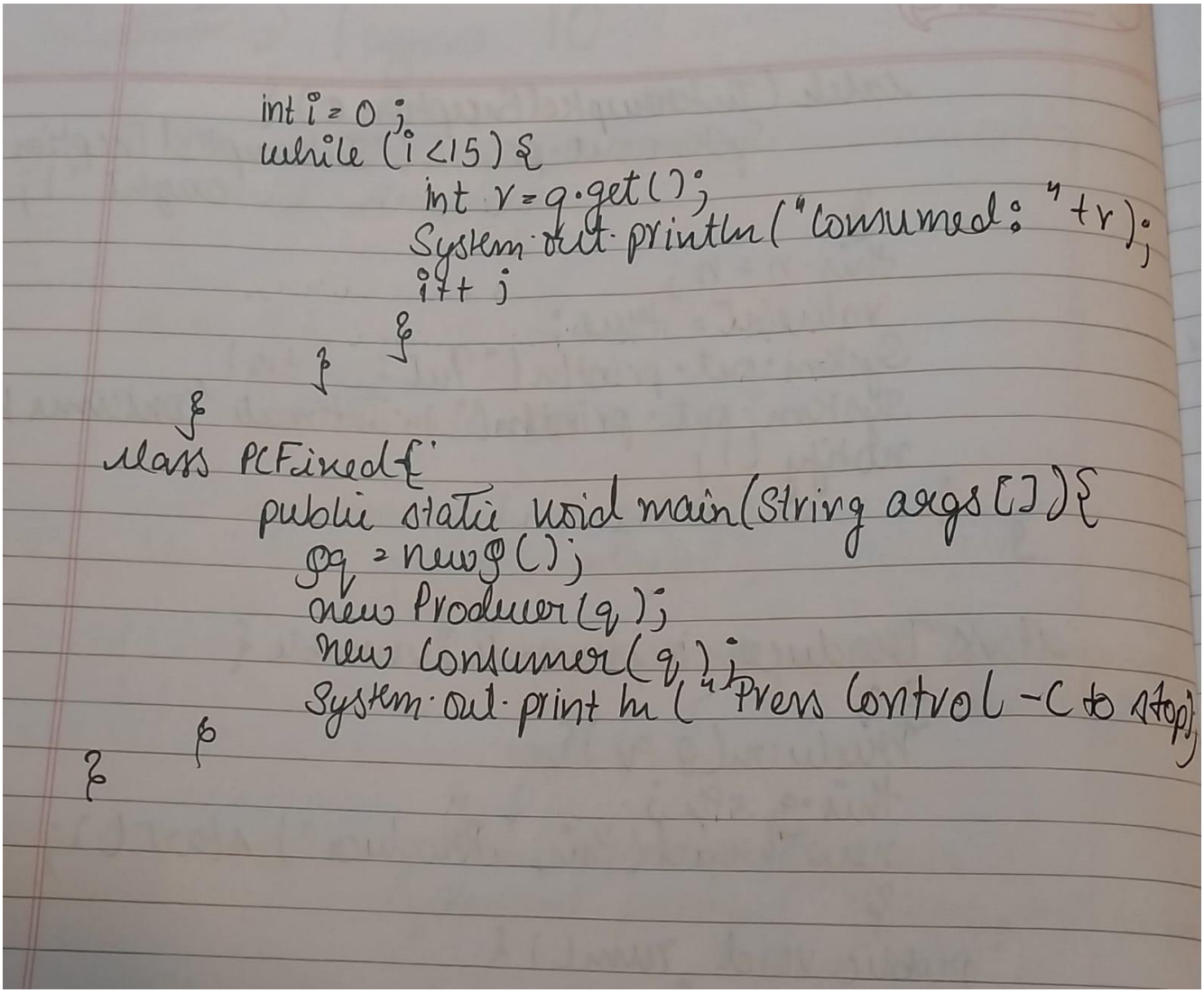


Pegm 10

Demonstrate Inter Process Communication and deadlock.

```
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("Interrupted Excep. caught");
        }
        System.out.println("Got : " + n);
        valueSet = false;
        System.out.println("\n Intimate Producer\n");
        notify();
        return n;
    }
    synchronized void put(int n) {
        while(valueSet)
        try {
            System.out.println("\n Producer wait
            wait();
        }
```

