VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

OBJECT ORIENTED JAVA PROGRAMMING(23CS3PCOOJ)

Submitted by

SUMANTH S SHETTY(1BM23CS348)

in partial fulfilment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Sep-2024 to Jan-2025

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled **"OBJECT ORIENTED JAVA PROGRAMMING" (23CS3PCOOJ)** carried out by **SUMANTH S SHETTY (1BM23CS348),** 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. The Lab report has been approved as it satisfies the academic requirements in respect of a **"OBJECT ORIENTED JAVA PROGRAMMING (23CS3PCOOJ)"**work prescribed for the said degree.

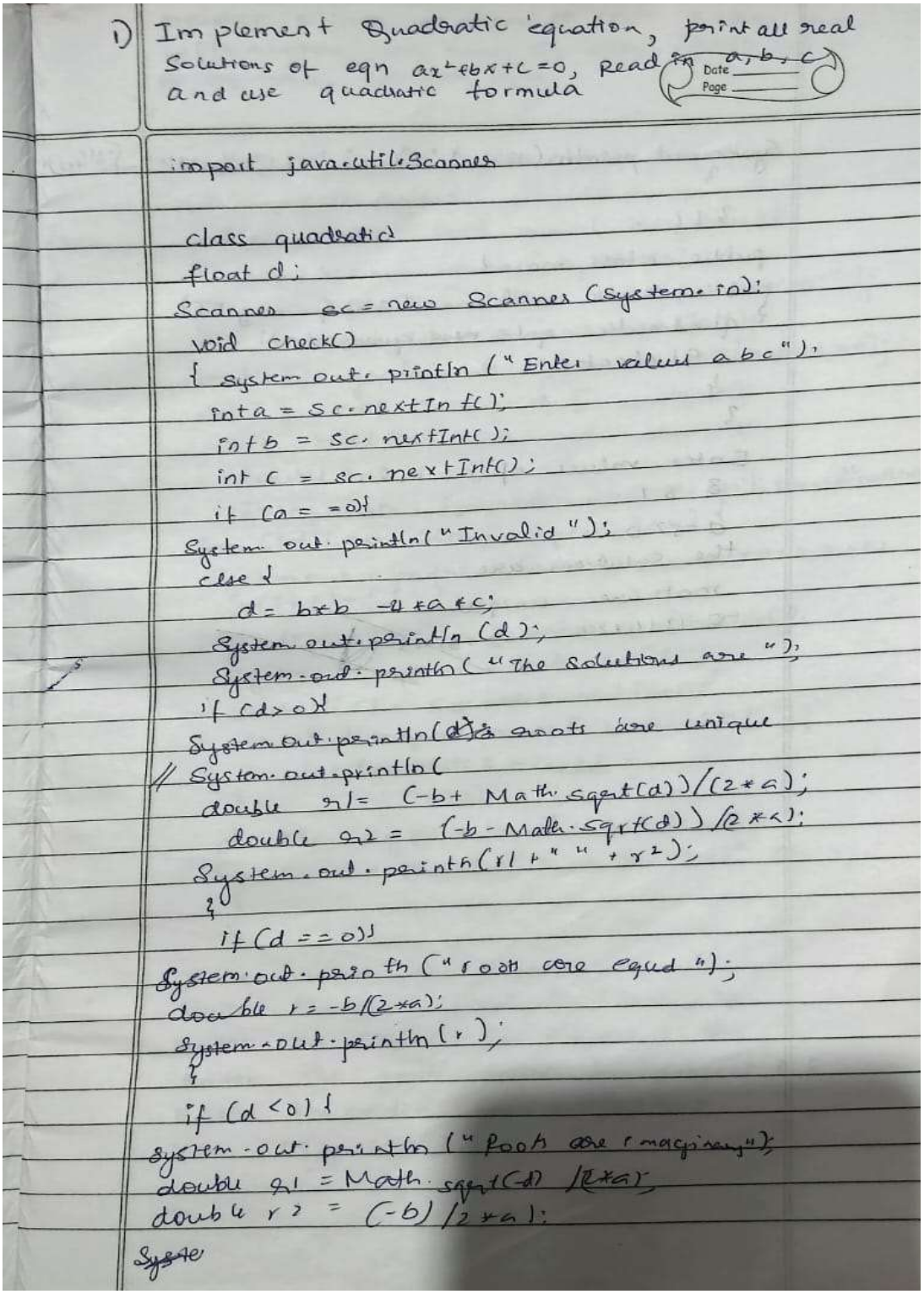
Dr. Nandhini Vineeth Associate Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor & HOD Department of CSE, BMSCE

Index

Sl. No.	Date	Experiment Title	Page No.
1	26-09-2024	Quadratic Equation Solution	1-4
2	3-10-2024	SGPA Calculation	5-9
3	19-10-2024	Library program: demonstration of toString() method	10-14
4	24-10-2024	Abstract Class demonstration program	15-18
5	7-11-2024	Inheritance demonstration program	19-32
6	14-11-2024	Packages in java demonstration	33-39
7	21-11-2024	Exception handling	40-45
8	28-11-2024	Multithreaded Programming	46-49
9	19-12-2024	Open ended exercise-1: Event Handling	50-56
10	19-12-2024	Open ended exercise-2: IPC and Deadlock	57-66

LABORATORY PROGRAM - 1

1) Develop a Java program that prints all real solutions to the quadratic equation ax2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions.



1) Implement Quadratic equation, print all real solutions of eqn $ax^2+bx+c=0$, Read a, b, c and use quadratic formula

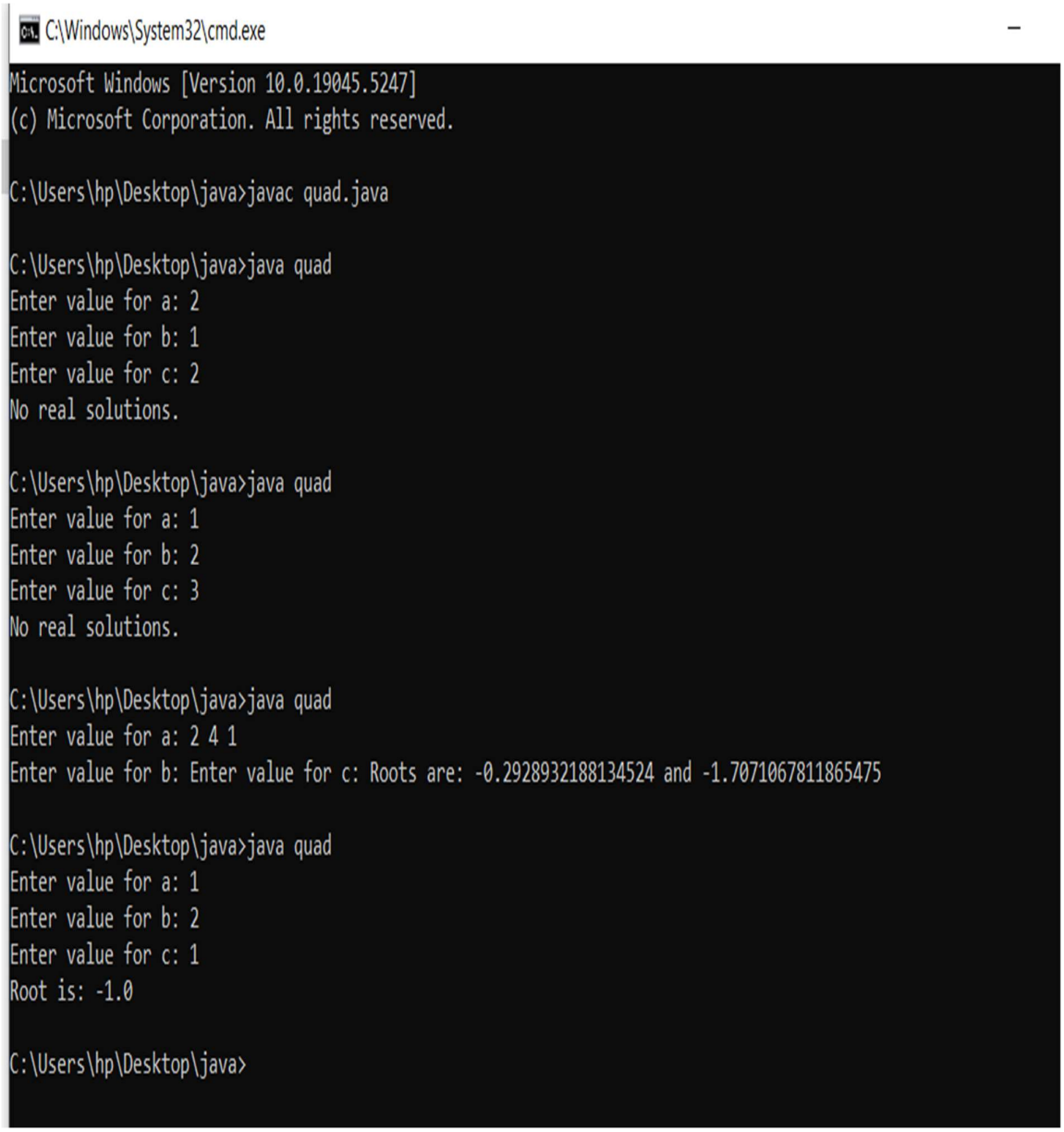
Date
Page

```
import java.util.Scanner

class quadratic
float d;
Scanner sc = new Scanner (System.in);
void check()
{ System.out.println ("Enter values a b c");
 int a = sc.nextInt();
 int b = sc.nextInt();
 int c = sc.nextInt();
 if (a == 0){
System.out.println("Invalid");
 else {
   d = b*b - 4*a*c;
 System.out.println (d);
 System.out.println ("The solutions are");
 if (d>0){
System.out.println(d) roots are unique
// System.out.println(
 double r1 = (-b+ Math.sqrt(d))/(2*a);
  double r2 = (-b - Math.sqrt(d))/(2*a);
System.out.println(r1 + " " + r2);
}
  if (d == 0){
System.out.println ("roots are equal");
 double r = -b/(2*a);
 System.out.println (r);
}
 if (d < 0) {
system.out.println ("Roots are imaginary");
double r1 = Math.sqrt(-d) /(2*a);
double r2 = (-b) /2*a);
Syste
```

System.out.println("x1 = " + x1 + " and x2 = " + x2);
}
}
}
public class main{
public static void main(String args[])
{ quadratic q1 = new quadratic()}
q1.check();
}
}

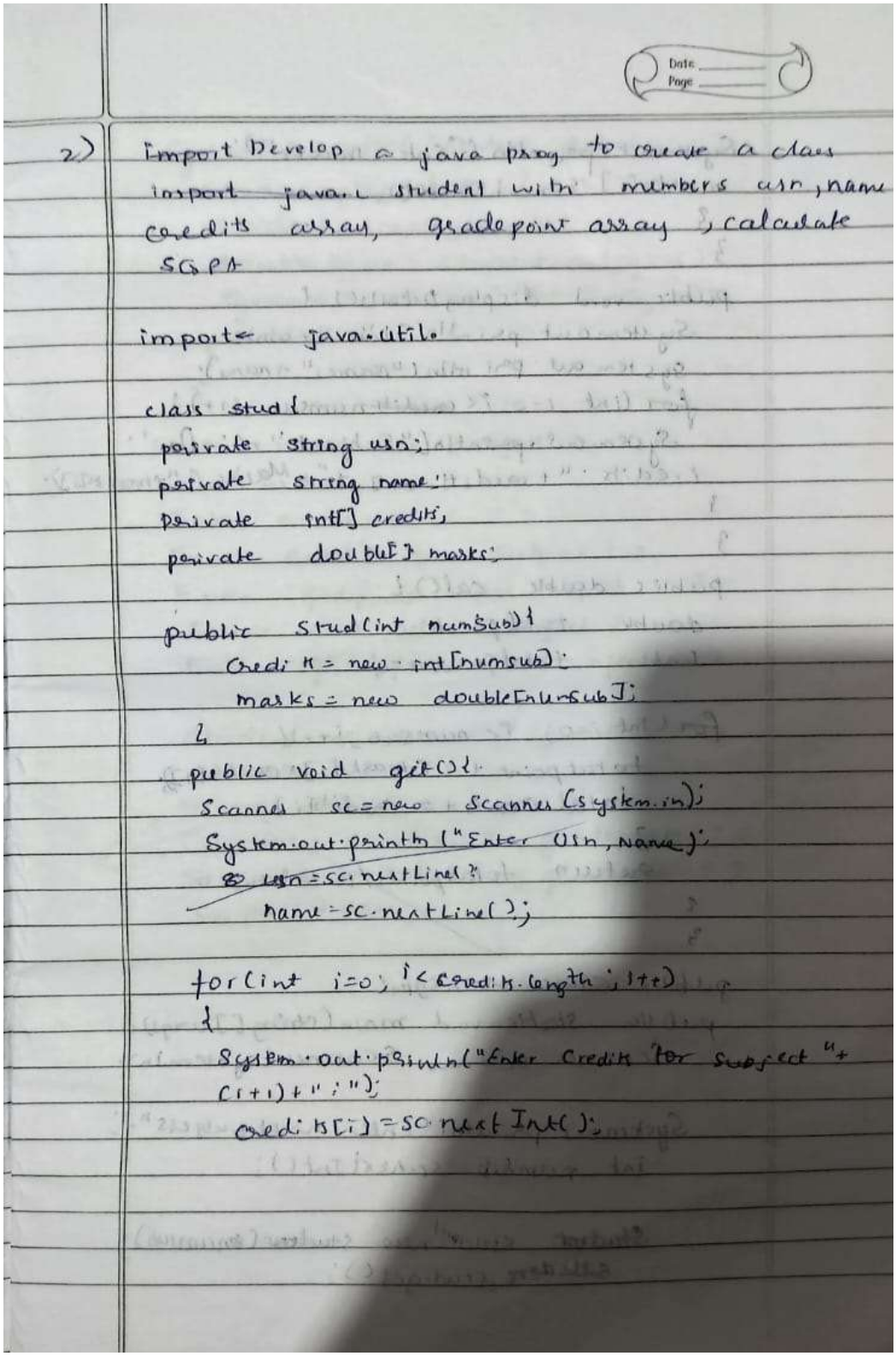
Enter values of a, b, c
3 8 1
d = 52.0
the solutions are
roots are unique
-0.1314129 -2.535183

```
import java.util.Scanner;

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

      System.out.print("Enter value for a: ");
      double a = sc.nextDouble();
      System.out.print("Enter value for b: ");
      double b = sc.nextDouble();
      System.out.print("Enter value for 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: " + root1 + " and " + root2);
      } else if (discriminant == 0) {
         double root = -b / (2 * a);
         System.out.println("Root is: " + root);
      } else {
         System.out.println("No real solutions.");
      }
   }}
```


