VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

K VEENA(1BM23CS135)

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 **K VEENA(1BM23CS135),** 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.

Lab faculty Incharge Name Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Sl. No.	Date	Experiment Title	Page No.
1	30/10/24	QUADRATIC EQUATION	1-5
2	7/10/24	SGPA CALCULATION	6-12
3	14/10/24	BOOK STORE	13-18
4	21/10/24	SHAPE AREA CALCULATOR	19-22
5	28/10/24	BANK	23-34
6	11/11/24	PACKAGE	35-38
7	28/11/24	EXCEPTIONS	39-41
8	28/11/24	THREADS	42-43
9	28/11/24	OPEN END:SWING DEMO	44-48
10	28/11/24	DEADLOCK	49-52

Github Link:
https://github.com/veenakenche/java-lab-programs.git

Program 1

Implement Quadratic Equation

Algorithm:

PAGE NO: 1
DATE: / /

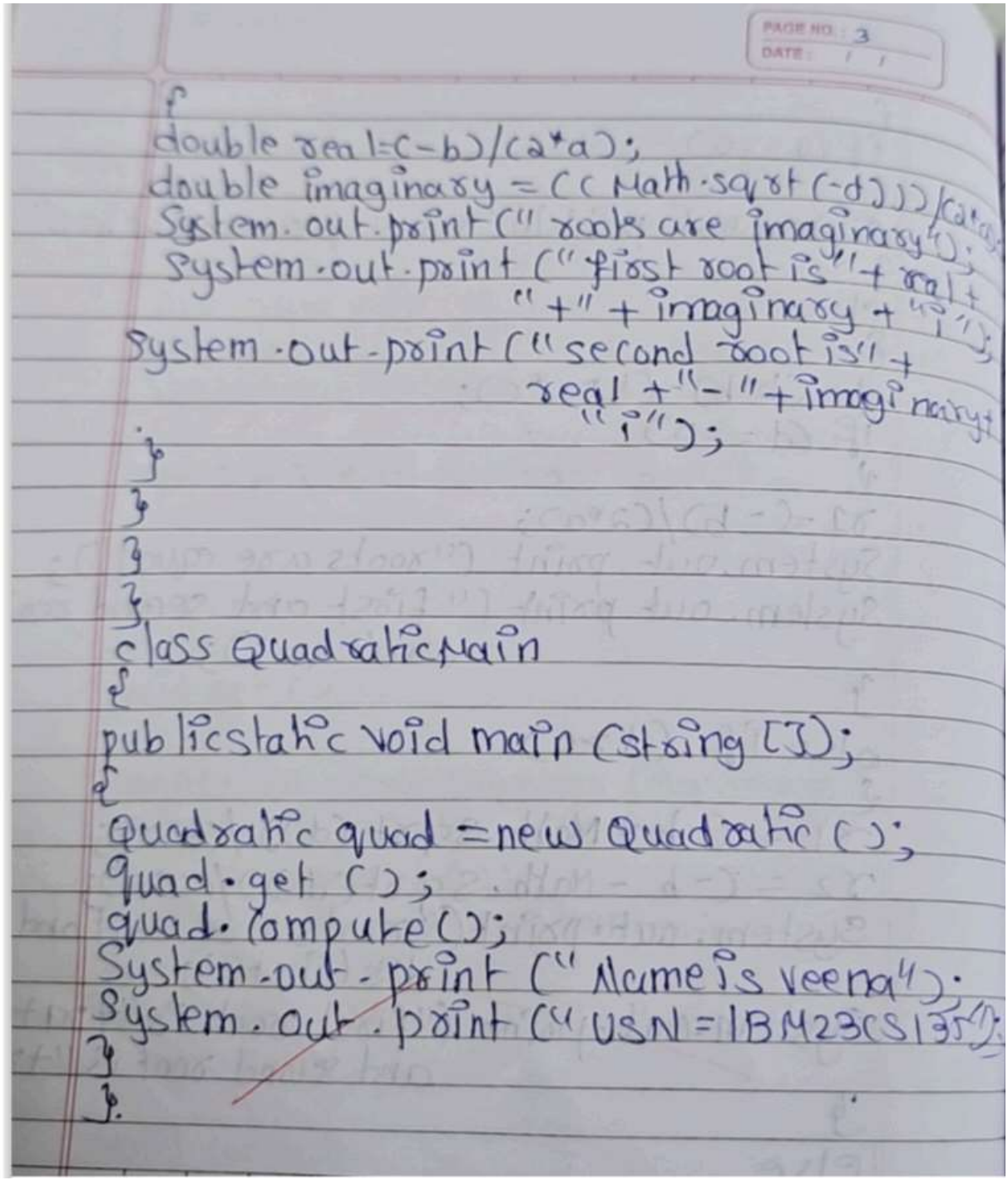
30/9/24

Lab program 1.

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

```
import java.util.Scanner;
class Quadratic
{
int a,b,c;
double r1,r2,d;
void get()
{
Scanner in =new Scanner(System.in);
System.out.println("enter a");
a =in.nextInt();
System.out.println("enter b");
b =in.nextInt();
System.out.println("enter c");
c =in.nextInt();
}
void compute()
```

```
{
if (a==0)
{
System.out.println("value of a cannot be 0");
}
else
{
d=(b*b)-(4*a*c);
if (d==0)
{
r1=(-b)/(2*a);
System.out.print("roots are equal");
System.out.print("first and second roots
                  are"+r1);
}
else if (d>0)
{
r1 = (-b + Math.Sqrt(d))/(2*a);
r2 = (-b - Math.Sqrt(d))/(2*a);
System.out.print("roots are real and
                  distinct");
System.out.print("first root is"+r1+"
                  and second root is"+r2
}
else
```



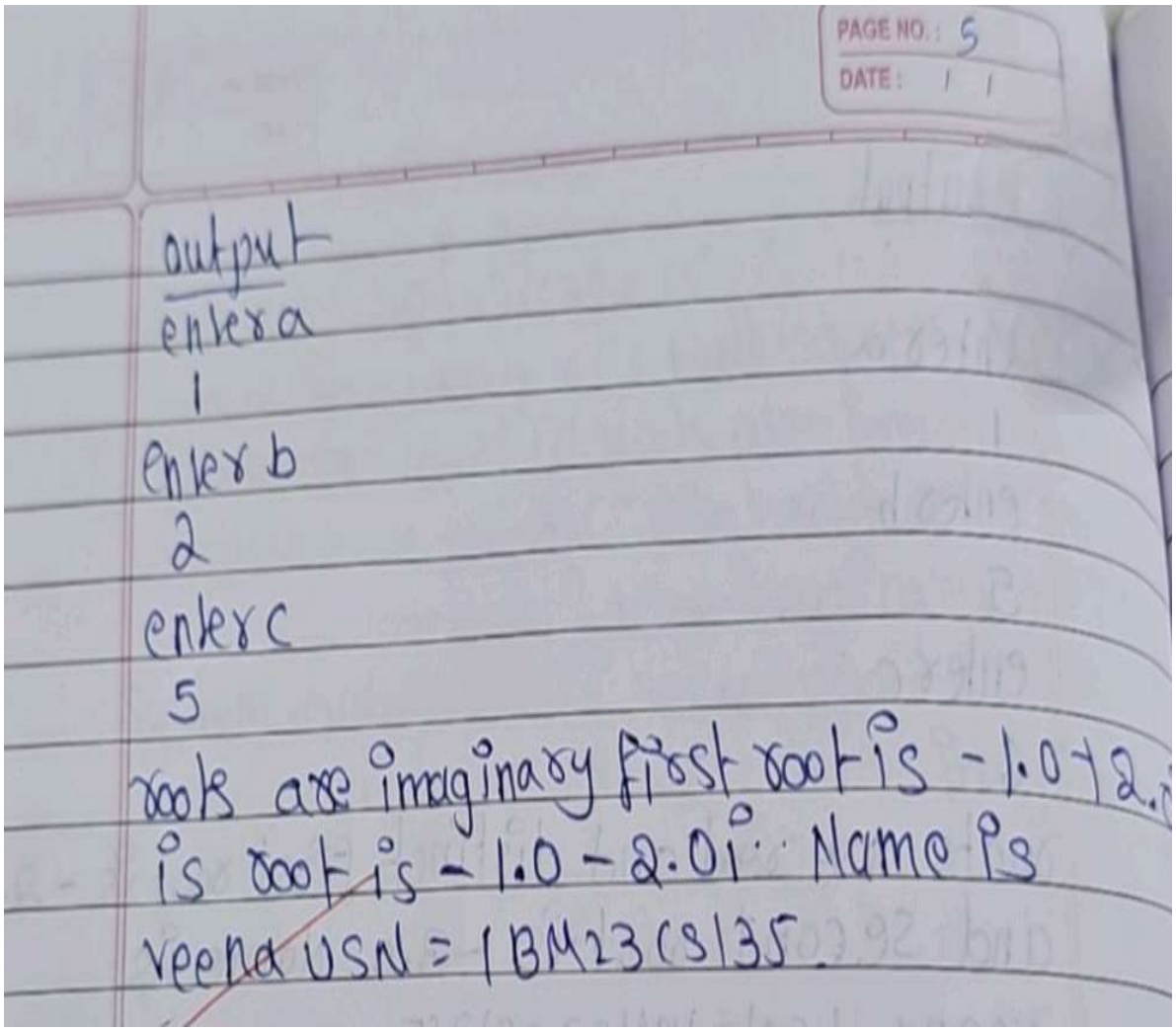
```
{
double real=(-b)/(2*a);
double imaginary = ((Math.sqrt(-d)))/(2*a);
System.out.print("roots are imaginary");
System.out.print("first root is"+real+
                 "+"+imaginary+"i");
System.out.print("second root is"+
                 real+"-"+imaginary+
                 "i");
}
}
}
}
class QuadraticMain
{
publicstatic void main(String[]);
{
Quadratic quad=new Quadratic();
quad.get();
quad.compute();
System.out.print("Name is veena");
System.out.print("USN=1BM23CS135");
}
}
```

output

enter a
1
enter b
5
enter c
6
roots are real and distinct first root is -2.0
and second root is -3.0 Name is
veena USN1=1BM23CS135

output.

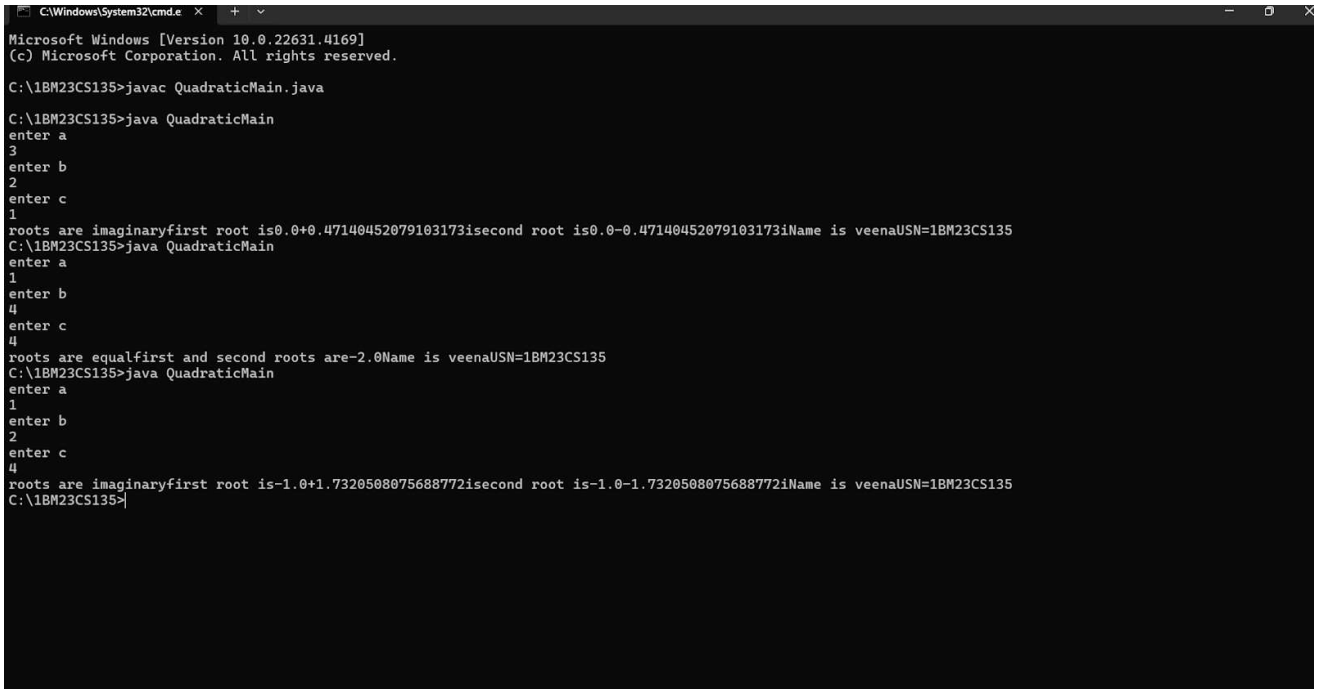
enter a
1
enter b
-2
enter c
1
roots are equal first and second roots are
1.0 Name is veena USN1=1BM23CS135



code:

```
    import java.util.Scanner;
class Quadratic
{
int a,b,c;
double r1,r2,d;
void get()
{
Scanner in=new Scanner(System.in);
System.out.println("enter a");
a=in.nextInt();
System.out.println("enter b");
b=in.nextInt();
System.out.println("enter c");
c=in.nextInt();
}
```

```
void compute()
{
if(a==0)
{
System.out.println("value of a cannot be 0");
}
else
{
d=(b*b)-(4*a*c);
if(d==0)
{
r1=(-b)/(2*a);
System.out.print("roots are equal");
System.out.print("first and second roots are"+r1);
}
else if (d>0)
{
r1=(-b + Math.sqrt(d))/(2*a);
r2=(-b - Math.sqrt(d))/(2*a);
System.out.print("roots are real and distinct");
System.out.print("first root is"+r1+"and second root is"+r2);
}
else
{
double real=(-b)/(2*a);
double imaginary=((Math.sqrt(-d)))/(2*a);
System.out.print("roots are imaginary");
System.out.print("first root is" + real +"+" + imaginary + "i");
System.out.print("second root is" + real +"-" + imaginary + "i");
}
}
}
}
class QuadraticMain
{
public static void main(strings[]);
{
Quadratic quad=new Quadratic();
quad.get();
quad.compute();
System.out.print("Name is veena");
System.out.print("USN=1BM23CS135");
}
}
```


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.
C:\1BM23CS135>javac QuadraticMain.java
C:\1BM23CS135>java QuadraticMain
enter a
3
enter b
2
enter c
1
roots are imaginaryfirst root is0.0+0.47140452079103173isecond root is0.0-0.47140452079103173iName is veenaUSN=1BM23CS135
C:\1BM23CS135>java QuadraticMain
enter a
1
enter b
4
enter c
4
roots are equalfirst and second roots are-2.0Name is veenaUSN=1BM23CS135
C:\1BM23CS135>java QuadraticMain
enter a
1
enter b
2
enter c
4
roots are imaginaryfirst root is-1.0+1.7320508075688772isecond root is-1.0-1.7320508075688772iName is veenaUSN=1BM23CS135
C:\1BM23CS135>

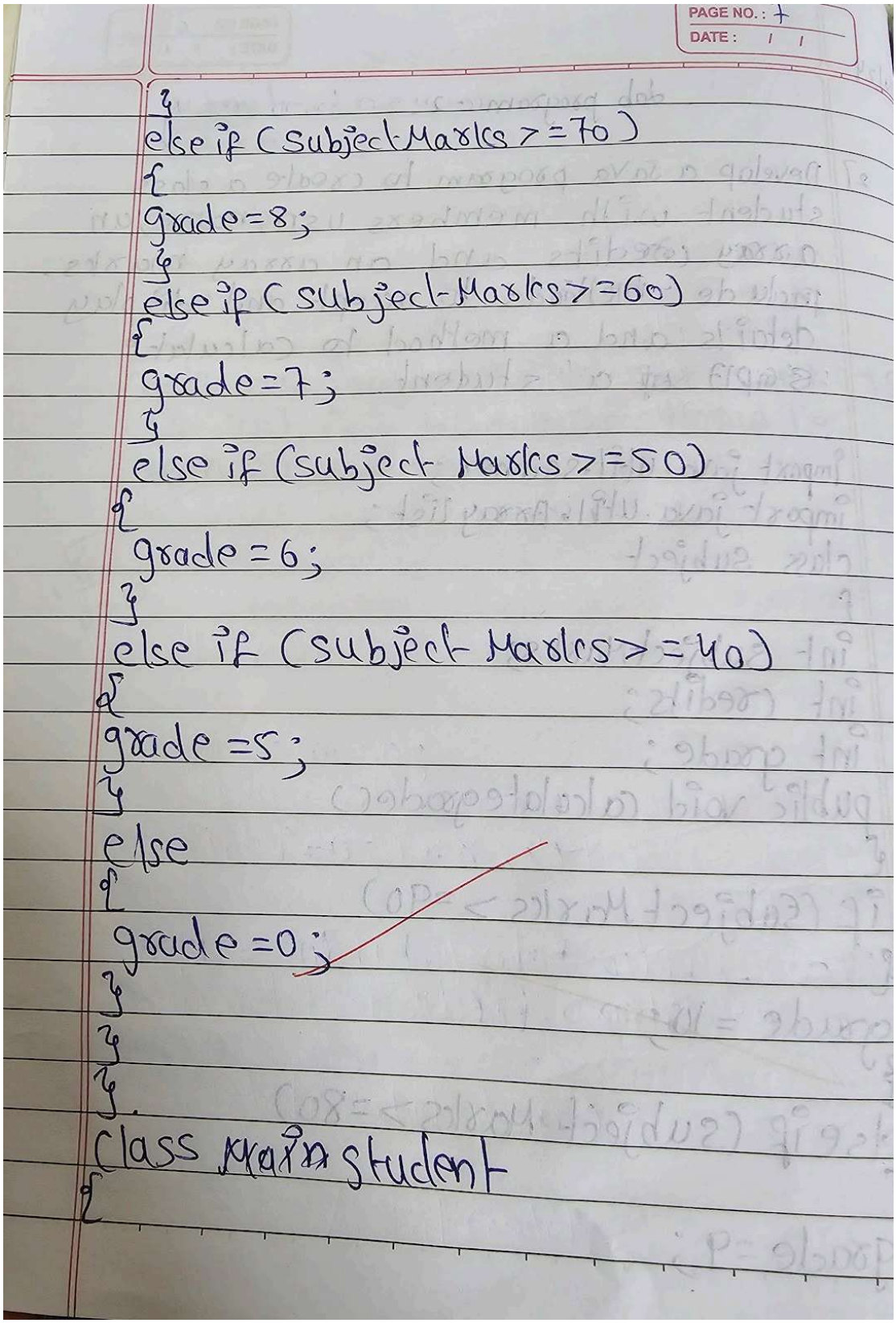
7/10/24

PAGE NO. : 6
DATE : / /

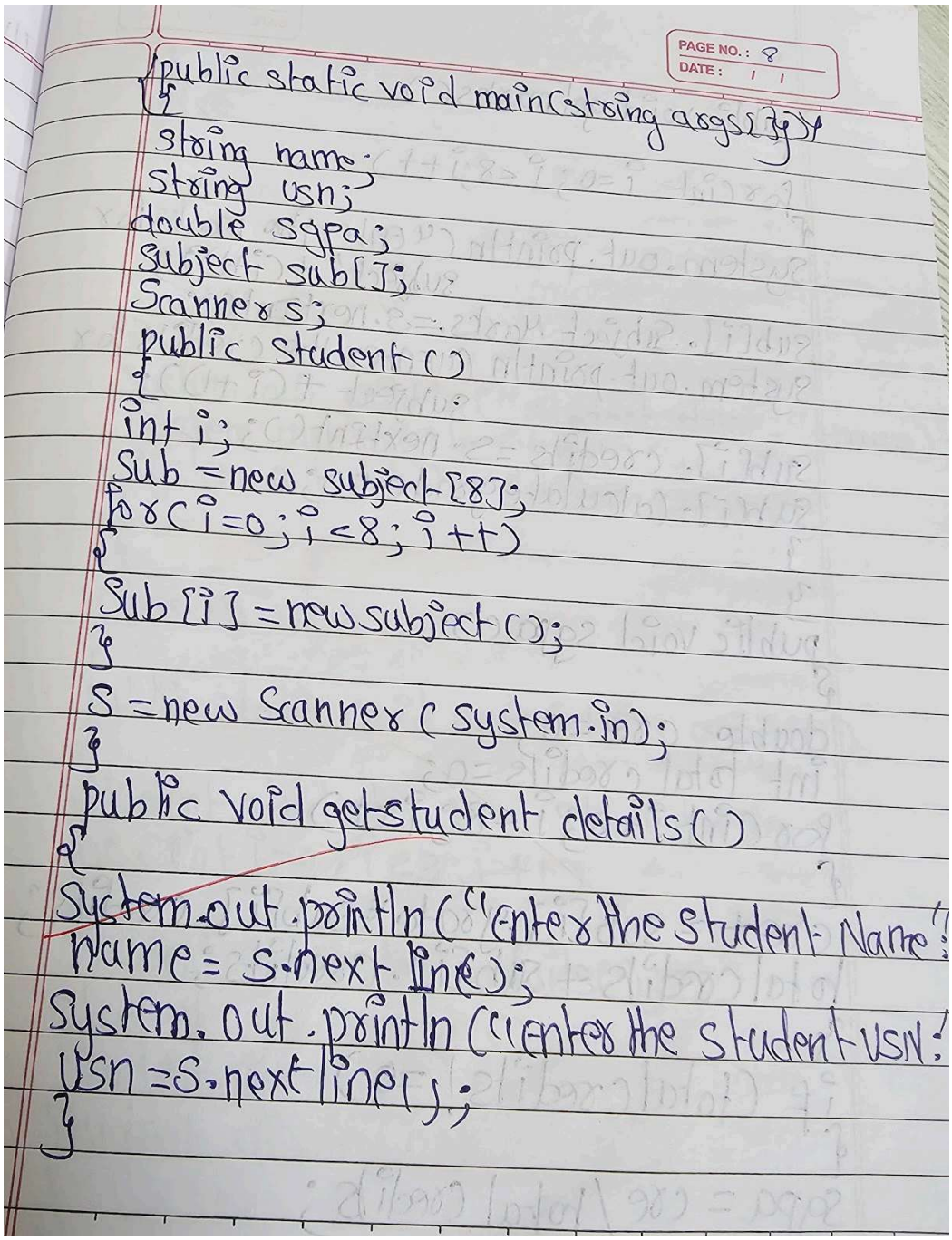
Lab program - 2.

