VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Ikshitha P (1BM23CS118)

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 **Ikshitha P (1BM23CS118),** who is a 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.

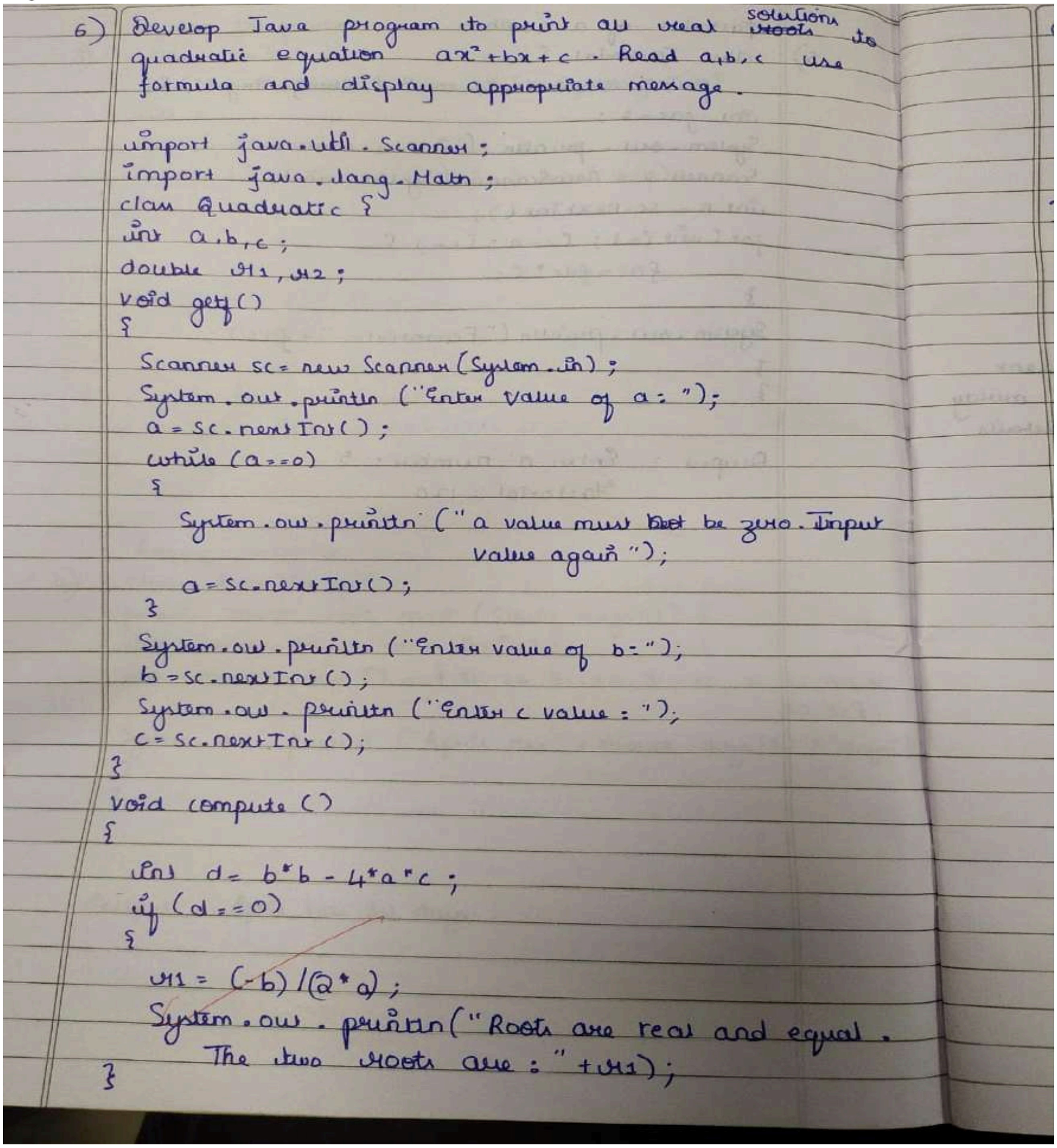
Dr. Seema Patil Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link:
https://github.com/Ikshitha-P/Java-lab-programs

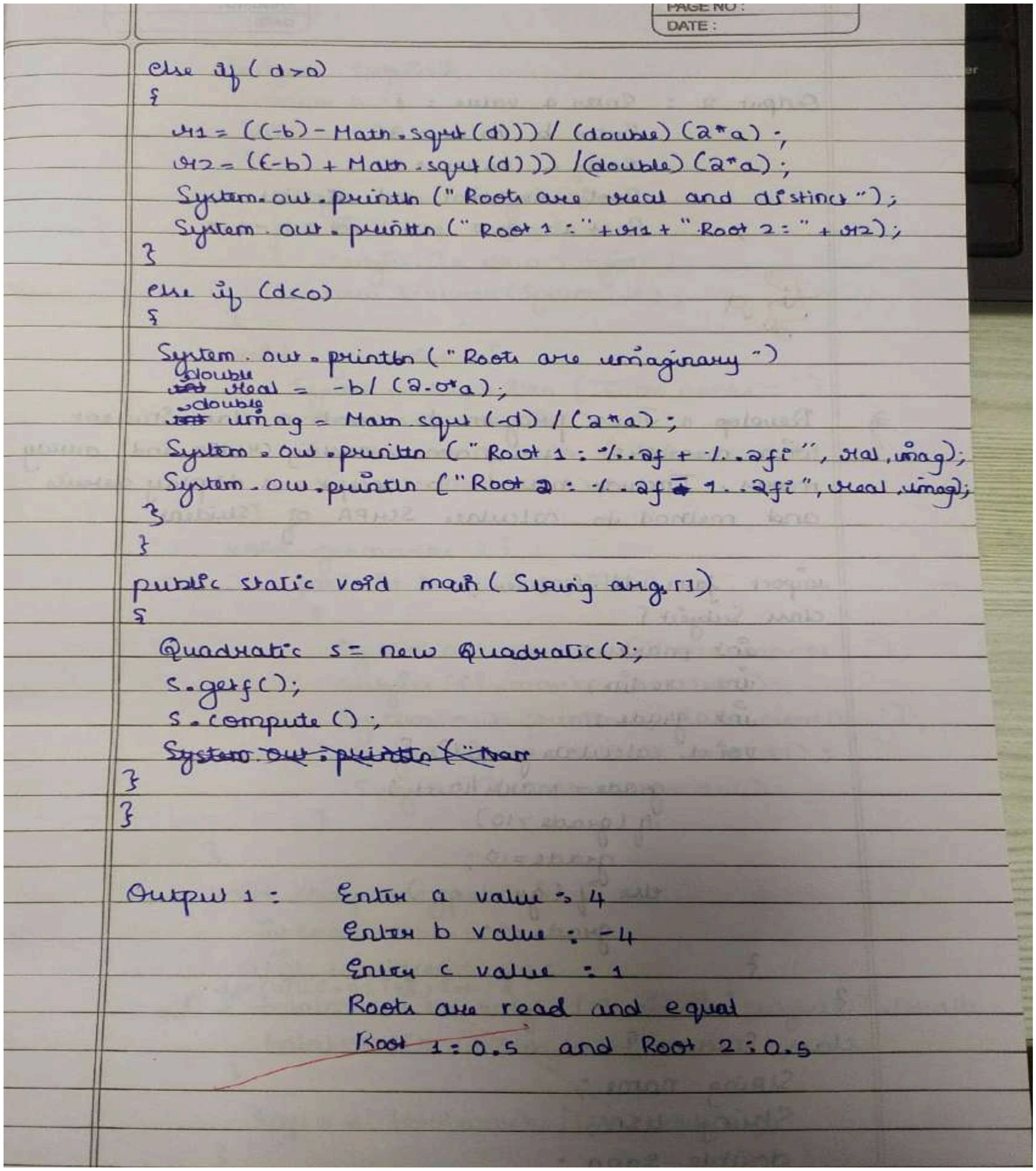
Program 1
Implement Quadratic Equation

Algorithm:



6) Develop Java program to print all real ~~roots~~ solutions to quadratic equation ax^2+bx+c. Read a,b,c use formula and display appropriate message.

```
import java.util.Scanner;
import java.lang.Math;
class Quadratic {
int a,b,c;
double r1, r2;
void get()
{
  Scanner sc= new Scanner(System.in);
  System.out.println("Enter value of a: ");
  a = sc.nextInt();
  while (a==0)
  {
     System.out.println("a value must ~~not~~ be zero. Input
                         value again");
     a=sc.nextInt();
  }
  System.out.println("Enter value of b:");
  b =sc.nextInt();
  System.out.println("Enter c value :");
  c=sc.nextInt();
}
void compute()
{
  int d= b*b - 4*a*c;
  if (d==0)
  {
   r1 = (-b)/(2*a);
   System.out.println("Roots are real and equal.
       The two roots are: "+r1);
  }
```



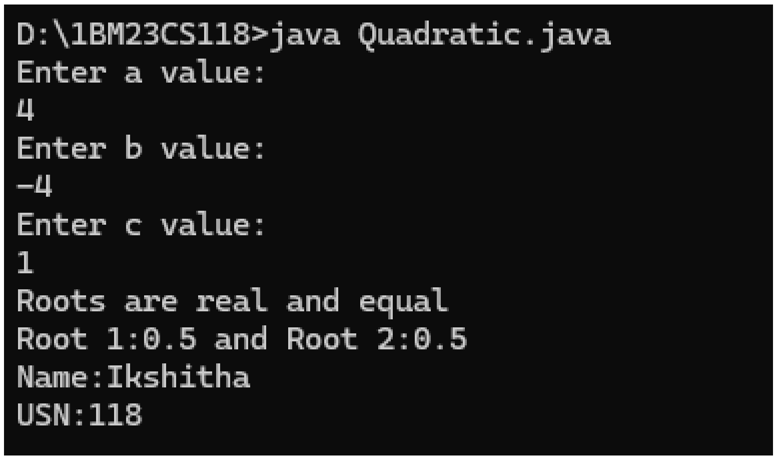
PAGE NO :
DATE :

```
else if (d>0)
{
  r1= ((-b)-Math.sqrt(d))) / (double) (2*a) ;
  r2= ((-b) + Math.sqrt(d))) /(double) (2*a);
  System.out.println ("Roots are real and distinct");
  System.out.println ("Root 1 :"+r1+ "Root 2:" + r2);
}
else if (d<0)
{
 System.out.println ("Roots are imaginary ")
 ~~int~~ double real = -b/ (2.0*a);
 ~~int~~ double imag = Math.sqrt (-d) / (2*a) ;
 System.out.println ("Root 1 : %.2f + %.2fi", real, imag);
 System.out.println ("Root 2 : %.2f - %.2fi", real, imag);
}
}
public static void main ( String args[])
{
 Quadratic s= new Quadratic();
 s.getf();
 s.compute () ;
 ~~System.out.println ("Num~~
}
}
```

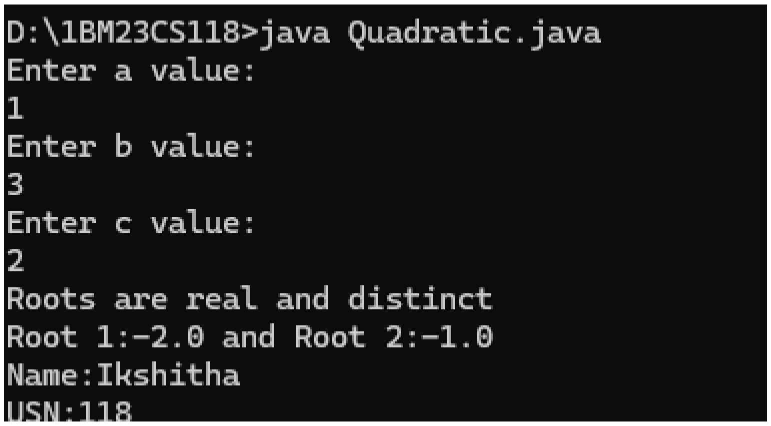
Output 1 :
```
Enter a value : 4
Enter b value : -4
Enter c value : 1
Roots are real and equal
Root 1:0.5  and  Root 2:0.5
```

Code:

```
import java.util.Scanner;
import java.lang.Math;
class Quadratic{
double a,b,c;
double r1,r2;
void getf()
{
Scanner sc=new Scanner(System.in);
System.out.println("Enter a value:");
a=sc.nextInt();
while(a==0)
{
System.out.println("a cannot be zero, enter A value again:");
a=sc.nextInt();
}

System.out.println("Enter b value:");
b=sc.nextInt();
System.out.println("Enter c value:");
c=sc.nextInt();
}
void compute()
{
double d=b*b-4*a*c;
if(d==0)
{
r1=(-b)/(2*a);
System.out.println("Roots are real and equal");
System.out.println("Root 1:"+r1+" and Root 2:"+r1);
}
else if(d>0)
{
r1 = ((-b) - (Math.sqrt(d)))/(double)(2*a);
r2=((-b) + (Math.sqrt(d)))/(double)(2*a);
System.out.println("Roots are real and distinct");

System.out.println("Root 1:"+r1+" and Root 2:"+r2);

}
else if(d<0)
{
 double realPart = -b / (2 * a);
            double imaginaryPart = Math.sqrt(-d) / (2 * a);
            System.out.println("Roots are imaginary: " + realPart + " + " + imaginaryPart + "i");
            System.out.println("Roots are imaginary: " + realPart + " - " + imaginaryPart + "i");
}
}
public static void main(String args[])
{
Quadratic s=new Quadratic();
s.getf();
s.compute();
System.out.println("Name:Ikshitha");

System.out.println("USN:118");
}
}
```