C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.
C:\Users\hp\Desktop\java>javac quad.java
C:\Users\hp\Desktop\java>java quad
Enter value for a: 2
Enter value for b: 1
Enter value for c: 2
No real solutions.
C:\Users\hp\Desktop\java>java quad
Enter value for a: 1
Enter value for b: 2
Enter value for c: 3
No real solutions.
C:\Users\hp\Desktop\java>java quad
Enter value for a: 2 4 1
Enter value for b: Enter value for c: Roots are: -0.2928932188134524 and -1.7071067811865475
C:\Users\hp\Desktop\java>java quad
Enter value for a: 1
Enter value for b: 2
Enter value for c: 1
Root is: -1.0
C:\Users\hp\Desktop\java>

LABORATORY 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.



```
2) import Develop a java prog to create a class
   import java.i student with members usn, name
   credits array, gradepoint array, calculate
   SGPA

import java.util.

class stud{
 private string usn;
 private string name;
 private int[] credits;
 private double[] marks;

 public Stud(int numSub){
     Credits = new int[numsub];
     marks = new double[numSub];
 }
 public void get(){
  Scanner sc = new Scanner(System.in);
  System.out.println("Enter Usn, Name");
  usn=sc.nextLine();
     name=sc.nextLine();

  for(int i=0; i<Credits.length; i++)
  {
   System.out.println("Enter Credits for Subject "+
   (i+1)+" :");
     Credits[i]=sc.nextInt();
```

```
System.out.println("Enter marks");
marks[i]=sc.nextDouble();
}
}

public void displayDetails() {
System.out.println("USN:"+usn);
System.out.println("Name:"+name);
for(int i=0;i<numsub;i++){
System.out.println("Subject"+(i+1)+"
Credits:"+credits[i]+", Marks:"+marks[i]);
}
}

public double cal(){
double totalpoints=0;
int totalcredits=0;

for(int i=0; i<numsub;i++){
totalpoint+=(marks[i]*credits[i]);
totalcredits+=credits[i];
}
return totalpoints/totalcredits;
}
}

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

System.out.println("Enter no of subjects");
int numsub=sc.nextInt();

Student stud=new Student(numsub);
stud.get();
```

```
System.out.println("Student details:");
Stud.displaydetails();
Student.displaydetails();
double sgpa = Student.calsgpa();
System.out.println("SGPA" +sgpa);
sc.close();
}
}
```

Enter number of subjects 2
Enter usn: 23
Enter name: ved
Enter Credits for subject 1: 2
Enter gradepoint for subject 1: 9
Enter credits for subject 2: 4
Enter gradepoint for subject 2: 7

Student details:
USN: 234
Name: ved
Subject 1 - Credit: 2, gradepoint: 9
Subject 2 - Credits: 4, gradepoint: 7
SGPA: 7.666

```
import java.util.Scanner;

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

   public Student(int numSubjects) {
      credits = new int[numSubjects];
      marks = new double[numSubjects];
   }

      public void acceptDetails() {
      Scanner sc = new Scanner(System.in);

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

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

      for (int i = 0; i < credits.length; i++) {
         System.out.print("Enter credits for subject " + (i + 1) + ": ");
         credits[i] = sc.nextInt();

         System.out.print("Enter gradepoint for subject " + (i + 1) + ": ");
         marks[i] = sc.nextDouble();
      }
   }

     public void displayDetails() {
      System.out.println("USN: " + usn);
      System.out.println("Name: " + name);
      for (int i = 0; i < credits.length; i++) {
         System.out.println("Subject " + (i + 1) + " - Credits: " + credits[i] + ",
gradepoints: " + marks[i]);
      }
   }

    public double calculatesgpa() {
      double totalPoints = 0;
      int totalCredits = 0;
```

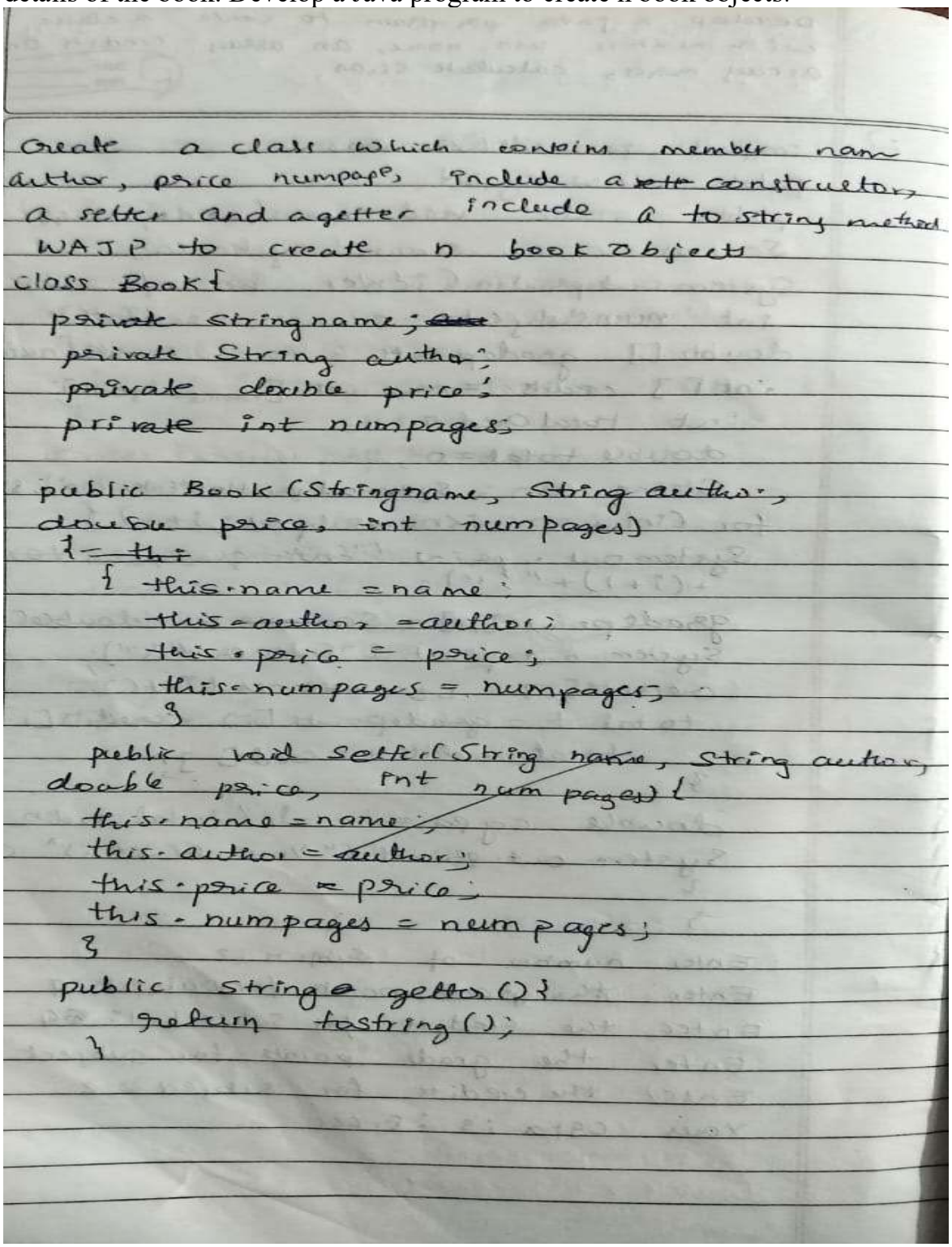
```
        for (int i = 0; i < credits.length; i++) {
            totalPoints += (marks[i] * credits[i]);
            totalCredits += credits[i];
        }

        return  totalPoints / totalCredits;
 //totalCredits == 0 ? 0 : totalPoints / totalCredits;
    }
}

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

        System.out.print("Enter the number of subjects: ");
        int numSubjects = sc.nextInt();

        Student student = new Student(numSubjects);
        student.acceptDetails();

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

        double sgpa = student.calculatesgpa();
        System.out.println("SGPA: " + sgpa);

        sc.close();
    }
}
```

output

```
C:\Users\hp\Desktop\java>java Mainsgpa
Enter the number of subjects: 3
Enter USN: 348
Enter Name: ritik
Enter credits for subject 1: 6
Enter gradepoint for subject 1: 8
Enter credits for subject 2: 2
Enter gradepoint for subject 2: 10
Enter credits for subject 3: 4
Enter gradepoint for subject 3: 5

Student Details:
USN: 348
Name: ritik
Subject 1 - Credits: 6, gradepoints: 8.0
Subject 2 - Credits: 2, gradepoints: 10.0
Subject 3 - Credits: 4, gradepoints: 5.0
SGPA: 7.333333333333333

C:\Users\hp\Desktop\java>
```

LABORATORY PROGRAM - 3

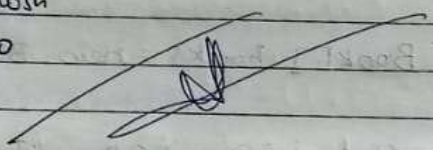
3) Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program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



```
Create a class which contains member name
author, price numpages, include a constructor,
a setter and a getter include a to string method
WAJP to create n book objects
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 setter(String name, String author,
double price, int numpages){
  this.name = name;
  this.author = author;
  this.price = price;
  this.numpages = numpages;
  }
public String getter(){
    return tostring();
  }
```

```
public String toString()
return " Book Name: " + name + " , Author: " +
author + " , price : Rs " + price + " , Pages - " +
numpages;
}
}
public class Bookmain
public static void main (String [] args)
Scanner sx = new Scanner (System.in);
System.out.println(" Enter no of books");
int n = sx.nextInt();

Book[] books = new Book[n];

for(int i=0; i<n; i++)
System.out.println(" Enter details of book " + (i+1))
System.out.println(" Enter Name, author, price,
no of pages");
String name = sx.nextLine();
String author = sx.nextline();
double price = sx.nextDouble();
int numpages = sx.nextInt();

books[i] = new Book( name, author, price,
numpages);
System.out.println (books[i].getter());
}
sx.close();
}
}
```

Enter the no of books
2
Enter details for book 1:
Name: Sujan
Author: ge
price : 3
Number of pages : 12
Book name: Sujan, Author: ge, Price: 33.0, pages: 12
Enter details for book 2
Name: dhasma
Author: Sughosh
price : 240



```
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 setDetails(String name, String author, double price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    public String getDetails() {
        return toString();
    }


    public String toString() {
        return "Book Name: " + name + ", Author: " + author + ", Price: rs" + price + ", Pages: " + numPages;
    }
}

public class Bookmain {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of books: ");
        int n = scanner.nextInt();
        scanner.nextLine();

        Book[] books = new Book[n];

        for (int i = 0; i < n; i++) {
            System.out.println("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[i] = new Book(name, author, price, numPages);
            System.out.println(books[i].getDetails());
        }
```

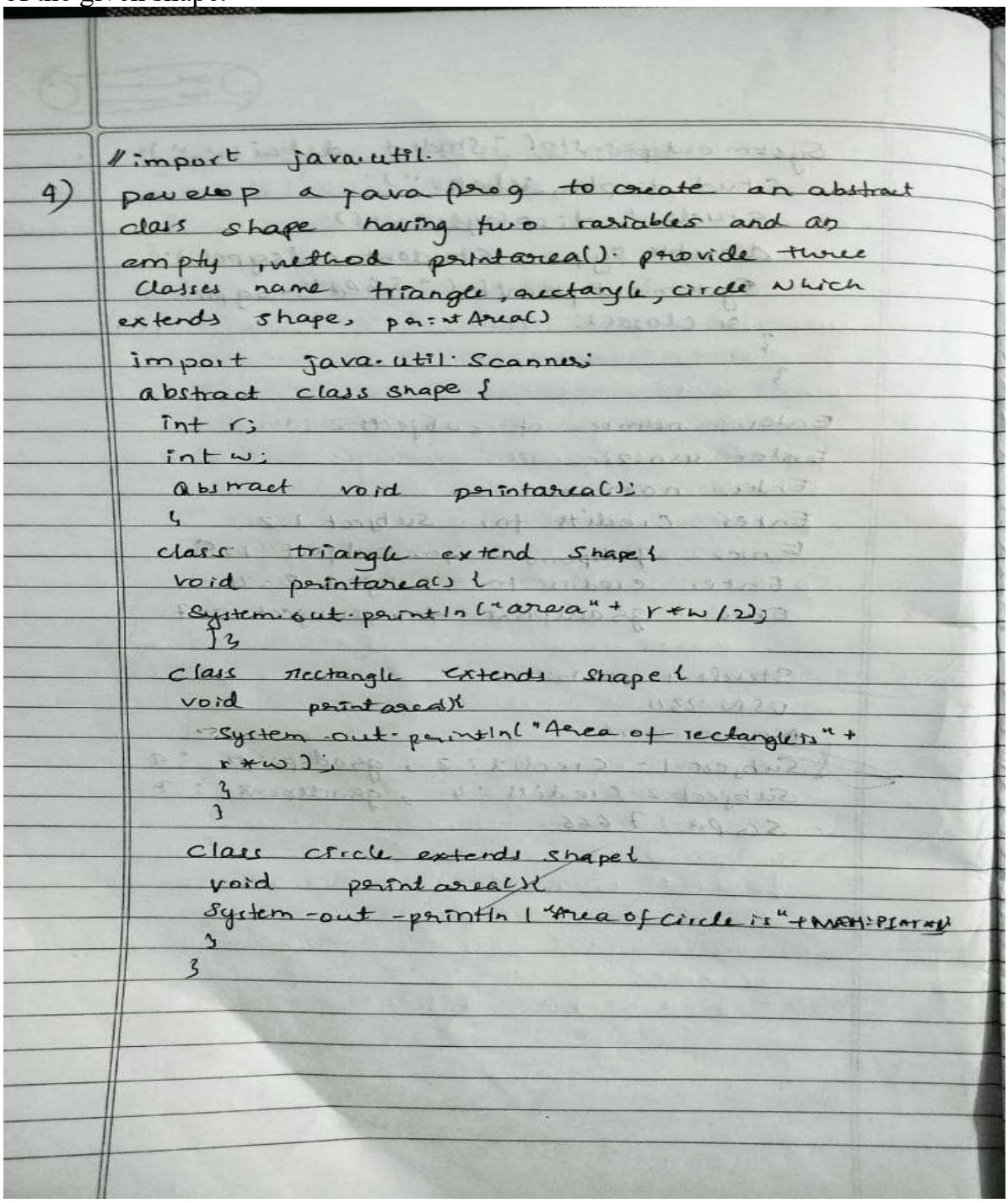
```
        scanner.close();
    }
}
```