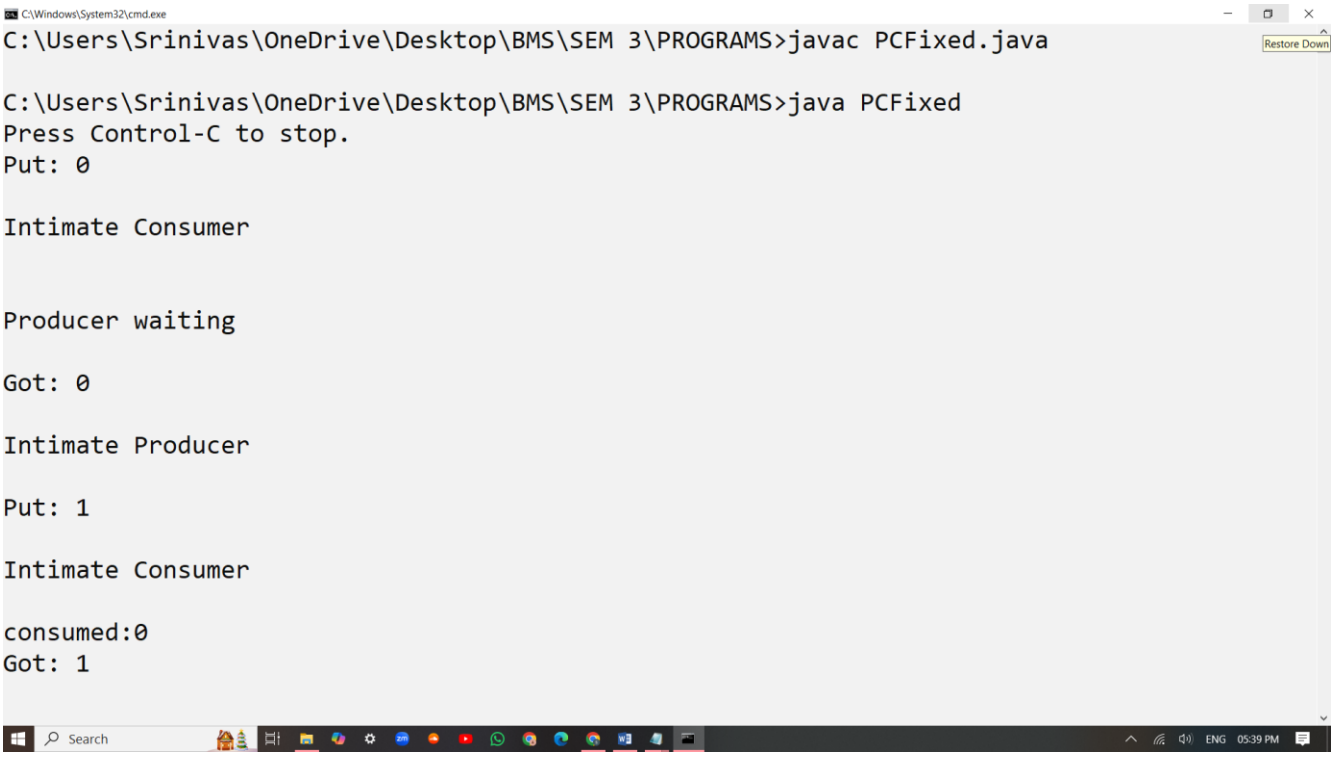
```
        catch (InterruptedException e) {
            System.out.println("InterruptedException
                                caught");
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("In Intimate Consumer");
        notify();
    }
}

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() {
```