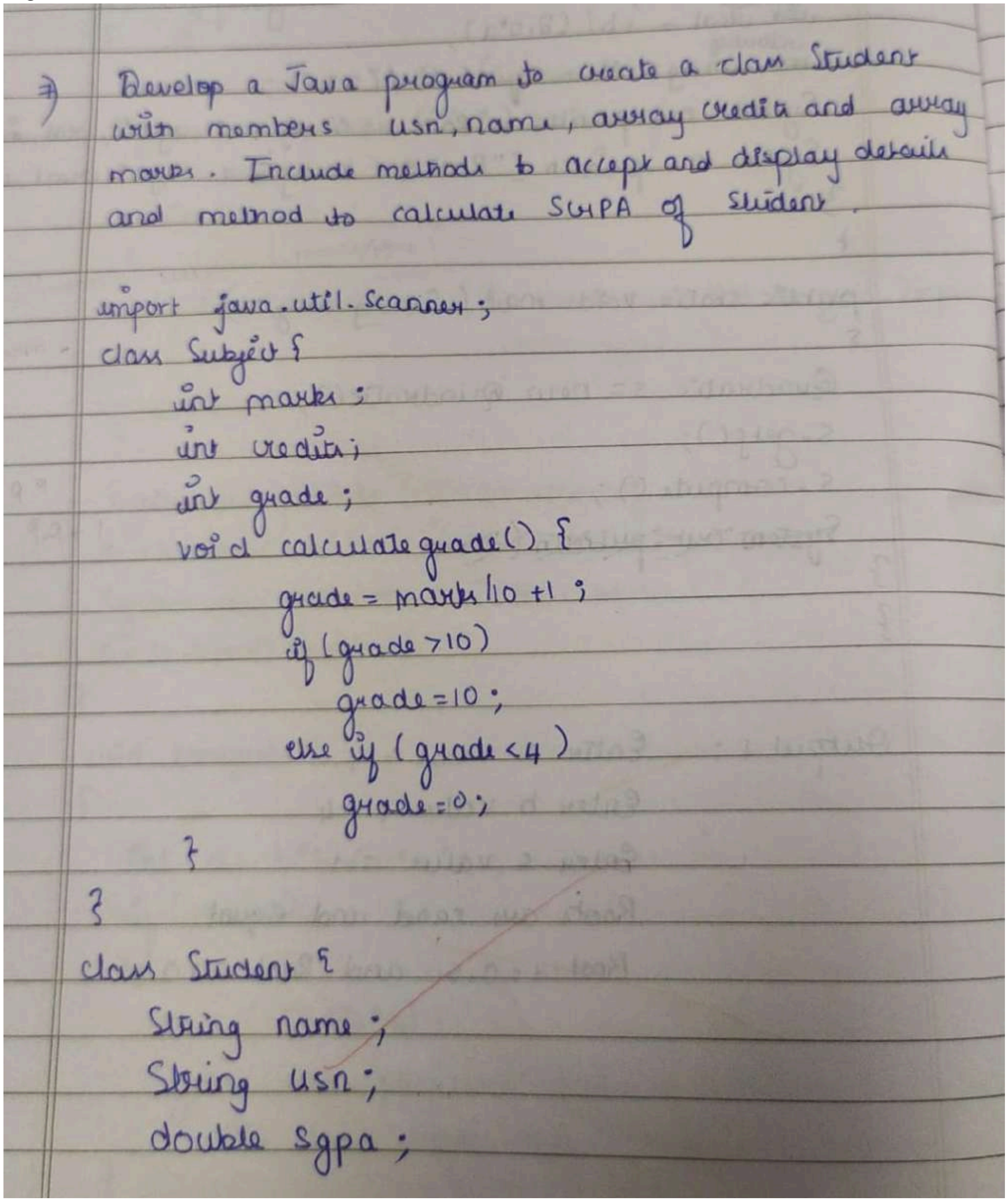
```
D:\1BM23CS118>java Quadratic.java
Enter a value:
4
Enter b value:
-4
Enter c value:
1
Roots are real and equal
Root 1:0.5 and Root 2:0.5
Name:Ikshitha
USN:118
```



```
D:\1BM23CS118>java Quadratic.java
Enter a value:
1
Enter b value:
3
Enter c value:
2
Roots are real and distinct
Root 1:-2.0 and Root 2:-1.0
Name:Ikshitha
USN:118
```

Program 2
Java program to create a class Student with members usn,name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

Algorithm:



7) Develop a Java program to create a class Student with members usn, name, array credits and array marks. Include methods to accept and display details and method to calculate SGPA of student.

```
import java.util.Scanner;
class Subject {
    int marks;
    int credits;
    int grade;
    void calculategrade() {
        grade = marks/10 +1;
        if (grade >10)
            grade =10;
        else if (grade <4)
            grade =0;
    }
}
class Student {
    String name;
    String usn;
    double sgpa;
```

```
Subject subject[];
 Scanner sc;
 Student()
 {
   subject = new Subject[3];
   for(int i=0; i<3; i++)
       subject[i]= new subject();
   sc= new Scanner(System.in);
 }
 void getdetails() {
     System.out.println("Enter name : ");
     name= sc.nextLine();
     System.out.println("Enter USN : ");
     usn = sc.nextLine();
  }
 void getmarks() {
     for(int i=0; i<3; i++)
      {
        System.out.println("Enter marks : ");
        subject[i].marks=snextInt();
        System.out.println("Enter credits : ");
        subject[i].credits= sc.nextInt();
        subject[i].calculategrade();
      }
}
void computesgpa() {
   int totalmarks=0;
   int totalcredits=0;
for(int i=0; i<3; i++){
   totalmarks += subject[i].~~marks~~grade * subject[i].credits
   totalcredits += subject[i].credits;
 }
   sgpa = totalmarks/totalcredits;
}
```

PAGE NO :
DATE :

```
public static void main ( String args[])
{
    Student S1 = new Student ()
    S1.getdetails();
    S1.getmarks();
    S1.computesgpa();
    System.out.println ('Name:', +S1.name);
    System.out.println ('USN :', +S1.usn);
    System.out.println ('SGPA:' + S1.sgpa);
}
}
```

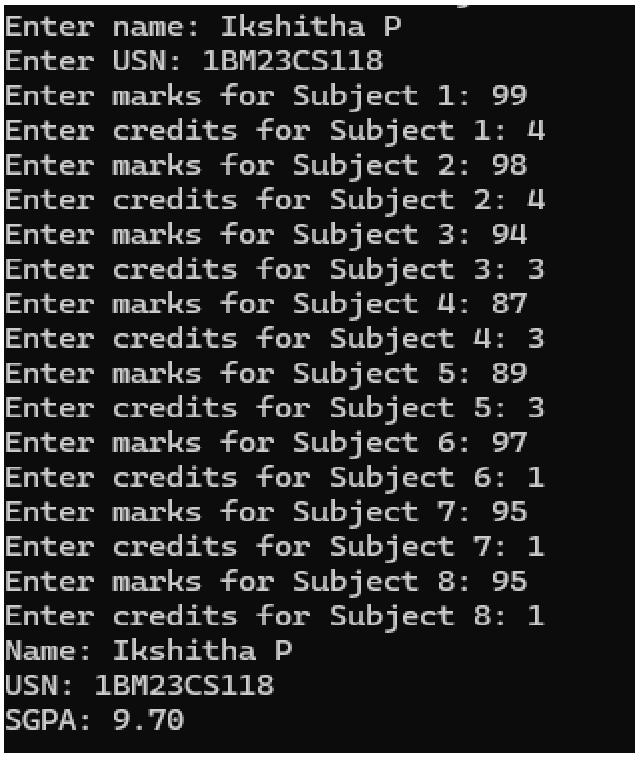
Output :

```
Enter name : Ikshitha
Enter USN : 1BM23CS118
Enter marks : 91
Enter credits : 4
Enter marks : 90
Enter credits : 3
Enter marks : 80
Enter credits : 2
Name : Ikshitha
USN : 1BM23CS118
SGPA : 9.77 .
```

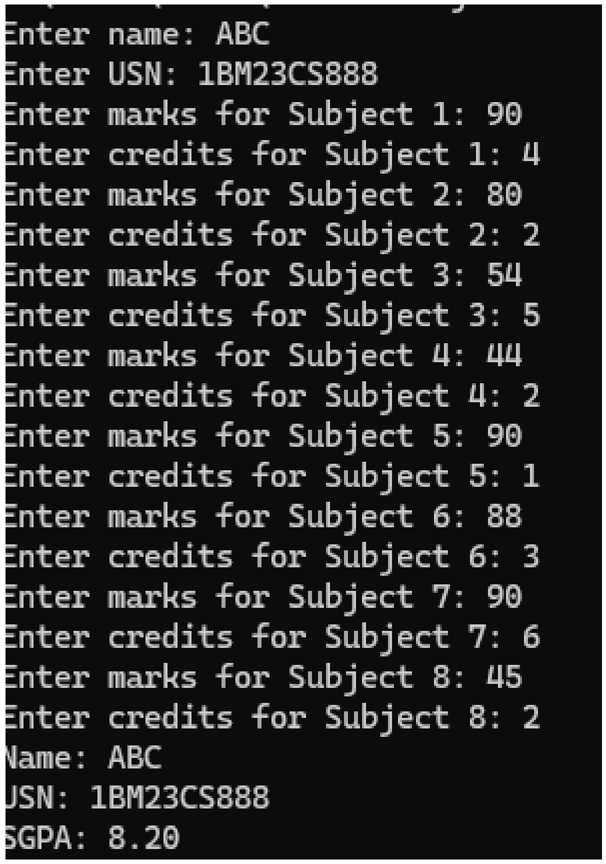
[illegible]
7-10-24

Code:

```
import java.util.Scanner;

class Subject {
    int marks;
    int credits;
    int grade;

      void calculateGrade() {
       grade=marks/10+1;
       if(grade>10)
       grade=10;
       else if(grade<4)
      grade=0;
    }
}
class Student {
    String name;
    String usn;
    double sgpa;
    Subject subject[];
    Scanner s;


    Student() {
        subject = new Subject[8];
        for (int i = 0; i < 8; i++) {
            subject[i] = new Subject(); // create array of Subject objects
        }
        s = new Scanner(System.in);
    }


    void getStudentDetails() {
        System.out.print("Enter name: ");
        name = s.nextLine();
        System.out.print("Enter USN: ");
        usn = s.nextLine();
    }

      void getMarks() {
        for (int i = 0; i < 8; i++) {
            System.out.print("Enter marks for Subject " + (i + 1) + ": ");
            subject[i].marks = s.nextInt();
            System.out.print("Enter credits for Subject " + (i + 1) + ": ");
            subject[i].credits = s.nextInt();
            subject[i].calculateGrade();

             }
    }

      void computeSGPA() {
       double totalGradePoints = 0;
       double totalCredits = 0;

       for (int i = 0; i < 8; i++) {
           totalGradePoints += subject[i].grade * subject[i].credits;
           totalCredits += subject[i].credits;
       }
       sgpa=totalGradePoints/totalCredits;

      }


    public static void main(String args[]) {
        Student s1 = new Student();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();


        System.out.println("Name: " + s1.name);
        System.out.println("USN: " + s1.usn);
       System.out.printf("SGPA: %.2f\n", s1.sgpa);
    }
}
```



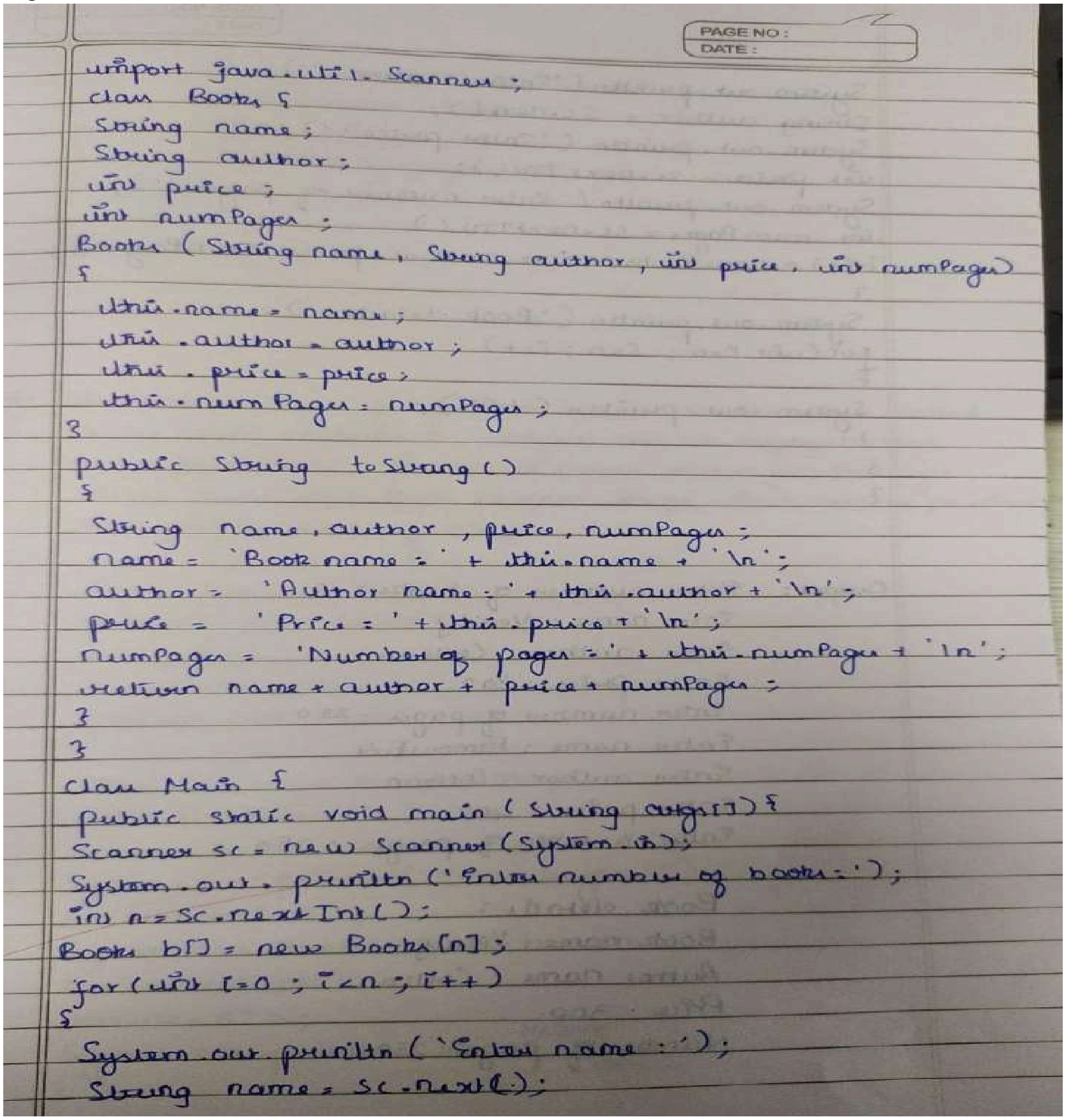
```
Enter name: Ikshitha P
Enter USN: 1BM23CS118
Enter marks for Subject 1: 99
Enter credits for Subject 1: 4
Enter marks for Subject 2: 98
Enter credits for Subject 2: 4
Enter marks for Subject 3: 94
Enter credits for Subject 3: 3
Enter marks for Subject 4: 87
Enter credits for Subject 4: 3
Enter marks for Subject 5: 89
Enter credits for Subject 5: 3
Enter marks for Subject 6: 97
Enter credits for Subject 6: 1
Enter marks for Subject 7: 95
Enter credits for Subject 7: 1
Enter marks for Subject 8: 95
Enter credits for Subject 8: 1
Name: Ikshitha P
USN: 1BM23CS118
SGPA: 9.70
```



```
Enter name: ABC
Enter USN: 1BM23CS888
Enter marks for Subject 1: 90
Enter credits for Subject 1: 4
Enter marks for Subject 2: 80
Enter credits for Subject 2: 2
Enter marks for Subject 3: 54
Enter credits for Subject 3: 5
Enter marks for Subject 4: 44
Enter credits for Subject 4: 2
Enter marks for Subject 5: 90
Enter credits for Subject 5: 1
Enter marks for Subject 6: 88
Enter credits for Subject 6: 3
Enter marks for Subject 7: 90
Enter credits for Subject 7: 6
Enter marks for Subject 8: 45
Enter credits for Subject 8: 2
Name: ABC
USN: 1BM23CS888
SGPA: 8.20
```

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

Algorithm:



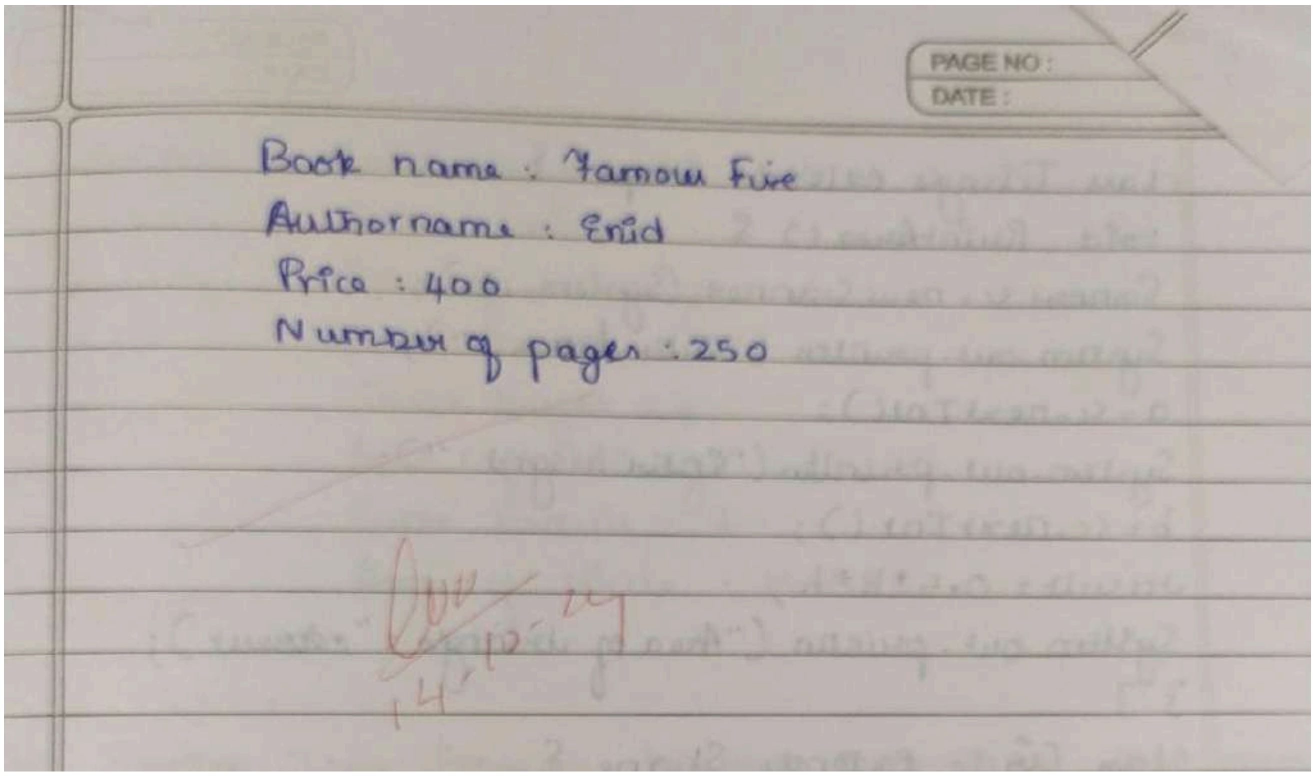
PAGE NO :
DATE :

```
import java.util.Scanner;
class Books {
String name;
String author;
int price;
int numPages;
Books (String name, String author, int price, int numPages)
{
 this.name = name;
 this.author = author;
 this.price = price;
 this.numPages = numPages;
}
public String toString()
{
 String name, author, price, numPages;
 name = 'Book name : ' + this.name + '\n';
 author = 'Author name : ' + this.author + '\n';
 price = 'Price : ' + this.price + '\n';
 numPages = 'Number of pages : ' + this.numPages + '\n';
 return name + author + price + numPages;
}
}
class Main {
 public static void main (String args[]) {
 Scanner sc = new Scanner (System.in);
 System.out.println ('Enter number of books:');
 int n = sc.nextInt();
Books b[] = new Books[n];
for(int i=0 ; i<n ; i++)
{
 System.out.println ('Enter name :');
 String name = sc.next();
```

DATE :

```
System.out.println('Enter author :');
String author = sc.next();
System.out.println ('Enter price :');
int price = sc.nextInt();
System.out.println('Enter number of pages :');
int numPages = sc.nextInt()
b[i] = new Books(name,author,price,numPages);
}
System.out.println ('Book details :');
for(int i=0; i<n; i++)
{
System.out.println (b[i]);
}
}
}
```

Output:
```
Enter number of books : 2
Enter name : Verity
Enter author : Colleen
Enter price : 300
Enter number of pages : 250
Enter name : FamousFive
Enter author : Colleen
Enter price : 400
Enter number of pages : 250

Book details :
Book name : Verity
Author name : Colleen
Price : 300
Number of pages : 250
```

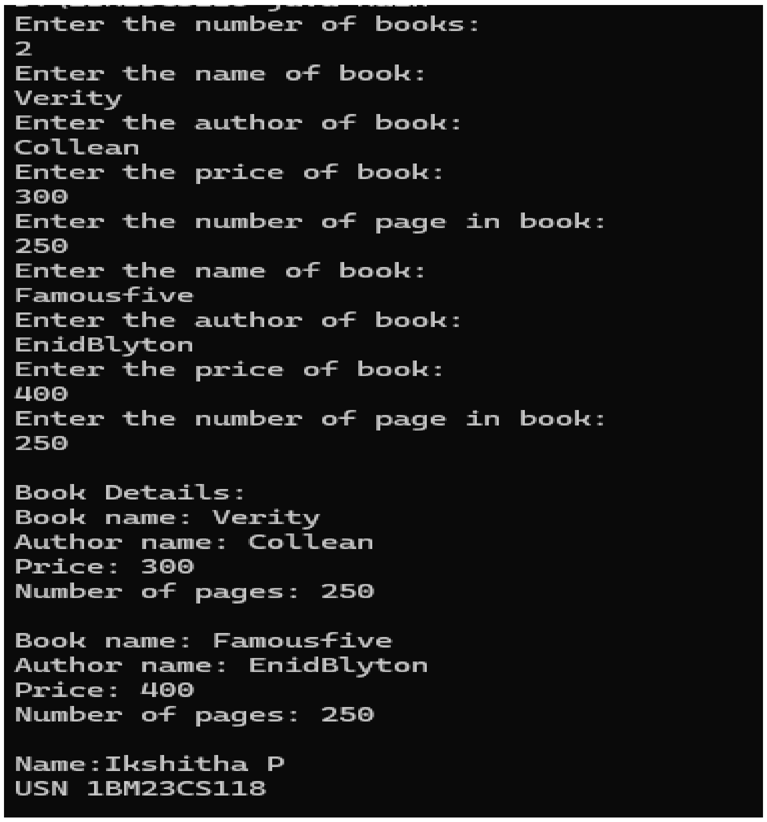


PAGE NO :
DATE :

```
Book name : Famous Five
Authorname : Enid
Price : 400
Number of pages : 250
```

Code:

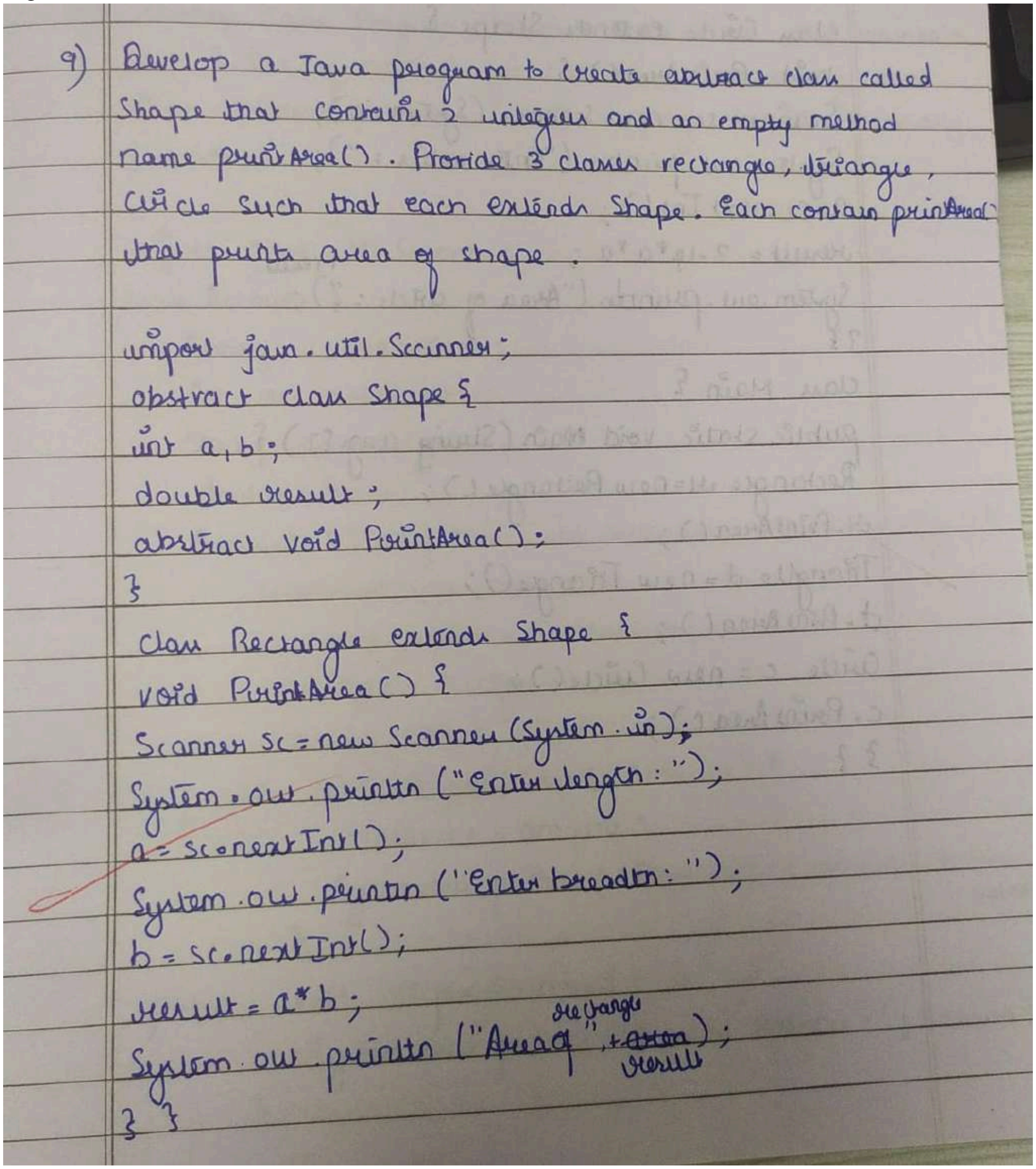
```
import java.util.Scanner;
class Books{
String name;
String author;
int price;
int numPages;
Books(String name, String author, int price, int numPages)
{
this.name = name;
this.author= author;
this.price = price;
this.numPages = numPages;
}
public String toString()
{
String name, author, price, numPages;
name = "Book name: " + this.name + "\n";
author = "Author name: " + this.author + "\n";
price = "Price: " + this.price + "\n";
numPages = "Number of pages: " + this.numPages + "\n";
return name + author + price + numPages;
}
}
class Main
{
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
int n;
String name;
String author;
int price;
int numPages;
System.out.println("Enter the number of books:");
n=sc.nextInt();
Books b[]=new Books[n];
for(int i=0;i<n;i++)
{
System.out.println("Enter the name of book:");
name=sc.next();
System.out.println("Enter the author of book:");
author=sc.next();
System.out.println("Enter the price of book:");
price=sc.nextInt();
System.out.println("Enter the number of page in book:");
numPages=sc.nextInt();
b[i] = new Books(name,author,price,numPages);
}
System.out.println("\nBook Details:");
for (int i = 0; i < n; i++)
{
System.out.println(b[i]);
}
}
}
```



```
Enter the number of books:
2
Enter the name of book:
Verity
Enter the author of book:
Collean
Enter the price of book:
300
Enter the number of page in book:
250
Enter the name of book:
Famousfive
Enter the author of book:
EnidBlyton
Enter the price of book:
400
Enter the number of page in book:
250

Book Details:
Book name: Verity
Author name: Collean
Price: 300
Number of pages: 250

Book name: Famousfive
Author name: EnidBlyton
Price: 400
Number of pages: 250

Name:Ikshitha P
USN 1BM23CS118
```

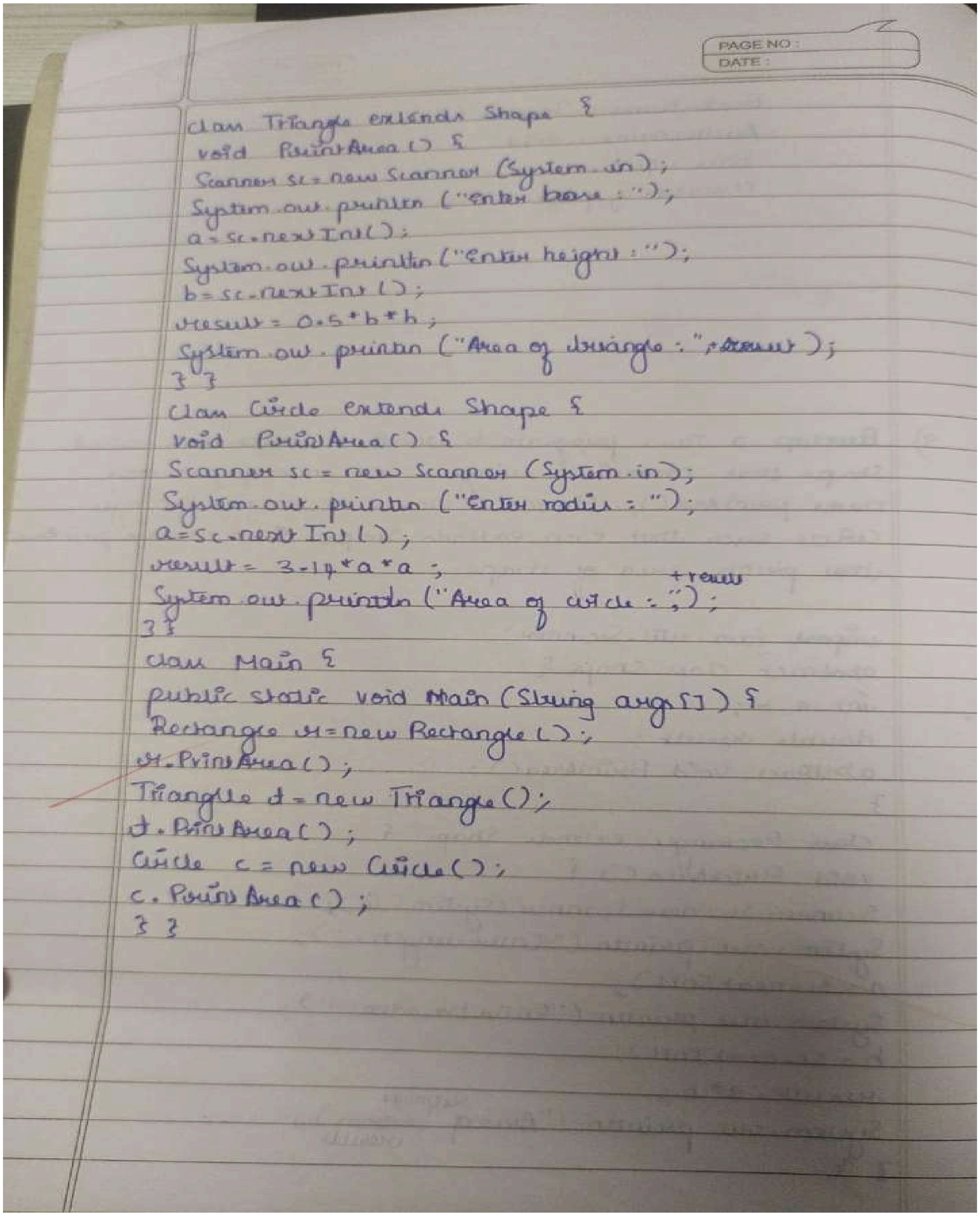
Program 4
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ).Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea( ) that prints the area of the given shape.