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.

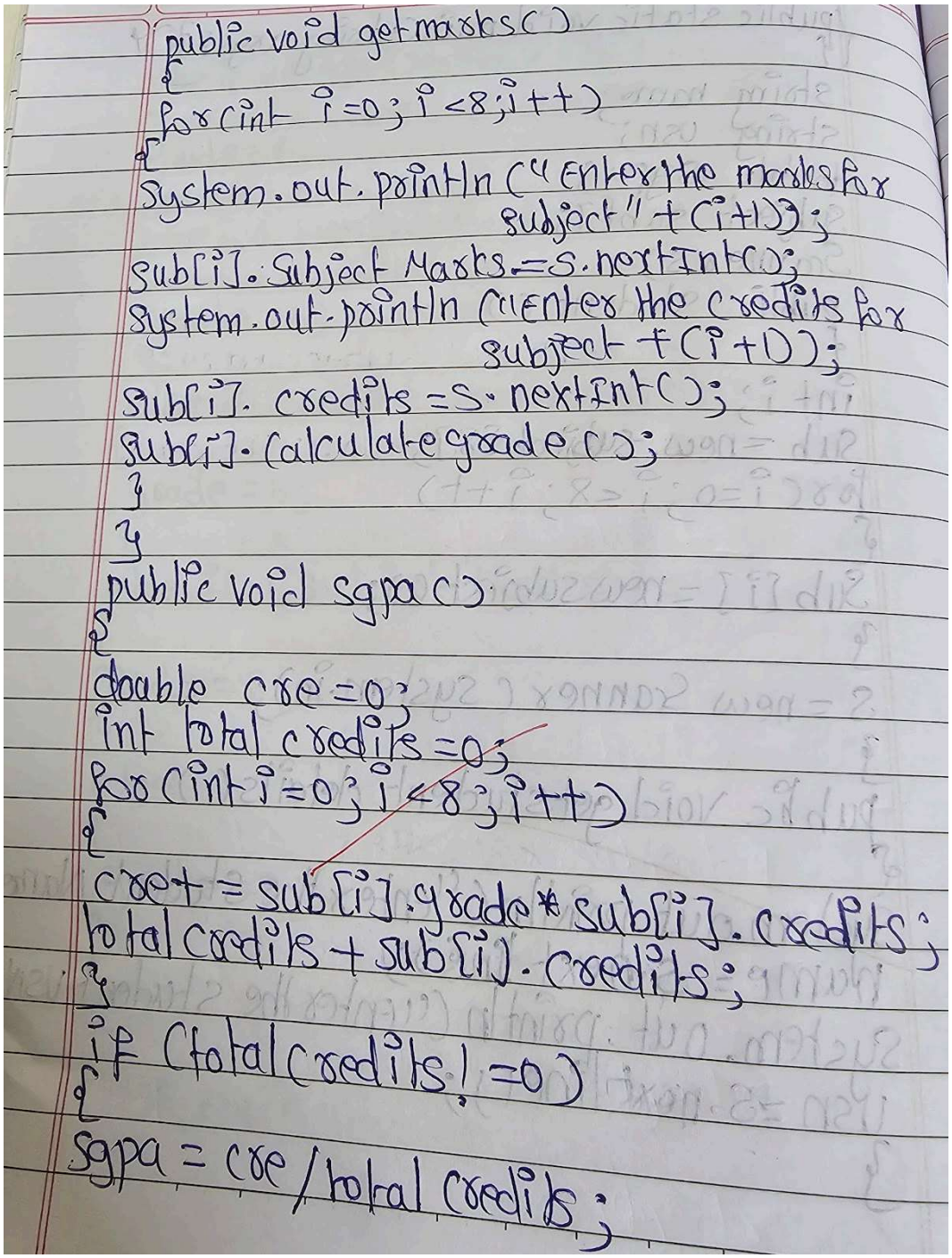
```
import java.util.Scanner;
import java.util.ArrayList;
class Subject
{
int subjectMarks;
int credits;
int grade;
public void calculategrade()
{
if (subjectMarks >=90)
{
grade = 10;
}
else if (subject Marks >=80)
{
grade=9;
```



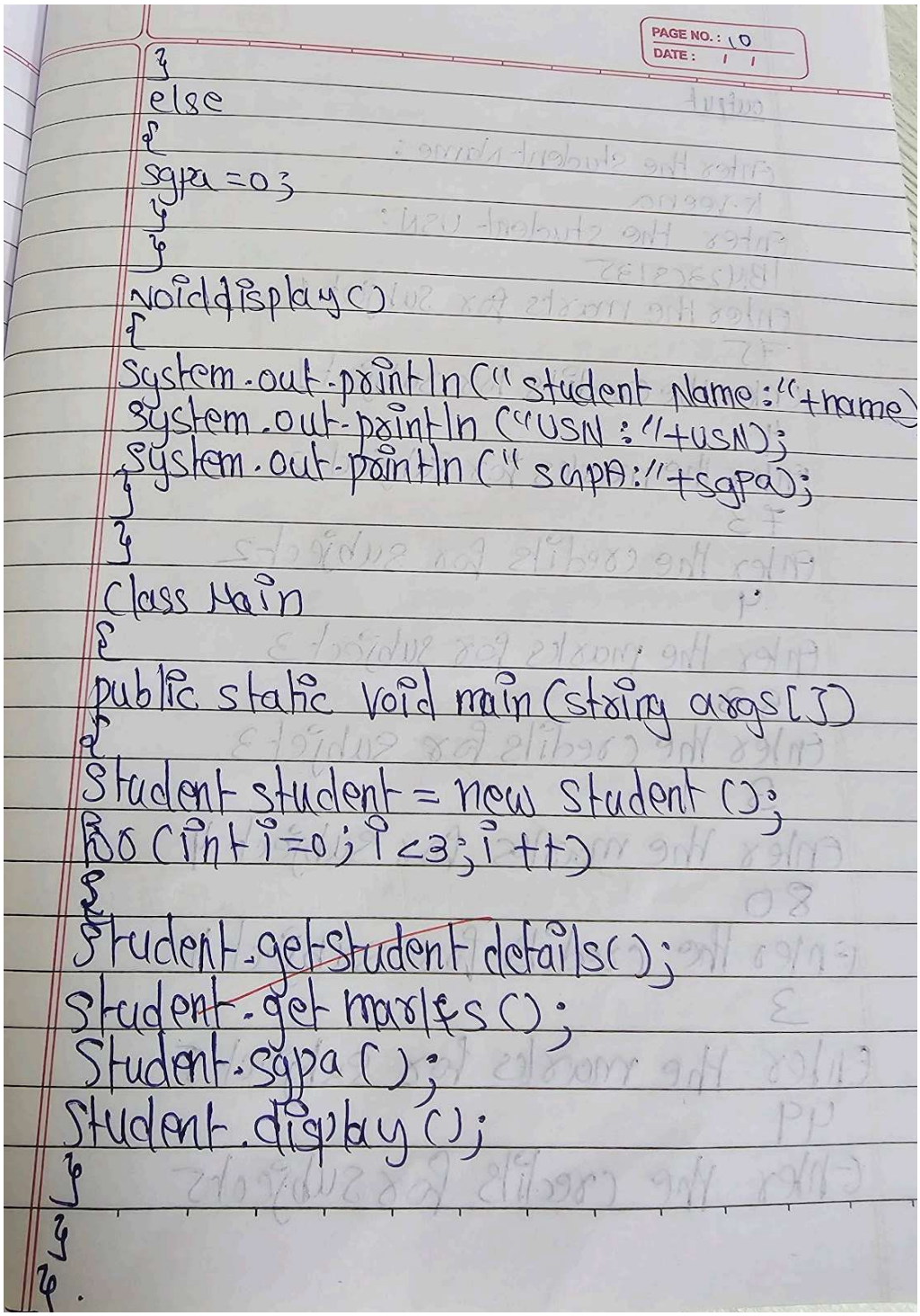
```
PAGE NO. : 7
DATE : / /

}
else if (subject-Marks >=70)
{
grade=8;
}
else if (subject-Marks>=60)
{
grade=7;
}
else if (subject Marks >=50)
{
grade=6;
}
else if (subject-Marks>=40)
{
grade=5;
}
else
{
grade=0;
}
}
}
class main student
{
```



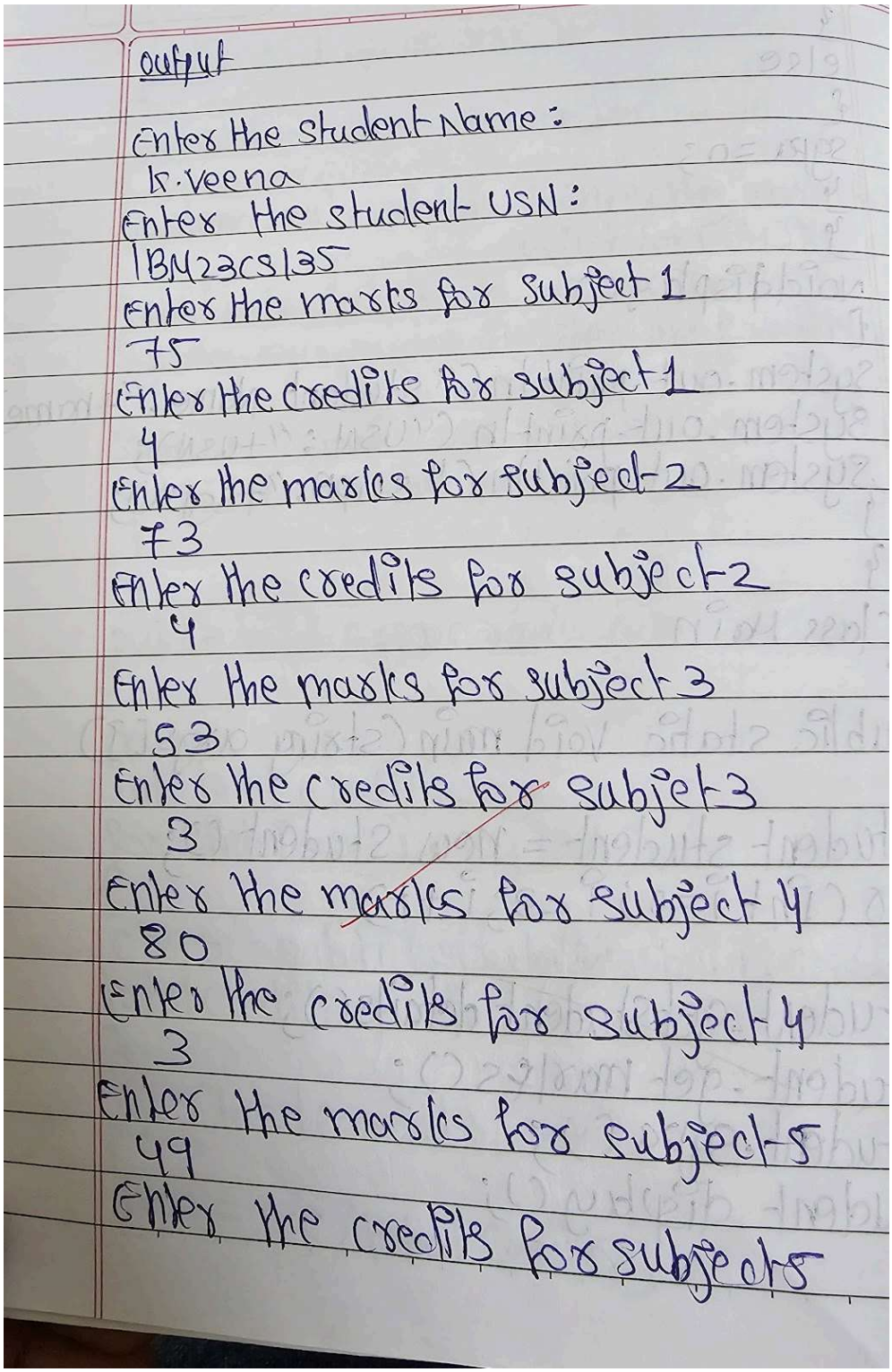
```
PAGE NO.: 8
DATE: / /

public static void main(String args[])
{
String name;
String usn;
double sgpa;
Subject sub[];
Scanner s;
public Student()
{
int i;
Sub = new subject[8];
for(i=0;i<8;i++)
{
Sub[i] = new subject();
}
S = new Scanner(System.in);
}
public void getstudent details()
{
System.out.println("Enter the student Name");
Name = S.next line();
System.out.println("Enter the student USN");
Usn = S.next line();
}
```



```
public void getmarks()
{
for(int i=0;i<8;i++)
{
System.out.println("Enter the marks for
                    subject"+(i+1));
sub[i].SubjectMarks=s.nextInt();
System.out.println("Enter the credits for
                    subject"+(i+1));
sub[i].credits=s.nextInt();
sub[i].calculategrade();
}
}
public void sgpa()
{
double cre=0;
int totalcredits=0;
for(int i=0;i<8;i++)
{
cret=sub[i].grade*sub[i].credits;
totalcredits+sub[i].credits;
}
if(totalcredits!=0)
{
sgpa=cre/totalcredits;
```

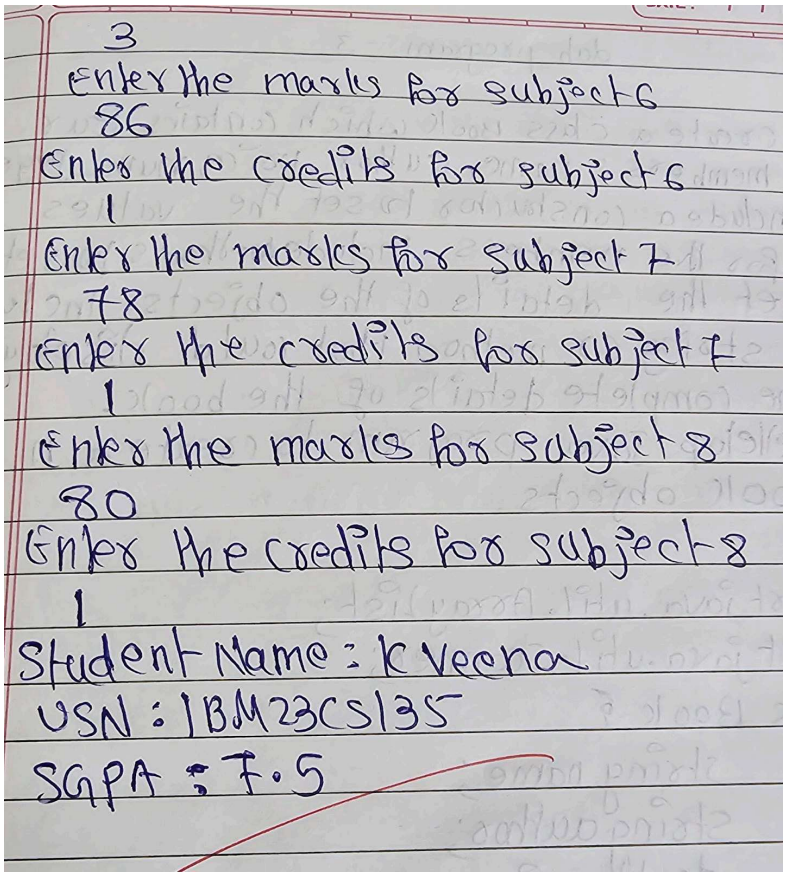


```
}
else
{
sgpa=0;
}
}
void display()
{
System.out.println("student Name:"+name);
System.out.println("USN:"+usn);
System.out.println("sgpa:"+sgpa);
}
}
class Main
{
public static void main(String args[])
{
Student student = new Student();
for(int i=0;i<3;i++)
{
student.getStudentdetails();
student.getmarks();
Student.sgpa();
Student.display();
}
}
}
```



output

Enter the student Name:
K.veena
Enter the student USN:
1BM23CS135
Enter the marks for subject 1
75
Enter the credits for subject 1
4
Enter the marks for subject-2
73
Enter the credits for subject-2
4
Enter the marks for subject 3
53
Enter the credits for subject 3
3
Enter the marks for subject 4
80
Enter the credits for subject 4
3
Enter the marks for subject-5
49
Enter the credits for subjects



code:

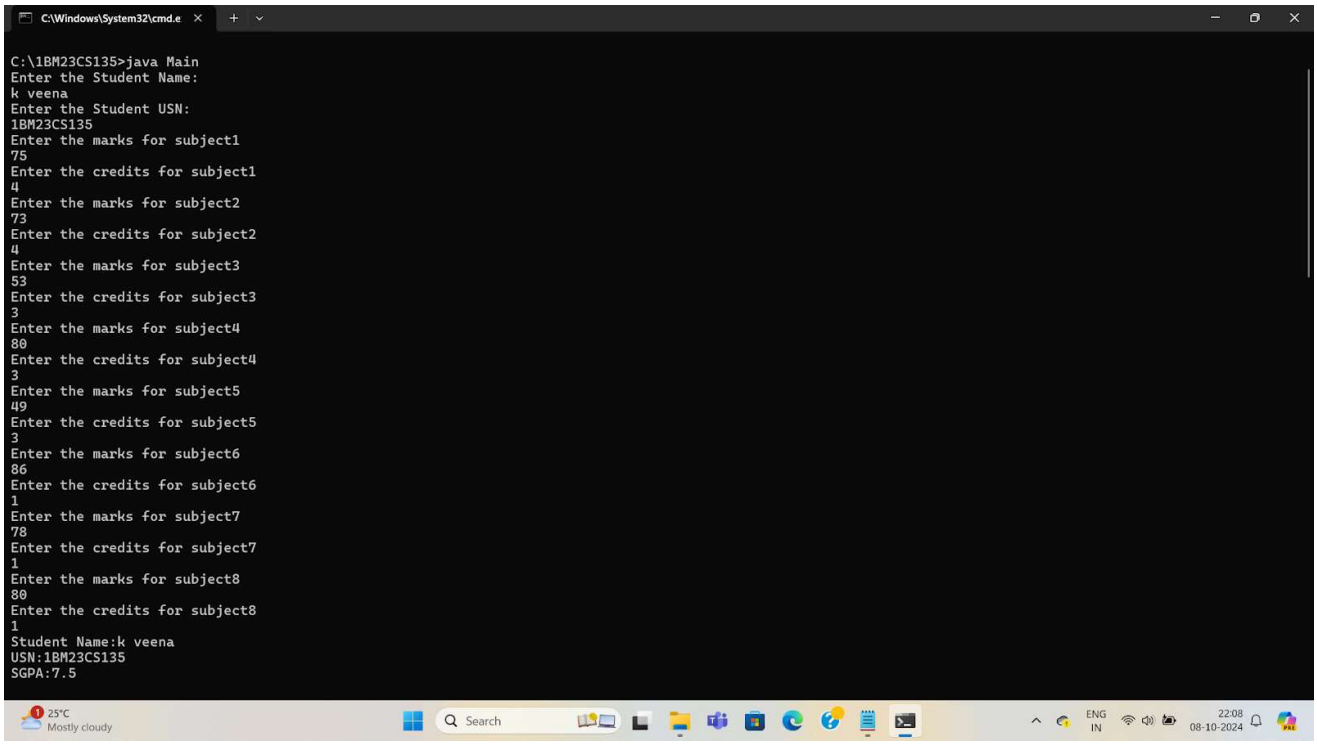
```
    import java.util.Scanner;
import java.util.ArrayList;
class Subject{
int SubjectMarks;
int credits;
int grade;
public void calculategrade(){
if(SubjectMarks>=90){
grade=10;
}
```

```
else if(SubjectMarks>=80){
grade=9;
}
else if(SubjectMarks>=70){
grade=8;
}
else if(SubjectMarks>=60){
grade=7;
}
else if(SubjectMarks>=50){
grade=6;
}
else if(SubjectMarks>=40){
grade=5;
}
else{
grade=0;
}}}
class Student{
String name;
String usn;
double sgpa;
Subject sub[];
Scanner s;
```

```
public Student()
{
int i;
sub=new Subject[8];
for(i=0;i<8;i++)
{
sub[i]=new Subject();
}
s=new Scanner(System.in);
}
public void getStudentdetails()
{
System.out.println("Enter the Student Name:");
name=s.nextLine();
System.out.println("Enter the Student USN:");
usn=s.nextLine();
}
public void getmarks(){
for(int i=0;i<8;i++)
{
System.out.println("Enter the marks for subject"+(i+1));
sub[i].SubjectMarks=s.nextInt();
System.out.println("Enter the credits for subject"+(i+1));
```

```
sub[i].credits=s.nextInt();
sub[i].calculategrade();
}
}
public void sgpa()
{
double cre=0;
int totalcredits=0;
for(int i=0;i<8;i++)
{
cre+=sub[i].grade*sub[i].credits;
totalcredits+=sub[i].credits;
}
if(totalcredits!=0)
{
sgpa=cre/totalcredits;
}
else
{
sgpa=0;
}
}
void display()
{
```

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

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

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

}

}

class Main

{

public static void main(String args[])

{

Student student=new Student();

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

student.getStudentdetails();

student.getmarks();

student.sgpa();

student.display();
}}}
```


C:\Windows\System32\cmd.e
C:\1BM23CS135>java Main
Enter the Student Name:
k veena
Enter the Student USN:
1BM23CS135
Enter the marks for subject1
75
Enter the credits for subject1
4
Enter the marks for subject2
73
Enter the credits for subject2
4
Enter the marks for subject3
53
Enter the credits for subject3
3
Enter the marks for subject4
80
Enter the credits for subject4
3
Enter the marks for subject5
49
Enter the credits for subject5
3
Enter the marks for subject6
86
Enter the credits for subject6
1
Enter the marks for subject7
78
Enter the credits for subject7
1
Enter the marks for subject8
80
Enter the credits for subject8
1
Student Name:k veena
USN:1BM23CS135
SGPA:7.5
25°C
Mostly cloudy
Search
ENG
IN
22:08
08-10-2024

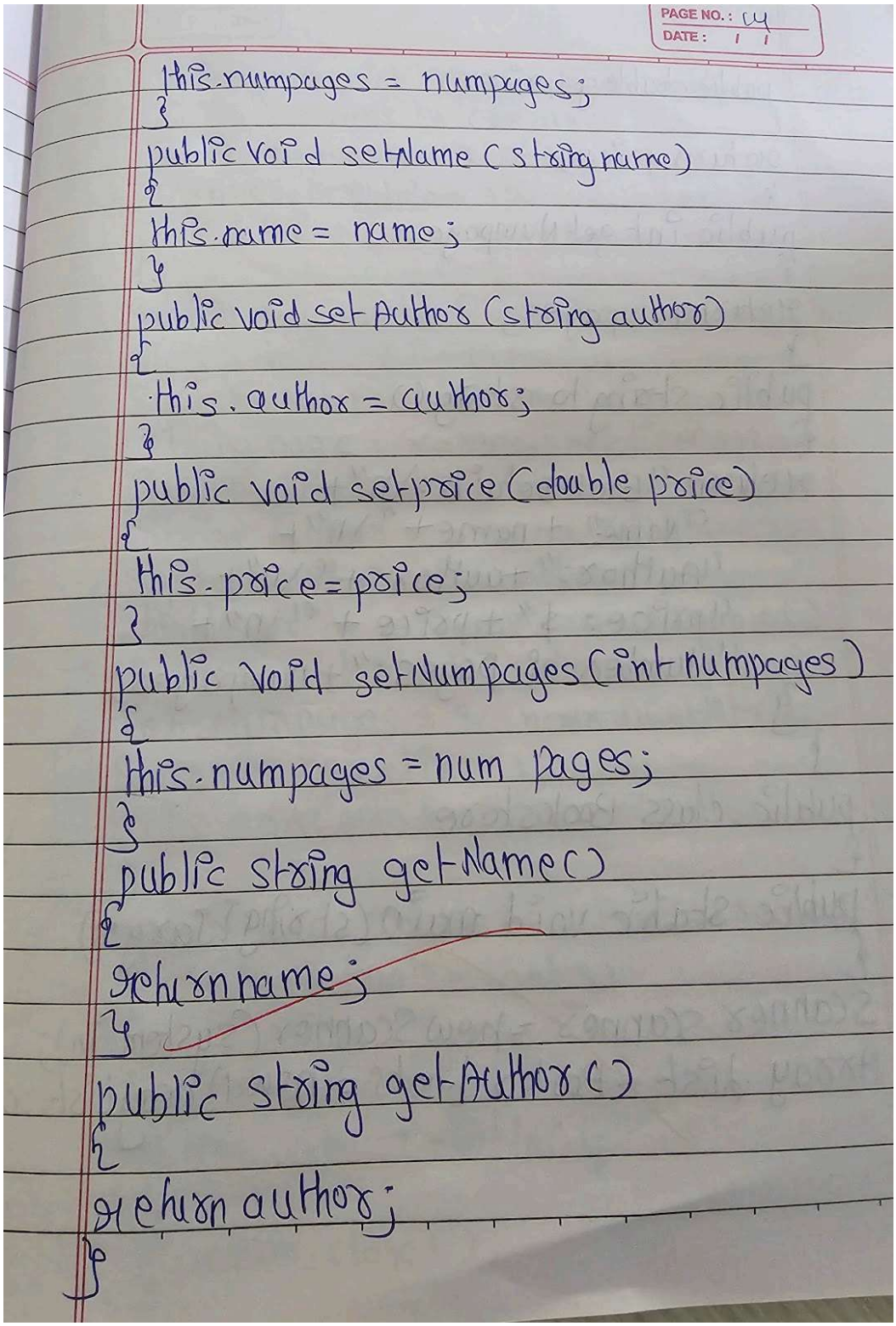
4/10/24

PAGE NO.: 13
DATE: / /

dab program - 3.

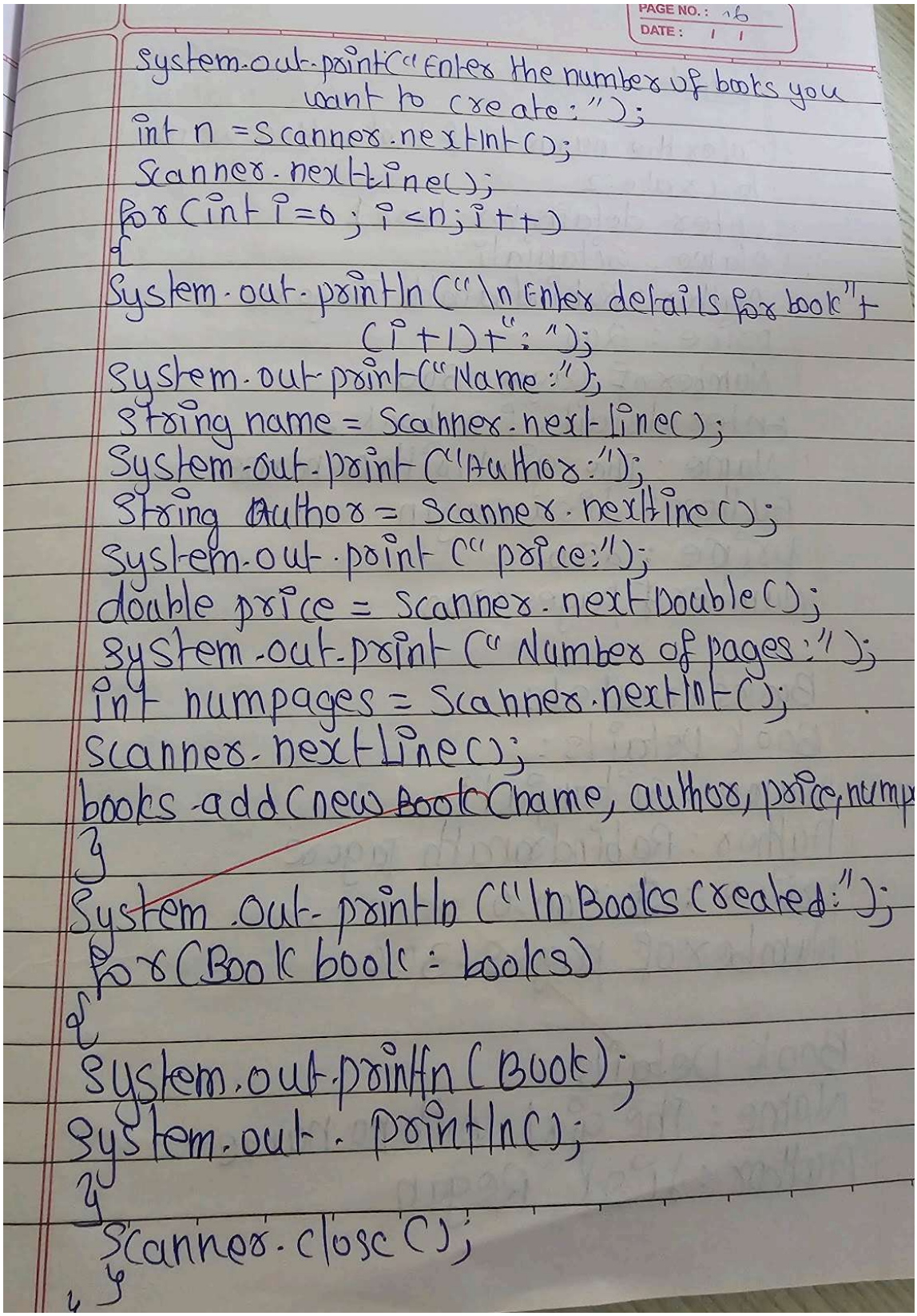
3] Create a class Book which contains four members: name, author, price, num_pages, Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include to string () method that could display the complete details of the book. Develop Java program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

```
import java.util.ArrayList;
import java.util.Scanner;
class Book {
    string name;
    string author;
    double price;
    int numpages;
public Book(string name, string author, doubl
        price, int numpages)
    {
this.name = name;
this.author = author;
this.price = price;
```

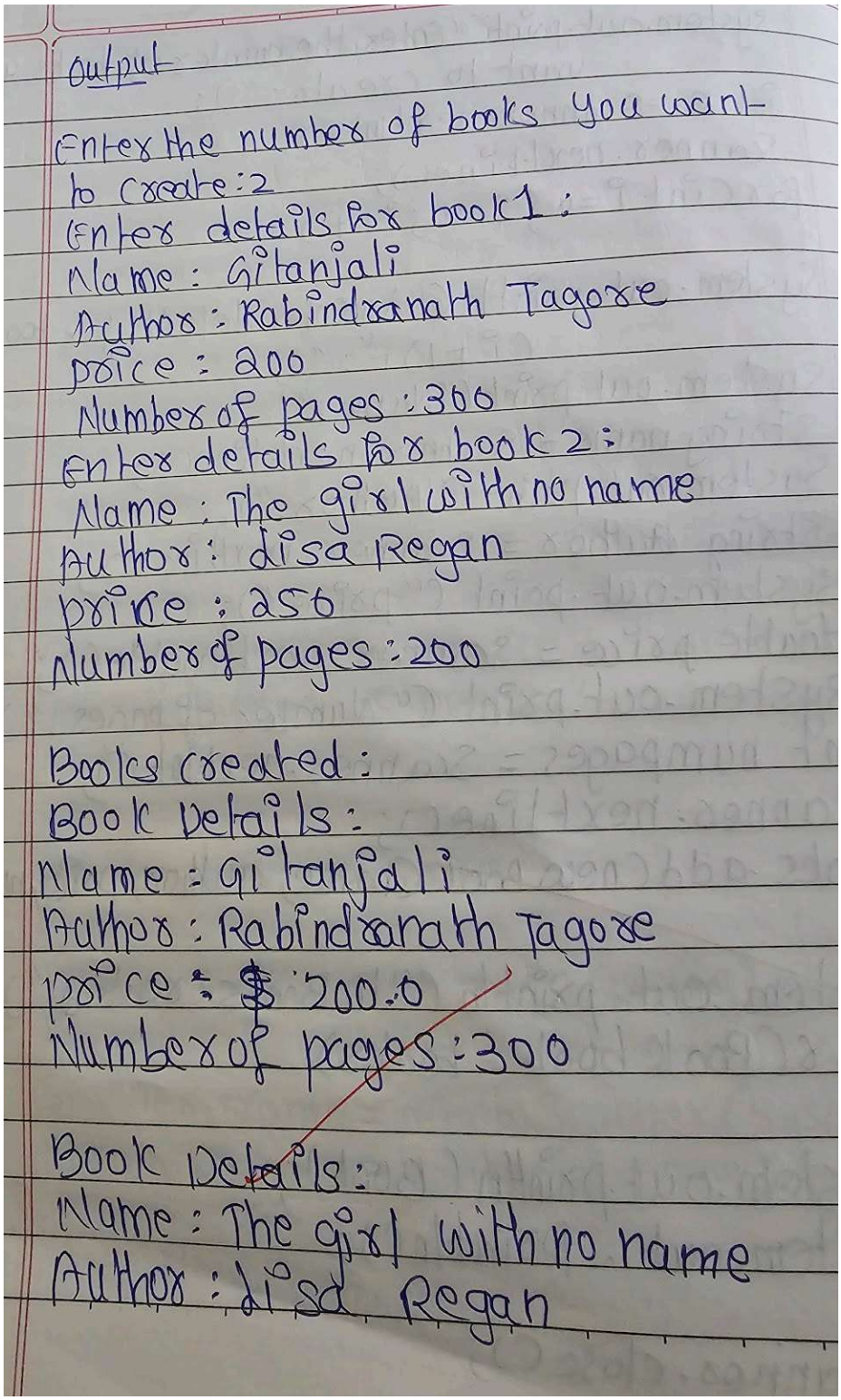


```
PAGE NO.: 14
DATE: / /

    this.numpages = numpages;
}
public void setName (string name)
{
    this.name = name;
}
public void setAuthor (string author)
{
    this.author = author;
}
public void setprice (double price)
{
    this.price=price;
}
public void setNumpages (int numpages)
{
    this.numpages = num pages;
}
public string getName()
{
    return name;
}
public string getAuthor()
{
    return author;
}
```

```
public double getPrice()
{
return price;
}
public int getNumpages()
{
return numpages;
}
public String toString()
{
return "Book details:\n" +
    "Name:" + name + "\n" +
    "Author:" + author + "\n" +
    "price: $" + price + "\n" +
    "Number of pages:" + numpages;
}
}
public class Bookstore
{
public static void main(String[]args)
{
Scanner scanner = new Scanner(System.in);
ArrayList<Book> books = new ArrayList<>
```



```
System.out.print("Enter the number of books you
        want to create: ");
int n = Scanner.nextInt();
Scanner.nextLine();
for(int i=0; i<n; i++)
{
System.out.println("\n Enter details for book"+
        (i+1)+": ");
System.out.print("Name: ");
String name = Scanner.nextLine();
System.out.print("Author: ");
String author = Scanner.nextLine();
System.out.print("price: ");
double price = Scanner.nextDouble();
System.out.print("Number of pages: ");
int numpages = Scanner.nextInt();
Scanner.nextLine();
books.add(new Book(name, author, price, nump
}
System.out.println("\n Books created: ");
for(Book book: books)
{
System.out.println(Book);
System.out.println();
}
Scanner.close();
}
```

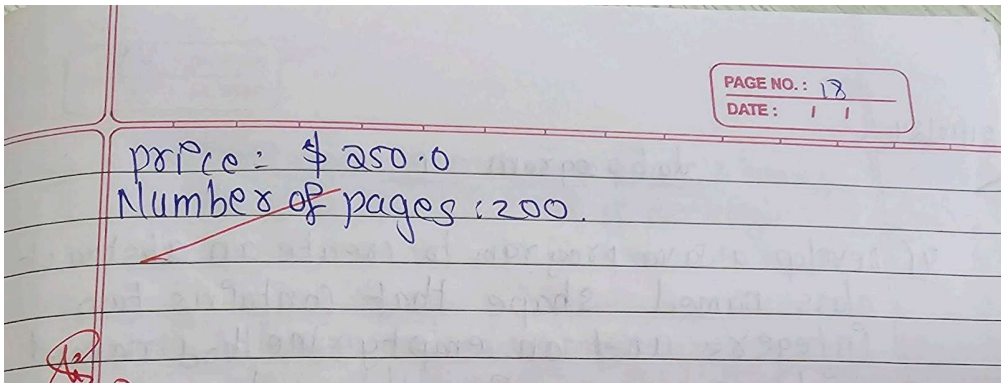


Output

Enter the number of books you want to create: 2
Enter details for book 1:
Name: Gitanjali
Author: Rabindranath Tagore
price: 200
Number of pages: 300
Enter details for book 2:
Name: The girl with no name
Author: disa Regan
price: 250
Number of pages: 200

Books created:
Book Details:
Name: Gitanjali
Author: Rabindranath Tagore
price: $ 200.0
Number of pages: 300

Book Details:
Name: The girl with no name
Author: disa Regan

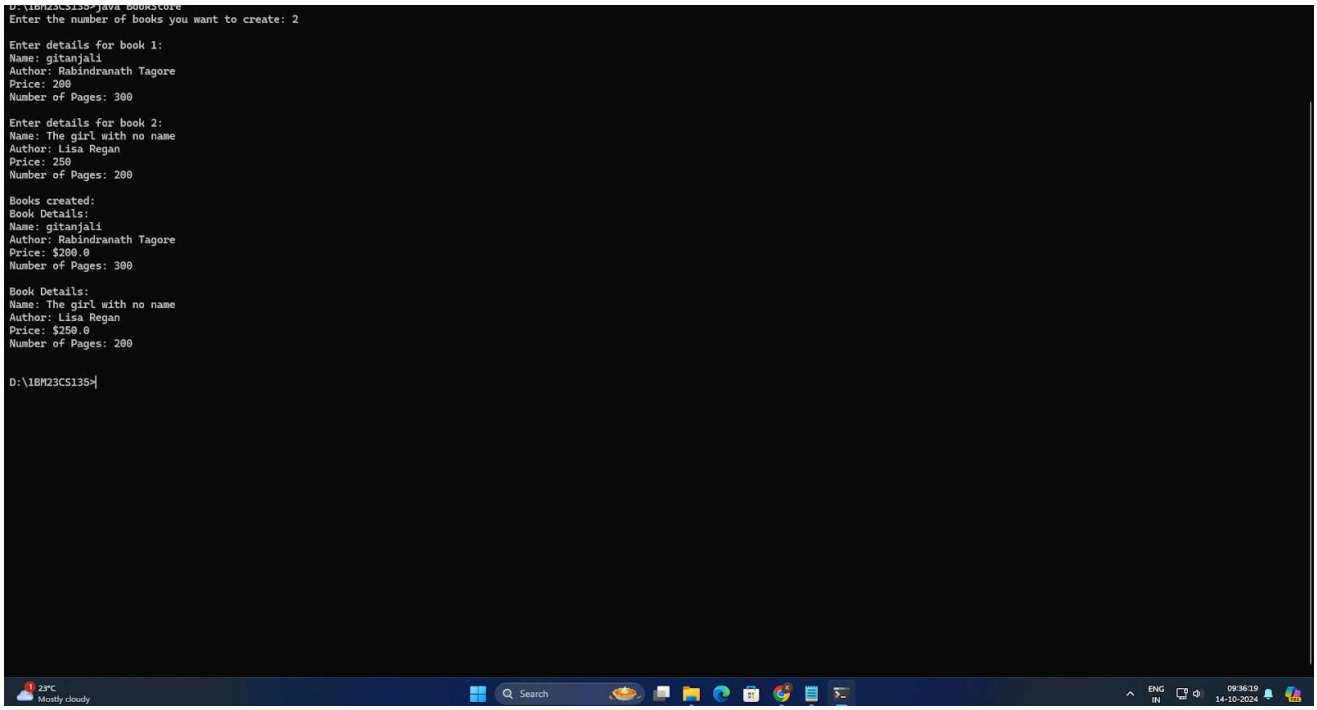


code:

```
   import java.util.ArrayList;
import java.util.Scanner;

class Book {
   private String name;
   private String author;
   private double price;
   private int numPages;
   public Book(String name, String author, double price, int numPages) {
      this.name = name;
      this.author = author;
      this.price = price;
      this.numPages = numPages;
   }
   public void setName(String name) {
      this.name = name;
   }

   public void setAuthor(String author) {
      this.author = author;
   }

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

   public void setNumPages(int numPages) {
      this.numPages = numPages;
   }
   public String getName() {
      return name;
   }
```

```
    public String getAuthor() {
        return author;
    }

    public double getPrice() {
        return price;
    }

    public int getNumPages() {
        return numPages;
    }
    public String toString() {
        return "Book Details:\n" +
                "Name: " + name + "\n" +
                "Author: " + author + "\n" +
                "Price: $" + price + "\n" +
                "Number of Pages: " + numPages;
    }
}

public class BookStore {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ArrayList<Book> books = new ArrayList<>();

        System.out.print("Enter the number of books you want to create: ");
        int n = scanner.nextInt();
        scanner.nextLine();

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for book " + (i + 1) + ":");

            System.out.print("Name: ");
            String name = scanner.nextLine();

            System.out.print("Author: ");
            String author = scanner.nextLine();

            System.out.print("Price: ");
            double price = scanner.nextDouble();

            System.out.print("Number of Pages: ");
            int numPages = scanner.nextInt();
            scanner.nextLine();

            books.add(new Book(name, author, price, numPages));
        }
```

```
        System.out.println("\nBooks created:");
        for (Book book : books) {
            System.out.println(book);
            System.out.println();
        }

        scanner.close();
    }
}
```



21/10/24

PAGE NO.: 19
DATE: / /

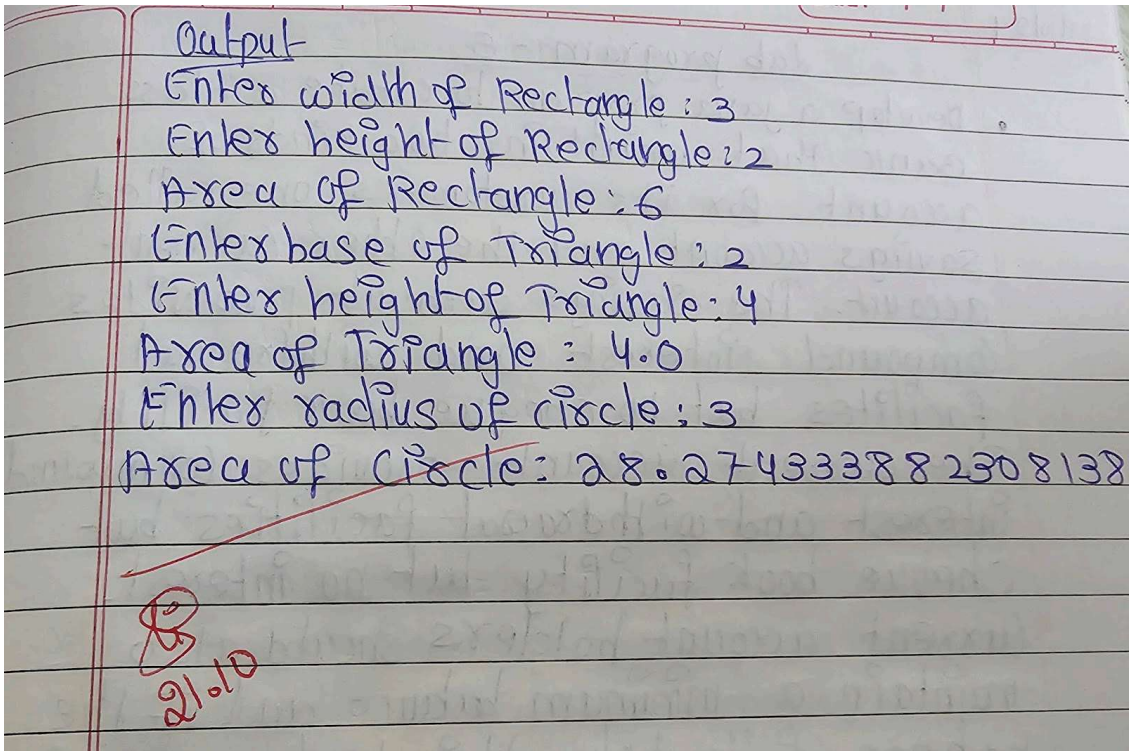
lab program - 4.

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.

```
import java.util.Scanner;
abstract class shape{
    int dimension1;
    int dimension2;
    abstract void printArea();
}
class Rectangle extends shape{
Rectangle(int width, int height){
    dimension1 = width;
    dimension2 = height;
}
void printArea(){
```

```
        int area = dimension1 * dimension2;
        System.out.println("Area of Rectangle:" + area);
    }
}
class Triangle extends Shape {
    Triangle(int base, int height) {
        dimension1 = base;
        dimension2 = height;
    }
    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Area of Triangle:" + area);
    }
}
class Circle extends Shape {
    Circle(int radius) {
        dimension1 = radius;
    }
    void printArea() {
        double area = Math.PI * dimension1 * dimensi
        System.out.println("Area of circle" + area)
    }
}
public class ShapeAreaCalculator {
    public static void main(String[] args
```

```
Scanner scanner = new Scanner(System.in);
System.out.print("Enter width of Rectangle:");
int rectangle width = scanner.nextInt();
System.out.print("Enter the height of Rectangle:");
int rectangle Height = scanner.nextInt();
Shape rectangle = new Rectangle(rectangle
                                width, rectangleHeight);
rectangle.printArea();
System.out.print("Enter base of Triangle:");
int triangleBase = scanner.nextInt();
System.out.print("Enter height of Triang
int triangleheight = scanner.nextInt();
shape triangle = new Triangle(triangleB
                              triangle Hei
triangle.printArea();
System.out.print("Enter radius of circle
int circleRadius = scanner.nextInt();
Shapecircle = new circle(circleRadi
Circle.printArea();
scanner.close();
}
}
```



code:

```
      import java.util.Scanner;
abstract class Shape {
   int dimension1;
   int dimension2;
   abstract void printArea();
}
class Rectangle extends Shape {
   Rectangle(int width,int height) {
      dimension1 = width;
      dimension2 = height;
   }
   void printArea() {
      int area = dimension1 * dimension2;
      System.out.println("Area of Rectangle: " + area);
   }
}
class Triangle extends Shape {
   Triangle(int base,int height) {
      dimension1 = base;
      dimension2 = height;
   }
   void printArea() {
      double area = 0.5 * dimension1 * dimension2;
```

```
        System.out.println("Area of Triangle: " + area);
    }
}
class Circle extends Shape {
    Circle(int radius) {
        dimension1 = radius;
    }
    void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Area of Circle: " + area);
    }
}
public class ShapeAreaCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter width of Rectangle: ");
        int rectangleWidth = scanner.nextInt();
        System.out.print("Enter height of Rectangle: ");
        int rectangleHeight = scanner.nextInt();
        Shape rectangle = new Rectangle(rectangleWidth, rectangleHeight);
        rectangle.printArea();
        System.out.print("Enter base of Triangle: ");
        int triangleBase = scanner.nextInt();
        System.out.print("Enter height of Triangle: ");
        int triangleHeight = scanner.nextInt();
        Shape triangle = new Triangle(triangleBase, triangleHeight);
        triangle.printArea();
        System.out.print("Enter radius of Circle: ");
        int circleRadius = scanner.nextInt();
        Shape circle = new Circle(circleRadius);
        circle.printArea();

        scanner.close();
    }
}
```


Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.
D:\1BM23CS135>javac ShapeAreaCalculator.java
D:\1BM23CS135>java ShapeAreaCalculator
Enter width of Rectangle: 3
Enter height of Rectangle: 2
Area of Rectangle: 6
Enter base of Triangle: 2
Enter height of Triangle: 4
Area of Triangle: 4.0
Enter radius of Circle: 3
Area of Circle: 28.274333882308138
D:\1BM23CS135>

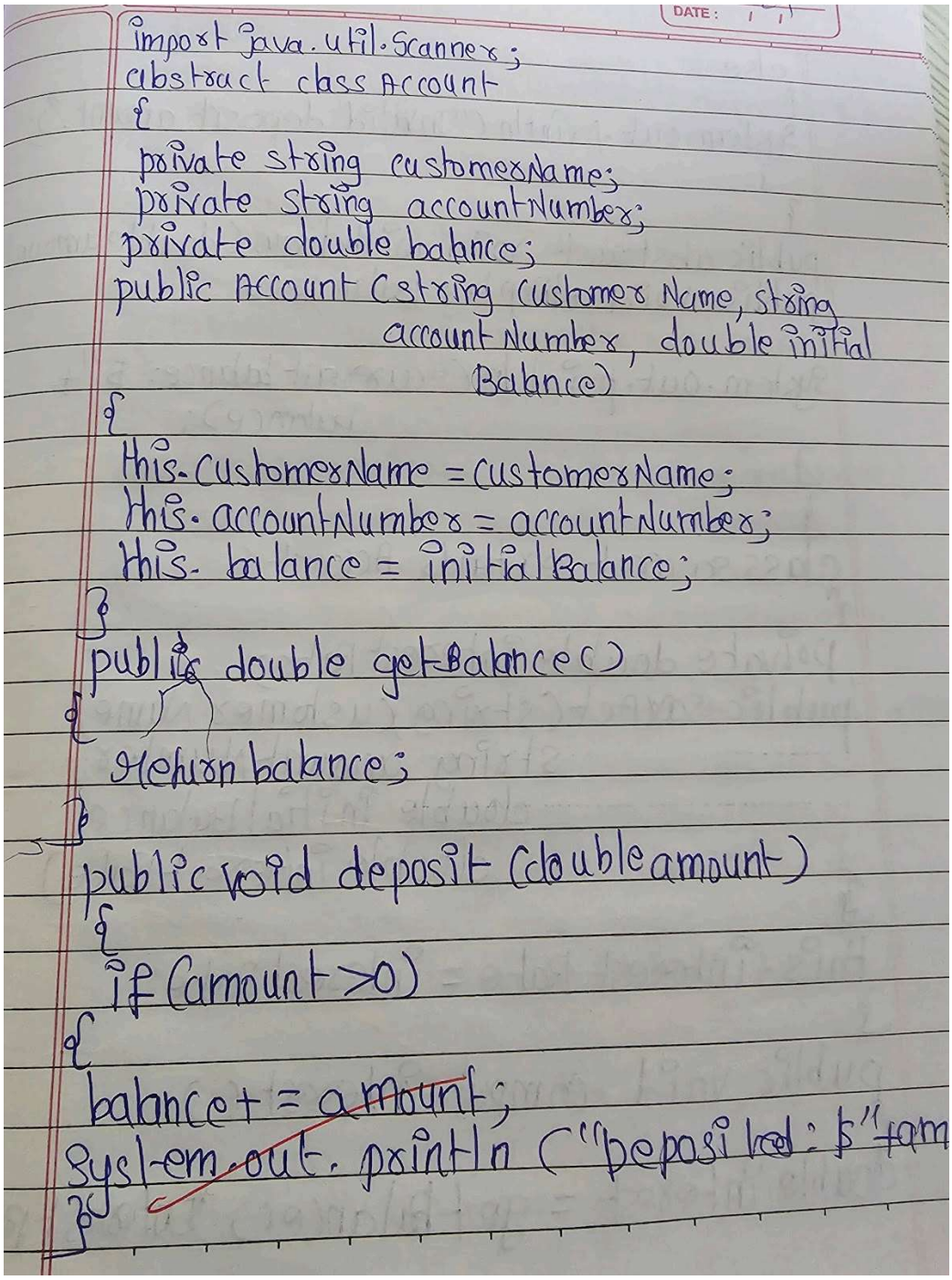
PAGE NO.: 13
DATE: / /

28/10/24

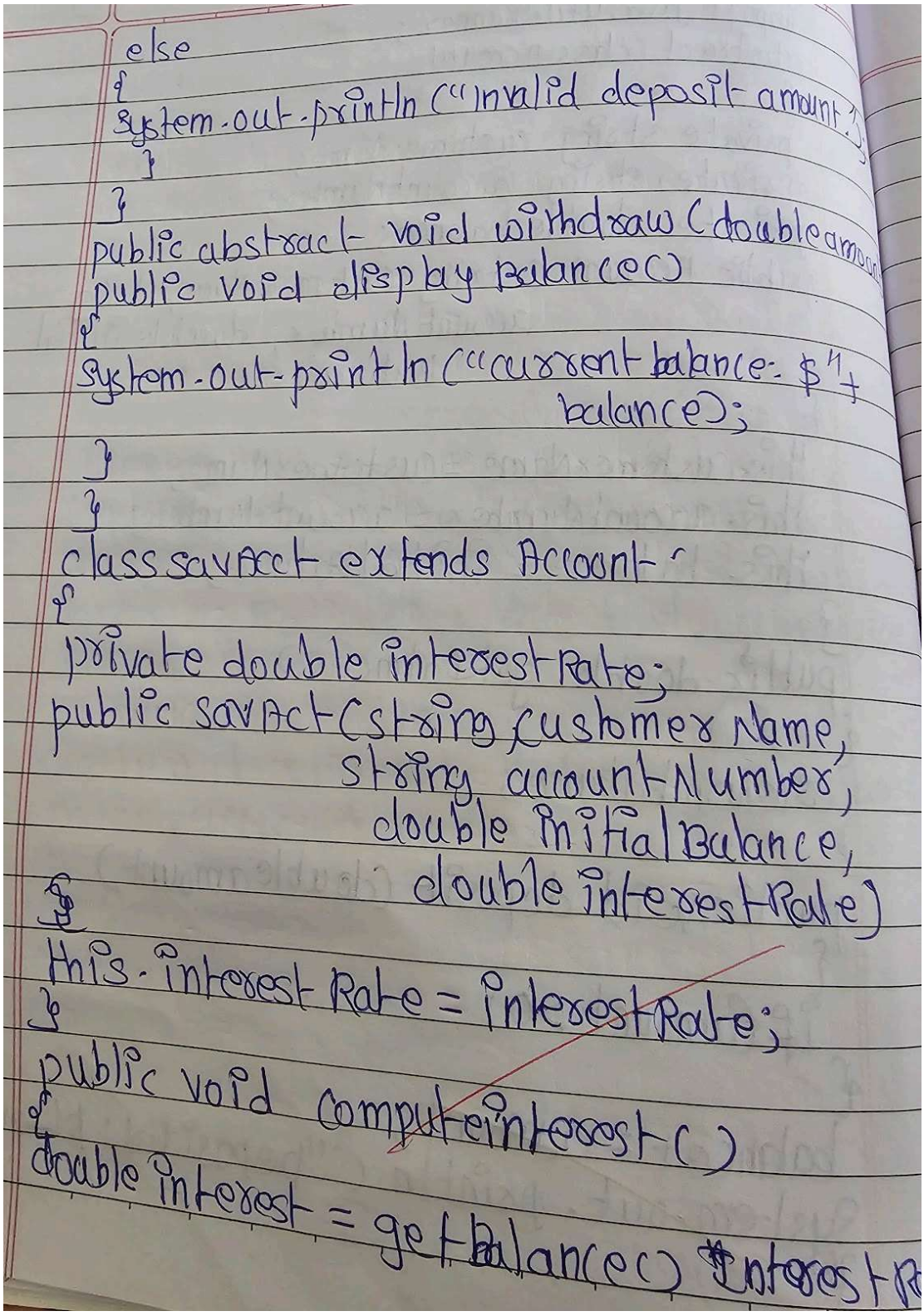
Lab program - 5

- 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 facilites but no cheque book facility. The current account provides ~~compound interest and withdrawal facilities but~~ cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a servi charge is imposed.

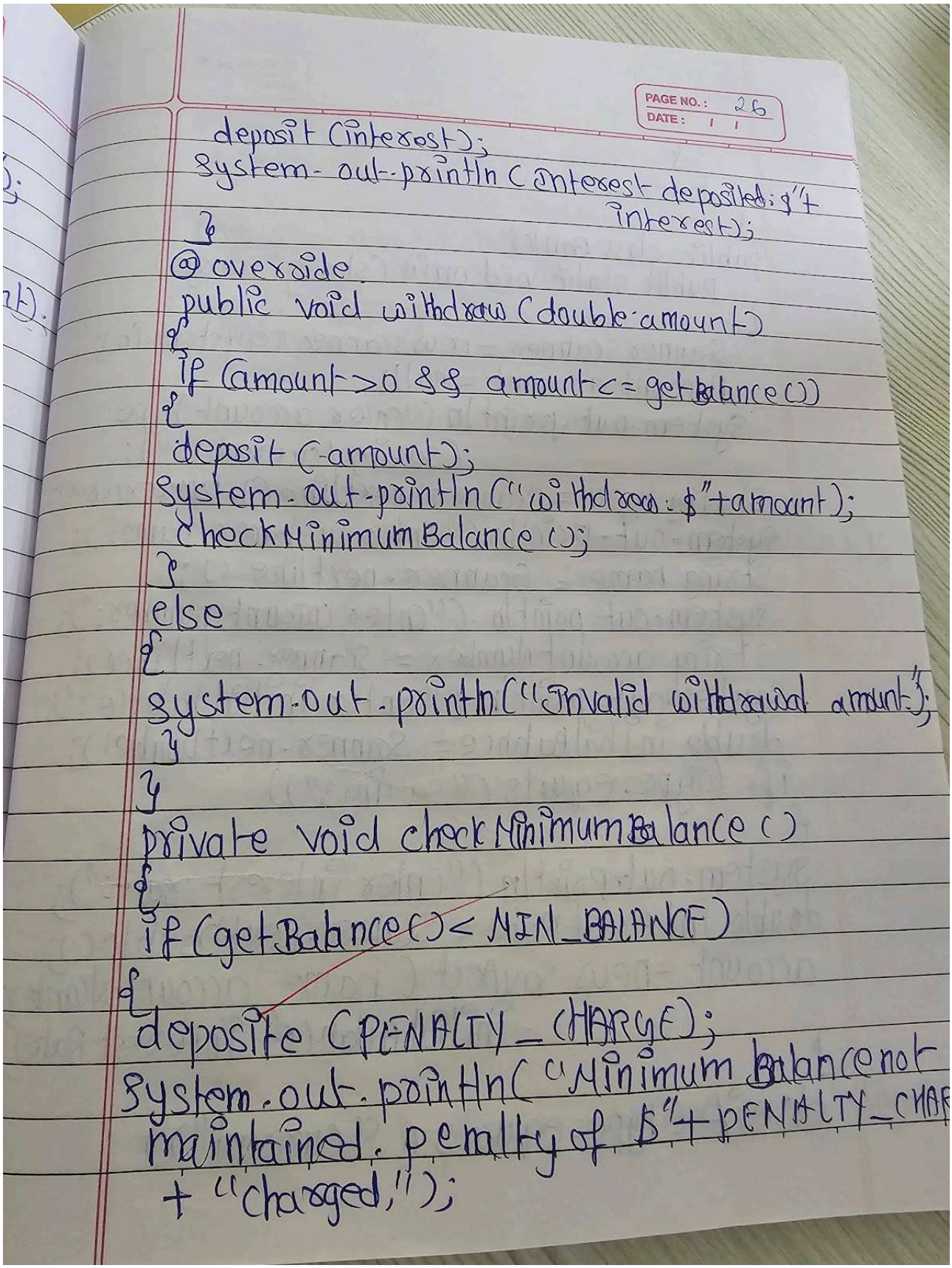
- create a class Account that stores cust name, account number