Output

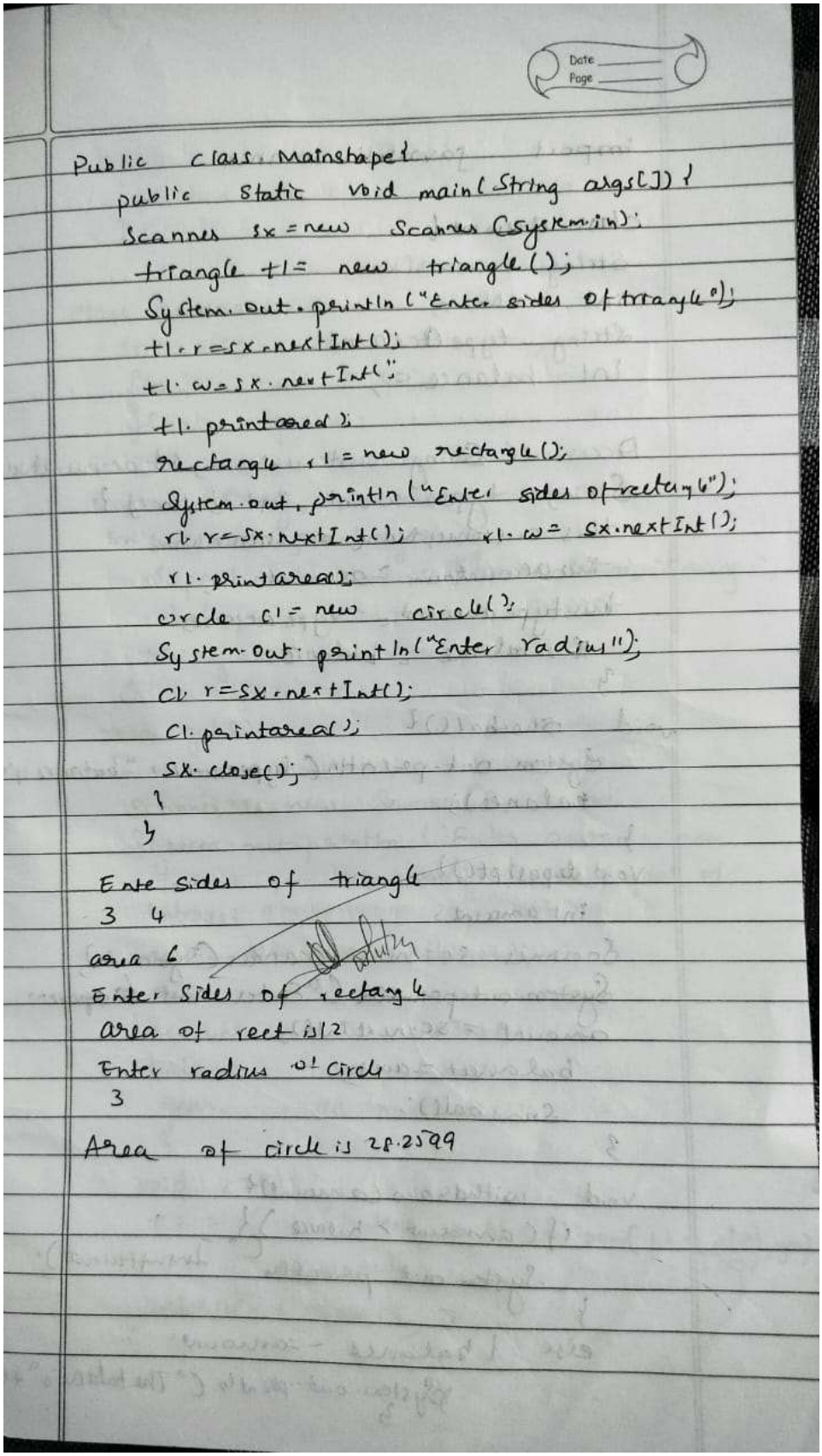
```
C:\Users\hp\Desktop\java>javac Bookmain.java

C:\Users\hp\Desktop\java>java Bookmain
Enter number of books: 2
Enter details for book 1:
Name: Atomoic Habits
Author: James Clear
Price: 349
Number of Pages: 208
Book Name: Atomoic Habits, Author: James Clear, Price: rs349.0, Pages: 208
Enter details for book 2:
Name: The Alchemist
Author: Paulo Coelho
Price: 262
Number of Pages: 409
Book Name: The Alchemist, Author: Paulo Coelho, Price: rs262.0, Pages: 409

C:\Users\hp\Desktop\java>
```

LABORATORY 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.
4) develop a java prog to create an abstract
class shape having two variables and an
empty method printarea(). provide three
classes name triangle, rectangle, circle which
extends shape, pass Area()
import java.util.Scanner;
abstract class shape {
 int r;
 int w;
 Abstract void printarea();
}
class triangle extend Shape {
 void printarea() {
  System.out.println("area" + r*w/2);
 }}
class rectangle extends shape {
 void printarea(){
  System.out.println("Area of rectangle is" +
  r*w);
 }
}
class circle extends shape{
 void printarea(){
 System.out.println("Area of circle is" +MATH.PI*r*r)
 }
}
```



```
Public class Mainshape{
  public static void main(String args[]){
  Scanner sx = new Scanner(System.in);
  triangle t1= new triangle();
  System.out.println("Enter sides of triangle");
  t1.r=sx.nextInt();
  t1.w=sx.nextInt();
  t1.printarea();
  rectangle r1 = new rectangle();
  System.out.println("Enter sides of rectangle");
  r1.r=sx.nextInt();   r1.w= sx.nextInt();
  r1.printarea();
  circle c1= new circle();
  System.out.println("Enter radius");
  c1.r=sx.nextInt();
  c1.printarea();
  sx.close();
  }
}
```

```
Ente sides of triangle
3 4
area 6
Enter sides of rectangle
area of rect is12
Enter radius of circle
3
Area of circle is 28.2599
```

```
import java.util.Scanner;
abstract class Shape{
   int r; int w;
   abstract void printarea();
}
class triangle extends  Shape{

   void printarea(){
      System.out.println("AREA OF TRIANGLE ="+ r*w/2);
   }}

class rectangle extends Shape{
void printarea(){
   System.out.println("AREA OF RECTANGLE =" + r*w);
}}
class circle extends Shape{
   void printarea(){
      System.out.println("AREA OF CIRCLE IS = "+ 3.14*r*r);
   }

}
public class mainshape{
   public static void main(String args[]){
      Scanner sx=new Scanner(System.in);
      triangle t1= new triangle();
      System.out.println("ENTER HEIGHT AND BASE  OF TRIANGLE");
      t1.r=sx.nextInt(); t1.w=sx.nextInt();
      t1.printarea();
      rectangle r1=new rectangle();
      System.out.println("ENTER SIDES OF RECTANGLE IE LENGTH
BREADTH");
      r1.r=sx.nextInt(); r1.w=sx.nextInt();

      r1.printarea();

      circle c1=new circle();
      System.out.println("ENTER RADIUS OF CIRCLE");
      c1.r=sx.nextInt(); //r1.w=sx.nextInt();
      c1.printarea();
      sx.close();


   }
```

}
output

```
C:\Users\hp\Desktop\java>javac mainshape.java

C:\Users\hp\Desktop\java>java mainshape
ENTER HEIGHT AND BASE  OF TRIANGLE
3
4
AREA OF TRIANGLE =6
ENTER SIDES OF RECTANGLE IE LENGTH BREADTH
3 4
AREA OF RECTANGLE =12
ENTER RADIUS OF CIRCLE
4
AREA OF CIRCLE IS = 50.24

C:\Users\hp\Desktop\java>
```

LABORATORY PROGRAM - 5

5)Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks: a) Accept deposit from customer and update the balance. b) Display the balance. c) Compute and deposit interest d) Permit withdrawal and update the balance Check for the minimum balance, impose penalty if necessary and update the balance

```
import java.util.Scanner;

class Account {
 String customerName;
 int accNum;
 String typeAccount;
 int balance=0;

 Account(String customerName, int accountNum,
 String typeaccount, int balance) {
  this.customerName = customerName;
  this.accountnum = accountnum;
  this.typeaccount = typeaccount;
  this.balance = balance;
 }
void showbal(){
   System.out.println(typeaccount + "balance is" +
   balance);
 }
 void deposipt(){
   int amount;
  Scanner sc= new Scanner(System.in);
  System.out.println("Enter amt to deposit");
  amount = sc.nextInt();
   balanceet =amt;
   showbal();
 }
 void withdraw(amount){
   if(amount > balance ){
    System.out.println("Insufficient");
  }
  else { balance -=amount;
    System.out.println("The balance is" +balance);
  }
```

```
boolean gscheque(){
    return fed x;
}
}
class curracc extends account{
  int minbal=50000;
curracc(String customerName, int accountnum,
int balance){
  super(customerName, accountnum, "current",
    balance);
}
void get(){
  Scanner sc=new Scanner(System.in);
  System.out.println("Enter customername,
account number, and balance for current
account:");
  customerName=sc.nextLine();
  accountnum=sc.nextInt();
}
void check(){
  if(balance<minbal)
    System.out.println("Low balance");
    balance-=500;
}
}

void withdraw(int amount){
  if(amount>balance){
  System.out.println("Insufficient");
}
else{ balance-=amount;
    System.out.print("The balance:"+balance);
    check();
}
}
```

```
boolean isCheque(){
    return false;
}

class saveacc extends account {
    int CI;
    floate rate = 7.0f;
    float years;

saveacc(String customerName, int accountnum,
int balance, float years){
    super(customername, int accountnum, "Savings",
      balance);
    this.year = years;
}
void get()
{
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter customer name,
    account number, balance and year of
    interest:");
    customerName = sc.nextLine();
    accountnum = sc.nextInt();
    balance = sc.nextInt();
    years = sc.nextFloat();
}
void CI(){
    CI = (int)(balance * Math.pow(1 + rate/100, years)
        - balance);
    balance += CI;
}
```

```
void ischeque()
{ System.out.println("Cheque available");
}

boolean ischeque() {
  return true
    }

}
class Banknew{
public static void main(String [] args){
  Scanner sc = new Scanner(System.in);
   int amt) char ch, sch;
   boolean exit= false;

  While( ! exit){
System.out.println\n" Enter account type:
 1. Savings 2. Current (0 to exit): ");
  ch = sc.nextInt();
    if (ch == 0){
     exit = true;
     continue;
     }
   switch(ch){
    case 1:
          Saveacc s = new Saveacc("",0,0);
           s.get();
           boolean Saving exit = false;
          While (! Savingexit)
          {
```

withdraw 3. Show 4. Check cheque 5. exit ");
int sch = sc.nextInt();
switch(sch)
{
case 1: c.deposit();
break;
case 2: System.out.println(" Enter amount to withdraw ");
amt = sc.nextInt();
c.withdraw(amt);
break;
case 3:
c.Showbal();
break;
case 4: if(c.ischeque())
{ System.out.println(" Available ");
case 5: case exit = true;
break;
default: System.out.println(" Invalid");
}
}
break;
default: System.out.println(" Invalid");
}
}
}

```
System.out.println("Enter choice: 1.deposit 2.
withdraw 3. show balance 4. compound Interest
5. check facility 6.Exit");
sch = sc.nextInt();

switch(sch){
case 1: s.deposit();
        break

case 2: System.out.println("Enter amt");
        amt = sc.nextInt();
        s.withdraw(amt);
        break;
case 3: showbal();
        break
case 4: s.CI();
        break;
case 5 : if (s.isCheque())
            System.out.println("Cheque available");
        break;
case 6 : savingsexit = true;
        break;
default: System.out.println("Invalid choice");
}
break;

case 2: CurrAcc c = new CurrAcc("...",0,0);
        c.get();
        boolean currexit = false;
        while(!currexit){
System.out.println("Enter choice 1.deposit 2.
```

Enter account type 1.savings 2.current 0 to exit:
1
Enter customers name account number, balance and years for interest
Madhu
34000
2
Enter Choice 1. deposit 2. withdraw 3. show balance
4. Compound Interest
1
Enter amount to deposit
2300
Savings balance is : 36300
Enter choice 1. deposit 2. withdraw 3. show balance
4. compound Interest
4
compound interest : 1254
Balance after compound interest 41554
Enter choice : 1. Deposit 2. withdraw 3. show balance
4. compound interest
2
Enter amount to withdraw
50000
Insufficient balance

14/11/24