Algorithm:



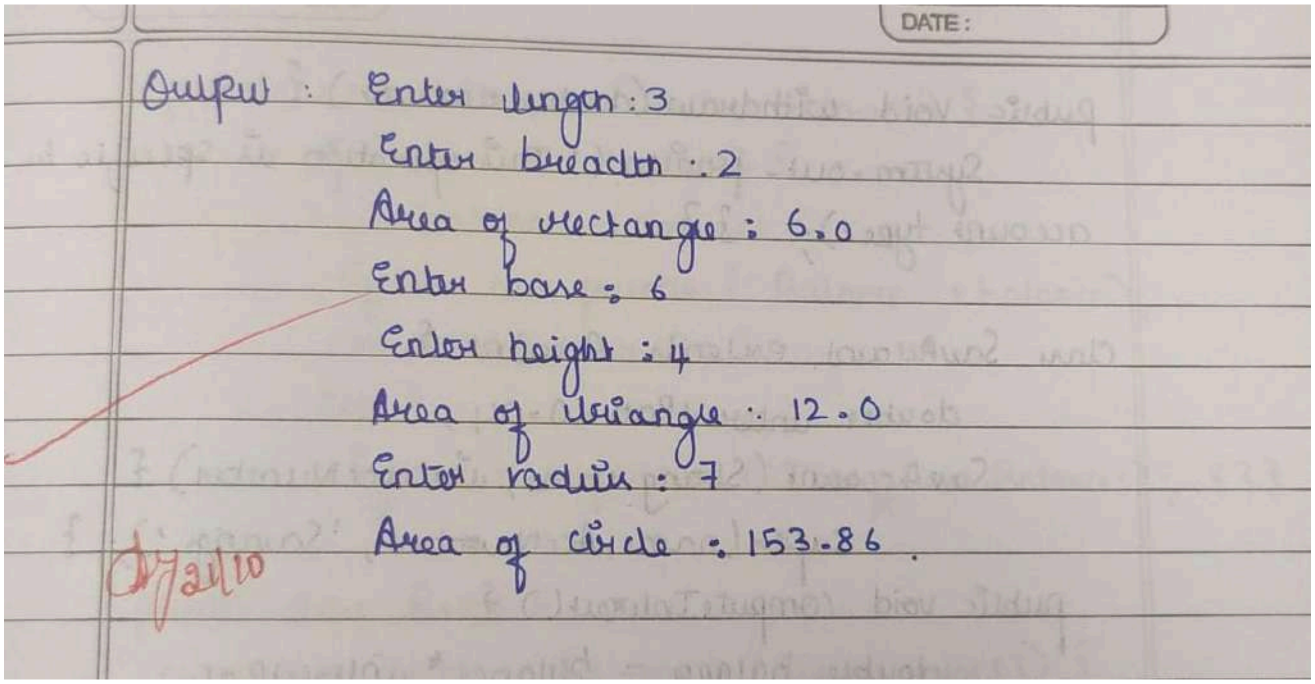
9) Develop a Java program to create abstract class called Shape that contains 2 integers and an empty method name printArea(). Provide 3 classes rectangle, triangle, circle such that each extends Shape. Each contain printArea() that prints area of shape.

```
import java.util.Scanner;
obstract class Shape {
int a, b;
double result;
abstract void PrintArea();
}
class Rectangle extends Shape {
void PrintArea() {
Scanner sc = new Scanner(System.in);
System.out.println("Enter length:");
a = sc.nextInt();
System.out.println("Enter breadth:");
b = sc.nextInt();
result = a*b;
System.out.println("Area of rectangle", + ~~area~~ result);
} }
```



PAGE NO :
DATE :

```
class Triangle extends Shape {
void PrintArea() {
Scanner sc = new Scanner (System.in);
System.out.println ("Enter base :");
a = sc.nextInt();
System.out.println ("Enter height :");
b = sc.nextInt();
result = 0.5*b*h;
System.out.println ("Area of triangle :"+result);
} }
class Circle extends Shape {
void PrintArea() {
Scanner sc = new Scanner (System.in);
System.out.println ("Enter radius :");
a = sc.nextInt();
result = 3.14*a*a;
System.out.println ("Area of circle :"+result);
} }
class Main {
public static void Main (String args[]) {
Rectangle r = new Rectangle();
r.PrintArea();
Triangle t = new Triangle();
t.PrintArea();
Circle c = new Circle();
c.PrintArea();
} }
```



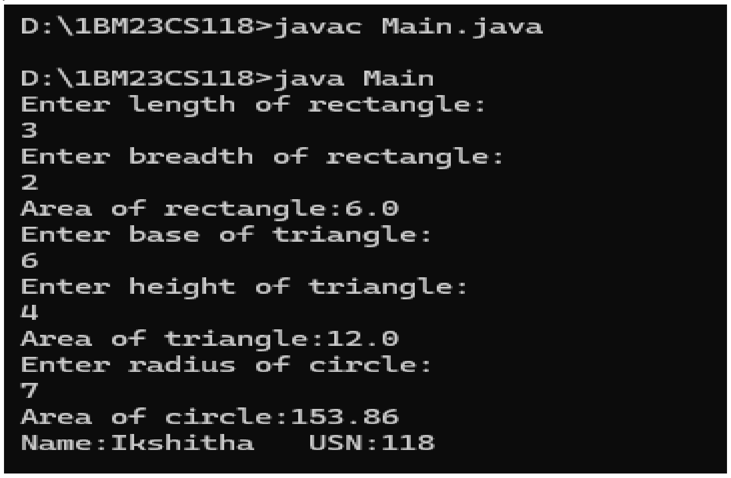
DATE :

```
Output : Enter length : 3
         Enter breadth : 2
         Area of rectangle : 6.0
         Enter base : 6
         Enter height : 4
         Area of triangle : 12.0
         Enter radius : 7
         Area of circle : 153.86 .
```

[illegible] 21/10

Code:

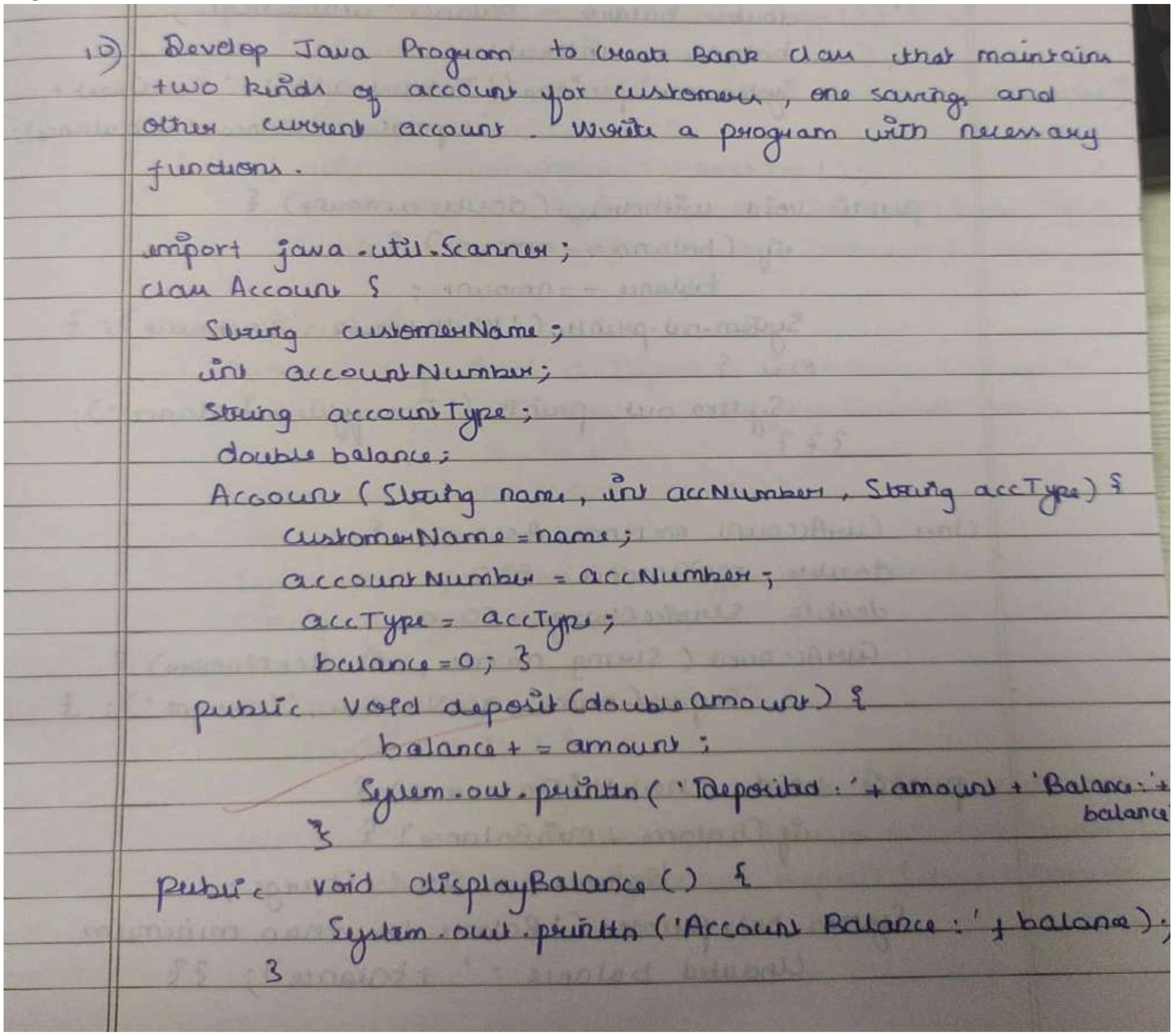
```
import java.util.Scanner;
abstract class Shape{
int a,b;
double result;
abstract void PrintArea();
}
class Rectangle extends Shape
{
void PrintArea()
{
Scanner sc=new Scanner(System.in);
System.out.println("Enter length of rectangle:");
a=sc.nextInt();
System.out.println("Enter breadth of rectangle:");
b=sc.nextInt();
result=a*b;
System.out.println("Area of rectangle:"+result);
}
}
class Triangle extends Shape
{
void PrintArea()
{
Scanner sc=new Scanner(System.in);
System.out.println("Enter base of triangle:");
a=sc.nextInt();
System.out.println("Enter height of triangle:");
b=sc.nextInt();
result=0.5*a*b;
System.out.println("Area of triangle:"+result);
}
}
class Circle extends Shape
{
void PrintArea()
{

Scanner sc=new Scanner(System.in);
System.out.println("Enter radius of circle:");
a=sc.nextInt();
result=3.14*a*a;
System.out.println("Area of circle:"+result);
}
}
class Main
{
public static void main(String args[])
{
Rectangle r=new Rectangle();
r.PrintArea();
Triangle t=new Triangle();
t.PrintArea();
Circle c=new Circle();
c.PrintArea();
System.out.println("Name:Ikshitha   USN:118");

}
}
```



```
D:\1BM23CS118>javac Main.java

D:\1BM23CS118>java Main
Enter length of rectangle:
3
Enter breadth of rectangle:
2
Area of rectangle:6.0
Enter base of triangle:
6
Enter height of triangle:
4
Area of triangle:12.0
Enter radius of circle:
7
Area of circle:153.86
Name:Ikshitha   USN:118
```

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

Algorithm:



10) Develop Java Program to create Bank class that maintains two kinds of account for customers, one saving and other current account. Write a program with necessary functions.

```
import java.util.Scanner;
class Account {
    String customerName;
    int accountNumber;
    String accountType;
     double balance;
    Account(String name, int accNumber, String accType) {
        customerName=name;
        accountNumber = accNumber;
         accType = accType;
         balance=0; }
   public void deposit(double amount) {
              balance += amount;
              System.out.println('Deposited :'+amount + 'Balance:'+
                                                              balance
         }
 public void displayBalance() {
       System.out.println('Account Balance :'+balance);
     }
```

PAGE NO :
DATE :

```
public void withdraw (double amount) {
        System.out.println ('This operation is specific to
 account type');   }}

class SavAccount extends Account {
        double interestRate = 0.04 ;
      SavAccount (String name, int accNumber) {
                super(name, AccNumber, 'Savings'); }
  public void computeInterest() {
        double balance = balance * interestRate ;
        balance += interest ;
        System.out.println ('Interest added :' +interest +
                         'Updated balance :' + balance); }

public void withdraw(double amount) {
        if (balance >= amount) {
                balance -= amount ;
        System.out.println('Withdrawn :'+amount); }
      else {
        System.out.println('Insufficient balance');
  }}}

class CurrAccount extends Account {
    double   minBalance = 500.0 ;
    double   serviceCharge = 50.0 ;
     CurrAccount ( String name , int AccNumber) {
            Super (name , accNumber , 'Current'); }

public void checkMinBalance () {
        if (balance < minBalance) {
                balance -= Service Charge ;
    System.out.println ('Balance less than minimum
        Updated balance : ' +balance); }}
```

PAGE NO :
DATE :

```
public void withdraw (double amount) {
        if (balance >= amount) {
            balance -= amount ;
          System.out.println ('Balance' + balance);
          checkMinBalance (); }
        else {
          System.out.println ('Insufficient Balance'); }}}

public class Bank {
    public static void main (String args[]) {
          Scanner sc = new Scanner (System.in );
        System.out.println ('Enter name and account number');
        String name = sc.next ();
        int accountnumber = sc.nextInt ();
        SavAccount savingAccount = new SavAccount (name, accountnumber);
        CurAccount currentAccount = new CurAccount ( name1, accountnumber);

while(True) {
    System.out.println(' 1. Deposit 2. Withdraw 3. Compute
                     Interest 4. Display Account Details
                     5. Exit ');
    System.out.println ('Enter choice');
    int choice = sc.nextInt ();
    System.out.println ('Ente type of account ');
    String accType = sc.next();

    if (accType.equals ('Saving')) {
      switch (choice) {
          case 1 : System.out.println ('Enter deposit amount'
                   double depositAmount = sc.nextDouble();
                   savingAccount.deposit (depositAmount);
                   break ;
```

DATE :

```
case 2 : System.out.println('Enter withdrawal amount');
    double withdrawalAmount = sc.nextDouble();
    savingsAccount.withdraw(withdrawalAmount);
    break;
case 3 : savingsAccount.computeInterest();
    break;
case 4 : System.out.println('Customer name : ' + savingsAccount.
                                                  customerName);

     System.out.println('Account number : ' +
                              savingsAccount.account
                                           Number);

     System.out.println('Type of Account : ' + savingsAccount.
                                           accountType);

     savingsAccount.displayBalance();
     breaks;
case 5 : System.exit(0);
     break;
default : System.out.println('Invalid choice');
     } }
else if (accType.equals('current')) {
  switch(choice) {
    case 1 : System.out.print('Enter amount : ');
         double amount = sc.nextDouble();
         currentAccount.deposit(amount);
         break;
    case 2 : System.out.print('Enter amount : ');
         double amount = sc.nextDouble();
         currentAccount.withdraw(amount);
         break;
    case 3 : System.out.println('No interest');
         break;
    case 4 : System.out.println('Name : ' + currentAccount.
                                            customerName);
         break;
```

PAGE NO :
DATE :

```
case 5 :  System.exit(0);
          break;
default :  System.out.println("Invalid choice");
 }
} else {
    System.out.println("Invalid account type");
 }
}  } }
```