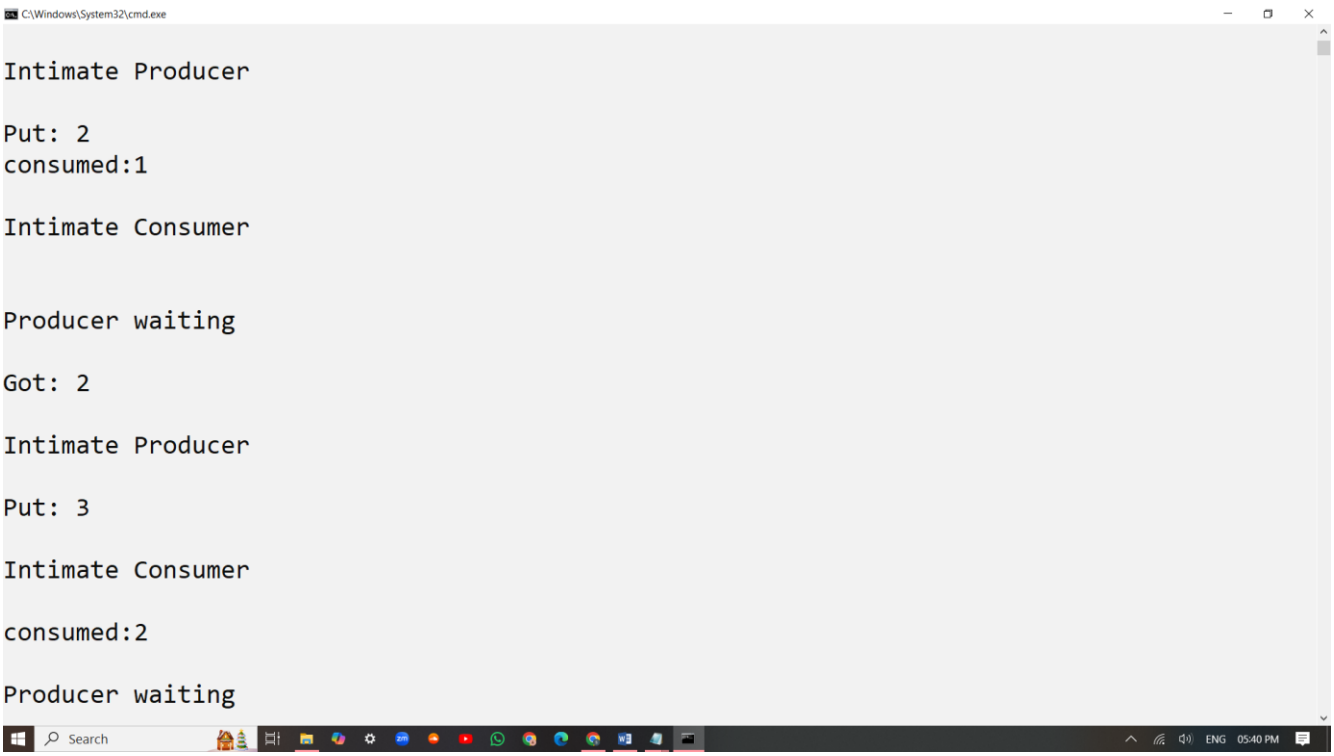


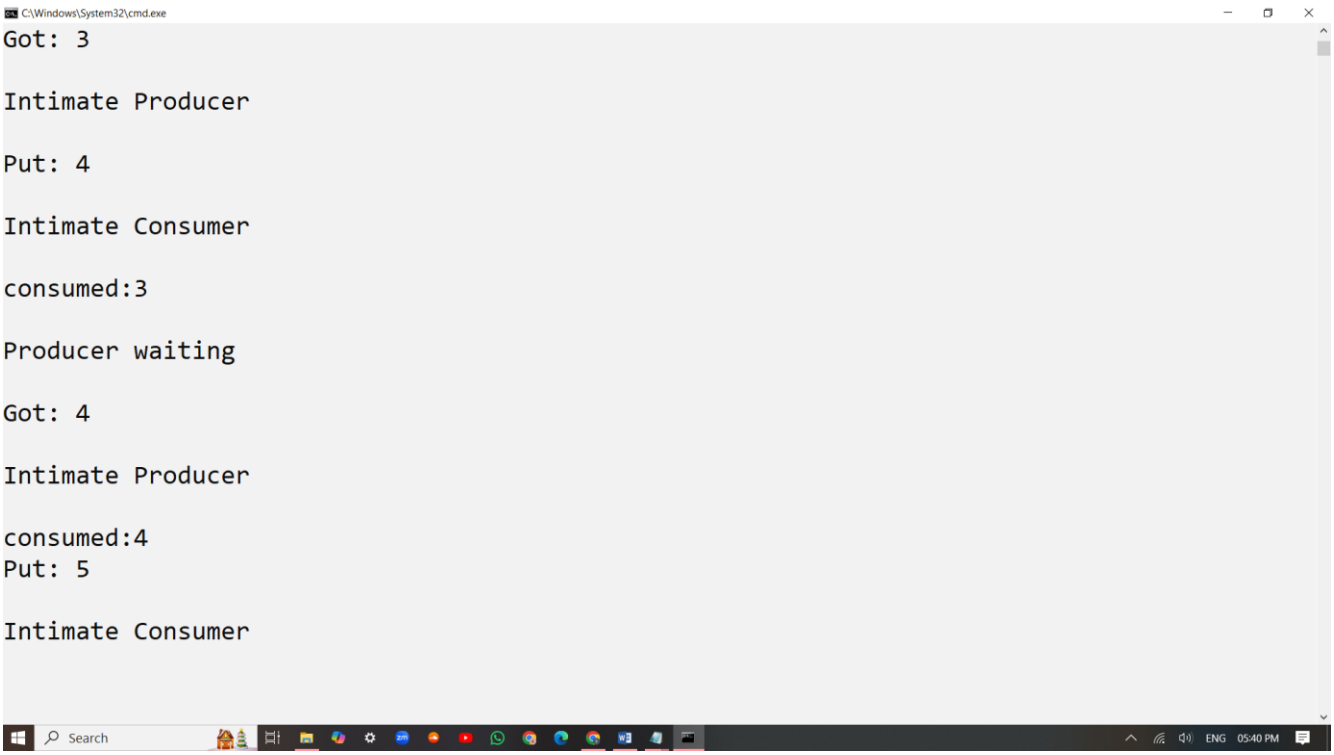

```
class Q {
int n;
boolean valueSet = false;