```
import java.util.Scanner;

class Account {
   String customerName;
   int accountNum;
   String typeAccount;
   int balance = 0;

   Account(String customerName, int accountNum, String typeAccount, int balance) {
      this.customerName = customerName;
      this.accountNum = accountNum;
      this.typeAccount = typeAccount;
      this.balance = balance;
   }

   void showBalance() {
      System.out.println(typeAccount + " balance is: " + balance);
   }

   void deposit() {
      int amount;
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter amount to deposit: ");
      amount = sc.nextInt();
      balance += amount;
      showBalance();
   }

   void withdraw(int amount) {
      if (amount > balance) {
         System.out.println("Insufficient balance.");
      } else {
         balance -= amount;
         System.out.println("The balance is: " + balance);
      }
   }
}

class SaveAcc extends Account {
   int compoundInterest;
   float rate = 7.0f; // 7% interest
   float year;

   SaveAcc(String customerName, int accountNum, int balance, float year) {
```

```
        super(customerName, accountNum, "Savings", balance);
        this.year = year;
    }

    void get() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter customer name, account number, balance, and years for
interest: ");
        customerName = sc.nextLine();
        accountNum = sc.nextInt();
        balance = sc.nextInt();
        year = sc.nextFloat();
    }

    void compoundInterest() {
        compoundInterest = (int)(balance * Math.pow((1 + rate / 100), year) - balance);
        balance += compoundInterest;
        System.out.println("Compound interest: " + compoundInterest);
        System.out.println("Balance after compound interest: " + balance);
    }
}

class CurrAcc extends Account {
    int minBalance = 50000;

    CurrAcc(String customerName, int accountNum, int balance) {
        super(customerName, accountNum, "Current", balance);
    }

    void get() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter customer name, account number, and balance for
current account: ");
        customerName = sc.nextLine();
        accountNum = sc.nextInt();
        balance = sc.nextInt();
    }

    void check() {
        if (balance < minBalance) {
            System.out.println("Low balance. Service charge of 500 imposed.");
            balance -= 500;
        }
    }
```

```
    void withdraw(int amount) {
        if (amount > balance) {
            System.out.println("Insufficient balance.");
        } else {
            balance -= amount;
            System.out.println("The balance is: " + balance);
            check();
        }
    }

    void isCheque() {
        System.out.println("Cheque facility is available.");
    }
}

public class newBank {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int amount, Ch, c1;

        System.out.println("Enter account type 1. Savings 2. Current");
        Ch = sc.nextInt();

        while(true) {
            if (Ch == 1) {
                SaveAcc s = new SaveAcc("", 0, 0, 0);
                s.get();
                System.out.println("Enter choice 1. Deposit 2. Withdraw 3. Show balance 4.
Compound interest");
                c1 = sc.nextInt();
                if (c1 == 1) {
                    s.deposit();
                } else if (c1 == 4) {
                    s.compoundInterest();
                } else if (c1 == 2) {
                    System.out.println("Enter amount to withdraw from savings account: ");
                    amount = sc.nextInt();
                    s.withdraw(amount);
                } else if (c1 == 3) {
                    s.showBalance();
                } else {
                    System.out.println("Invalid choice");
                }
```

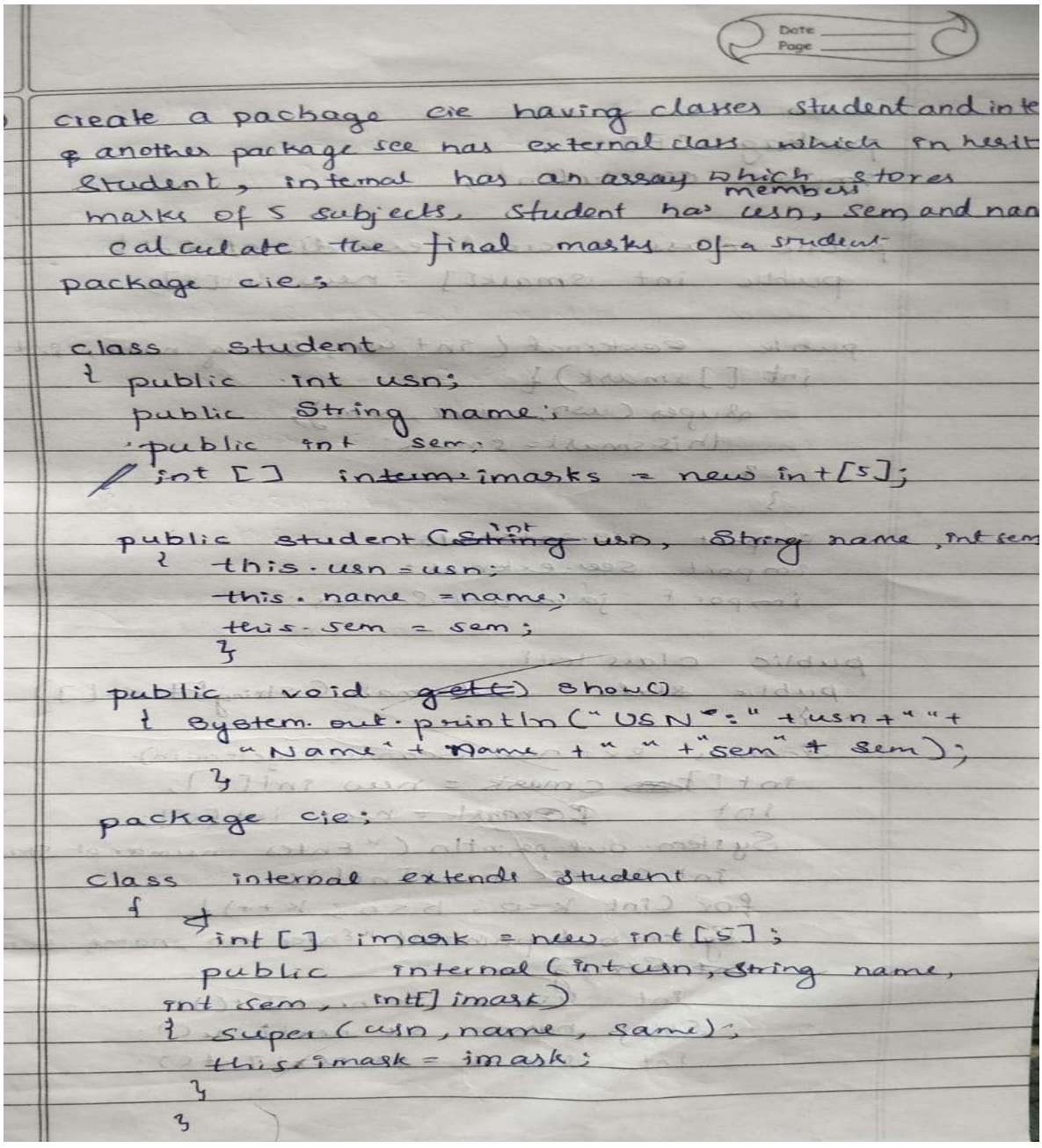
```
        } else if (Ch == 2) {
            CurrAcc c = new CurrAcc("", 0, 0);
            c.get();
            System.out.println("Enter choice 1. Deposit 2. Withdraw 3. Show balance");
            c1 = sc.nextInt();
            if (c1 == 1) {
                c.deposit();
            } else if (c1 == 2) {
                System.out.println("Enter amount to withdraw from current account: ");
                amount = sc.nextInt();
                c.withdraw(amount);
            } else if (c1 == 3) {
                c.showBalance();
            } else {
                System.out.println("Invalid choice");
            }
        } else {
            System.out.println("Invalid choice");
        }

        System.out.println("Do you want to continue? (1. Yes / 0 or any chracter. No)");
        int continueChoice = sc.nextInt();
        if (continueChoice !=1) {
            break;
        }

    }

    sc.close();
  }
}
```

```
Enter account type: 1. Savings 2. Current (0 to exit):
1
Enter customer name, account number, balance, and years for interest:
madhu
23
34000
2
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Compound interest 5. Check Cheque facility 6. Exit
3
Savings balance is: 34000
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Compound interest 5. Check Cheque facility 6. Exit
1
Enter amount to deposit:
2300
Savings balance is: 36300
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Compound interest 5. Check Cheque facility 6. Exit
4
Compound interest: 5259
Balance after compound interest: 41559
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Compound interest 5. Check Cheque facility 6. Exit
2
Enter amount to withdraw from savings account:
50000
Insufficient balance.
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Compound interest 5. Check Cheque facility 6. Exit
2
Enter amount to withdraw from savings account:
23000
The balance is: 18559
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Compound interest 5. Check Cheque facility 6. Exit
5
Cheque facility is not available for Savings account.
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Compound interest 5. Check Cheque facility 6. Exit
6
Enter account type: 1. Savings 2. Current (0 to exit):
2
Enter customer name, account number, and balance for current account:
vasisshta
345
120000
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Check Cheque facility 5. Exit
1
Enter amount to deposit:
24567
Current balance is: 144567
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Check Cheque facility 5. Exit
2
Enter amount to withdraw from current account:
90000
The balance is: 54567
```

```
The balance is: 24567
Low balance. Service charge of 500 imposed.
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Check Cheque facility 5. Exit
3
Current balance is: 24067
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Check Cheque facility 5. Exit
4
Cheque facility is available.
Enter choice: 1. Deposit 2. Withdraw 3. Show balance 4. Check Cheque facility 5. Exit
```

LABORATORY PROGRAM - 6

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



```
create a package cie having classes student and inte
& another package see has external class which in herit
student, internal has an array which stores
marks of 5 subjects. student has usn, sem and nam
calculate the final masks of a student
package cie;

class student
{ public int usn;
  public String name;
  public int sem;
  int [] interm; imarks = new int[5];

  public student (String usn, String name, int sem
    { this.usn = usn;
      this.name = name;
      this.sem = sem;
    }

public void getS() show()
  { System.out.println("USN":" + usn + " " +
    "Name" + name + " " + "sem" + sem);
    }

package cie;

Class internal extends student
  {
    int [] imark = new int[5];
    public internal (int usn, String name,
  int sem, int[] imark)
  { super(usn, name, same);
    this.imark = imark;
    }
  }
```

```
package SEE;
import cie.student;

public class external extends student{
  public int smark[] = new int[5];

  public external( int usn, String name, int sem
    int []smark){
      super(usn, name, sem);
       this.smark= smark;
    }
  }

  import cie.internals;
  import see.external;
  import java.util.Scanner;

public class test{
  public static void main(String xx[])
  {
  Scanner sx= new Scanner(System.in);
    int [] cmark = new int[5];
    int [] emark = new int[5];
    System.out.println("Enter number of Students");
      int n= sc.nextInt();
    for(int k=0; k<n; k++){
    System.out.println("Enter usn, name, sem");
      String
        int usn = sc.nextInt();
      String name = sx.nextLine();
        int sem = sx.nextInt();
```

```
System.out.println("enter 5 subjects mark for interna
for(int i=0; i<5 ;i++)
{ cmask[i]= nextInt();
}
System.out.println ("enter see marks of 5 subjects");
for(int i=0; i<5 ;i++)
{ emask[i]=nextInt();
}
internal il= new internal(usn, name, sem,
cmask);
external el= new external(usn, name, sem,
emask);
System.out.println("details"); el.show();
for(int i=0; i<=4; i++){ student s=new student()
System.out.println("Total marks of student");
// el.show();
System.out.println(il.imask[i] + el.smask[i]/2);
}
}}
```

O/P

```
Enter no of students
1
Enter usn, name, sem
23
Bujay
3
enter 5 subject marks for internal
38
37
30
34
23
```

Enter see marks of 5 subjects

78

89

96

98

60

details

USN:23 name: ujay sem:3

Total mark in subject

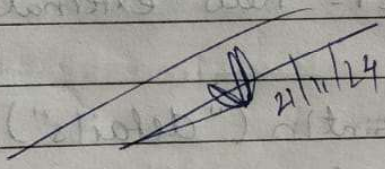
77

81

78

83

63



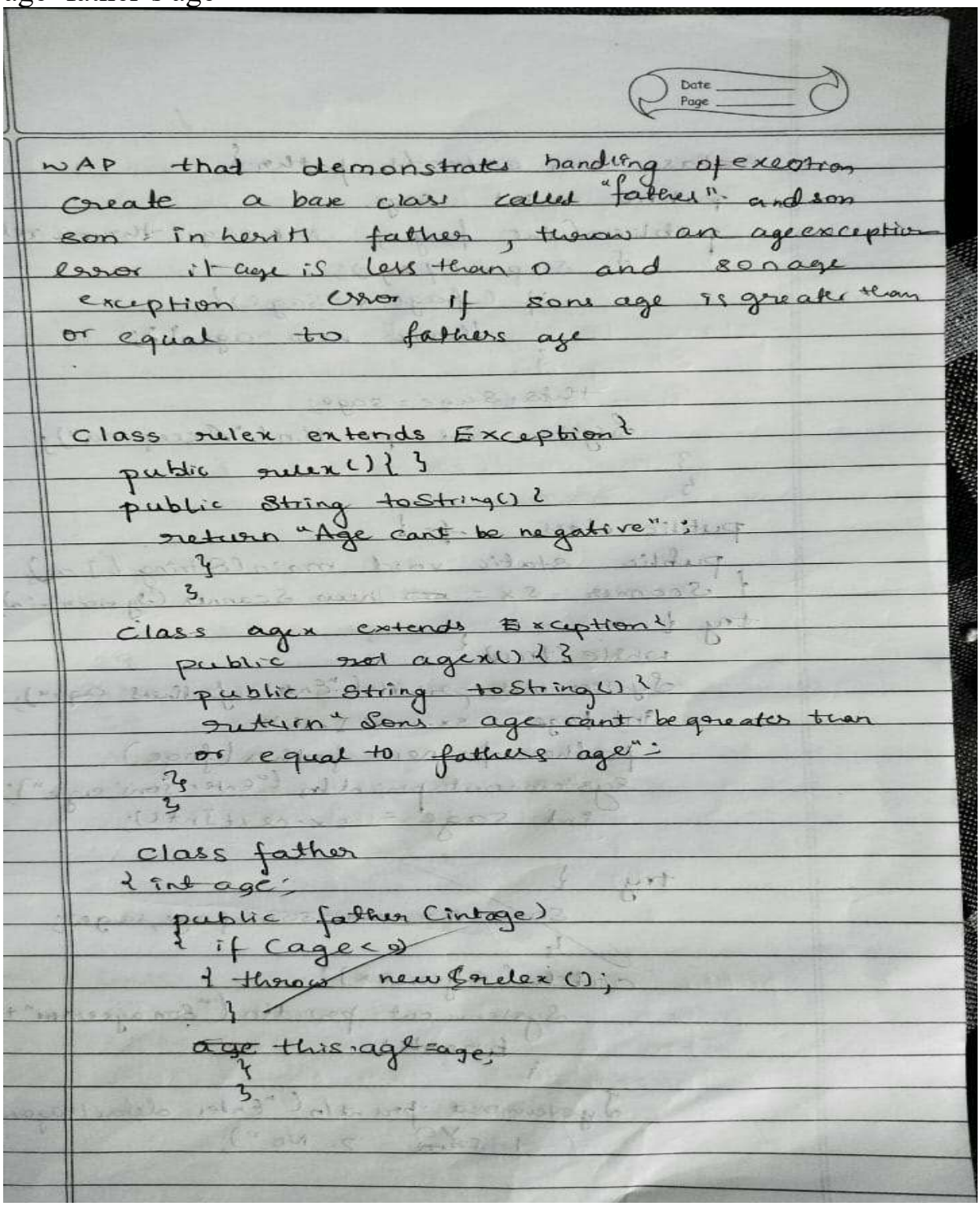
```
package CIE;