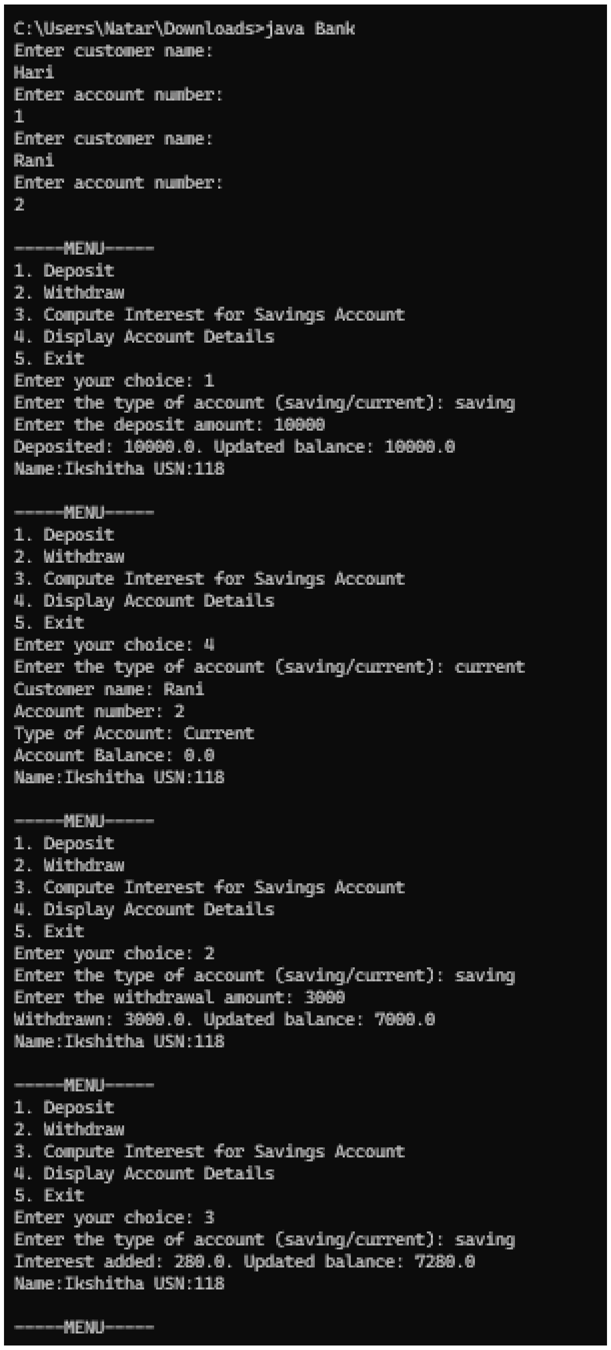
OUTPUT :

```
Enter customer name : Hari
Enter account number : 1
Enter customer name : Rani
Enter account number : 2
1. Deposit
2. Withdraw
3. Compute interest
4. Display account details
5. Exit
Enter choice : 1
Enter type of account : saving
Enter deposit amount : 10000
Deposited 10000.0  Updated balance : 10000
```

[illegible] 28-10-2021

Code:

```
import java.util.Scanner;

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


    Account(String name, int accNumber, String accType) {
        customerName = name;
        accountNumber = accNumber;
        accountType = accType;
        balance = 0;
    }


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

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


    public void withdraw(double amount) {
        System.out.println("This operation is specific to account type.");
    }
}


class SavAccount extends Account {
    double interestRate = 0.04;  // 4% annual interest rate

    SavAccount(String name, int accNumber) {
        super(name, accNumber, "Savings");
    }


    public void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest added: " + interest + ". Updated balance: " + balance);
    }


    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ". Updated balance: " + balance);
        } else {
            System.out.println("Insufficient balance.");
        }
    }
}
class CurAccount extends Account {
    double minBalance = 500.0;
    double serviceCharge = 50.0;

    CurAccount(String name, int accNumber) {
        super(name, accNumber, "Current");
    }

    public void checkMinBalance() {
        if (balance < minBalance) {
            balance -= serviceCharge;
            System.out.println("Balance below minimum. Service charge imposed: " + serviceCharge + ".
Updated balance: " + balance);
        }
    }


    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ". Updated balance: " + balance);
            checkMinBalance();
        } else {
            System.out.println("Insufficient balance.");
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter customer name:");
        String name=sc.next();
        System.out.println("Enter account number:");
        int accountnumber=sc.nextInt();
        SavAccount savingsAccount = new SavAccount(name, accountnumber);
System.out.println("Enter customer name:");
        String name1=sc.next();
        System.out.println("Enter account number:");
        int accountnumber1=sc.nextInt();
        CurAccount currentAccount = new CurAccount(name1, accountnumber1);

        while (true) {
            System.out.println("\n-----MENU-----");
            System.out.println("1. Deposit\n2. Withdraw\n3. Compute Interest for Savings Account\n4.
Display Account Details\n5. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();

            System.out.print("Enter the type of account (saving/current): ");
            String accType = sc.next();

            if (accType.equals("saving")) {
                switch (choice) {
                    case 1:
                        System.out.print("Enter the deposit amount: ");
                        double depositAmount = sc.nextDouble();
                        savingsAccount.deposit(depositAmount);
                        break;
                    case 2:
                        System.out.print("Enter the withdrawal amount: ");
                        double withdrawalAmount = sc.nextDouble();
                        savingsAccount.withdraw(withdrawalAmount);
                        break;
                    case 3:
                        savingsAccount.computeInterest();
                        break;
                    case 4:
                        System.out.println("Customer name: " + savingsAccount.customerName);
                        System.out.println("Account number: " + savingsAccount.accountNumber);
                        System.out.println("Type of Account: " + savingsAccount.accountType);
                        savingsAccount.displayBalance();
                        break;
                    case 5:
                        System.exit(0);
                        break;
                    default:
                        System.out.println("Invalid choice.");
                }
            } else if (accType.equals("current")) {
                switch (choice) {
                    case 1:
                        System.out.print("Enter the deposit amount: ");
                        double depositAmount = sc.nextDouble();
                        currentAccount.deposit(depositAmount);
                        break;
                    case 2:
                        System.out.print("Enter the withdrawal amount: ");
                        double withdrawalAmount = sc.nextDouble();
                        currentAccount.withdraw(withdrawalAmount);
                        break;
                    case 3:
                        System.out.println("Current accounts do not earn interest.");
                        break;
                    case 4:
                        System.out.println("Customer name: " + currentAccount.customerName);
                        System.out.println("Account number: " + currentAccount.accountNumber);
                        System.out.println("Type of Account: " + currentAccount.accountType);
                        currentAccount.displayBalance();
                        break;
                    case 5:
                        System.exit(0);
                        break;
                    default:
                        System.out.println("Invalid choice.");
                }
            } else {
                System.out.println("Invalid account type.");
            }
        }
    }
}
```

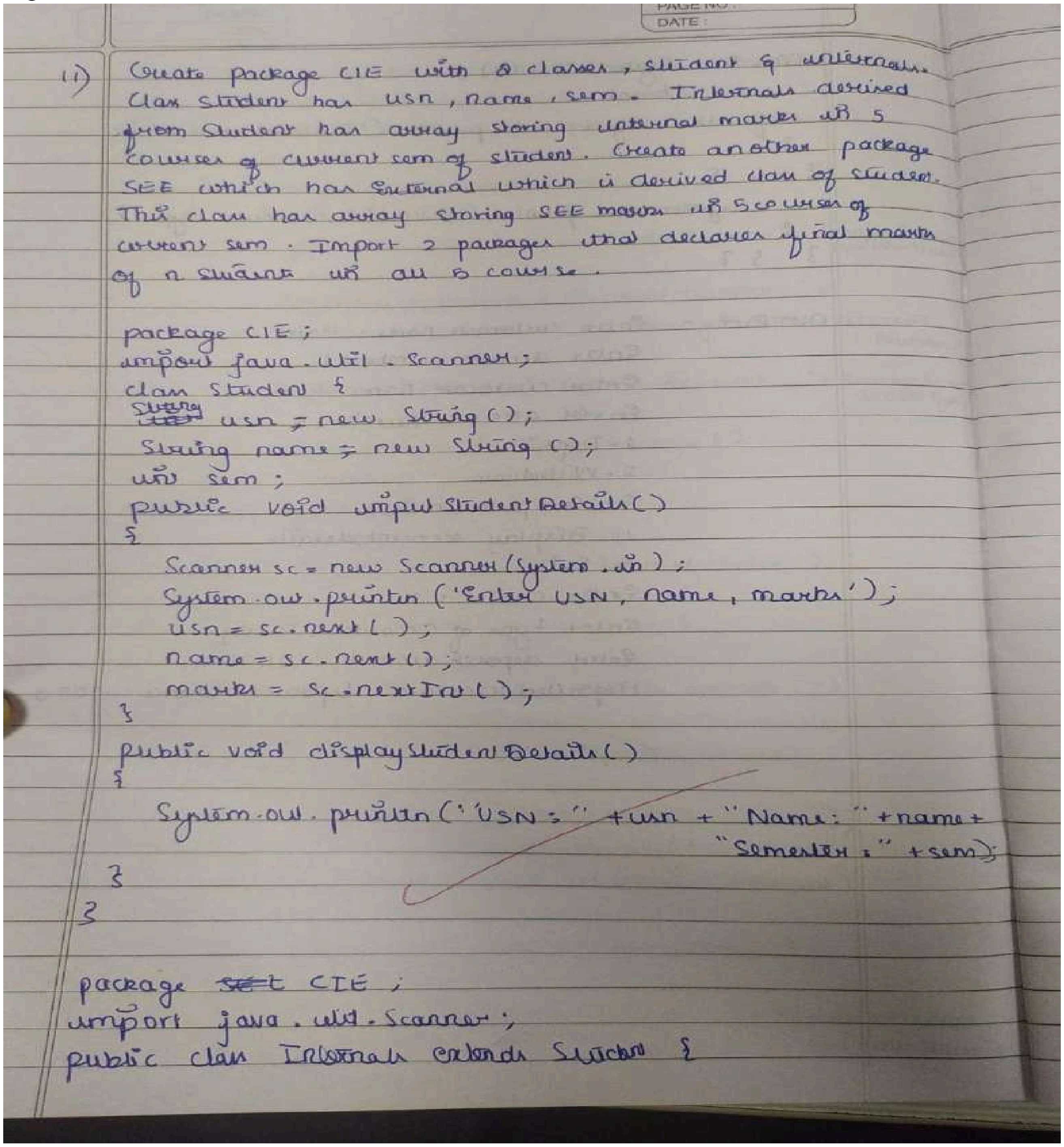


```
C:\Users\Natar\Downloads>java Bank
Enter customer name:
Hari
Enter account number:
1
Enter customer name:
Rani
Enter account number:
2

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): saving
Enter the deposit amount: 10000
Deposited: 10000.0. Updated balance: 10000.0
Name:Ikshitha USN:118

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 4
Enter the type of account (saving/current): current
Customer name: Rani
Account number: 2
Type of Account: Current
Account Balance: 0.0
Name:Ikshitha USN:118

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account (saving/current): saving
Enter the withdrawal amount: 3000
Withdrawn: 3000.0. Updated balance: 7000.0
Name:Ikshitha USN:118

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 3
Enter the type of account (saving/current): saving
Interest added: 280.0. Updated balance: 7280.0
Name:Ikshitha USN:118

-----MENU-----
```

Program 6

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

Algorithm:



PAGE NO.:
DATE:

1.) Create package CIE with 2 classes, student & internals. Class Student has usn, name, sem. Internals derived from Student has array storing internal marks in 5 courses of current sem of student. Create another package SEE which has External which is derived class of student. This class has array storing SEE marks in 5 courses of current sem. Import 2 packages that declares final marks of n student in all 5 course.

```
package CIE;
import java.util.Scanner;
class Student {
 String usn = new String();
 String name = new String();
 int sem;
public void inputStudentDetails()
{
   Scanner sc = new Scanner(System.in);
   System.out.println('Enter USN, name, marks');
   usn = sc.next();
   name = sc.next();
   marks = sc.nextInt();
}
public void displayStudentDetails()
{
   System.out.println("USN: " +usn + "Name: " +name +
                              "Semester: " +sem);
}
}

package ~~SEE~~ CIE;
import java.util.Scanner;
public class Internals extends Student {
```

PAGE NO :
DATE :

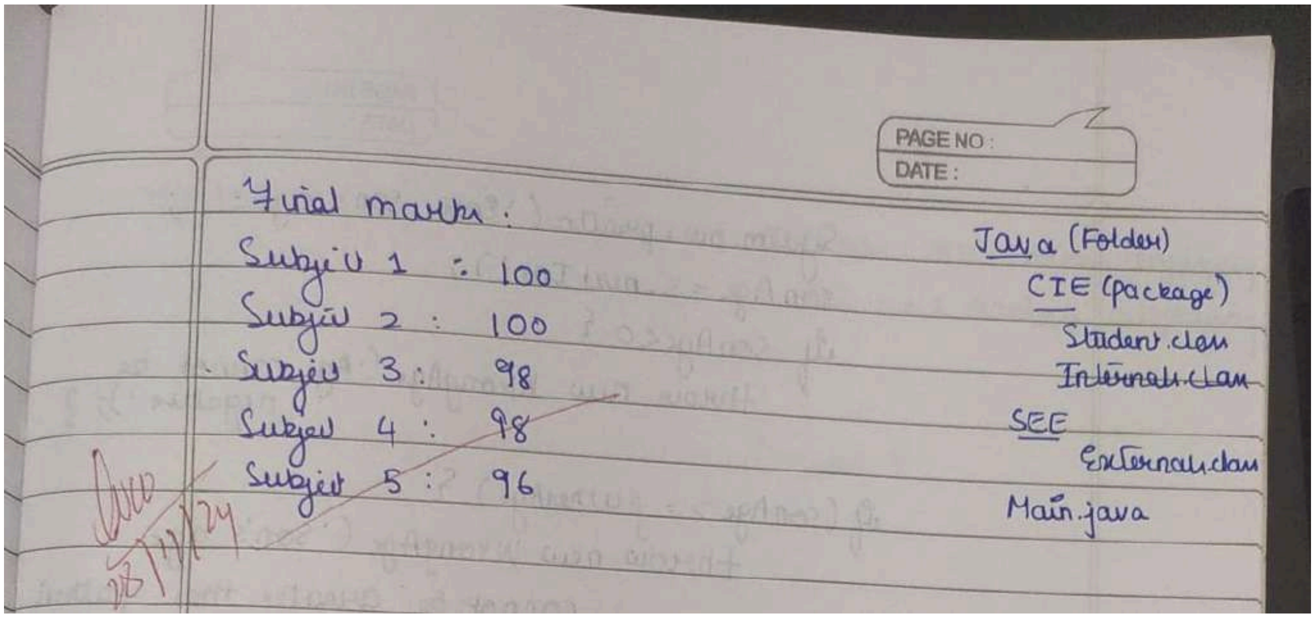
```
int marks[] = new int [5] ;
void input inputCIEmarks () {
 Scanner sc= new Scanner (System.in);
  for (int i=0; i<5 ; i++) {
          System.out.println ('Marks :');
          marks[i]= sc.nextInt (); }
  } }

package SEE ;
import CIE.~~marks~~Internals ;
import java.util.Scanner;
public class Externals extends Internals {
       int marks[] = new int [5] ;
       int final [] = new int [5] ;
    Externals () {
         marks= new int [5] ;
          final = new int [5] ;   }
public void inputSEEmarks () {
       Scanner sc=new Scanner (System.in);
      for ( int i=0 ; i<5 ; i++) {
            Sopln ('Marks :');
            marks[i] = sc.next Int ;
             final.marks[i] ~~+~~= marks[i]  + this.marks[i];}}
public void displayFinal () {
      display student ();
      for (int i=0 ; i<5 ; i++ ) {
            System.out.println ( final marks[i]); }}}

import SEE.Externals;
import java.util.Scanner;
class Main {
     Void main ( String args[]) {
      Scanner sc=new Scanner (System.in);
```