```
import java.util.Scanner;
abstract class Account
{
 private string customerName;
 private string accountNumber;
 private double balance;
 public Account (string customerName, string
                 accountNumber, double initial
                      Balance)
{
 this.customerName = customerName;
 this.accountNumber = accountNumber;
 this.balance = initialBalance;
}
 public double getBalance()
{
 return balance;
}
 public void deposit (double amount)
 {
  if (amount > 0)
{
 balance += amount;
System.out.println ("Deposited: $" +am
}
```



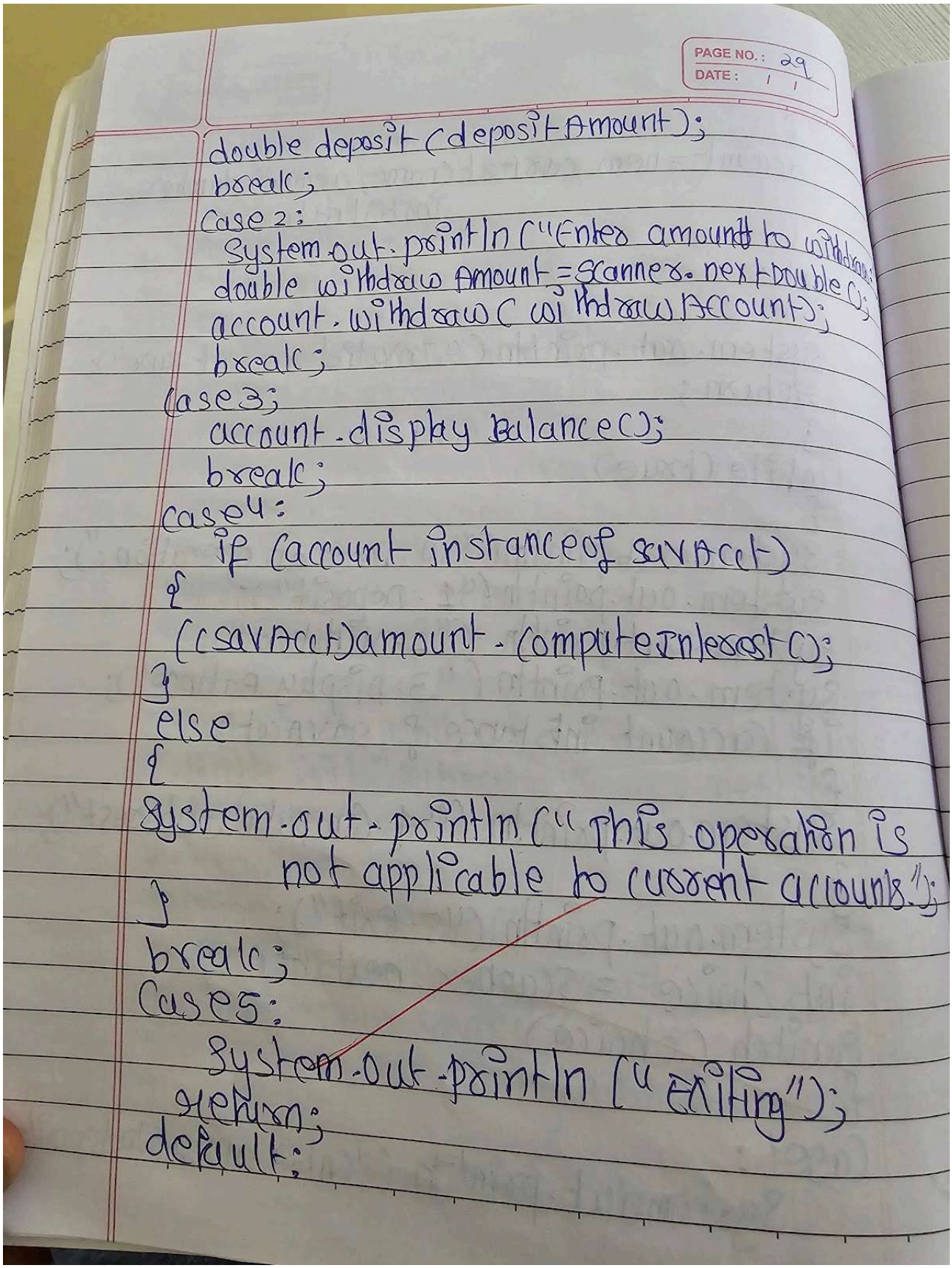
```
else
{
System.out.println("Invalid deposit amount");
}
}
public abstract void withdraw(double amount);
public void displayBalance()
{
System.out.println("current balance: $" +
                   balance);
}
}
class savAct extends Account
{
private double interestRate;
public savAct(String customerName,
              String accountNumber,
              double initialBalance,
              double interestRate)
{
this.interestRate = interestRate;
}
public void computeInterest()
{
double interest = getBalance() *interestR
```



```
deposit(interest);
System.out.println("Interest deposited: $" +
                    interest);
}
@Override
public void withdraw(double amount)
{
if(amount>0 && amount<=getBalance())
{
deposit(-amount);
System.out.println("Withdraw: $"+amount);
checkMinimumBalance();
}
else
{
System.out.println("Invalid withdrawal amount");
}
}
private void checkMinimumBalance()
{
if(getBalance()<MIN_BALANCE)
{
deposite(PENALTY_CHARGE);
System.out.println("Minimum Balance not
maintained. penalty of $"+PENALTY_CHAR
+ "charged.");
```

```
    }
  }
}
public class Bank{
   public static void main (String[] args)
{
   Scanner scanner = new scanner (System.in);
   Account account = null;
  System.out.println ("Enter account type
                    (saving / current): ");
 String type = scanner.nextLine().toLowerCase();
 system.out.println ("Enter customer name:");
 String name = Scanner.nextline ();
 System.out.println ("Enter account number:");
 string accountNumber = scanner.nextLine();
 System.out.println ("Enter inital balance:");
 double inialBalance = Scanner.nextDouble();
 if (type.equals ("savings"))
{
system.out.println ("Enter interest rate:");
double interestRate = Scanner.nextDouble();
account = new SavAcct (name, accountNumber,
                    inialBalance, interestRate);
}
else if (type.equals ("current"))
```

```
{
account = new curAcct(name, accountNumber,
                      initialBalance);
}
else
{
System.out.println("Invalid account type");
return;
}
while (true)
{
System.out.println("\n choose an operation:");
System.out.println("1. Deposit");
System.out.println("2. Withdraw");
System.out.println("3. Display Balance");
if (account instance of savAcct)
{
System.out.println("4. Compute Interest");
}
System.out.println("5. Exit");
int choice = scanner.nextInt();
switch (choice)
{
case 1:
   System.out.println("Enter amount to deposit");
```


PAGE NO.: 29
DATE: / /

```
double deposit (depositAmount);
break;
Case 2:
  System.out.println("Enter amount to withdraw:");
  double withdrawAmount = scanner.nextDouble();
  account.withdraw(withdrawAccount);
  break;
Case 3;
  account.display Balance();
  break;
case 4:
  if (account instanceof savAcct)
{
 (savAcct)amount.computeInterest();
}
else
{
System.out.println("This operation is
      not applicable to current accounts.");
}
break;
Case 5:
  System.out.println("Exiting");
  return;
default:
```

PAGE NO.: 30
DATE: / /

```
System.out.println("Invalid choice Try again");
        }
    }
}
```

Output-

```
Enter Account type (savings / current):
current
Enter customer name:
veena
Enter account number:
1
Enter initial balance:
5000
choose an operation:
1. Deposit
2. withdraw
3. Display Balance
4. EXIT
1
Enter amount to deposit:
400
Deposited: $400.0
```

PAGE NO.: 31
DATE: / /

choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
2
Enter amount to withdraw:
1000
Withdrew: $1000.0
choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
3.
current balance: $5400.0
choose an operation
1. Deposit
2. Withdraw
3. Display Balance
4. EXIT
4.
Exiting.

PAGE NO.: 32
DATE: / /

Enter account type(saving/current):
savings
Enter customer name:
veena
Enter account number
1
Enter intial balance:
5000
Enter interest rate:
4

Choose an operation:
1. Deposit
2. withdraw
3. Display Balance
4. Compute Interest
5. EXIT

1
Enter amount to deposit: 1000
Deposited: $ 1000.0

2.
Enter amount to withdraw: 1000
withdrew $1000.0

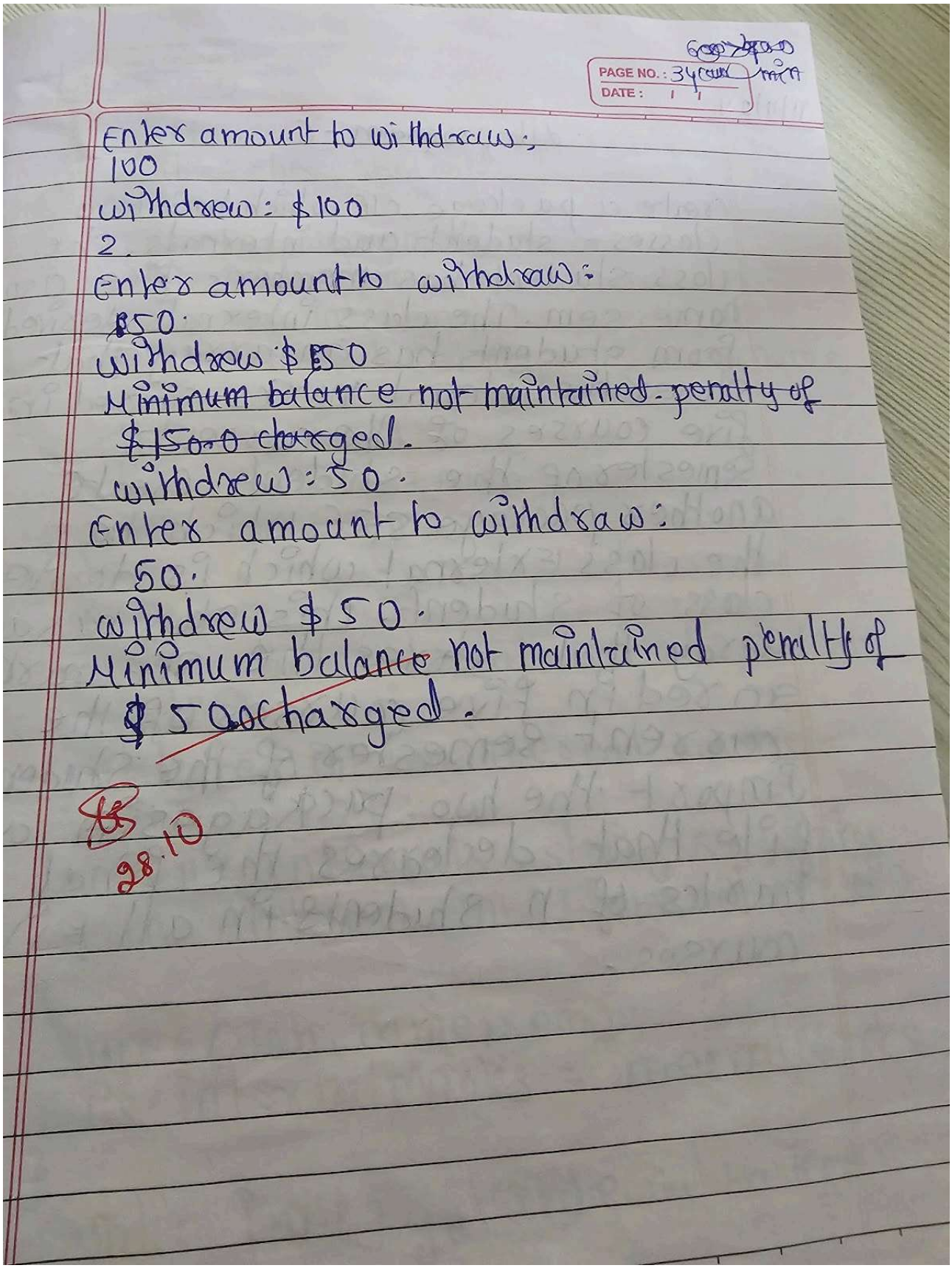
3.
current balance: $6000.0
4.
Deposited : $240.0
Interest deposited : $ 240.0.
5.
Exiting.

28.10

Enter account type (savings/current):
current
Enter customer name:
veena
Enter account number
3
Enter inital balance:
600

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Exit

2.


PAGE NO.: 34
DATE: / /
Enter amount to withdraw:
100
withdrew: $100
2.
Enter amount to withdraw:
$50.
withdrew $50
Minimum balance not maintained. penalty of
$150.0 charged.
withdrew: 50.
Enter amount to withdraw:
50.
withdrew $50.
Minimum balance not maintained penalty of
$50.0 charged.
28.10

code:

```
import java.util.Scanner;

abstract class Account {
    private String customerName;
    private String accountNumber;
    private double balance;

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

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: $" + amount);
        } else {
            System.out.println("Invalid deposit amount.");
        }
    }

    public abstract void withdraw(double amount);

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

class SavAcct extends Account {
    private double interestRate;

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

    public void computeInterest() {
        double interest = getBalance() * interestRate / 100;
        deposit(interest);
        System.out.println("Interest deposited: $" + interest);
```

```
    public void withdraw(double amount) {
        if (amount > 0 && amount <= getBalance()) {
            deposit(-amount);
            System.out.println("Withdrew: $" + amount);
        } else {
            System.out.println("Invalid withdrawal amount.");
        }
    }
}

class CurAcct extends Account {
    private static final double MIN_BALANCE = 500.0;
    private static final double PENALTY_CHARGE = 50.0;

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

    @Override
    public void withdraw(double amount) {
        if (amount > 0 && amount <= getBalance()) {
            deposit(-amount);
            System.out.println("Withdrew: $" + amount);
            checkMinimumBalance();
        } else {
            System.out.println("Invalid withdrawal amount.");
        }
    }

    private void checkMinimumBalance() {
        if (getBalance() < MIN_BALANCE) {
            deposit(-PENALTY_CHARGE);
            System.out.println("Minimum balance not maintained. Penalty of $" + PENALTY_CHARGE
+ " charged.");
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Account account = null;
        System.out.println("Enter account type (savings/current): ");
        String type = scanner.nextLine().toLowerCase();
        System.out.println("Enter customer name: ");
        String name = scanner.nextLine();
        System.out.println("Enter account number: ");
```

```
String accountNumber = scanner.nextLine();
System.out.println("Enter initial balance: ");
double initialBalance = scanner.nextDouble();

if (type.equals("savings")) {
    System.out.println("Enter interest rate: ");
    double interestRate = scanner.nextDouble();
    account = new SavAcct(name, accountNumber, initialBalance, interestRate);
} else if (type.equals("current")) {
    account = new CurAcct(name, accountNumber, initialBalance);
} else {
    System.out.println("Invalid account type.");
    return;
}
while (true) {
    System.out.println("\nChoose an operation: ");
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Display Balance");
    if (account instanceof SavAcct) {
        System.out.println("4. Compute Interest");
    }
    System.out.println("5. Exit");
    int choice = scanner.nextInt();

    switch (choice) {
        case 1:
            System.out.println("Enter amount to deposit: ");
            double depositAmount = scanner.nextDouble();
            account.deposit(depositAmount);
            break;
        case 2:
            System.out.println("Enter amount to withdraw: ");
            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("This operation is not applicable to current accounts.");
            }
            break;
        case 5:
```

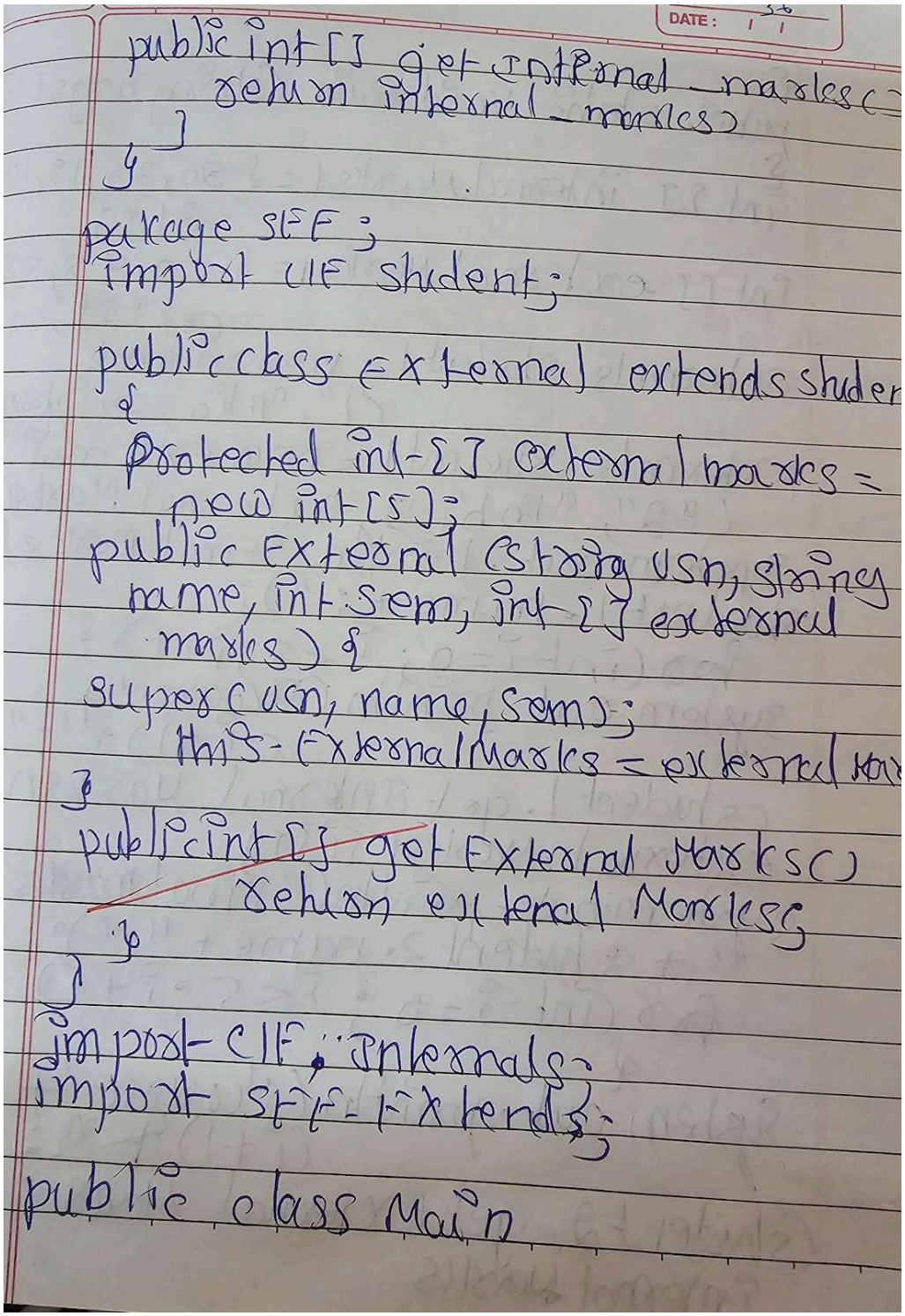
```
                System.out.println("Exiting.");
                return;
            default:
                System.out.println("Invalid choice. Try again.");
        }
    }
}
}
```

```
D:\1BM23CS135>javac Bank.java

D:\1BM23CS135>java Bank
Enter account type (savings/current):
current
Enter customer name:
veena
Enter account number:
1
Enter initial balance:
5000