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

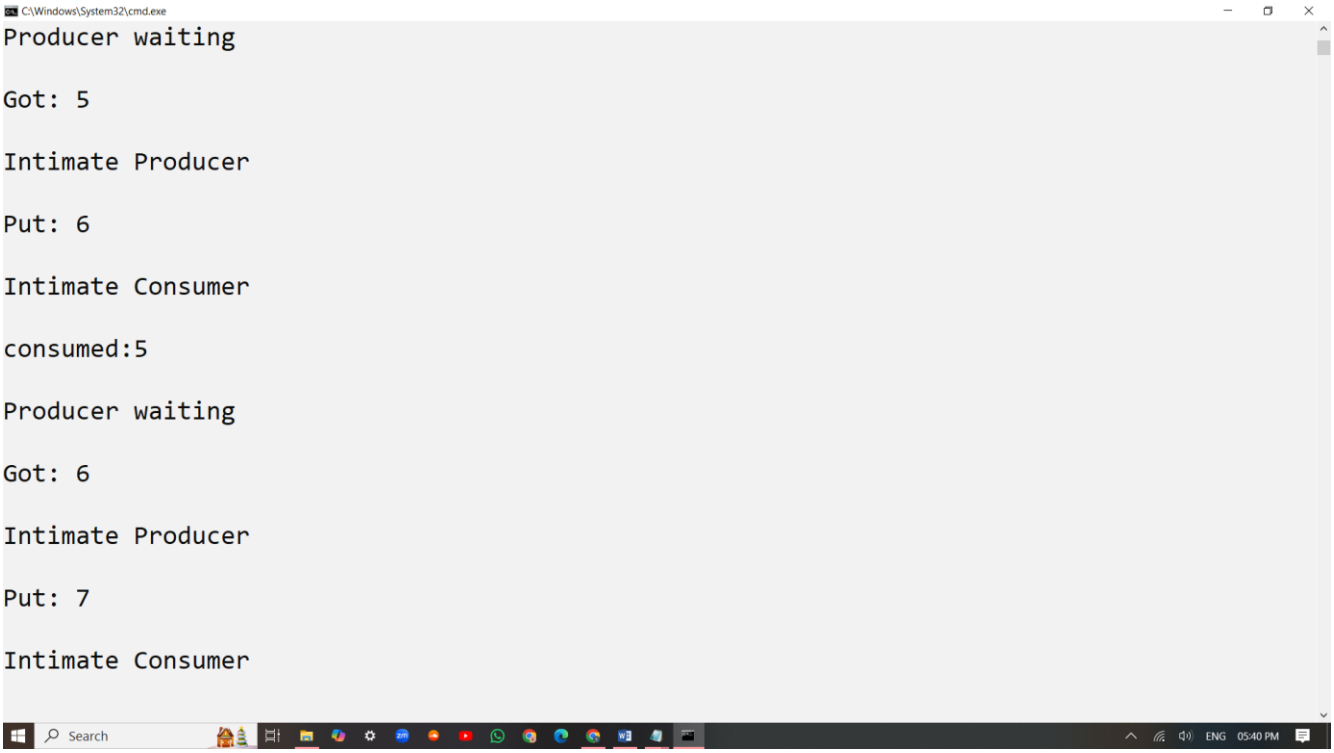
```
return n;
}