public class Student {
    public String usn;
    public String name;
    public int sem;
}

package CIE;

public class Internals extends Student {
    public int[] internalMarks = new int[5];
}
package SEE;
import CIE.Student;

public class External extends Student {
    public int[] seeMarks = new int[5];
}
import CIE.Internals;
import SEE.External;
import java.util.Scanner;

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

        Internals[] internalStudents = new Internals[n];
        External[] externalStudents = new External[n];

        for (int i = 0; i < n; i++) {
            internalStudents[i] = new Internals();
            externalStudents[i] = new External();

            System.out.println("Enter details for student " + (i + 1));
            System.out.print("USN: ");
            internalStudents[i].usn = sx.next();
            externalStudents[i].usn = internalStudents[i].usn;

            System.out.print("Name: ");
            internalStudents[i].name = sx.next();
```

```
            externalStudents[i].name = internalStudents[i].name;

            System.out.print("Semester: ");
            internalStudents[i].sem = sx.nextInt();
            externalStudents[i].sem = internalStudents[i].sem;

            System.out.println("Enter internal marks (max 50) for 5 subjects:");
            for (int j = 0; j < 5; j++) {
                int internalMark;
                do {
                    System.out.print("Subject " + (j + 1) + ": ");
                    internalMark = sx.nextInt();
                    if (internalMark < 0 || internalMark > 50) {
                        System.out.println("Invalid input! Marks should be between 0 and 50.");
                    }
                } while (internalMark < 0 || internalMark > 50);
                internalStudents[i].internalMarks[j] = internalMark;
            }

            System.out.println("Enter semester marks (SEE) for 5 subjects:");
            for (int j = 0; j < 5; j++) {
                System.out.print("Subject " + (j + 1) + ": ");
                externalStudents[i].seeMarks[j] = sx.nextInt();
            }
        }

        System.out.println("Final Marks of Students:");
        for (int i = 0; i < n; i++) {
            System.out.println("Student " + (i + 1) + " (" + internalStudents[i].usn + " - " +
internalStudents[i].name + ")");
            for (int j = 0; j < 5; j++) {
                // Calculate total marks as internal marks + semester marks / 2
                int totalMarks = internalStudents[i].internalMarks[j] +
(externalStudents[i].seeMarks[j] / 2);
                System.out.println("Subject " + (j + 1) + " Total Marks: " + totalMarks);
            }
        }
        sx.close();
    }
}
```

```
C:\Users\hp\Desktop\java>java FinalMarks
Enter number of students: 2
Enter details for student 1
USN: 12
Name: serine
Semester: 3
Enter internal marks (max 50) for 5 subjects:
Subject 1: 34
Subject 2: 45
Subject 3: 43
Subject 4: 34
Subject 5: 44
Enter semester marks (SEE) for 5 subjects:
Subject 1: 89
Subject 2: 90
Subject 3: 78
Subject 4: 66
Subject 5: 88
Enter details for student 2
USN: 13
Name: reser
Semester: 2
Enter internal marks (max 50) for 5 subjects:
Subject 1: 50
Subject 2: 43
Subject 3: 21
Subject 4: 34
Subject 5: 33
Enter semester marks (SEE) for 5 subjects:
Subject 1: 78
Subject 2: 55
Subject 3: 44
Subject 4: 34
Subject 5: 33
Final Marks of Students:
Student 1 (12 - serine)
Subject 1 Total Marks: 78
Subject 2 Total Marks: 90
Subject 3 Total Marks: 82
Subject 4 Total Marks: 67
Subject 5 Total Marks: 88
Student 2 (13 - reser)
Subject 1 Total Marks: 89
Subject 2 Total Marks: 70
Subject 3 Total Marks: 43
Subject 4 Total Marks: 51
Subject 5 Total Marks: 49
```

LABORATORY PROGRAM - 7

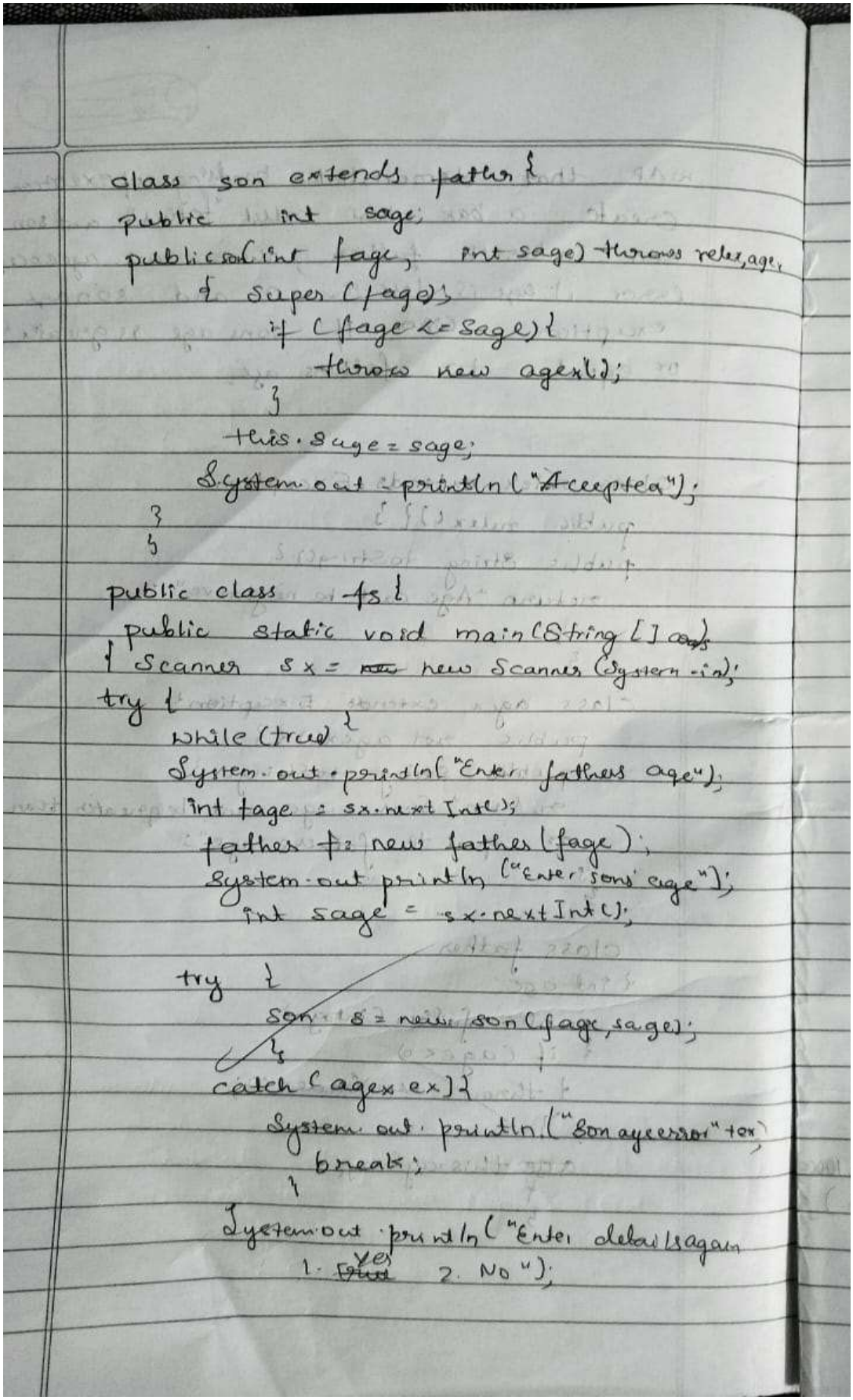
7)Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age=father’s age



Date
Page

WAP that demonstrates handling of exceptions
Create a base class called "father" and son
son inherit father, throw an age exception
error if age is less than 0 and son age
exception error if sons age is greater than
or equal to fathers age

```
Class relex extends Exception{
   public relex(){}}
   public String toString(){
     return "Age cant be negative";
     }
   }
Class agex extends Exception{
    public notagex(){}}
    public String toString(){
    return" Sons age cant be greater than
    or equal to fathers age";
  }
}
Class father
{ int age;
   public father(int age)
   { if(age<0)
     { throw new relex();
     }
     age this.age=age;
     }
   }
```



```
class son extends father {
Public int sage;
public son(int fage, int sage) throws reter,age;
    { super (fage);
        if (fage <= sage) {
            throw new agex();
        }
        this.sage = sage;
    System.out.println("Accepted");
}
}

public class fs {
 public static void main(String [] args
{ Scanner sx = new Scanner (System.in);
try {
    while (true) {
    System.out.println("Enter fathers age");
    int fage = sx.nextInt();
    father f = new father (fage);
    System.out.println ("Enter sons age");
     int sage = sx.nextInt();

try {
    son s = new son (fage, sage);
    }
    catch (agex ex) {
        System.out.println("Son age error" +ex);
        break;
    }
System.out.println("Enter detail again
    1. Yes  2. No");
```

```
if (choice != 1)
    break;
  }
}
catch (NegativeException ex) {
    System.out.println (ex);
}
finally {
    sx.close();
  }
}
```

O/P

Enter fathers age
29
Enter sons age
39
Sons age error sons age cant be greater than or equal to fathers age

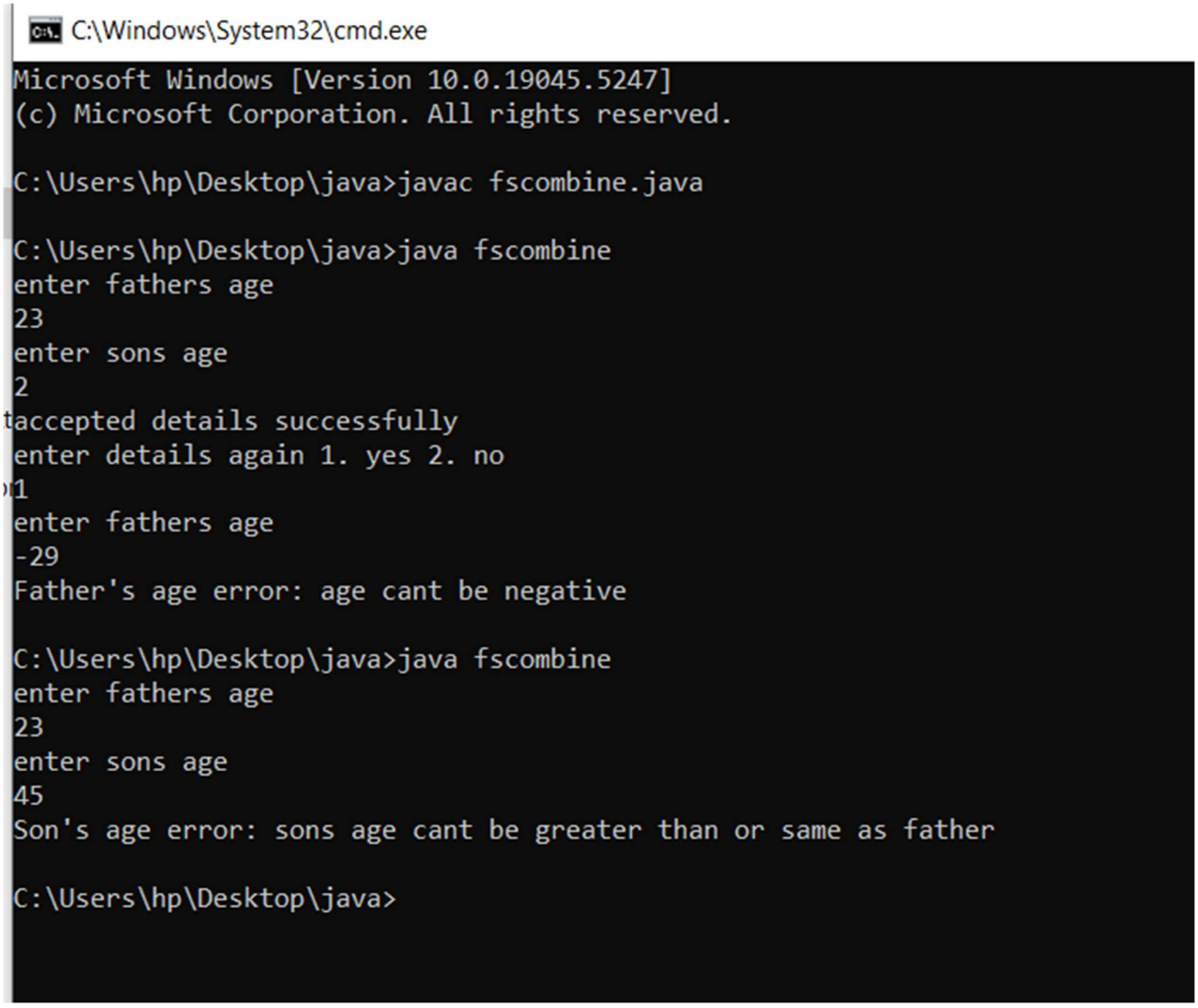
Enter fathers age
-30
Age cant be negative

Enter fathers age
50
Enter sons age
3
Accepted