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
1
Enter amount to deposit:
400
Deposited: $400.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
2
Enter amount to withdraw:
1000
Invalid deposit amount.
Withdrew: $1000.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
3
Current balance: $5400.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
5
Exiting.
```

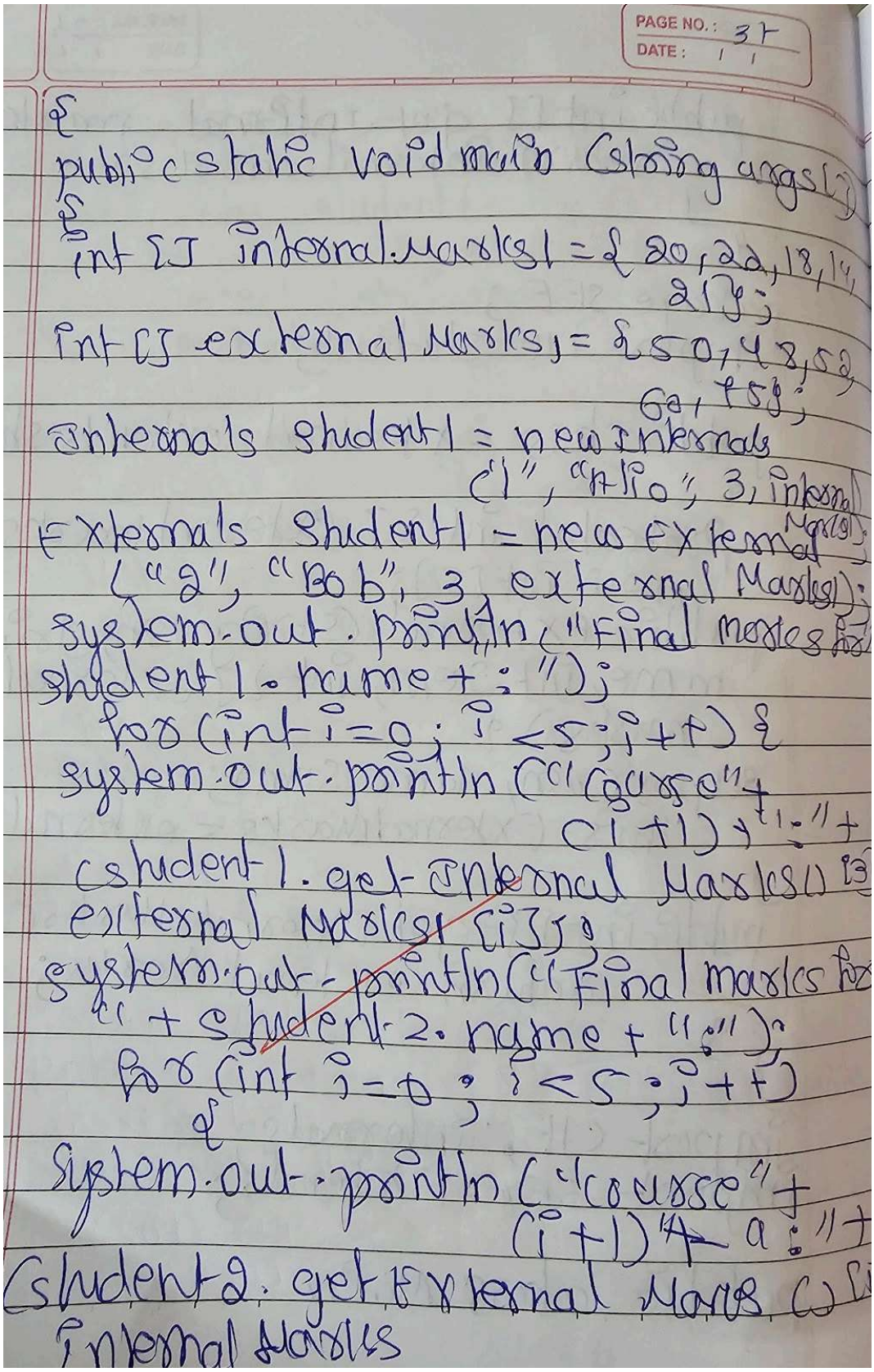
```
D:\1BM23CS135>javac Bank.java

D:\1BM23CS135>java Bank
Enter account type (savings/current):
savings
Enter customer name:
veena
Enter account number:
1
Enter initial balance:
5000
Enter interest rate:
4

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest
5. Exit
1
Enter amount to deposit:
1000
Deposited: $1000.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest
5. Exit
2
Enter amount to withdraw:
1000
Invalid deposit amount.
Withdrew: $1000.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest
5. Exit
3
Current balance: $6000.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
```

```
5. Exit
4
Deposited: $240.0
Interest deposited: $240.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest
5. Exit
5
Exiting.
```

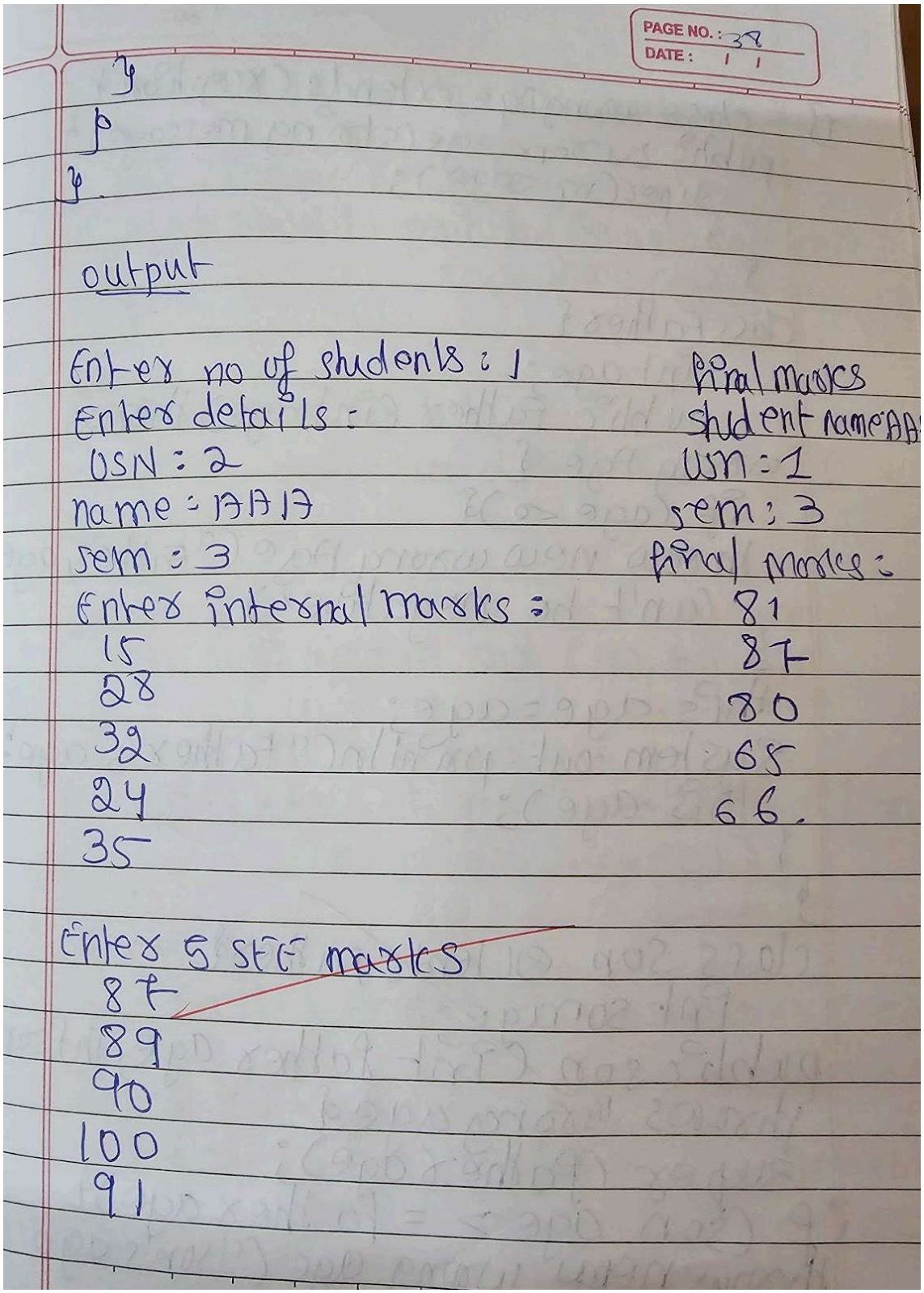
6.
```
package CIE;
import java.util.Scanner;
public class student{
    protected string USN;
    protected string name;
    protected int sem;

public student(string usn, string name,
                int sem)
{
  this.usn = usn;
  this.name = name;
  this.sem = sem;
}
package CIE;
public class Internals extends student
{
protected int[] internal_marks =
new int[5];
public Internals(String usn, string
name, int sem, int[] internal_marks)
  super(usn, name, sem);
  this.internal_marks = internal_
  marks;
}
```



```
public int[] getInternal_marks()
    return internal_marks;
}

package SEE;
import CIE.Student;

public class External extends Student
{
  protected int[] external marks =
      new int[5];
public External(String usn, String
  name, int sem, int[] external
    marks) {
super(usn, name, sem);
    this.ExternalMarks = external mar
}
public int[] getExternal Marks()
      return external Marks;
  }
}

import CIE.Internals;
import SEE.Extends;

public class Main
```


PAGE NO.: 37
DATE: / /

{
public static void main (String args[])
{
int [] internal Marks1 = { 20, 22, 18, 14, 21};
int [] external Marks1 = { 50, 48, 52, 60, 55};
Internals student1 = new Internals ("1", "Alice", 3, internal Marks1);
Externals student1 = new External ("2", "Bob", 3, external Marks1);
System.out.println("Final marks for student1.name + ":");
for (int i=0; i<5; i++) {
System.out.println("Course" + (i+1) + ": " + student1.getInternal Marks()[i] + external Marks1[i]);
System.out.println("Final marks for " + student2.name + ":");
for (int i=0; i<5; i++)
{
System.out.println("Course" + (i+1) + ": " + student2.getExternal Marks()[i] + Internal Marks



code:

```
package CIE;
```

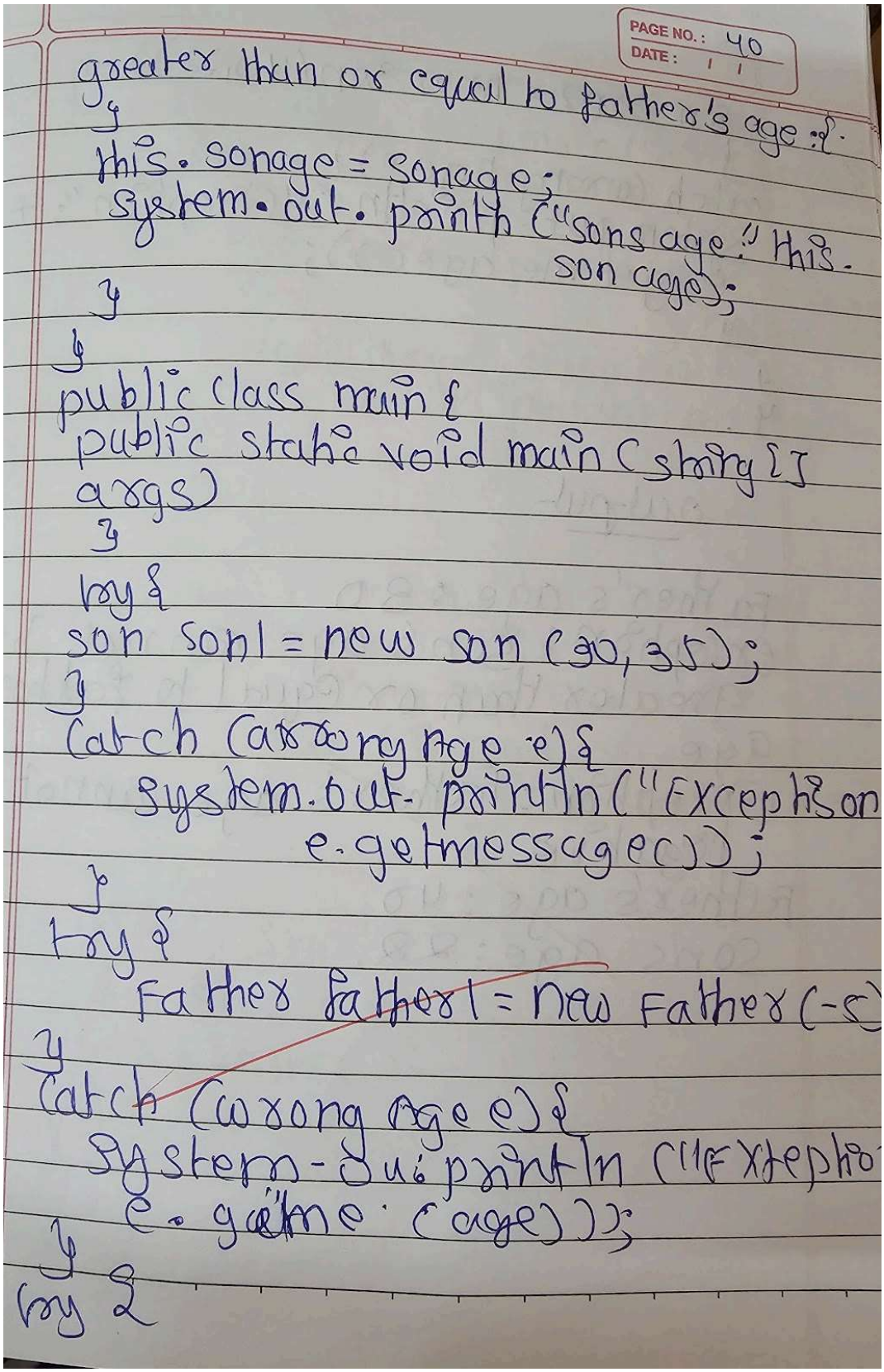
```
import java.util.Scanner;
public class Student
{
public String usn;
 public String name;
 public int sem;
public Student( String usn,
 String name,  int sem)
{
this.usn=usn;
this.name=name;
this.sem=sem;
}
}
package CIE;
public class Internals extends Student
{

 int internalmarks[]=new int[5];
public Internals( String usn,String name,int sem,int internalmarks[]){
super(usn,name,sem);
this.internalmarks=internalmarks;
}
public int[] getInternalmarks(){
return internalmarks;}
}
package SEE;
import CIE.Student;
public class External extends Student
{
 int[] externalmarks=new int[5];
public External ( String usn,
 String name,
 int sem,int externalmarks[])
{
super(usn,name,sem);
this.externalmarks=externalmarks;
}
public int[] getExternalMarks(){
return externalmarks;
}
}

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

```
public class Main1{
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);


      System.out.print("Enter the number of students: ");
      int n = scanner.nextInt();
      scanner.nextLine();

          for (int i = 0; i < n; i++) {
      System.out.println("\nEnter details for student " + (i + 1));

        System.out.print("Enter USN: ");
      String usn = scanner.nextLine();

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

      System.out.print("Enter Semester: ");
      int sem = scanner.nextInt();
      scanner.nextLine();
               int[] internalMarks = new int[5];
      System.out.println("Enter internal marks for 5 courses: ");
      for (int j = 0; j < 5; j++) {
         System.out.print("Course " + (j + 1) + ": ");
         internalMarks[j] = scanner.nextInt();
      }
      scanner.nextLine();

      int[] externalMarks = new int[5];
      System.out.println("Enter external marks for 5 courses: ");
      for (int j = 0; j < 5; j++) {
         System.out.print("Course " + (j + 1) + ": ");
         externalMarks[j] = scanner.nextInt();
      }
      scanner.nextLine();
      Internals internStudent = new Internals(usn, name, sem, internalMarks);
      External externalStudent = new External(usn, name, sem, externalMarks);

      System.out.println("\nFinal marks for " + internStudent.name + ":");
      for (int j = 0; j < 5; j++) {
         int finalMark = internStudent.getInternalmarks()[j] + externalMarks[j];
        System.out.println("Course " + (j + 1) + ": " + finalMark);
      }

           }
```

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

```
D:\131>javac CIE/Student.java CIE/Internals.java SEE/External.java Main1.java

D:\131>java Main1
Enter the number of students: 1

Enter details for student 1
Enter USN: 2
Enter Name: l
Enter Semester: 1
Enter internal marks for 5 courses:
Course 1: 23
Course 2: 23
Course 3: 23
Course 4: 23
Course 5: 23
Enter external marks for 5 courses:
Course 1: 56
Course 2: 56
Course 3: 56
Course 4: 56
Course 5: 56

Final marks for l:
Course 1: 79
Course 2: 79
Course 3: 79
Course 4: 79
Course 5: 79

D:\131>
```

#)
```
class wrongAge extends Exception {
public wrongage(string message){
    super(message);
  }
}
class Father{
    int age;
    public Father (int age) throws
 wrong Age {
  if (age <0){
 throw new wrong Age ("Father, age
   (can't be negative");
 }
 this.age = age;
 System.out.println("Father's age:"
   this.age);
}
}
class son extends Father{
    int sonage;
public son (int father age, int sonag
 throws wrong age{
 super (fatherage);
if (son age >= father age){
throw new wrong age ("son's age can
```



PAGE NO.: 40
DATE: / /

```
greater than or equal to father's age :(.
    }
    this.sonage = sonage;
     System.out.println("sons age:" this.
                                son age);
    }
}
public class main {
 public static void main (string [ ]
 args)
  }
 try {
 son son1 = new son (30, 35);
 }
 Catch (arrongAge e){
    System.out.println("Exception
            e.getmessage());
  }
 try {
    Father father1 = new Father(-5)
}
Catch (wrongAge e){
  System.out.println("Exception
    e.getMe.(age));
 }
try {
```

```
Son son2 = new Son(40,22))
}
catch (wrongAge e){
System.out.println("Exception:"+
e.getmeage());
}
}
}
```

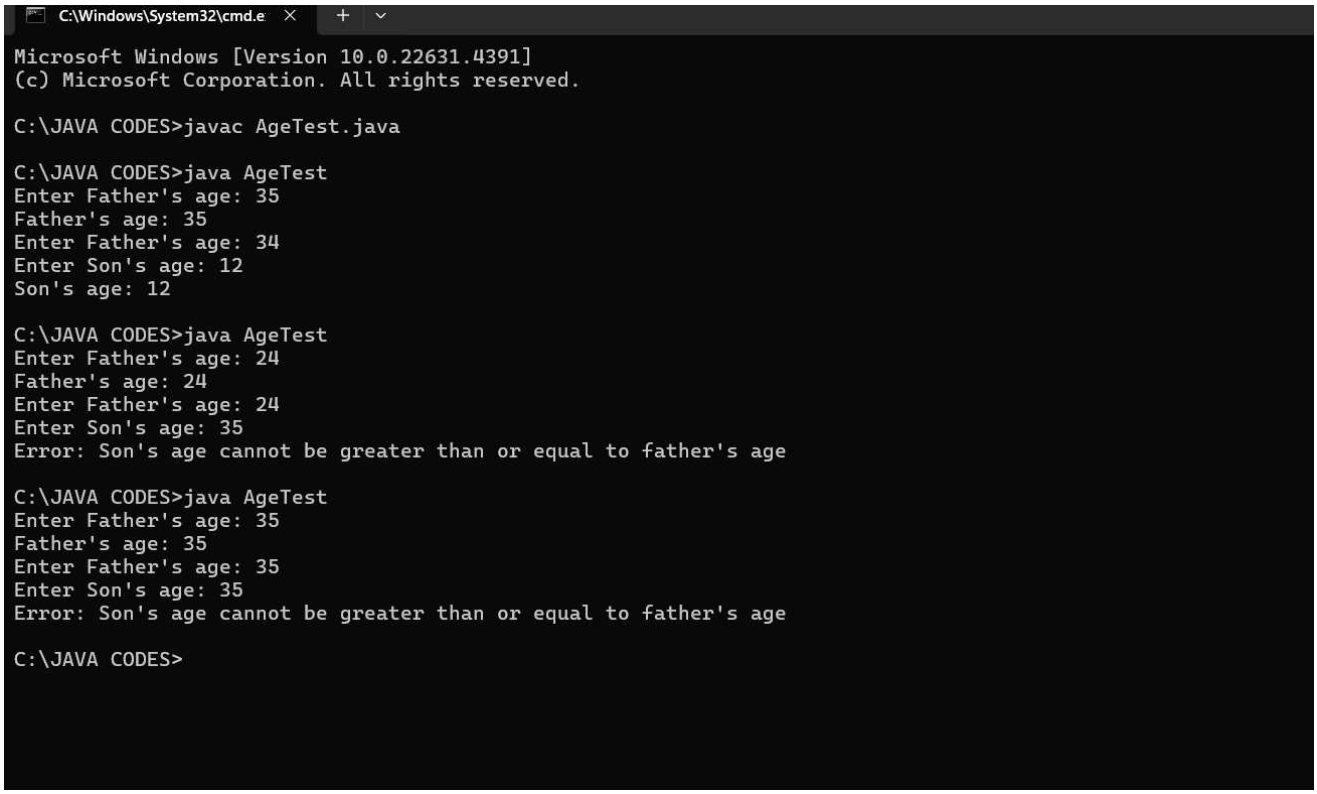
output

Father's age : 30
Exception: Son's age cannot be greater than or equal to father's age.
Exception: Father's age cannot be negative
Father's age : 40
Son's age : 22.