synchronized void put(int n) {
while(valueSet)
try {
System.out.println("\nProducer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
this.n = n;
valueSet = true;
System.out.println("Put: " + n);
System.out.println("\nIntimate Consumer\n");
notify();
}
}

class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
public void run() {
int i = 0;
while(i<15) {
q.put(i++);
}
}
}

class Consumer implements Runnable {
Q q;
Consumer(Q q) {
this.q = q;
new Thread(this, "Consumer").start();
}
public void run() {
        int i=0;
while(i<15) {
int r=q.get();
```

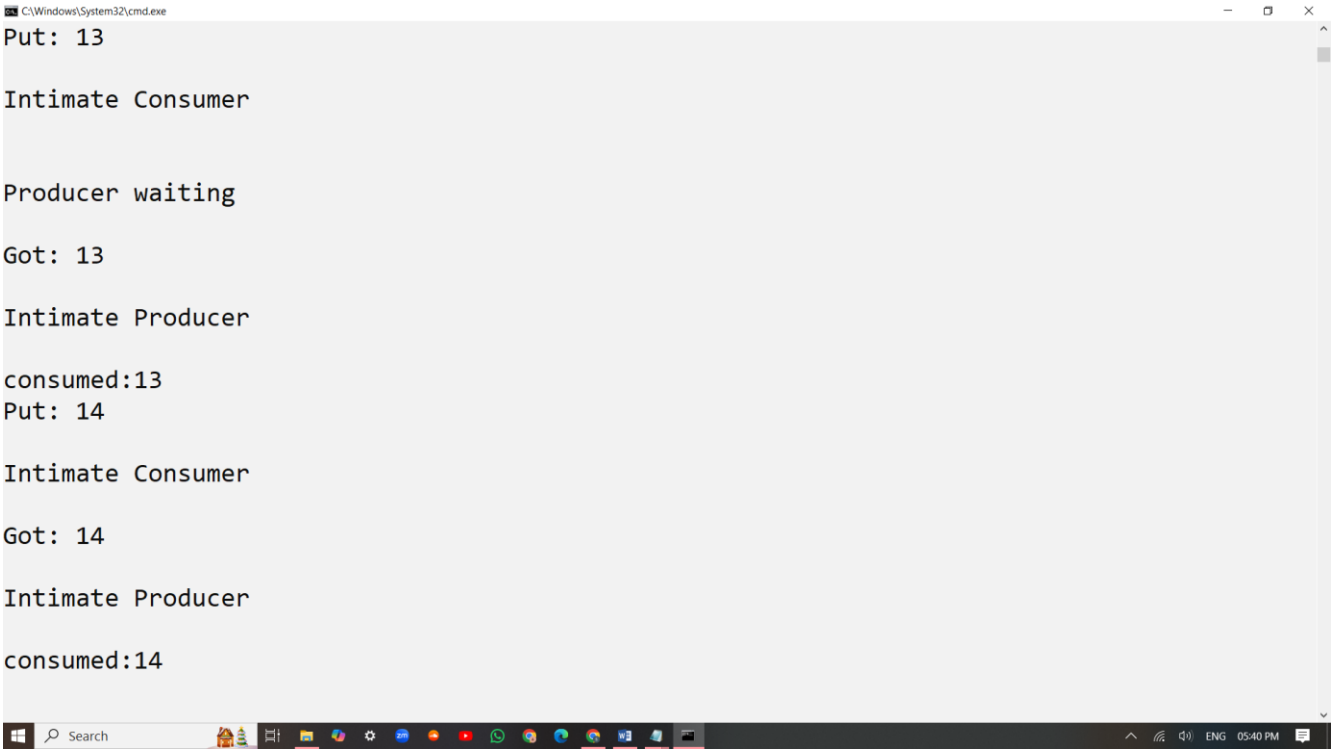
```
System.out.println("consumed:"+r);
i++;
}
}
}