```
import java.util.*;

class relex extends Exception {
    public relex() {}
    public String toString() {
        return "age cant be negative";
    }
}

class agex extends Exception {
    public agex() {}
    public String toString() {
        return "sons age cant be greater than or same as father";
    }
}

class father {
    public int age;

    public father(int age) throws relex {
        if (age < 0) {
            throw new relex();
        }
        this.age = age;
    }
}

class son extends father {
    public son(int fage, int sage) throws agex, relex {
        super(fage);
        if (fage <= sage) {
            throw new agex();
        }
        age = sage;
        System.out.println("accepted details successfully");
    }
}

class fscombine {
    public static void main(String xx[]) {
        Scanner sx = new Scanner(System.in);
        try {
            while (true) {
                System.out.println("enter fathers age");
```

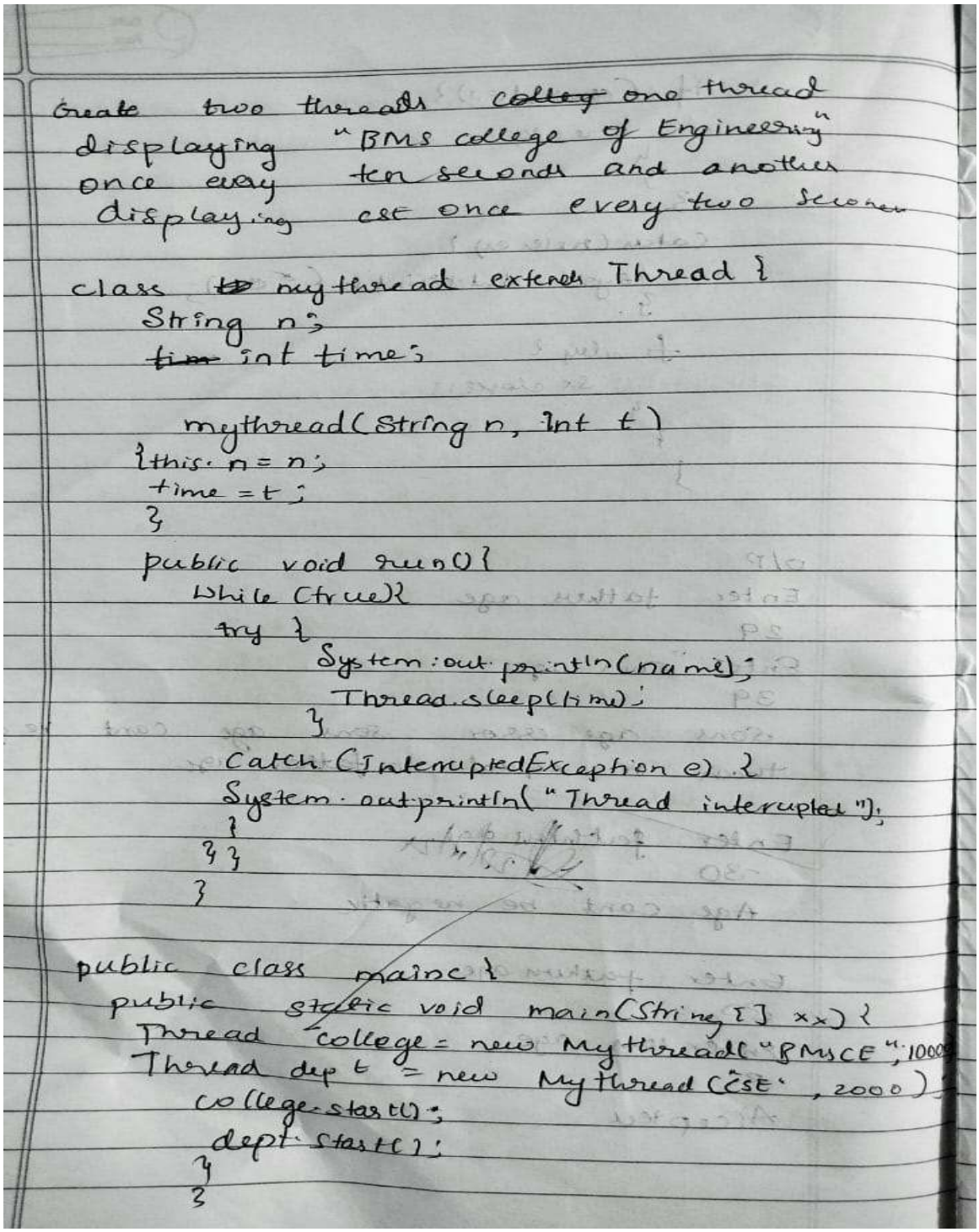
```
            int fage = sx.nextInt();
            father f = new father(fage);
            System.out.println("enter sons age");
            int sage = sx.nextInt();
            son s = new son(fage, sage);
            System.out.println("enter details again 1. yes 2. no");
            int ch = sx.nextInt();
            if (ch != 1) {
               break;
            }
         }
      } catch (relex ex) {
         System.out.println("Father's age error: " + ex);
      } catch (agex ex) {
         System.out.println("Son's age error: " + ex);
      }
      sx.close();
   }
}
```

Output:

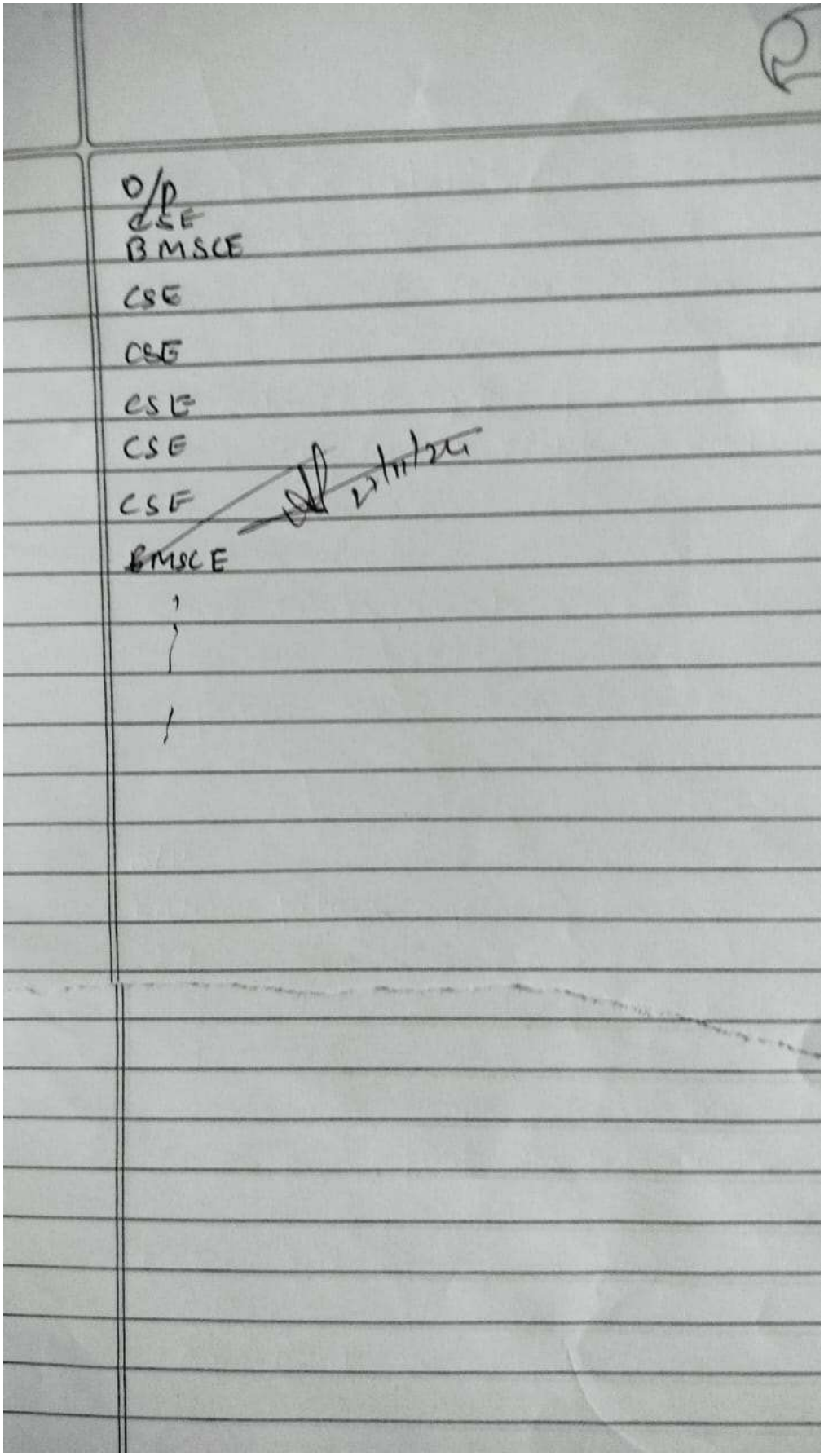

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.
C:\Users\hp\Desktop\java>javac fscombine.java
C:\Users\hp\Desktop\java>java fscombine
enter fathers age
23
enter sons age
2
accepted details successfully
enter details again 1. yes 2. no
1
enter fathers age
-29
Father's age error: age cant be negative
C:\Users\hp\Desktop\java>java fscombine
enter fathers age
23
enter sons age
45
Son's age error: sons age cant be greater than or same as father
C:\Users\hp\Desktop\java>

LABORATORY PROGRAM - 8

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.

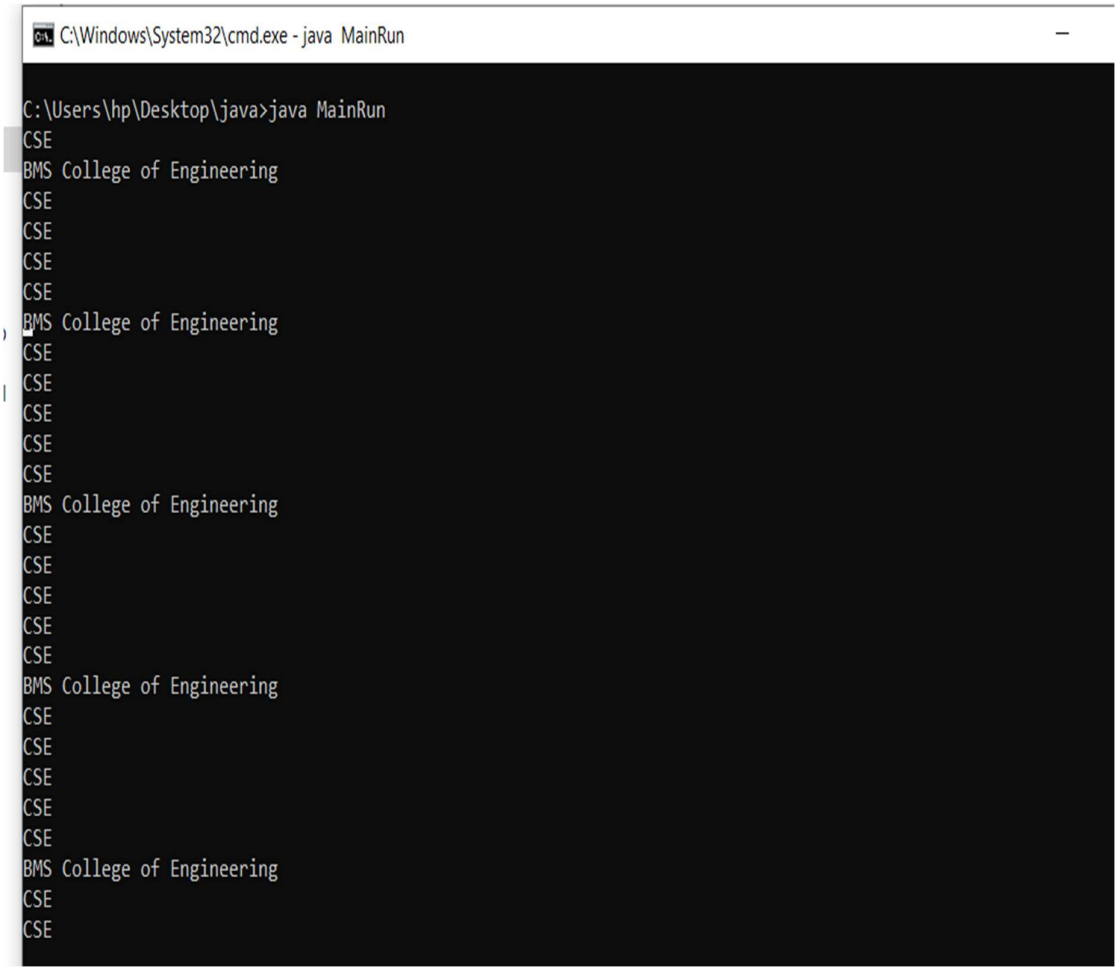