PAGE NO :
DATE :

```
    int n=sc.nextInt();
    Externals s[] = new Externals [n]
   for(int i=0; i<n; i++)  {
      ~~Students~~ s[i]=new Externals();
      s[i].inputStudent();
      s[i]. inputCIEmarks();
      s[i]. inputSEEmarks();
      s[i]. calcFinalmarks();
   }
   Sopln ('Final');
   for(int i=0; i<n; i++)
       s[i].displayFinalMarks();
} }
```

Output

```
Enter number of students : 1
 USN : 1BM23CS118
 Name : Ikshitha . P
 Semester : 3
 CIE marks :
 Subject 1 : 48
 Subject 2 : 48
 Subject 3 : 50
 Subject 4 : 47
 Subject 5 : 48
SEE marks :
 Subject 1 : 50
 Subject 2 :  50
 Subject 3 :  49
 Subject 4 :  49
 Subject 5  48
```


PAGE NO :
DATE :

Final marks :
Subject 1 : 100
Subject 2 : 100
Subject 3 : 98
Subject 4 : 98
Subject 5 : 96

Java (Folder)
CIE (package)
Student.class
Internals.class
SEE
External.class
Main.java

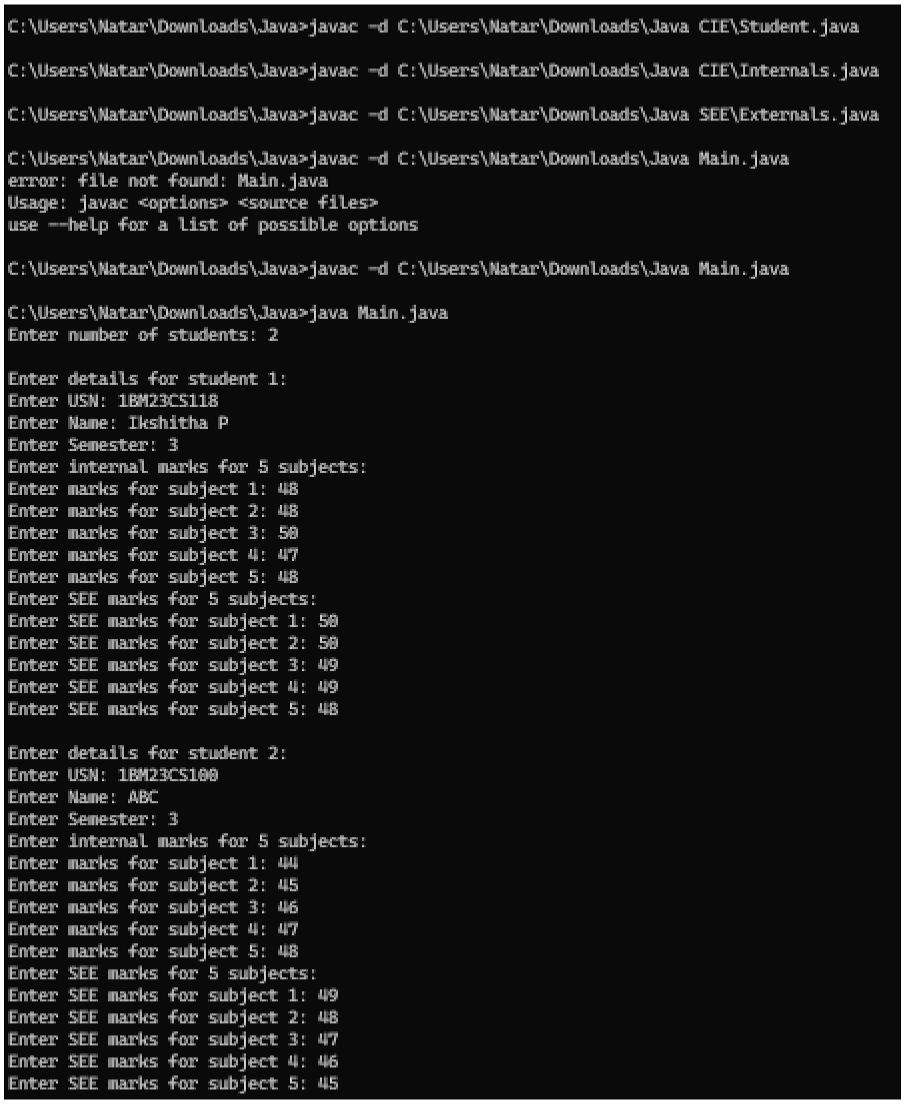
[illegible]
28/11/24

Code:

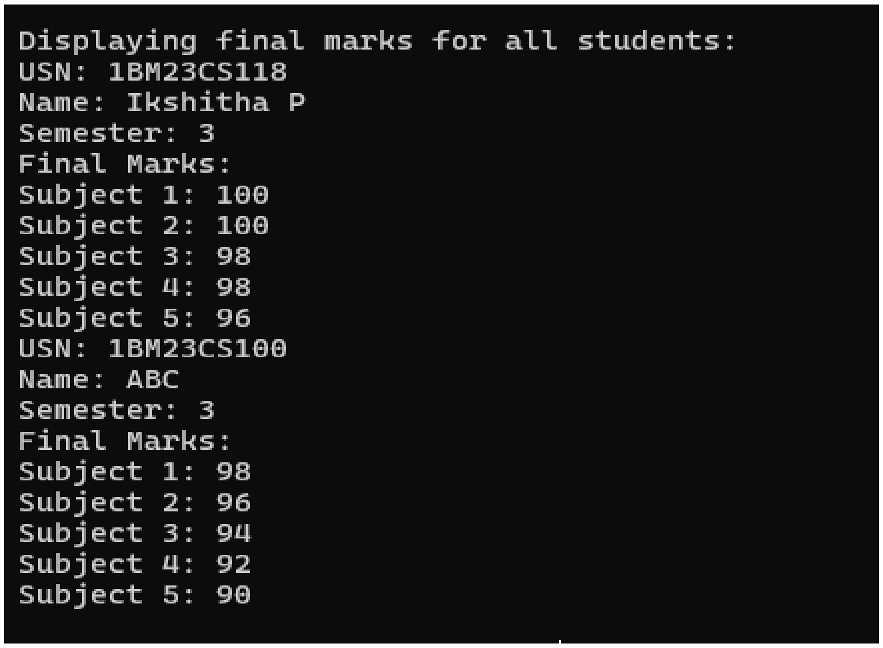
```
package CIE;

import java.util.Scanner;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;
    public void inputStudentDetails() {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = s.nextLine();
        System.out.print("Enter Name: ");
        name = s.nextLine();
        System.out.print("Enter Semester: ");
        sem = s.nextInt();
    }

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


package CIE;
```

```
import java.util.Scanner;

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

    // Method to input internal marks
    public void inputCIEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter internal marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = s.nextInt();
        }
    }
}
```

```
package SEE;

import CIE.Internals;
import java.util.Scanner;

public class Externals extends Internals {
    protected int[] marks = new int[5];
    protected int[] finalMarks = new int[5];

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

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

    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = marks[i] + this.marks[i]; // Final marks = internal + external
        }
    }
    public void displayFinalMarks() {
        displayStudentDetails();  // Display student details (inherited from Student)
        System.out.println("Final Marks:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Subject " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}



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

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

        // Input number of students
        System.out.print("Enter number of students: ");
        int n = s.nextInt();
        s.nextLine(); // Consume newline

        Externals[] students = new Externals[n];

        for (int i = 0; i < n; i++) {
            students[i] = new Externals();
            System.out.println("\nEnter details for student " + (i + 1) + ":");
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
            students[i].calculateFinalMarks();
        }

        System.out.println("\nDisplaying final marks for all students:");
        for (int i = 0; i < n; i++) {
            students[i].displayFinalMarks();
        }

        s.close();
    }
}
```



```
C:\Users\Natar\Downloads\Java>javac -d C:\Users\Natar\Downloads\Java CIE\Student.java

C:\Users\Natar\Downloads\Java>javac -d C:\Users\Natar\Downloads\Java CIE\Internals.java

C:\Users\Natar\Downloads\Java>javac -d C:\Users\Natar\Downloads\Java SEE\Externals.java

C:\Users\Natar\Downloads\Java>javac -d C:\Users\Natar\Downloads\Java Main.java
error: file not found: Main.java
Usage: javac <options> <source files>
use --help for a list of possible options

C:\Users\Natar\Downloads\Java>javac -d C:\Users\Natar\Downloads\Java Main.java

C:\Users\Natar\Downloads\Java>java Main.java
Enter number of students: 2

Enter details for student 1:
Enter USN: 1BM23CS118
Enter Name: Ikshitha P
Enter Semester: 3
Enter internal marks for 5 subjects:
Enter marks for subject 1: 48
Enter marks for subject 2: 48
Enter marks for subject 3: 50
Enter marks for subject 4: 47
Enter marks for subject 5: 48
Enter SEE marks for 5 subjects:
Enter SEE marks for subject 1: 50
Enter SEE marks for subject 2: 50
Enter SEE marks for subject 3: 49
Enter SEE marks for subject 4: 49
Enter SEE marks for subject 5: 48

Enter details for student 2:
Enter USN: 1BM23CS100
Enter Name: ABC
Enter Semester: 3
Enter internal marks for 5 subjects:
Enter marks for subject 1: 44
Enter marks for subject 2: 45
Enter marks for subject 3: 46
Enter marks for subject 4: 47
Enter marks for subject 5: 48
Enter SEE marks for 5 subjects:
Enter SEE marks for subject 1: 49
Enter SEE marks for subject 2: 48
Enter SEE marks for subject 3: 47
Enter SEE marks for subject 4: 46
Enter SEE marks for subject 5: 45
```

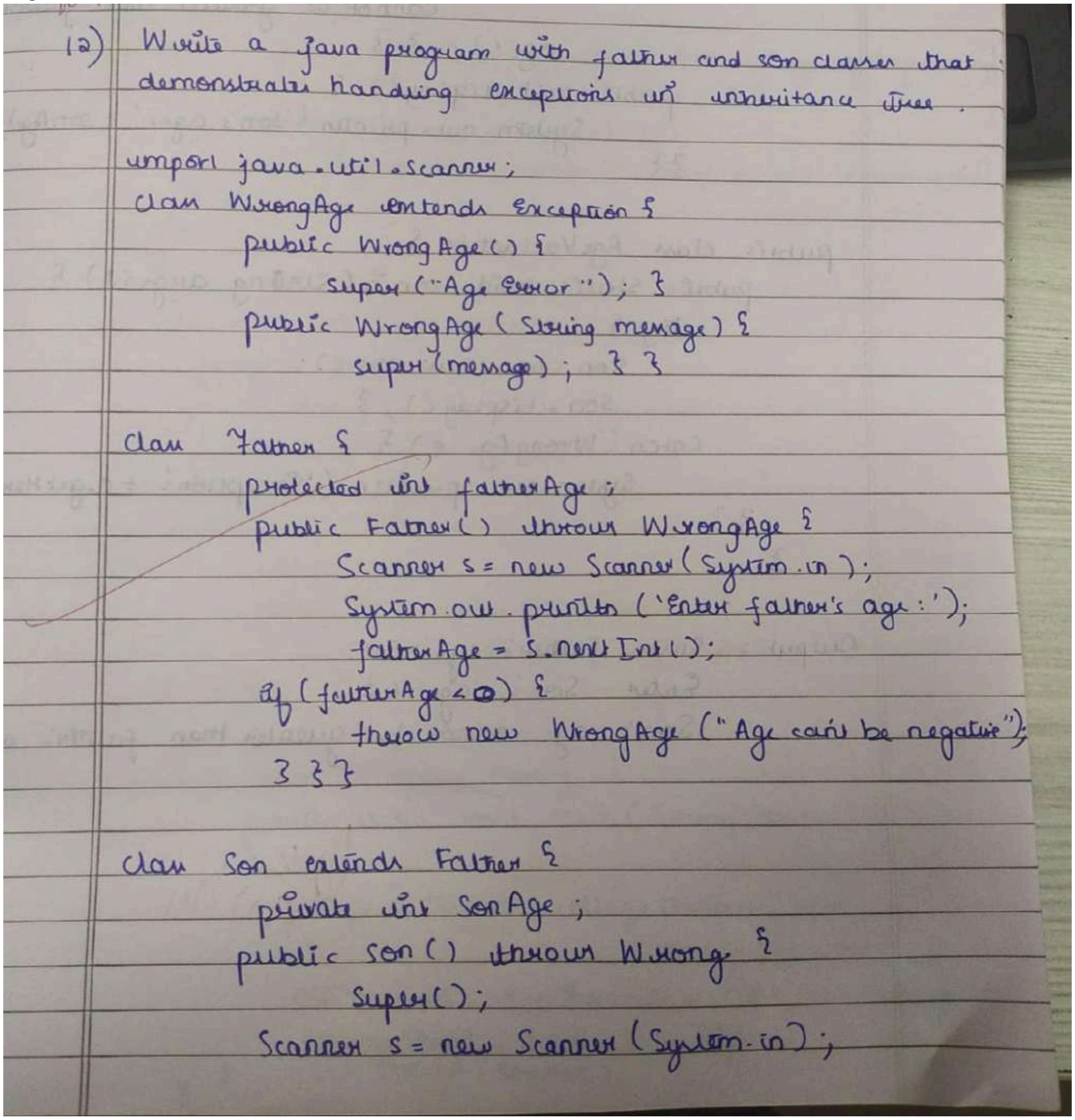


```
Displaying final marks for all students:
USN: 1BM23CS118
Name: Ikshitha P
Semester: 3
Final Marks:
Subject 1: 100
Subject 2: 100
Subject 3: 98
Subject 4: 98
Subject 5: 96
USN: 1BM23CS100
Name: ABC
Semester: 3
Final Marks:
Subject 1: 98
Subject 2: 96
Subject 3: 94
Subject 4: 92
Subject 5: 90
```

Program 7

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is >=father’s age.