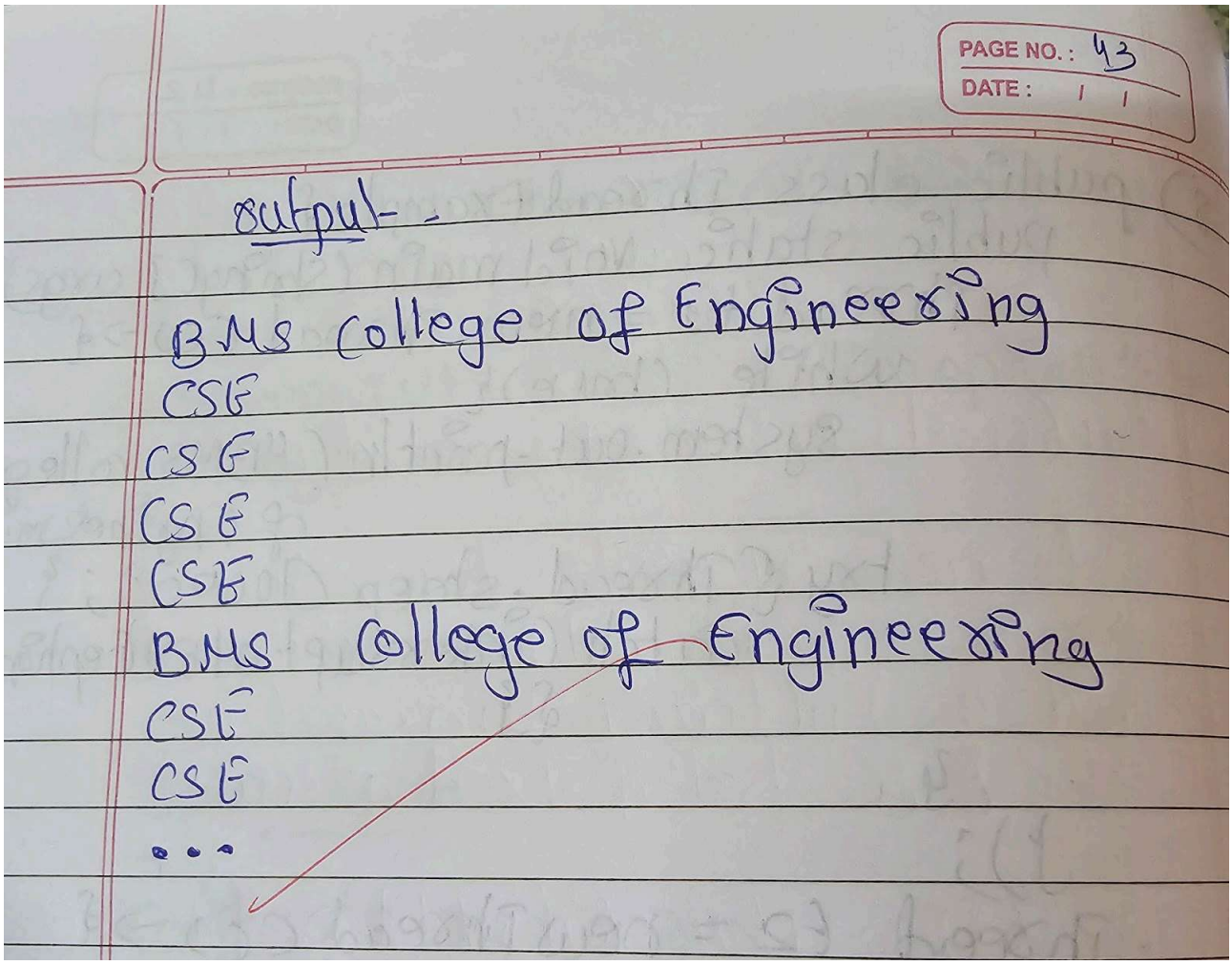
code:

```
import java.util.Scanner;

class WrongAge extends Exception {

    public WrongAge() {
        super("Age Error");
    }


    public WrongAge(String message) {
        super(message);
    }
}


class InputScanner {
    Scanner s = new Scanner(System.in);


    public int readAge() {
        return s.nextInt();
    }
}


class Father extends InputScanner {
    protected int fatherAge;


    public Father() throws WrongAge {
        System.out.print("Enter father's age: ");
        fatherAge = readAge();
        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }


    public void displayFatherAge() {
        System.out.println("Father's age: " + fatherAge);
    }
}


class Son extends Father {
```

```
    private int sonAge;


    public Son() throws WrongAge {

        super();
        System.out.print("Enter son's age: ");
        sonAge = readAge();


        if (sonAge > fatherAge) {
            throw new WrongAge("Son's age cannot be greater than father's age");
        } else if (sonAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }


    public void displaySonAge() {
        System.out.println("Son's age: " + sonAge);
    }
}

public class Main {
    public static void main(String[] args) {
        try {

            Son son = new Son();


            son.displayFatherAge();
            son.displaySonAge();
        } catch (WrongAge e) {

            System.out.println(e.getMessage());
        }
    }
}
```


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.
C:\JAVA CODES>javac AgeTest.java
C:\JAVA CODES>java AgeTest
Enter Father's age: 35
Father's age: 35
Enter Father's age: 34
Enter Son's age: 12
Son's age: 12
C:\JAVA CODES>java AgeTest
Enter Father's age: 24
Father's age: 24
Enter Father's age: 24
Enter Son's age: 35
Error: Son's age cannot be greater than or equal to father's age
C:\JAVA CODES>java AgeTest
Enter Father's age: 35
Father's age: 35
Enter Father's age: 35
Enter Son's age: 35
Error: Son's age cannot be greater than or equal to father's age
C:\JAVA CODES>

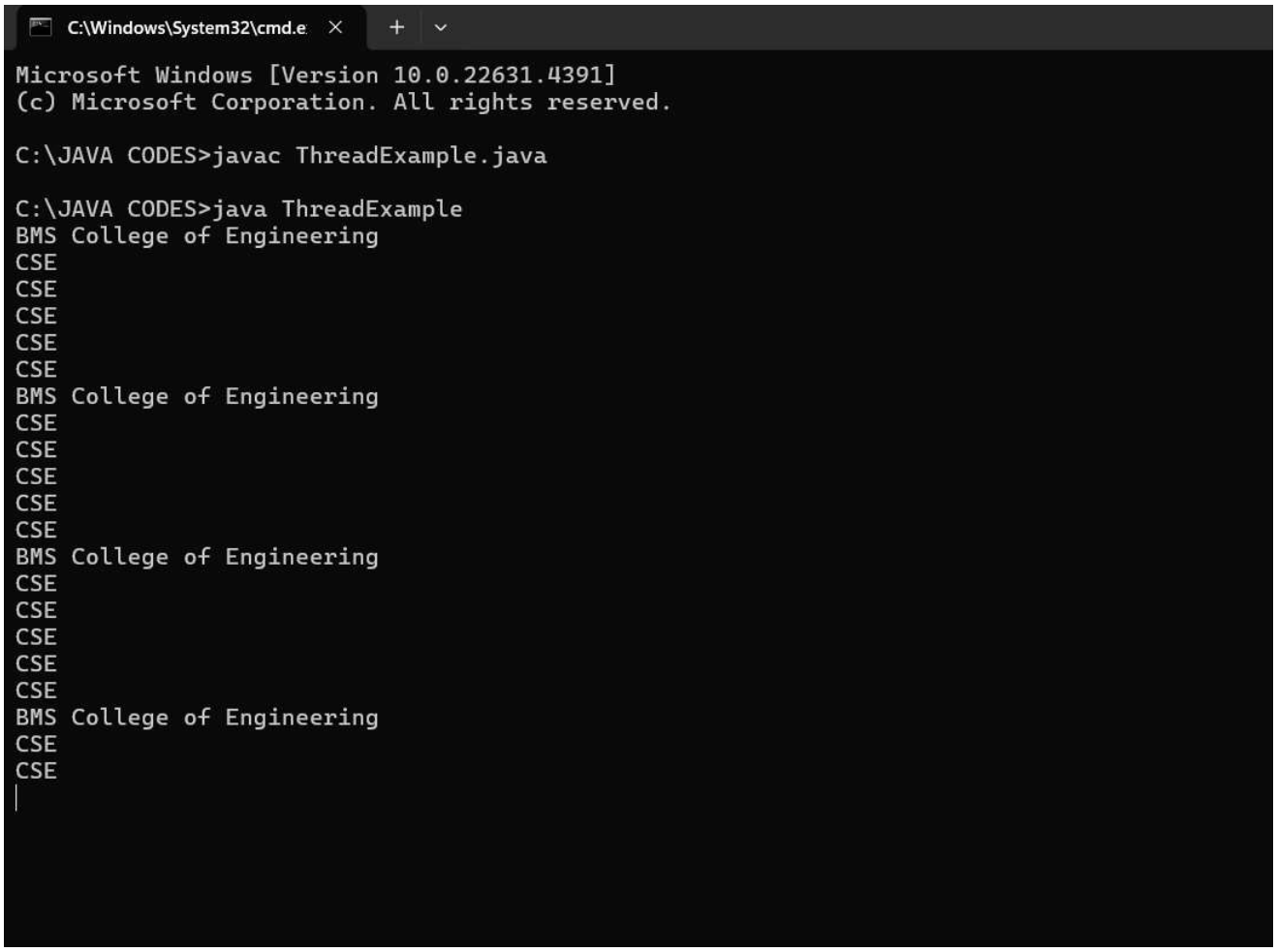
8)
```
public class ThreadExample {
    public static void main(String[] args){
        Thread t1 = new Thread(() -> {
            while (true){
                System.out.println("BMS colleg
                                    of Engineeri
                try { Thread.sleep(10000); }
                catch (InterruptedException e
                    {}
            }
        });
        Thread t2 = new Thread(() -> {
            while (true){
                System.out.println("CSE");
                try { Thread.sleep(2000);
                catch (InterruptedExcep
                    {}
            }
        });
        t1.start();
        t2.start();
    }
}
```



code:

```
public class ThreadExample {
   static class BMSPrinter extends Thread {
      public void run() {
         try {
            while (true) {
               System.out.println("BMS College of Engineering");
               Thread.sleep(10000);
            }
         } catch (InterruptedException e) {
            System.out.println(e);
         }
      }
   }
   static class CSEPrinter extends Thread {
      public void run() {
         try {
```

```
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }
  }

  public static void main(String[] args) {

    Thread t1 = new BMSPrinter();
    Thread t2 = new CSEPrinter();


    t1.start();
    t2.start();
  }
}
```

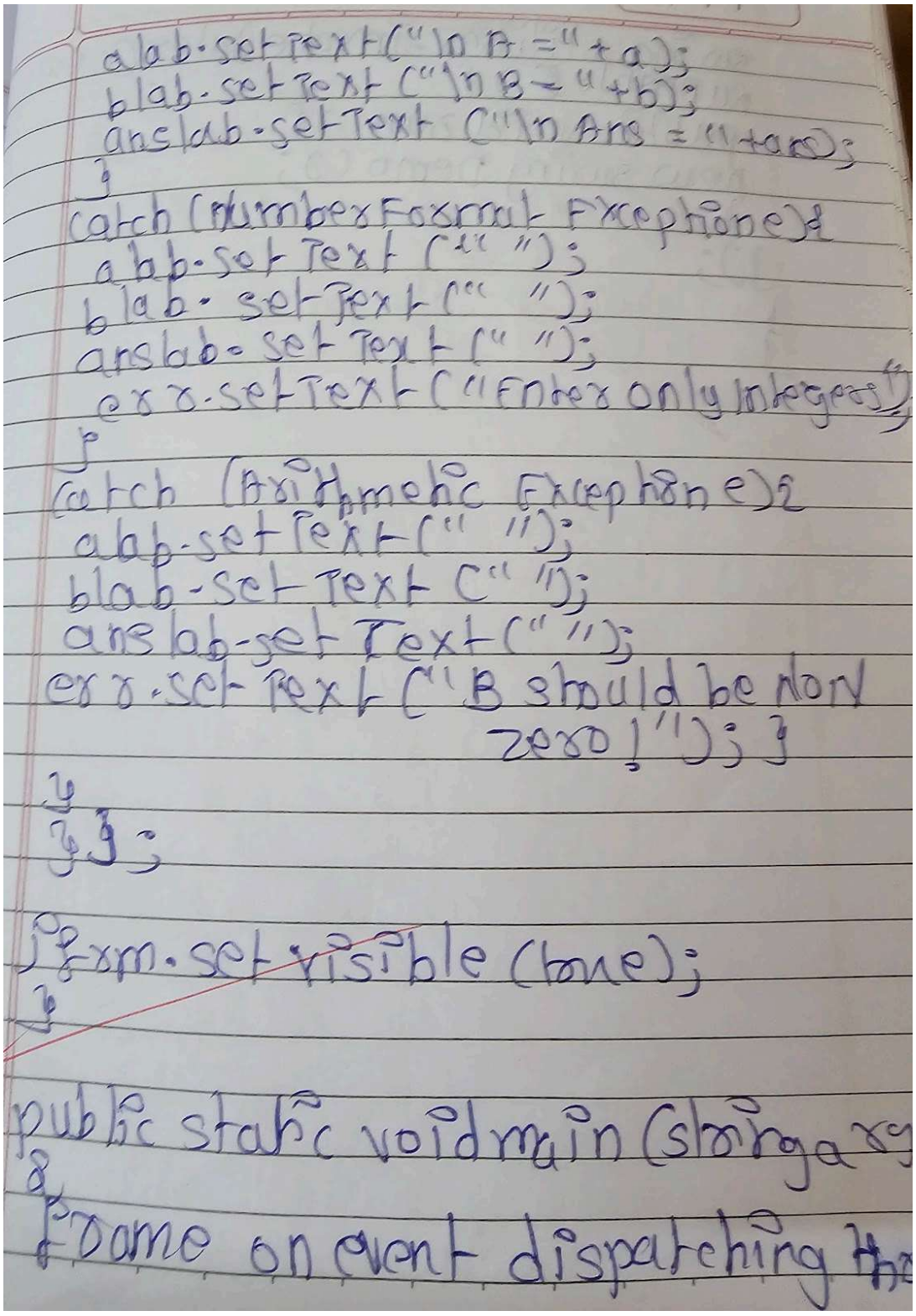

C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.
C:\JAVA CODES>javac ThreadExample.java
C:\JAVA CODES>java ThreadExample
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE

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

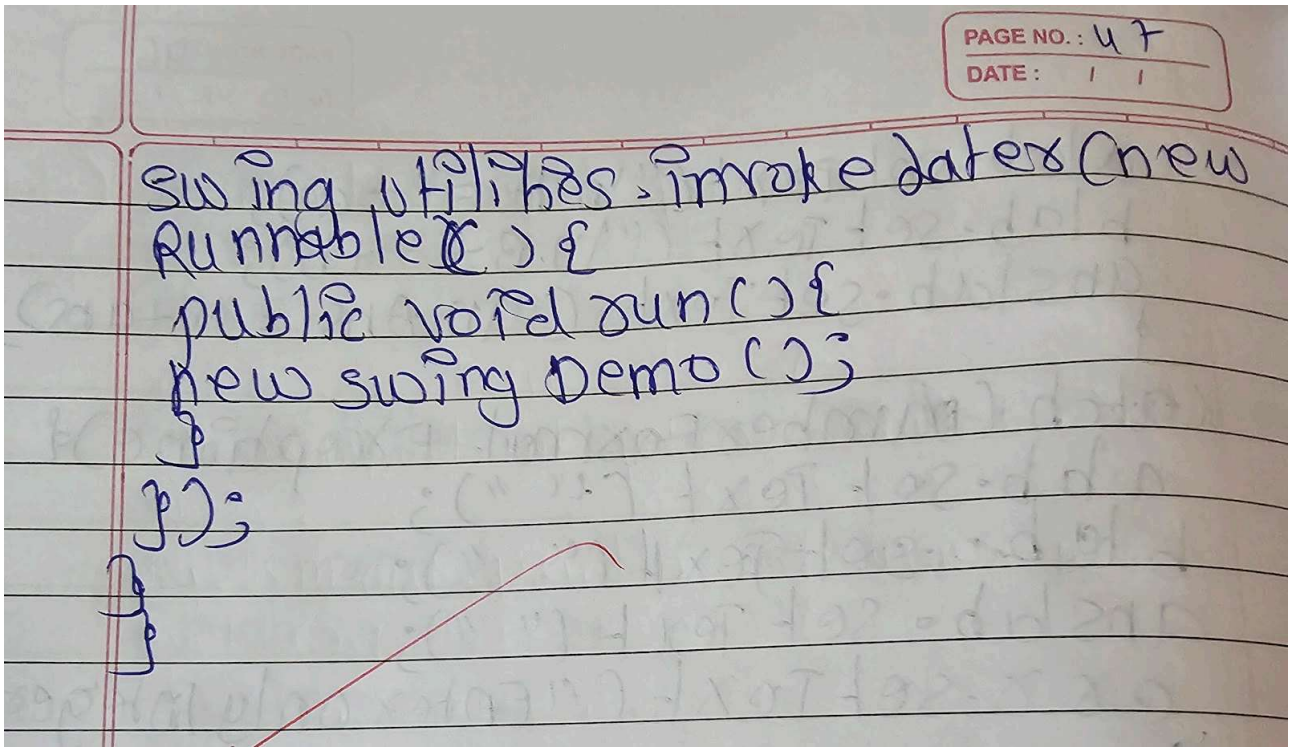
class swingDemo{
swingDemo(){
JFrame jfrm = new JFrame("Divider App");
jfrm.setLayout(new FlowLayout());
jfrm.setSize(275,150);
jfrm.setDefaultCloseOperation
(JFrame.EXIT_ON_CLOSE);
JLabel jlab = new JLabel("Enter the
dividend and divident:");
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8
JButton button = new JButton("Calcul
JLabel err = new JLabel();
JLabel ablab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();

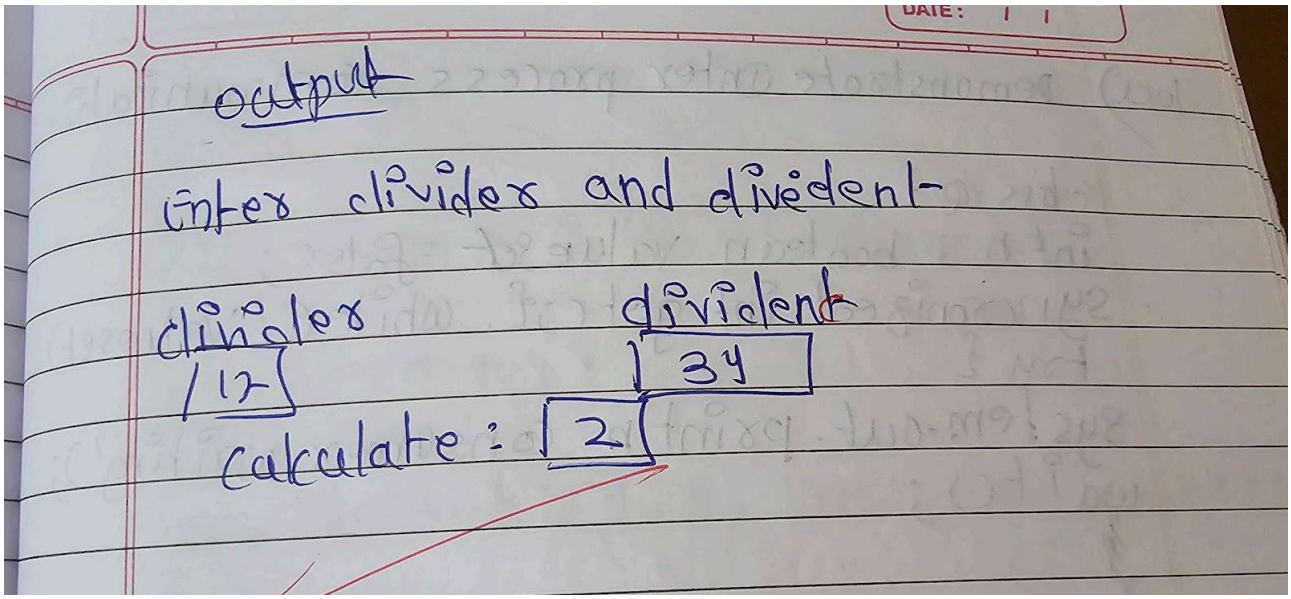
jfrm.add(err);
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);

```
jfrm.add(a|ab);
jfrm.add(b|ab);
jfrm.add(ans|ab);

Action listener l = new Action listener
{
public void action performed
(Action event evt){
 system.out.println("Action ever
 from a text-filed"); }
};
a jtf.add Action listner(l);
b jtf.add Action listener(l);

button.add Action listener(new
 Action listener(){
public void action performed
(Action event evt) { try {
int a = integer.parseInt
(a jtf.get Text());  int b =
integer.parseInt(b jtf.get Tex
 int ans = a/b;
```



```
alab.setText("\n A = " + a);
blab.setText("\n B = " + b);
anslab.setText("\n Ans = " + ans);
}
catch (NumberFormatException e){
abb.setText("");
blab.setText("");
anslab.setText("");
erro.setText("Enter only Integers");
}
catch (ArithmeticException e){
abb.setText("");
blab.setText("");
anslab.setText("");
erro.setText("B should be non
zero!");}
}
}};

frm.setVisible(true);
}

public static void main(String arg
{
Frame1 on event dispatching th
```