```
Create two threads college one thread
displaying "BMS college of Engineering"
once every ten seconds and another
displaying cse once every two seconds

class mythread extends Thread {
    String n;
    int time;

    mythread(String n, int t)
{this.n = n;
time = t;
}
public void run(){
    while(true){
        try {
            System.out.println(name);
            Thread.sleep(time);
        }
        catch(InterruptedException e) {
        System.out.println("Thread interupted");
        }
}}
}

public class mainc {
  public static void main(String[] xx) {
  Thread college = new Mythread("BMSCE", 1000);
  Thread dept = new Mythread("CSE", 2000);
    college.start();
    dept.start();
  }
}
```


O/P
BMSCE
CSE
CSE
CSE
CSE
CSE
BMSCE

```
class CollegeRunnable implements Runnable {
   String name;
   int time;
   CollegeRunnable(String n, int t) {
      name = n;
      time = t;
   }

   public void run() {
      while (true) {
         try {
            System.out.println(name);
            Thread.sleep(time);

         } catch (InterruptedException e) {
            System.out.println("Thread was interrupted");
         }
      }
   }
}

public class MainRun {
   public static void main(String[] args) {
      Runnable college = new CollegeRunnable("BMS College of Engineering",
10000);
      Runnable dept = new CollegeRunnable("CSE", 2000);
      Thread collegeThread = new Thread(college);
      Thread deptThread = new Thread(dept);
      collegeThread.start();
      deptThread.start();
   }
}
```

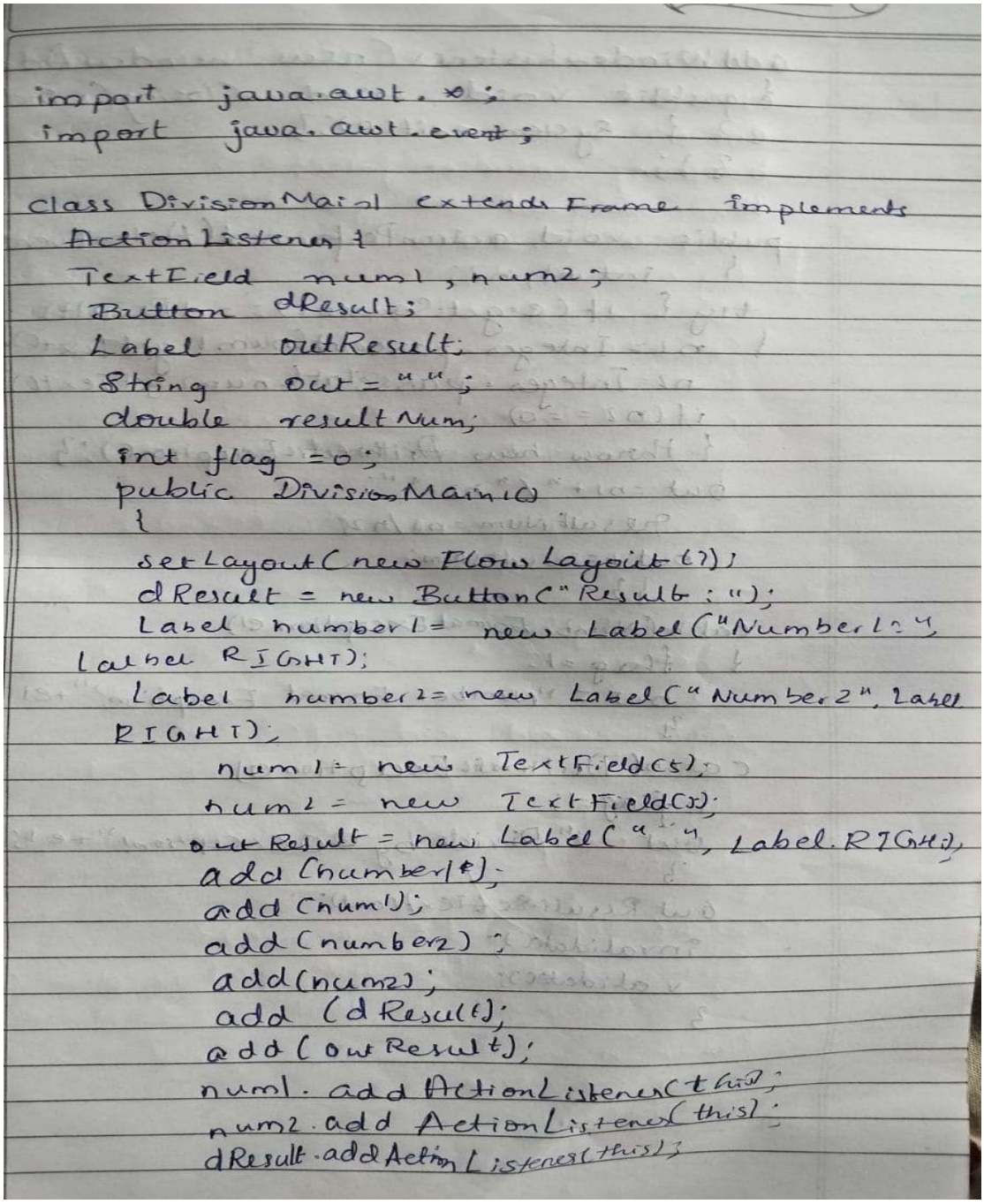
Output:


C:\Windows\System32\cmd.exe - java MainRun
C:\Users\hp\Desktop\java>java MainRun
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
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE

LABORATORY PROGRAM - 9

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

y



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

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("Number1:",
Label.RIGHT);
  Label number2 = new Label("Number2", Label
RIGHT);
      num1 = new TextField(5);
      num2 = new TextField(5);
      outResult = new Label("", Label.RIGHT);
      add(number1);
      add(num1);
      add(number2);
      add(num2);
      add(dResult);
      add(outResult);
      num1.addActionListener(this);
      num2.addActionListener(this);
      dResult.addActionListener(this);
```

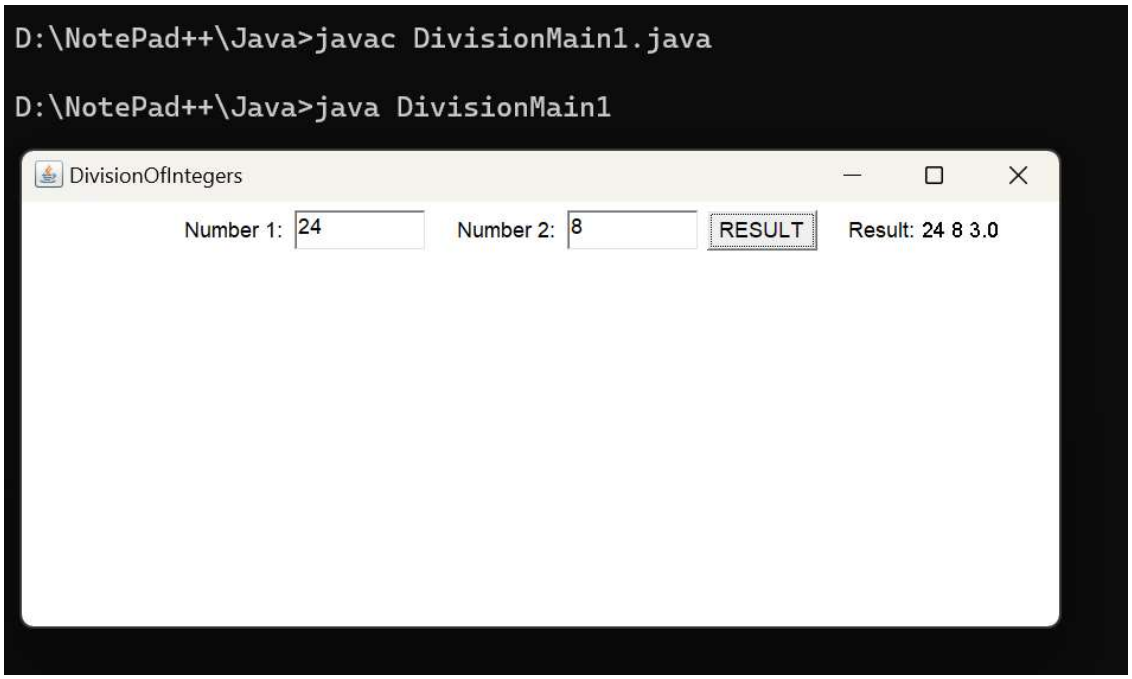
```
addWindowListener(new WindowAdapter()
{ public void windowClosing(WindowEvent
e) { System.exit(0); }
  });
}
public void actionPerformed(ActionEvent e)
  {   int n1, n2;
try { if(e.getSource() == dResult)
{  n1 = Integer.parseInt(num1.getText());
   n2 = Integer.parseInt(num2.getText());
   if(n2 == 0)
   { throw new ArithmeticException(); }
  Out = n1 + " / " n2 + " ";
    resultNum = n1/n2;
   Out += resultNum;
    }}
catch (NumberFormatException e1)
{  flag = 1;
   Out = "Number Format Exception!" + e1;
  }
catch( ArithmeticException e)
  { flag = 1;
  Out = " Divide by @0 exception!" + e1;
   }
  Out Result.setText(Out);
  invalidate();
   validate();
}
```

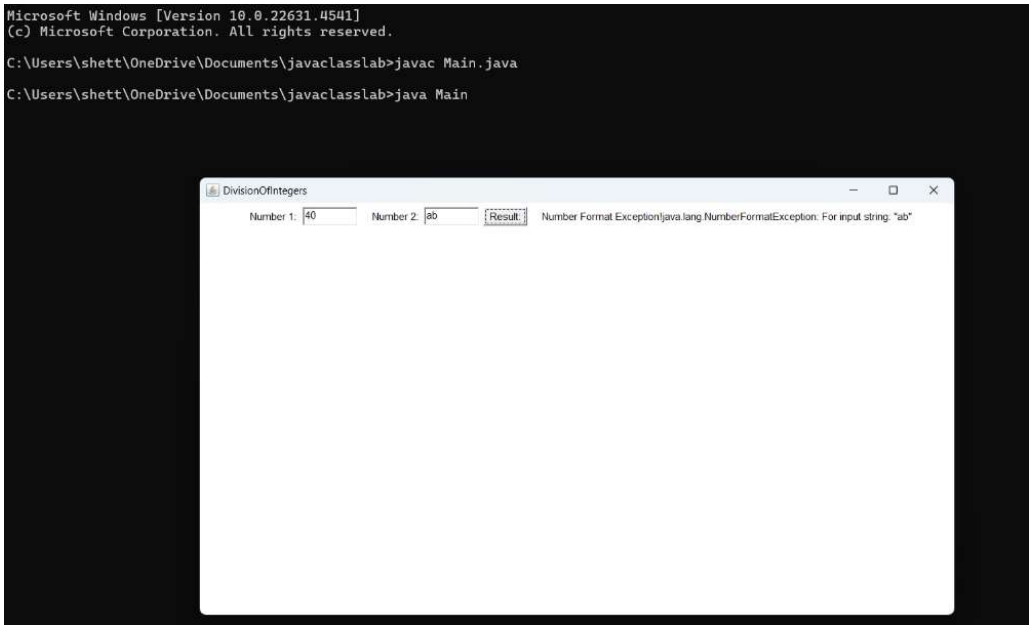
```
public class Main
{   public static void main(String args[])
    {
    DivisionMain1 obj = new DivisionMain1();
     obj.setSize(new Dimension(800,400));
      Obj.SetTitle("Division Of Integers");
      Obj.setVisible(true);
    }
}
```

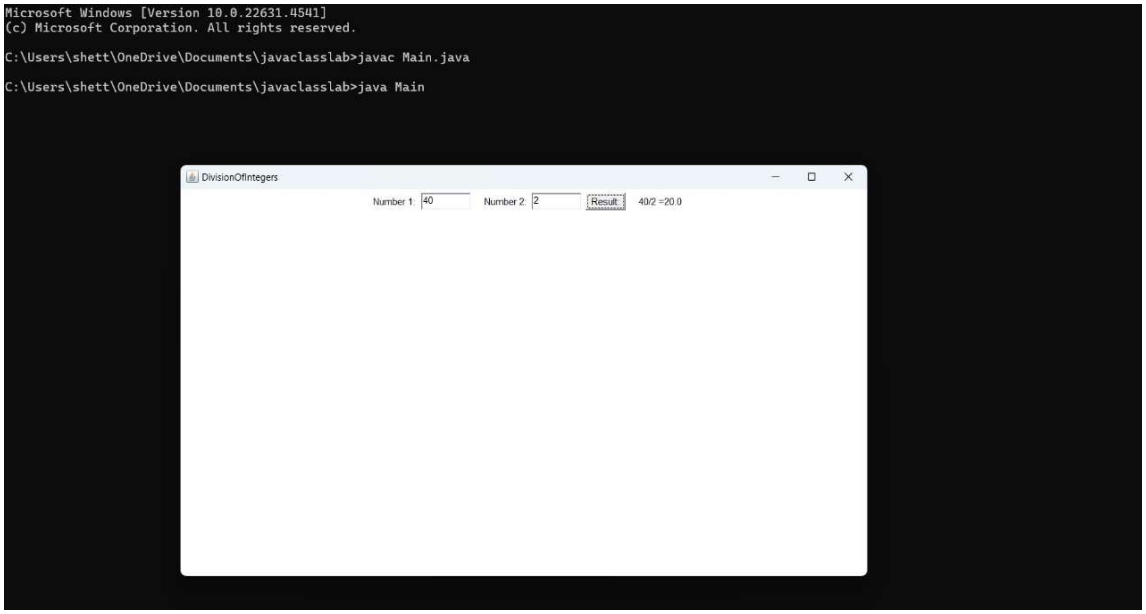
```
import.java.awt.* ;
    public class DivisionMain1 extends Frame implements ActionListener
    {
            TextField
            num1,num2; Button
            dResult;
            Label
            outResult;
            String out="";
            double
            resultNum; int
            flag=0;

            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()
                    {
```