Algorithm:



12) Write a java program with father and son classes that demonstrate handling exceptions in inheritance tree.

```
import java.util.Scanner;
class WrongAge extends Exception {
        public WrongAge() {
                super ("Age Error"); }
        public WrongAge (String message) {
                super (message);   } }

class Father {
        protected int fatherAge;
        public Father() throws WrongAge {
                Scanner s = new Scanner(System.in);
                System.out.println ('Enter father's age:');
                fatherAge = s.nextInt();
        if (fatherAge < 0) {
                throw new WrongAge ("Age can't be negative");
        } } }

class Son extends Father {
        private int SonAge;
        public son() throws Wrong {
                super();
        Scanner s = new Scanner(System.in);
```

PAGE NO :
DATE :

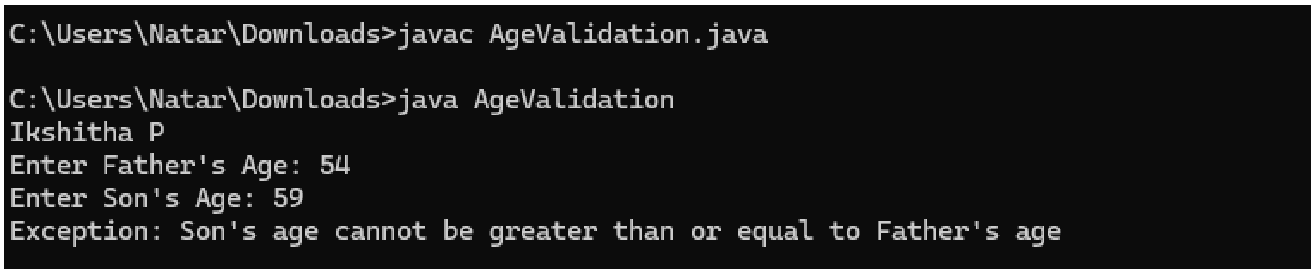
```
System.out.println('Enter son's age :');
son Age = s.nextInt();
if sonAge<0 {
    throw new WrongAge('Age cannot be
                       negative'); }

if (sonAge > = fatherAge) {
    throw new WrongAge('Son's age
              cannot be greater than father's
        age'); } }
public void display() {
    System.out.println('Son's age:' + sonAge);
} }

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

Output :
```
Enter Father's age : 51
Enter Son's age : 62
Son's age cannot be greater than father's age.
```

[illegible]
28/11/24

Code:

```
import java.util.Scanner;
class WrongAge extends Exception {
public WrongAge() {
        super("Age Error");
    }
public WrongAge(String message) {
     super(message);
    }
}

class Father {
    protected int fatherAge;
public Father() throws WrongAge {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter Father's Age: ");
        fatherAge = s.nextInt();
 if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }
}

class Son extends Father {
    private int sonAge;
 public Son() throws WrongAge {
        super();
Scanner s = new Scanner(System.in);
   System.out.print("Enter Son's Age: ");
        sonAge = s.nextInt();

      if (sonAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to Father's age");
        }
    }


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

public class AgeValidation {
    public static void main(String[] args) {
System.out.println("Ikshitha P");
        try {

            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Exception: " + e.getMessage());

        }
    }
}
```

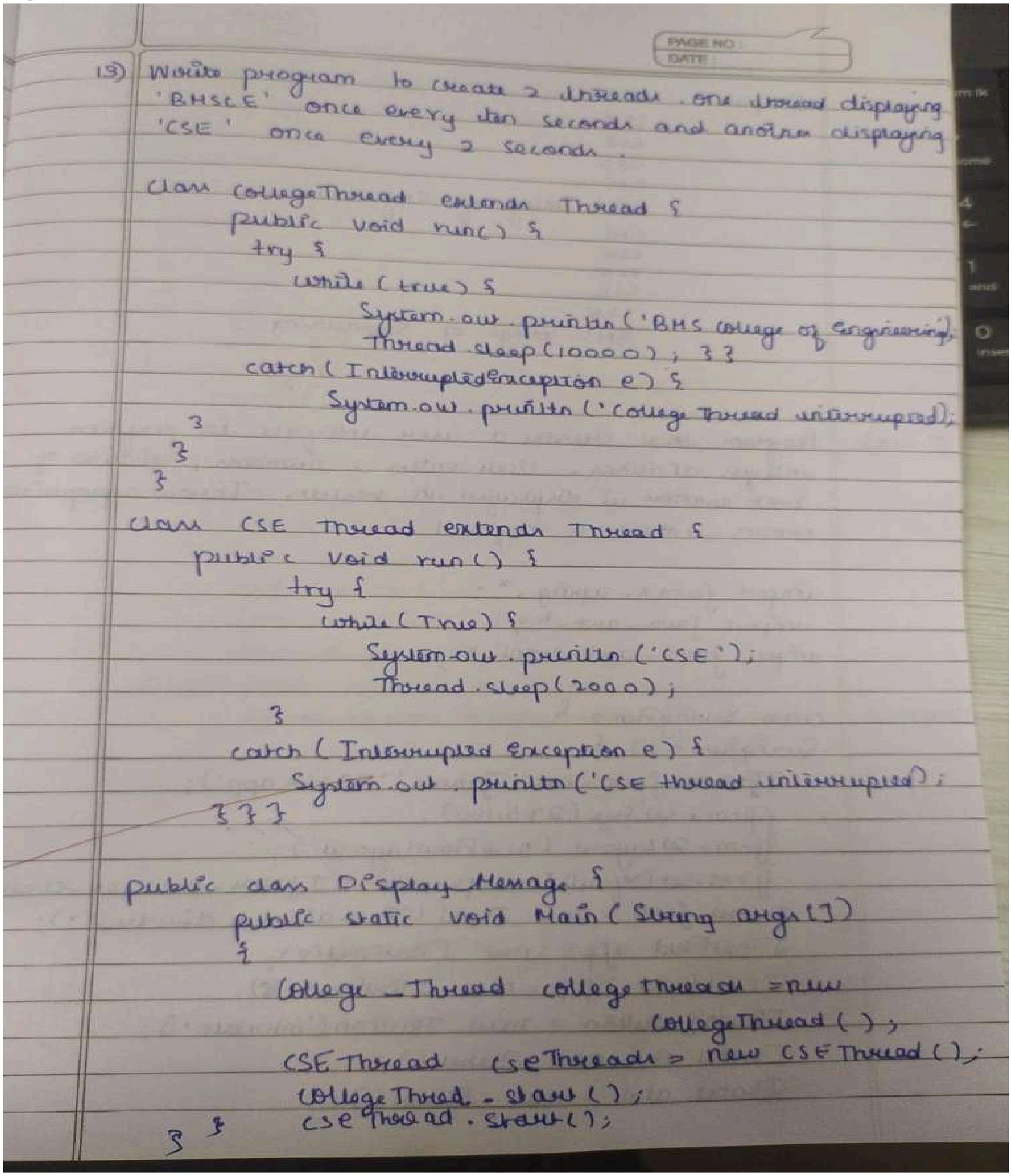


```
C:\Users\Natar\Downloads>javac AgeValidation.java

C:\Users\Natar\Downloads>java AgeValidation
Ikshitha P
Enter Father's Age: 54
Enter Son's Age: 59
Exception: Son's age cannot be greater than or equal to Father's age
```

Program 8

Write a program which creates two threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds.

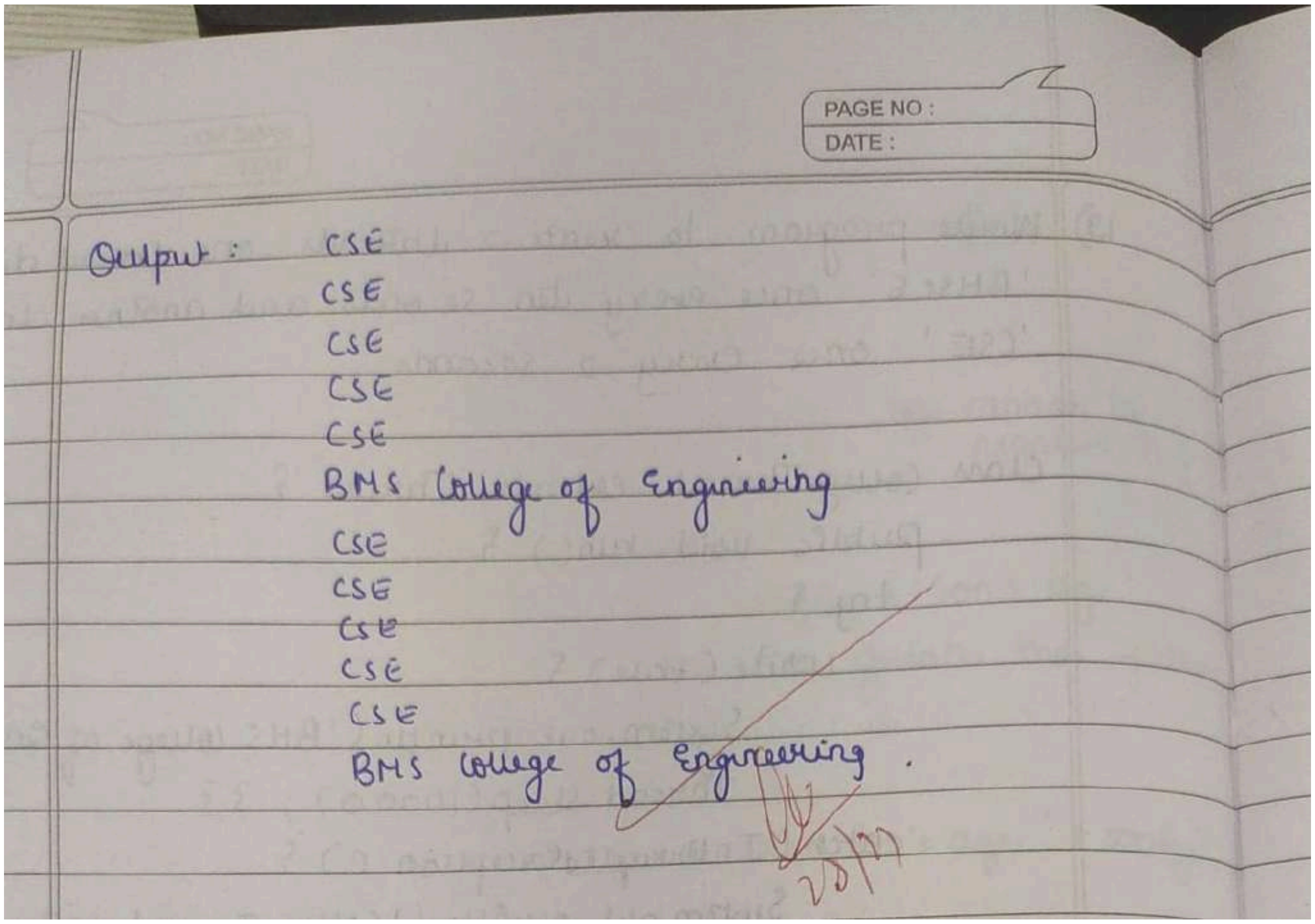
Algorithm:



PAGE NO:
DATE:

13) Write program to create 2 threads. one thread displaying 'BMSCE' once every ten seconds and another displaying 'CSE' once every 2 seconds.

```
class CollegeThread extends Thread {
    public void run() {
      try {
         while (true) {
             System.out.println('BMS college of Engineering');
             Thread.sleep(10000); }}
      catch (InterruptedException e) {
           System.out.println('College Thread interrupted);
      }
   }
}

class CSE Thread extends Thread {
  public void run() {
       try {
         while(True) {
             System.out.println('CSE');
             Thread.sleep(2000);
       }
      catch (Interrupted Exception e) {
          System.out.println('CSE thread interrupted);
   }}}

public class DisplayMessage {
     public static void Main (String args[])
     {
        College_Thread collegeThreadsu = new
                               CollegeThread();
        CSEThread cseThreads = new CSEThread();
        CollegeThread.start();
        cseThread.start();
}  }
```



PAGE NO :
DATE :

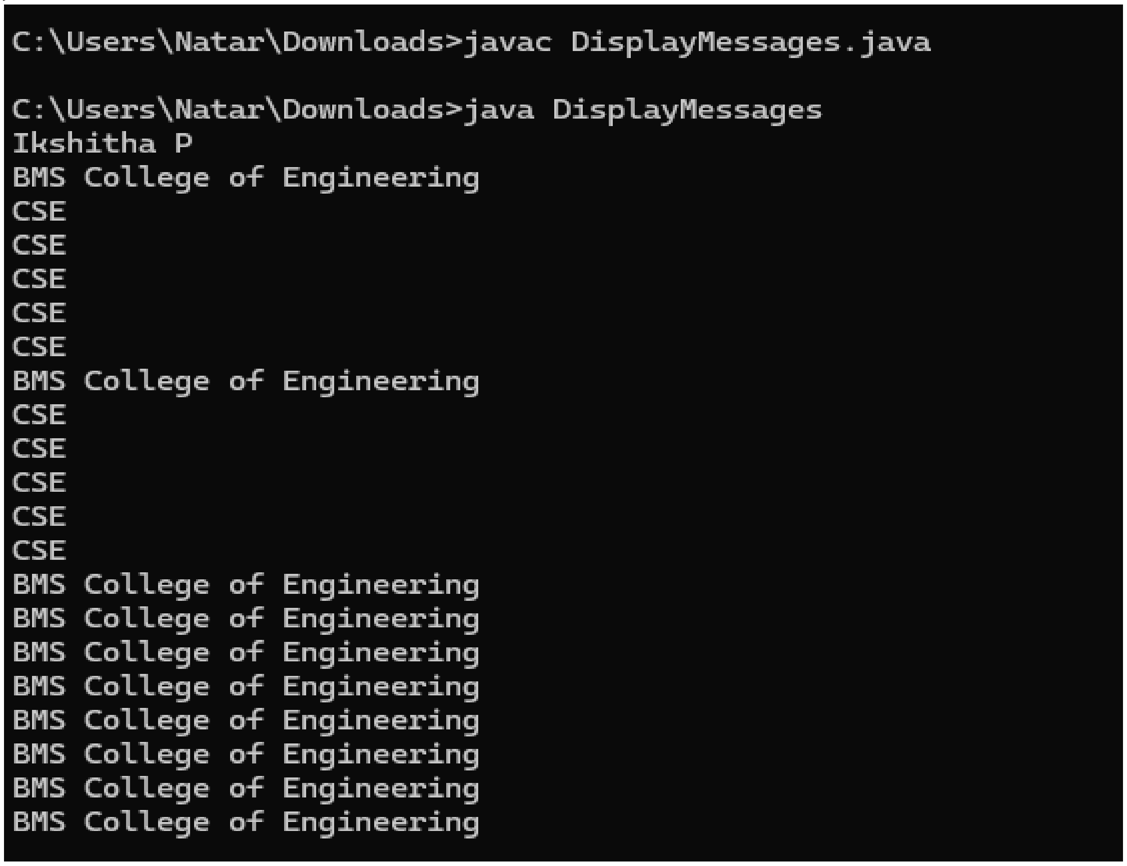
Output :

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

20/12

Code:

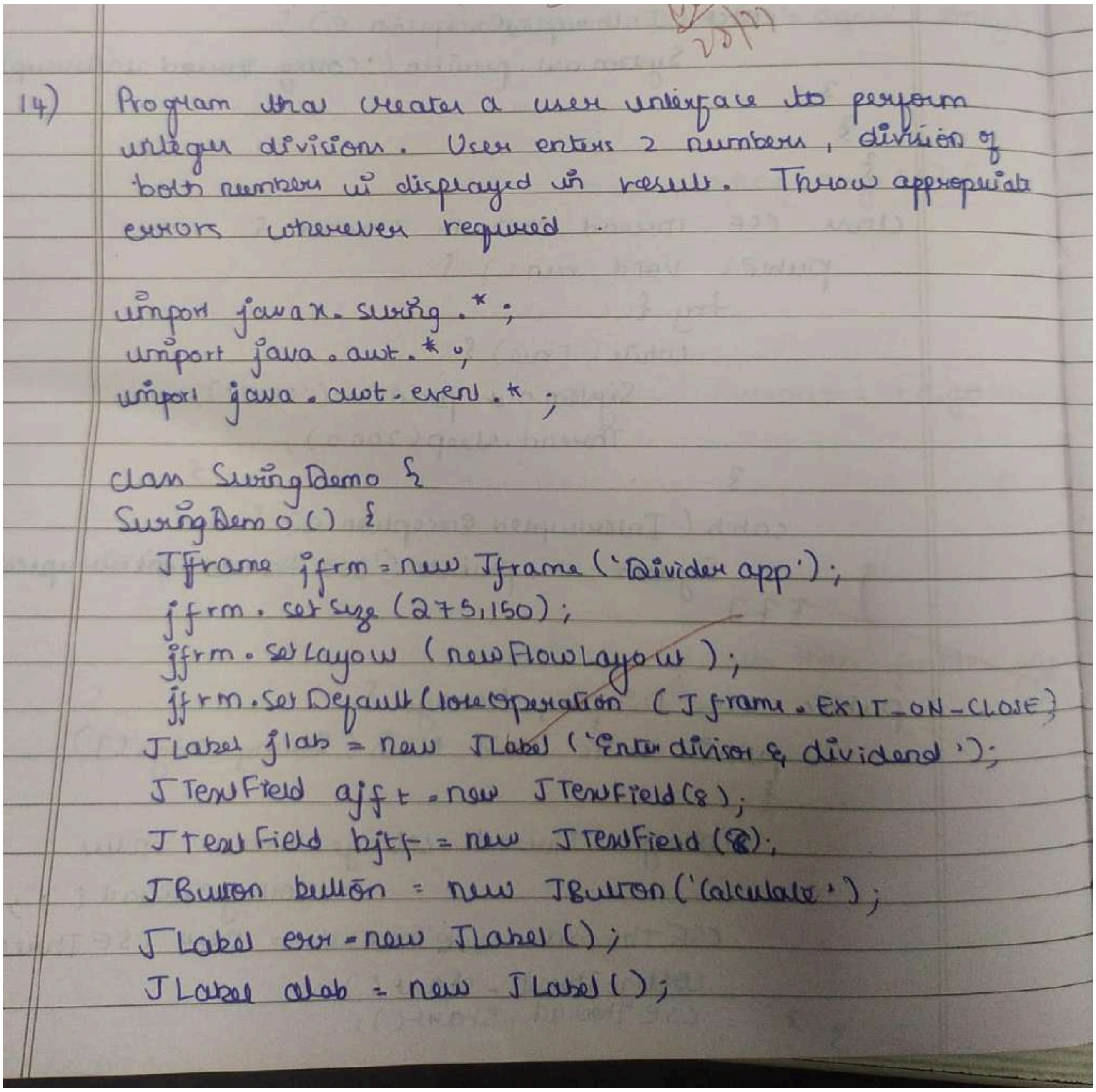
```
class CollegeThread extends Thread {
   public void run() {
      try {
         for(int i=1;i<=10;i++){
            System.out.println("BMS College of Engineering");
            Thread.sleep(10000);
         }
      } catch (InterruptedException e) {
         System.out.println("CollegeThread interrupted: " + e.getMessage());
      }
   }
}