class PCFixed {
public static void main(String args[]) {
Q q = new Q();
new Producer(q);
new Consumer(q);
System.out.println("Press Control-C to stop.");
}
}
```

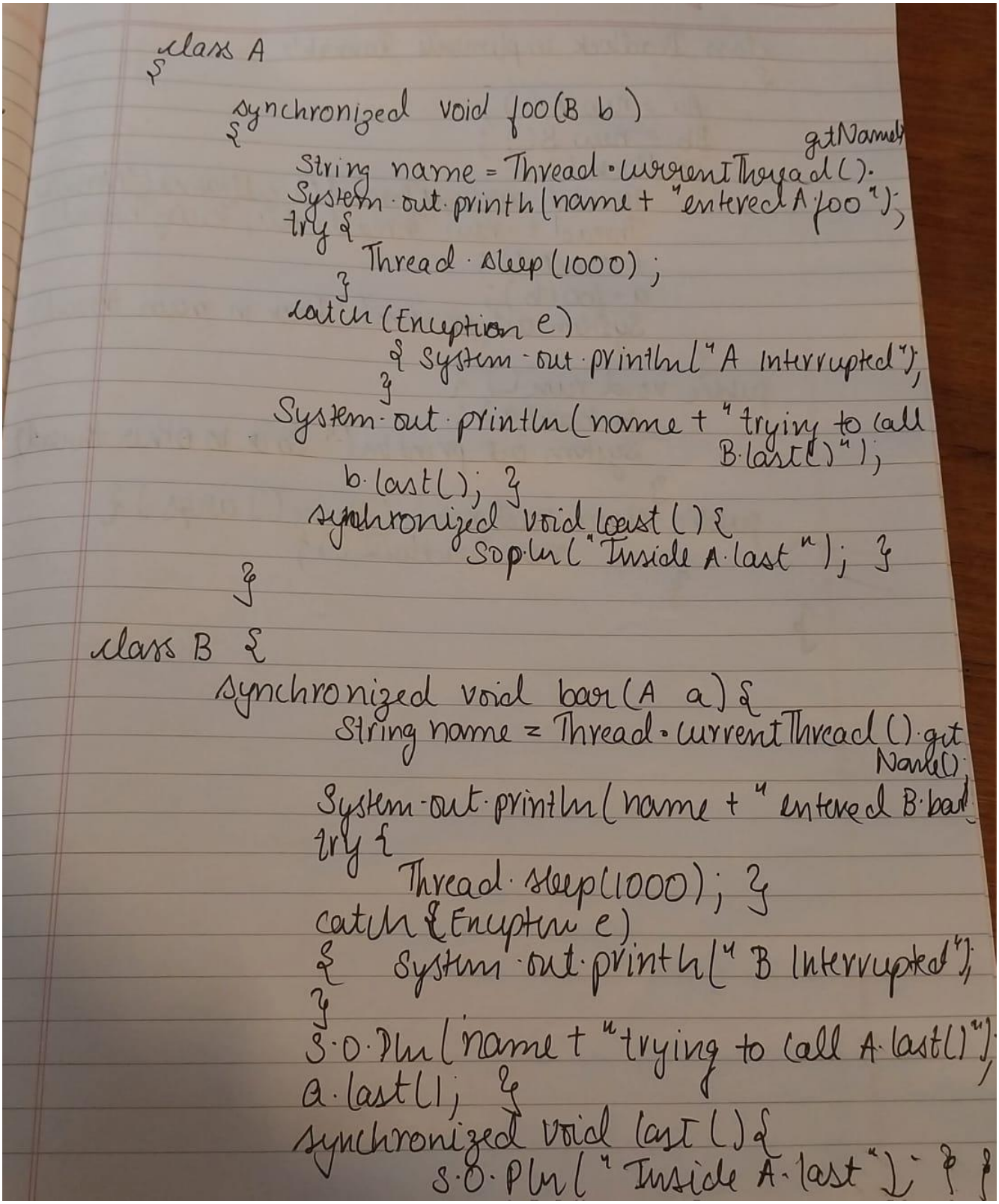



C:\Windows\System32\cmd.exe
Intimate Producer
Put: 2
consumed:1
Intimate Consumer
Producer waiting
Got: 2
Intimate Producer
Put: 3
Intimate Consumer
consumed:2
Producer waiting
Search
ENG 05:40 PM

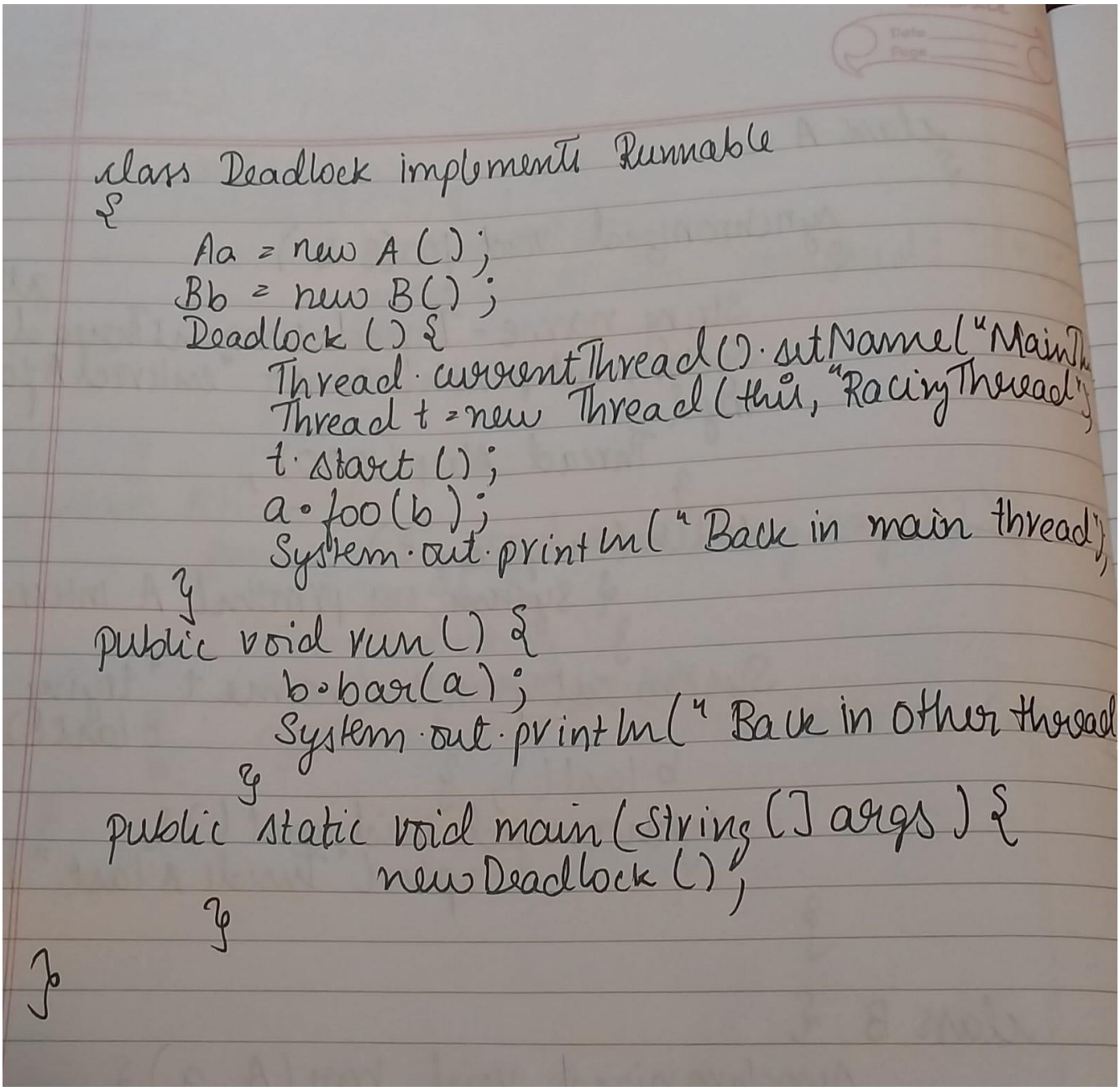

C:\Windows\System32\cmd.exe
Got: 3
Intimate Producer
Put: 4
Intimate Consumer
consumed:3
Producer waiting
Got: 4
Intimate Producer
consumed:4
Put: 5
Intimate Consumer
Search
ENG 05:40 PM


C:\Windows\System32\cmd.exe
Producer waiting
Got: 5
Intimate Producer
Put: 6
Intimate Consumer
consumed:5
Producer waiting
Got: 6
Intimate Producer
Put: 7
Intimate Consumer
Search
ENG 05:40 PM


C:\Windows\System32\cmd.exe
Put: 13
Intimate Consumer
Producer waiting
Got: 13
Intimate Producer
consumed:13
Put: 14
Intimate Consumer
Got: 14
Intimate Producer
consumed:14
Search
ENG 05:40 PM