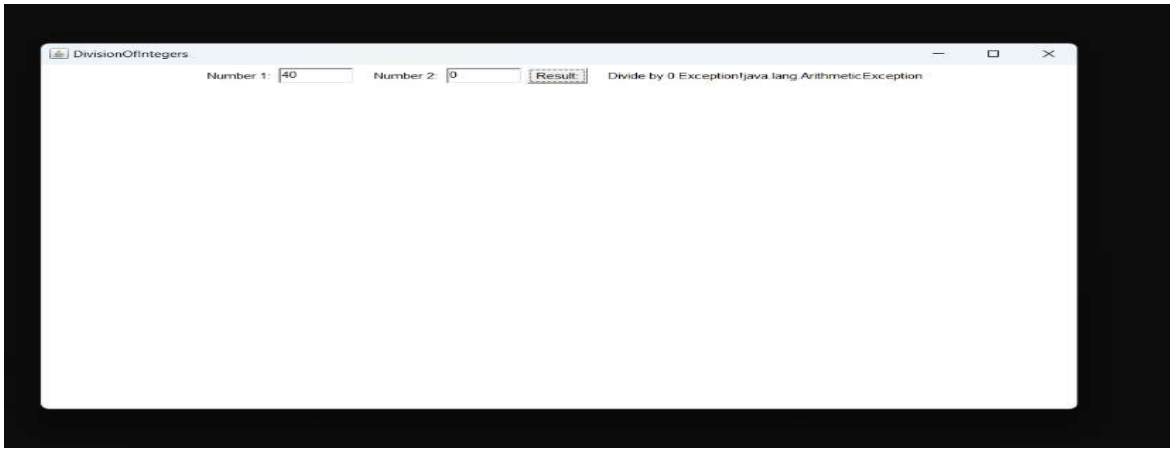
```
                public void windowClosing(WindowEvent we)
                {
                        System.exit(0);
                }
        });
}
public void actionPerformed(ActionEvent ae)
{
        int
        n1,n2;
        try
        {
                if (ae.getSource() == dResult)
                {
                        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(resultN
                        um); 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;
    }
    public static void main(String[] args)
    {
        DivisionMain1 dm=new
        DivisionMain1(); dm.setSize(new
        Dimension(800,400));
        dm.setTitle("DivisionOfIntegers");
        dm.setVisible(true);
    }
}
```

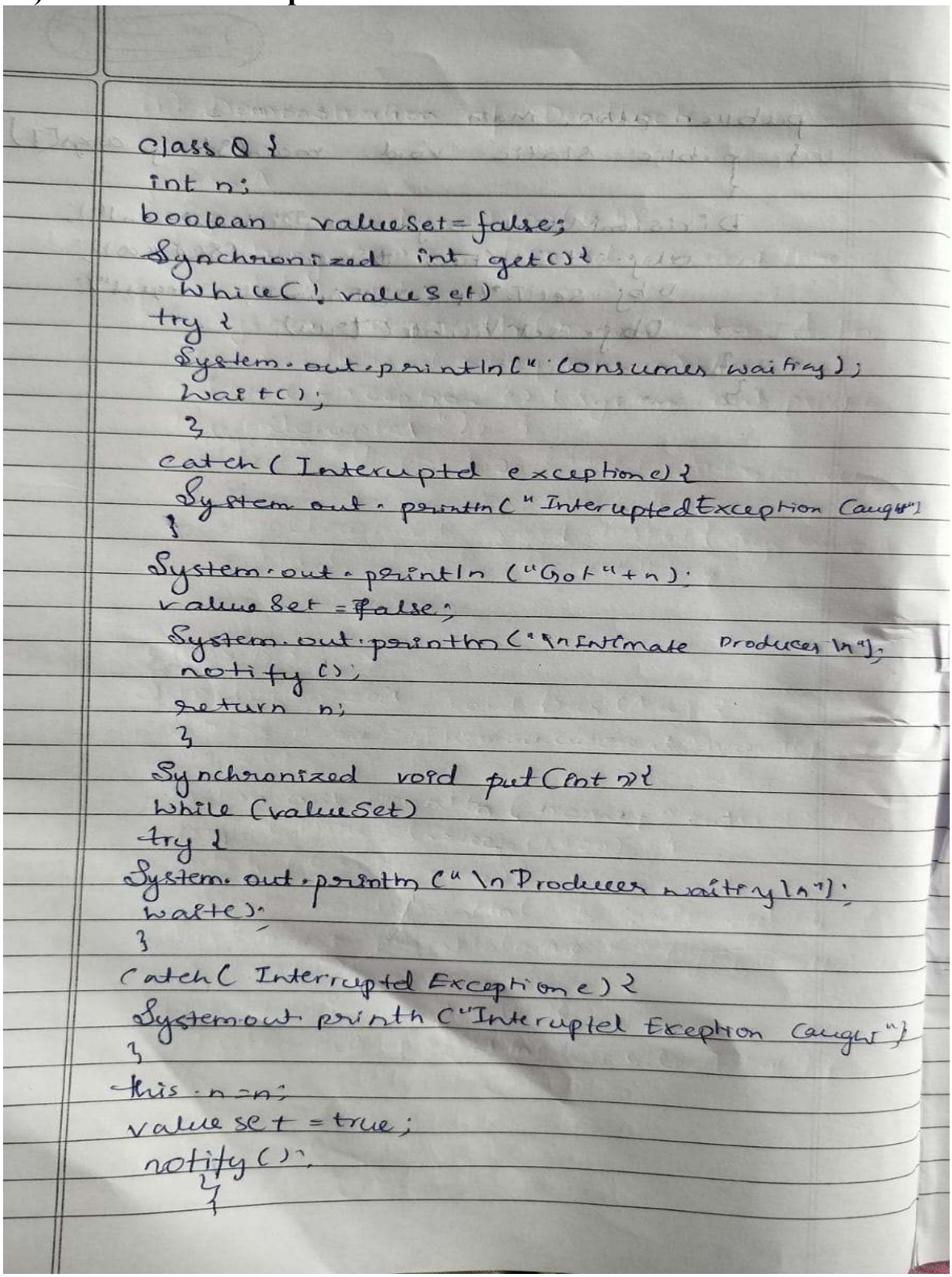



Microsoft Windows [Version 10.0.22631.4541]
(c) Microsoft Corporation. All rights reserved.
C:\Users\shett\OneDrive\Documents\javaclasslab>javac Main.java
C:\Users\shett\OneDrive\Documents\javaclasslab>java Main
DivisionOfIntegers
Number 1: 40
Number 2: ab
Result
Number Format Exception!java.lang.NumberFormatException: For input string: "ab"

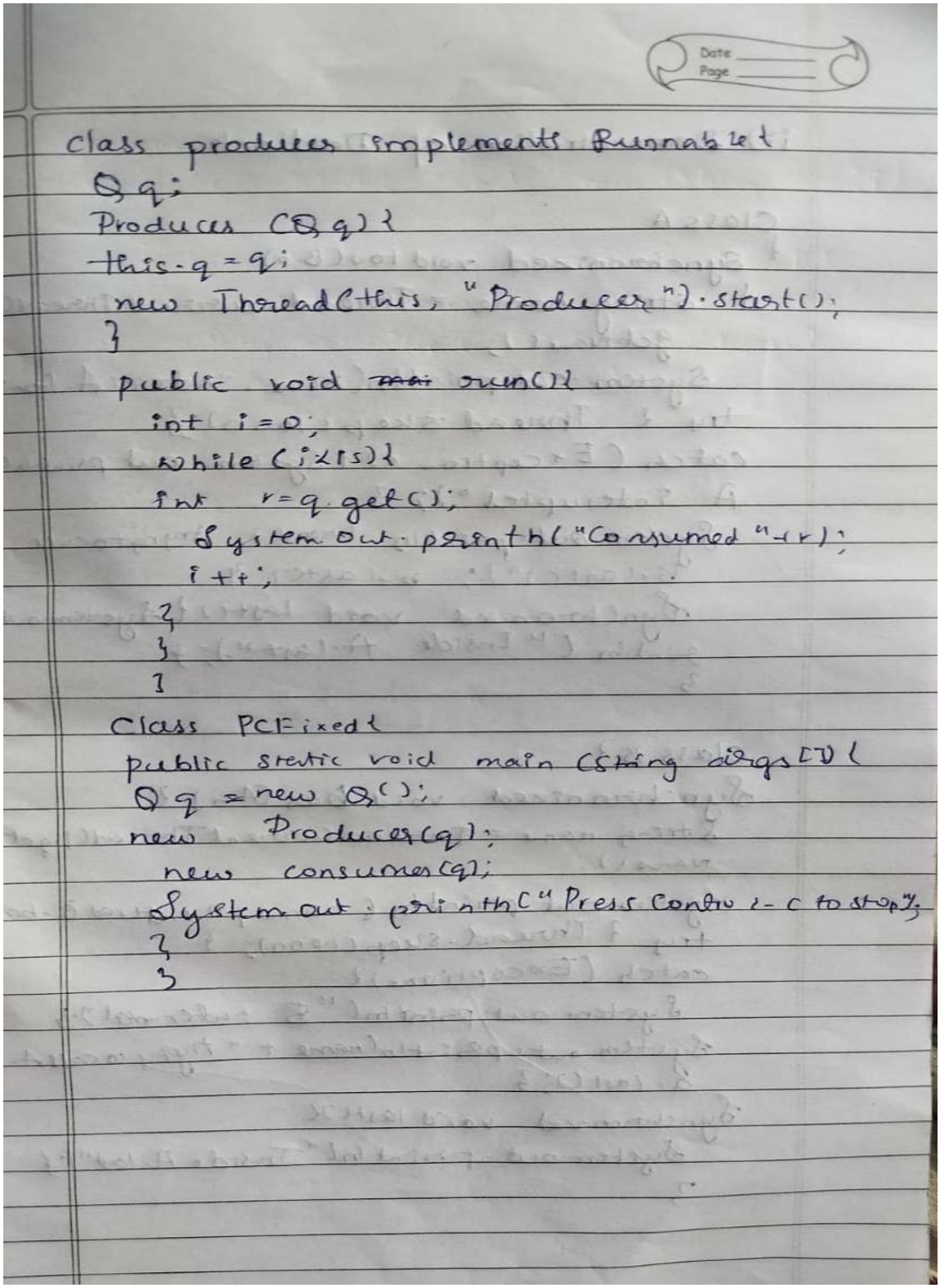

Microsoft Windows [Version 10.0.22631.4541]
(c) Microsoft Corporation. All rights reserved.
C:\Users\shett\OneDrive\Documents\javaclasslab>javac Main.java
C:\Users\shett\OneDrive\Documents\javaclasslab>java Main
DivisionOfIntegers
Number 1: 40
Number 2: 2
Result
40/2 =20.0


DivisionOfIntegers
Number 1: 40
Number 2: 0
Result
Divide by 0 Exception!java.lang.ArithmeticException

10)Demonstrate Interprocess communication and deadlock



```
class Q {
int n;
boolean valueSet = false;
Synchronized int get(){
 While(! valueSet)
try {
 System.out.println(" Consumes wait fag);
 wait();
 }
catch ( Interupted exception e){
 System out . println(" InteruptedException 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){
Systemout.println("Interupted Exception Caught");
}
this.n=n;
valueSet = true;
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) {
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");
}
}
```

ii) Demonstration of Deadlock

```
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 bas(A a) {
  String name = Thread.currentThread().get
  Name();
System.out.println(name+" entered B.bas");
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 implement Runnable
{
  A a = new A();
  B b = new B();
  Deadlock() {
  Thread.currentThread().setName("Main_Thread");
  Thread t = new Thread(this, "Racing Thread");
  t.start();
  a.foo(b);
  System.out.println("Back in main thread");
  }
  public void run() { b.bar(a);
  System.out.prith("Back in other thread");
  }
  public static void main(String args[])
  { new Deadlock(); }
```

```
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.");
}
}
```

ii. Demonstration of deadlock

```
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(); }
}
```

```
C:\Users\shett\OneDrive\Documents\javaclasslab>javac Deadlock.java

C:\Users\shett\OneDrive\Documents\javaclasslab>java Deadlock
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()


C:\Users\shett\OneDrive\Documents\javaclasslab>javac sync.java

C:\Users\shett\OneDrive\Documents\javaclasslab>java sync
Press Control-C to stop.
Put: 0

Intimate Consumer


Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer


Producer waiting

Consumed: 0
Got: 1

Intimate Producer

Consumed: 1
Put: 2

Intimate Consumer


Producer waiting

Got: 2

Intimate Producer

Consumed: 2
Put: 3

Intimate Consumer


Producer waiting

Got: 3

Intimate Producer

Consumed: 3
Put: 4

Intimate Consumer
```

```
Consumed: 3
Put: 4

Intimate Consumer


Producer waiting

Got: 4

Intimate Producer

Consumed: 4
Put: 5

Intimate Consumer


Producer waiting

Got: 5

Intimate Producer

Consumed: 5
Put: 6

Intimate Consumer


Producer waiting

Got: 6

Intimate Producer

Consumed: 6
Put: 7

Intimate Consumer


Producer waiting

Got: 7

Intimate Producer

Consumed: 7
Put: 8

Intimate Consumer


Producer waiting
```

```
Intimate Producer

Consumed: 7
Put: 8

Intimate Consumer


Producer waiting

Got: 8

Intimate Producer

Consumed: 8
Put: 9

Intimate Consumer


Producer waiting

Got: 9

Intimate Producer

Consumed: 9
Put: 10

Intimate Consumer


Producer waiting

Got: 10

Intimate Producer

Consumed: 10
Put: 11

Intimate Consumer


Producer waiting

Got: 11

Intimate Producer

Consumed: 11
Put: 12

Intimate Consumer
```

```
Intimate Consumer


Producer waiting

Got: 11

Intimate Producer

Consumed: 11
Put: 12

Intimate Consumer


Producer waiting

Got: 12

Intimate Producer

Consumed: 12
Put: 13

Intimate Consumer


Producer waiting

Got: 13

Intimate Producer

Consumed: 13
Put: 14

Intimate Consumer

Got: 14

Intimate Producer

Consumed: 14

C:\Users\shett\OneDrive\Documents\javaclasslab>
```