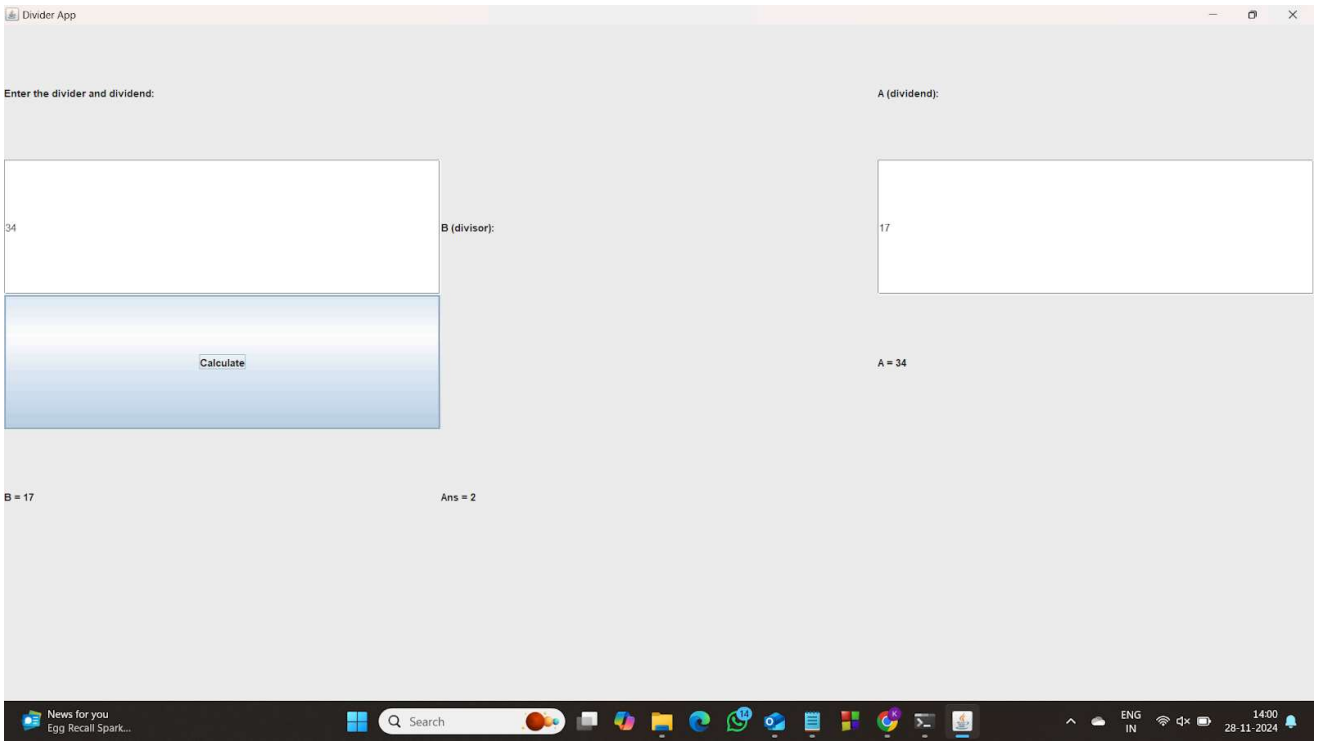


code:

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

class SwingDemo {
    SwingDemo() {

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

```
jfrm.setSize(275, 150);
jfrm.setLayout(new GridLayout(5, 2));


jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


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


JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);


JButton button = new JButton("Calculate");

JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();


jfrm.add(jlab);
jfrm.add(new JLabel()); // empty space
jfrm.add(new JLabel("A (dividend):"));
jfrm.add(ajtf);
jfrm.add(new JLabel("B (divisor):"));
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(err);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);


button.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
      try {

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


         if (b == 0) {
            throw new ArithmeticException("B should be NON zero!");
         }
```

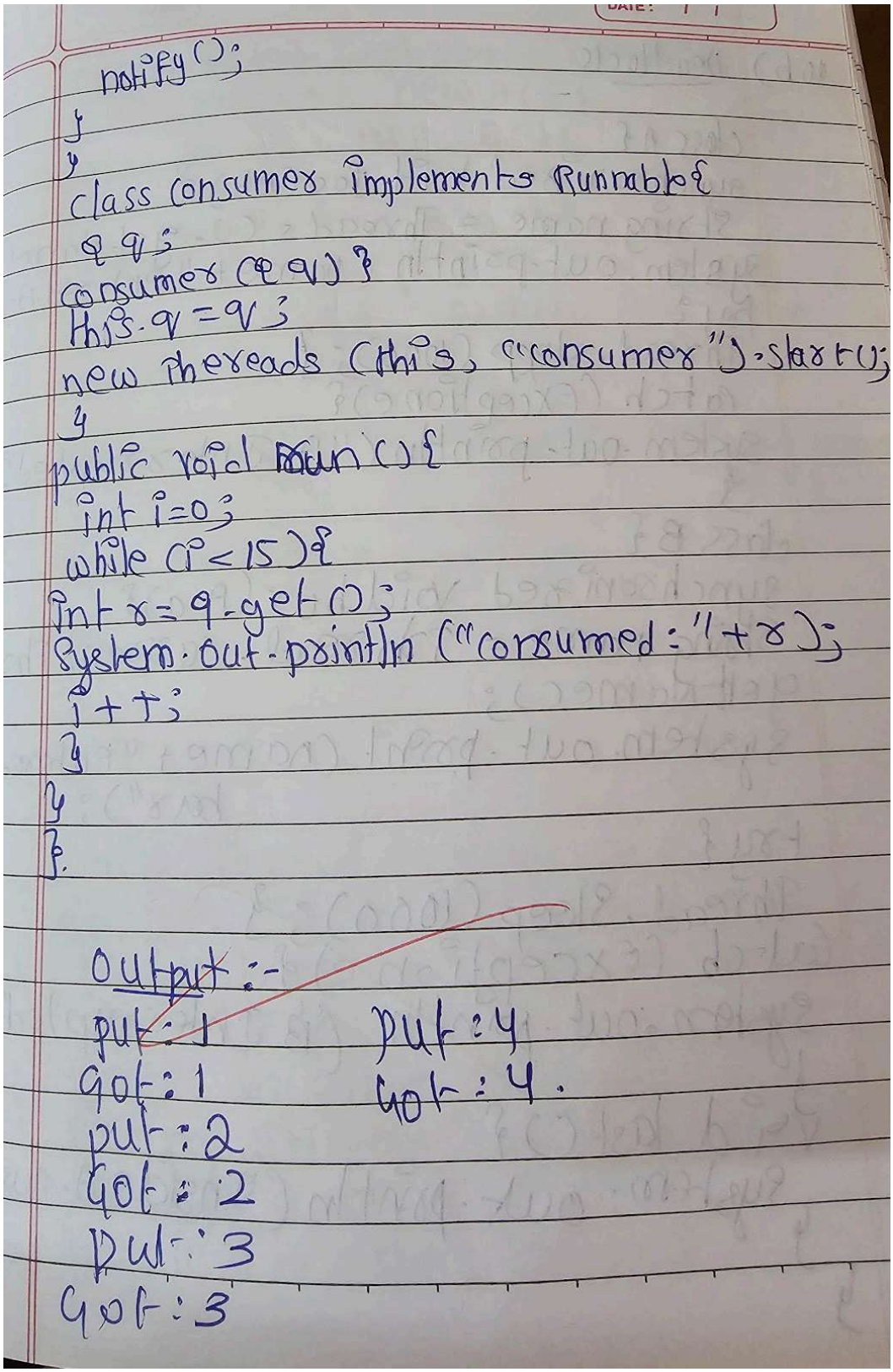
```
                int ans = a / b;
                alab.setText("A = " + a);
                blab.setText("B = " + b);
                anslab.setText("Ans = " + ans);
                err.setText(""); // Clear any previous error message
            } catch (NumberFormatException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("Enter Only Integers!");
            } catch (ArithmeticException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText(e.getMessage()); // Display arithmetic error message
            }
        }
    });


    jfrm.setVisible(true);
  }

  public static void main(String args[]) {

    SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        new SwingDemo();
      }
    });
  }
}
```


Divider App
Enter the divider and dividend:
34
B (divisor):
17
Calculate
A (dividend):
A = 34
B = 17
Ans = 2
News for you
Egg Recall Spark...
Search
ENG
IN
14:00
28-11-2024

DATE : / /

10a) Demonstrate Inter process communication:

```
class Q {
int n ; boolean valueset = false;
synchronized int get(){ while(! valueset);
 try {
system.out.println ("consumer waiting");
wait();
}
}
wait();
} catch (InterruptedException e){
system.out.println ("Interrupted
Exception caught");
}
synchronized void put (int n){
while(value set)
 try {
 System.out.println("Interrupted E
caught")}
this.n = n;
value set = true;
system.out.println ("put:" + n);
system.out.println ("Intimate cons
```



```
notify();
}
}
class consumer implements Runnable{
Q q;
consumer(Q q){
this.q=q;
new Threads(this,"consumer").start();
}
public void run(){
int i=0;
while(i<15){
int r=q.get();
System.out.println("consumed:"+r);
i++;
}
}
}
```

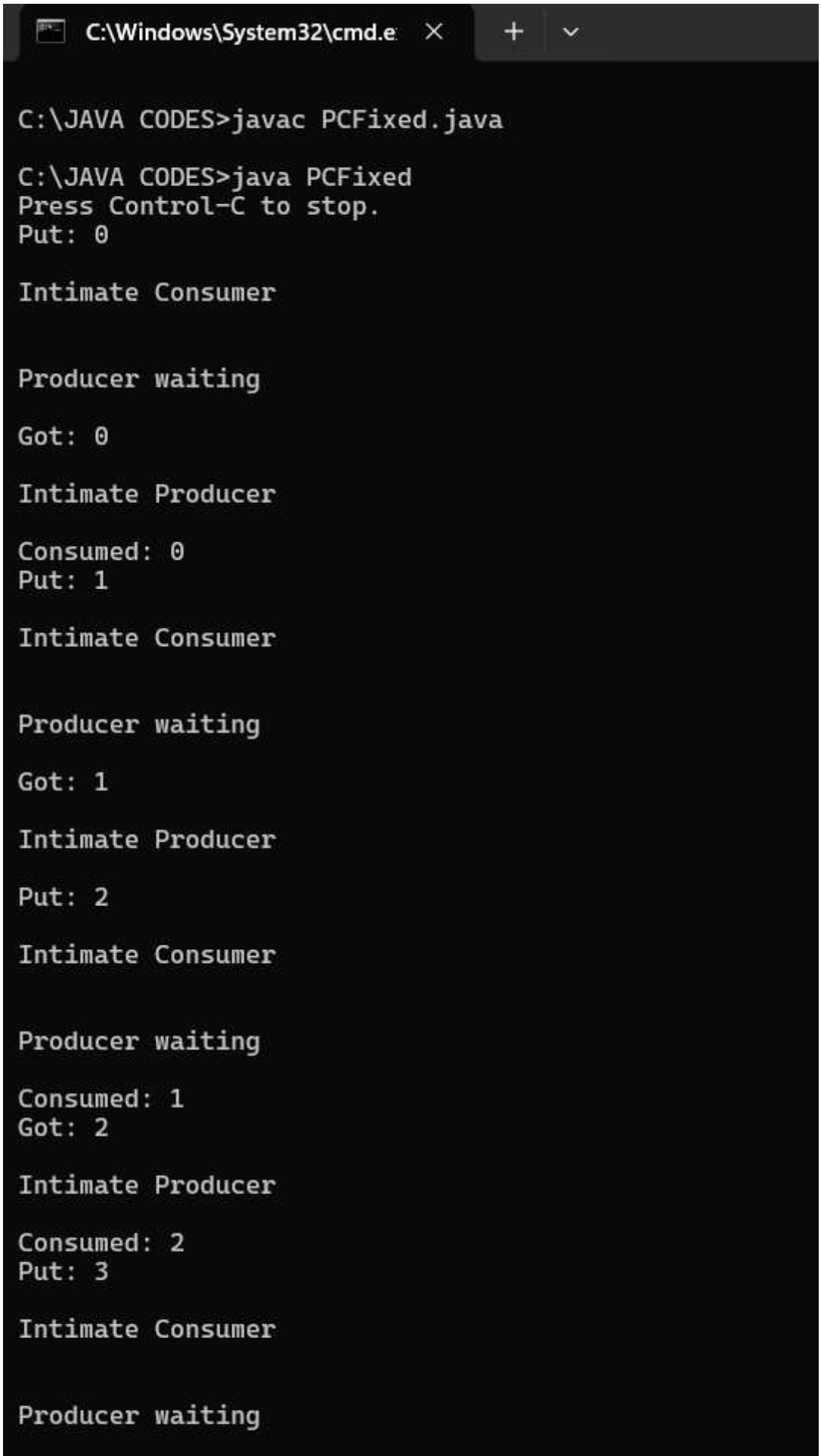
Output :-

```
put:1        put:4
got:1        got:4.
put:2
got:2
put:3
got:3
```

code:

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

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

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

class Producer implements Runnable {
    Q q;

    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }

    public void run() {
        int i = 0;
```

```
        while (i < 15) {
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Q q;

    Consumer(Q q) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }

    public void run() {
        int i = 0;
        while (i < 15) {
            int r = q.get();
            System.out.println("Consumed: " + r);
            i++;
        }
    }
}

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


C:\Windows\System32\cmd.e
C:\JAVA CODES>javac PCFixed.java
C:\JAVA CODES>java PCFixed
Press Control-C to stop.
Put: 0
Intimate Consumer
Producer waiting
Got: 0
Intimate Producer
Consumed: 0
Put: 1
Intimate Consumer
Producer waiting
Got: 1
Intimate Producer
Put: 2
Intimate Consumer
Producer waiting
Consumed: 1
Got: 2
Intimate Producer
Consumed: 2
Put: 3
Intimate Consumer
Producer waiting

PAGE NO.: 57
DATE : / /

40 b) Deadlocks

```
class A{
synchronized void foo(B b){
 String name = Thread.s().get-name();
System.out.println(name+"entered A.foo
try{
 Thread.sleep(1000);}
 catch(Exception e){
System.out.println("A Interrupted");
 }
class B{
 synchronized void box(A a){
 String name = Thread.currentThre
 getName();
 System.out.print(name+"entered
                        box");
 try{
 Thread.Sleep(1000);}
Catch(Exception e){
System.out.println("B Interrupted
 }
 void last(){
 System.out.println("Inside A.la
 }
}
```

PAGE NO. : 52
DATE : / /

```
class Deadlock implements Runable{
    A a = new A();
    B b = new B();
    Deadlock(){
    public void run(){
     b.bar(a);
     System.out.println("Back in
other Thread");
}
public static void main(String arg
                                ()){
new Deadlock();
}
}
```

Output

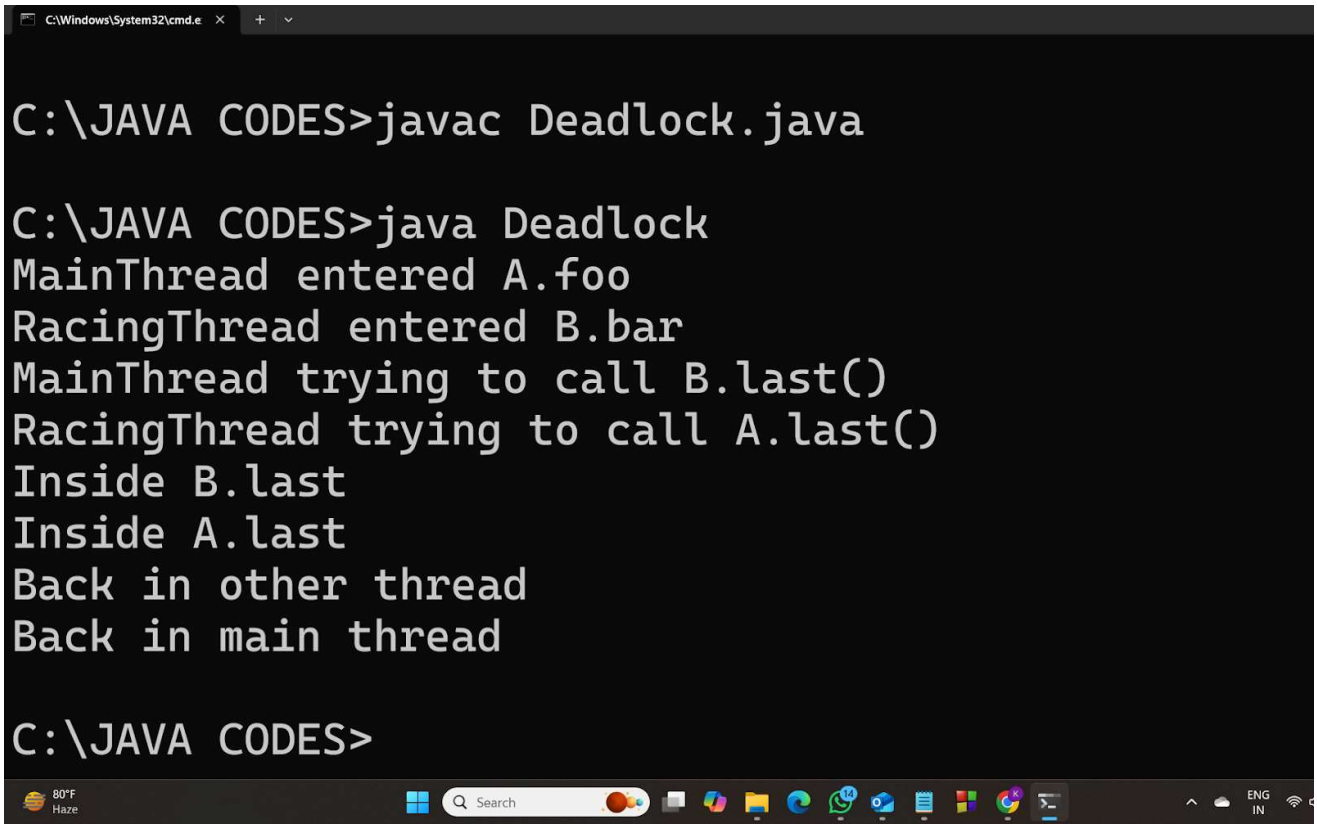
Main Thread entered A foo
Racing Thread entered B.bar
Main Thread trying to call B.b
Inside B.last
Back in main Thread
Racing Thread trying to call A.
Inside A.last
Back in other Thread.

02.12

code:

```
class A {

   synchronized void foo(B b) {

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

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

      try {
         Thread.sleep(1000);
      } catch (Exception e) {
         System.out.println("A Interrupted");
      }

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

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

class B {

   synchronized void bar(A a) {

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

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

      try {
         Thread.sleep(1000);
      } catch (Exception e) {
         System.out.println("B Interrupted");
      }

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

   void last() {
      System.out.println("Inside B.last");
   }
}
```

```
class Deadlock implements Runnable {

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

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


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


      a.foo(b);
      System.out.println("Back in main thread");
   }

   public void run() {

      b.bar(a);
      System.out.println("Back in other thread");
   }

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


C:\Windows\System32\cmd.e
C:\JAVA CODES>javac Deadlock.java
C:\JAVA CODES>java Deadlock
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
RacingThread trying to call A.last()
Inside B.last
Inside A.last
Back in other thread
Back in main thread
C:\JAVA CODES>
80°F
Haze
Search
ENG
IN