```
class A
{
    synchronized void foo(B b)
    {
        String name = Thread.currentThread().getName();
        System.out.println(name + "entered A.foo");
        try {
            Thread.sleep(1000);
        }
        catch (Exception e)
        { System.out.println("A interrupted");
        }
        System.out.println(name + "trying to call
                                   B.last()");
        b.last(); }
        synchronized void last() {
            SOP.ln("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 (Enception e)
        { System.out.println("B interrupted");
        }
        S.O.Pln(name + "trying to call A.last()");
        a.last(); }
        synchronized void last() {
            S.O.Pln("Inside A.last"); } }
```




```
class A
{
  synchronized void foo(B b)
   { String name = Thread.currentThread().getName();
     System.out.println(name + " entered A.foo");
     try { Thread.sleep(1000); }
     catch(Exception e) { System.out.println("A Interrupted"); }
     System.out.println(name + " trying to call B.last()"); b.last(); }
     synchronized void last() { System.out.println("Inside A.last"); }
```

```
}

 class B {
   synchronized void bar(A a) {
   String name = Thread.currentThread().getName();
   System.out.println(name + " entered B.bar");
   try { Thread.sleep(1000); }
catch(Exception e) { System.out.println("B Interrupted"); }
System.out.println(name + " trying to call A.last()"); a.last(); }
synchronized void last() { System.out.println("Inside A.last"); }

}


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(); }
}
```