class CSEThread extends Thread {
   public void run() {
      try {
         for(int i=1;i<=10;i++){
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println("CSEThread interrupted: " + e.getMessage());
        }
    }
}

// Main class to run the threads
public class DisplayMessages {
    public static void main(String[] args) {

        CollegeThread collegeThread = new CollegeThread();
        CSEThread cseThread = new CSEThread();


        collegeThread.start();
        cseThread.start();
System.out.println("Ikshitha P");
    }
}
```



```
C:\Users\Natar\Downloads>javac DisplayMessages.java

C:\Users\Natar\Downloads>java DisplayMessages
Ikshitha P
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
```

Program 9
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

Algorithm:

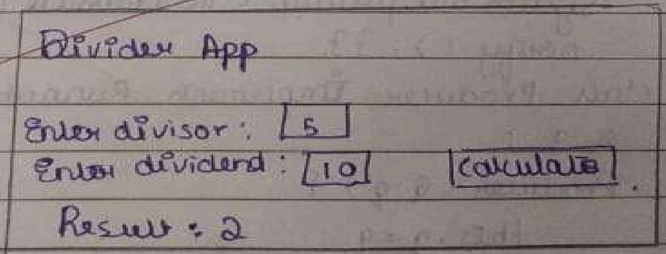


14) Program that creates a user interface to perform integer divisions. User enters 2 numbers, division of both numbers is displayed in result. Throw appropriate errors wherever required.

```
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 FlowLayout);
   jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)
  JLabel jlab = new JLabel('Enter divisor & dividend');
  JTextField ajft = new JTextField(8);
  JTextField bjtf = new JTextField(8);
  JButton button = new JButton('Calculate');
  JLabel err = new JLabel();
  JLabel alab = new JLabel();
```

PAGE NO :
DATE :

```
JLabel blab = new JLabel();
JLabel anslab = new JLabel();
jfrm.add(err);
jfrm.add(flab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);
ActionListener L = new ActionListener() {
        public voidactionPerformed(ActionEvent evt) {
                System.out.println('Action event from a text field"); }}
ajtf.addActionListener(l);
bjtf.addActionListener(l)
catch (ArithmeticException e) {
        alab.setText('');
        blab.setText('');
        anslab.setText('');
        err.setText('B cant be zero'); }});
public static void main (String args[]) {
        ~~frame on even~~ SwingUtilities.invokeLater(new Runnable(){
                public void run() {
                        new SwingDemo(); }}); }}.
```



Divider App

Enter divisor : 5

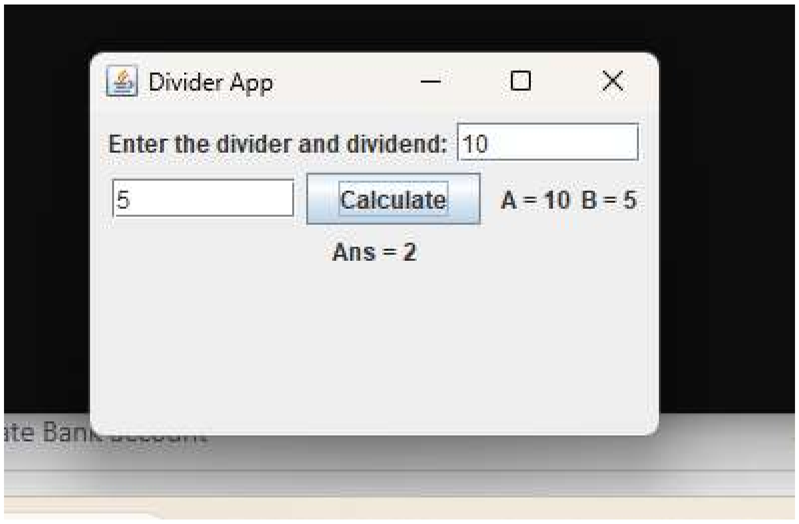
Enter dividend : 10   Calculate

Result : 2

[illegible]

Code:

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

class SwingDemo {
   SwingDemo() {
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(300, 200);
      jfrm.setLayout(new FlowLayout());
 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);
      ajtf.setToolTipText("Enter an integer as the dividend");
      bjtf.setToolTipText("Enter an integer as the divider");
      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(ajtf);
      jfrm.add(bjtf);
      jfrm.add(button);
      jfrm.add(err); // Error label
      jfrm.add(alab); // Label for dividend
      jfrm.add(blab); // Label for divider
      jfrm.add(anslab); // Label for result
      button.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent evt) {
            try {
               int a = Integer.parseInt(ajtf.getText());
               int b = Integer.parseInt(bjtf.getText());
               int ans = a / b;
               alab.setText("A = " + a);
               blab.setText("B = " + b);
               anslab.setText("Ans = " + ans);
               err.setText("");
            } catch (NumberFormatException e) {
               // Handle invalid number format
               alab.setText("");
               blab.setText("");
               anslab.setText("");
               err.setText("Error: Please enter valid integers!");
            } catch (ArithmeticException e) {
                // Handle division by zero
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("Error: Divider (B) cannot be zero!");
            }
        }
    });

    // Display the frame
    jfrm.setVisible(true);
  }

  public static void main(String args[]) {
    // Create the frame on the Event Dispatch Thread
    SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        new SwingDemo();
      }
    });
  }
}
```



Divider App

Enter the divider and dividend: 10

5 Calculate A = 10 B = 5

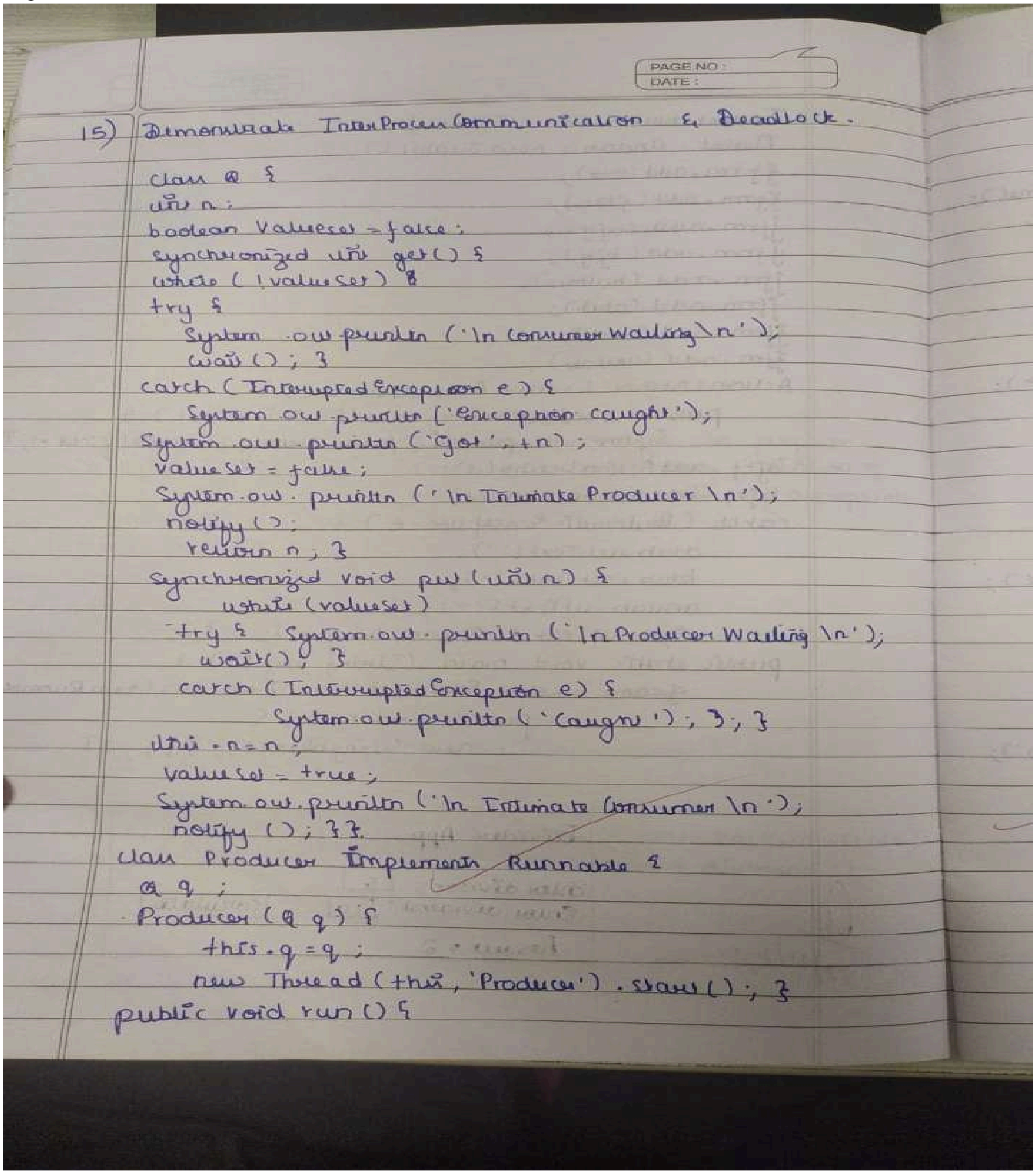
Ans = 2

Program 10

Demonstrate Inter process Communication and deadlock

IPC:

Algorithm:



PAGE NO :
DATE :

15) Demonstrate InterProcess Communication & Deadlock.

```
class Q {
int n;
boolean Valueset = false;
synchronized int get() {
while ( !valueSet) {
try {
  System.out.println ('\n Consumer Waiting\n');
  wait (); }
catch (InterruptedException e) {
  System.out.println ('Exception caught');
System.out.println ('Got', +n);
 valueset = false;
 System.out.println ('\n Intimate Producer \n');
  notify ();
   return n; }
 synchronized void put (int n) {
      while (valueset)
  try {  System.out.println ('\n Producer Waiting \n');
    wait(); }
    catch (InterruptedException e) {
           System.out.println ('caught'); }; }
this.n=n;
 valueset = true;
 System.out.println ('\n Intimate Consumer \n');
  notify (); }}.
class Producer Implements Runnable {
  Q q;
 Producer (Q q) {
      this.q=q;
      new Thread (this, 'Producer') . start (); }
public void run () {
```

PAGE NO :
DATE :

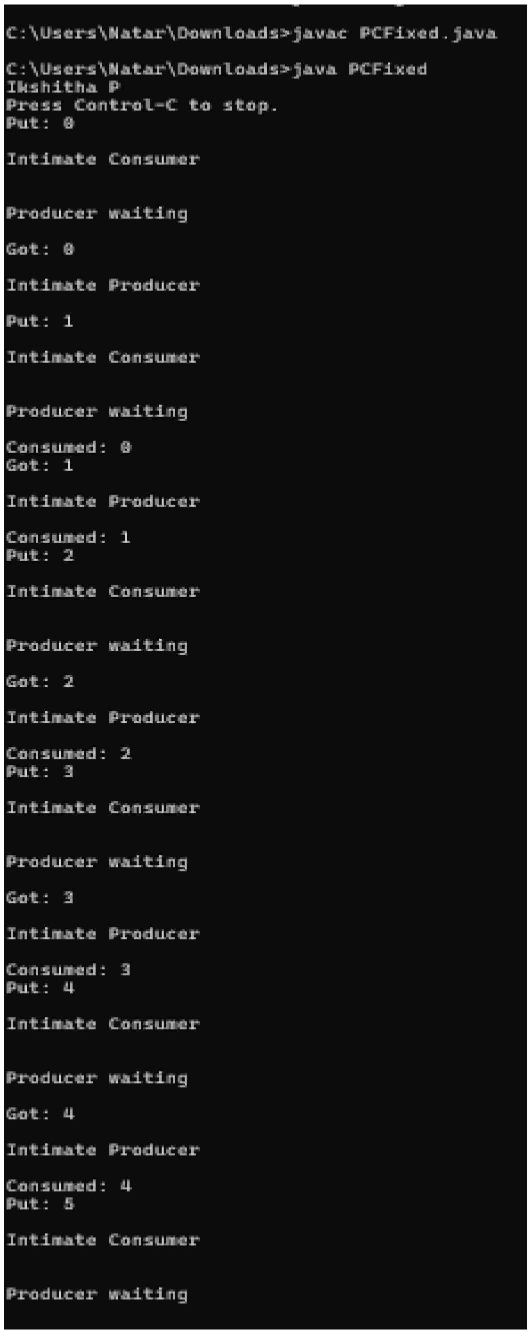
```
int i=0;
while (i<15) { q.put(i++); }}}
class Consumer implements Runnable {
 Q q;
 Consumer (Q q) {
         this.q = q;
  new Thread (this,'Consumer'). start (); }
public void run() {
      int i=0;
     while (i<15) {
    int r=q.get ();
      i++; }}}
class PCFixed {
  public static void main (String args[])
{     Q q = new Q();
      new Producer(q);
      new Consumer(q);
  }}
```

Output :

```
Put :1
Got : 1
Put :2
Got : 2
Put :3
Got :3
Put :4
Got :4
Put :5
Got : 5
```

Code:

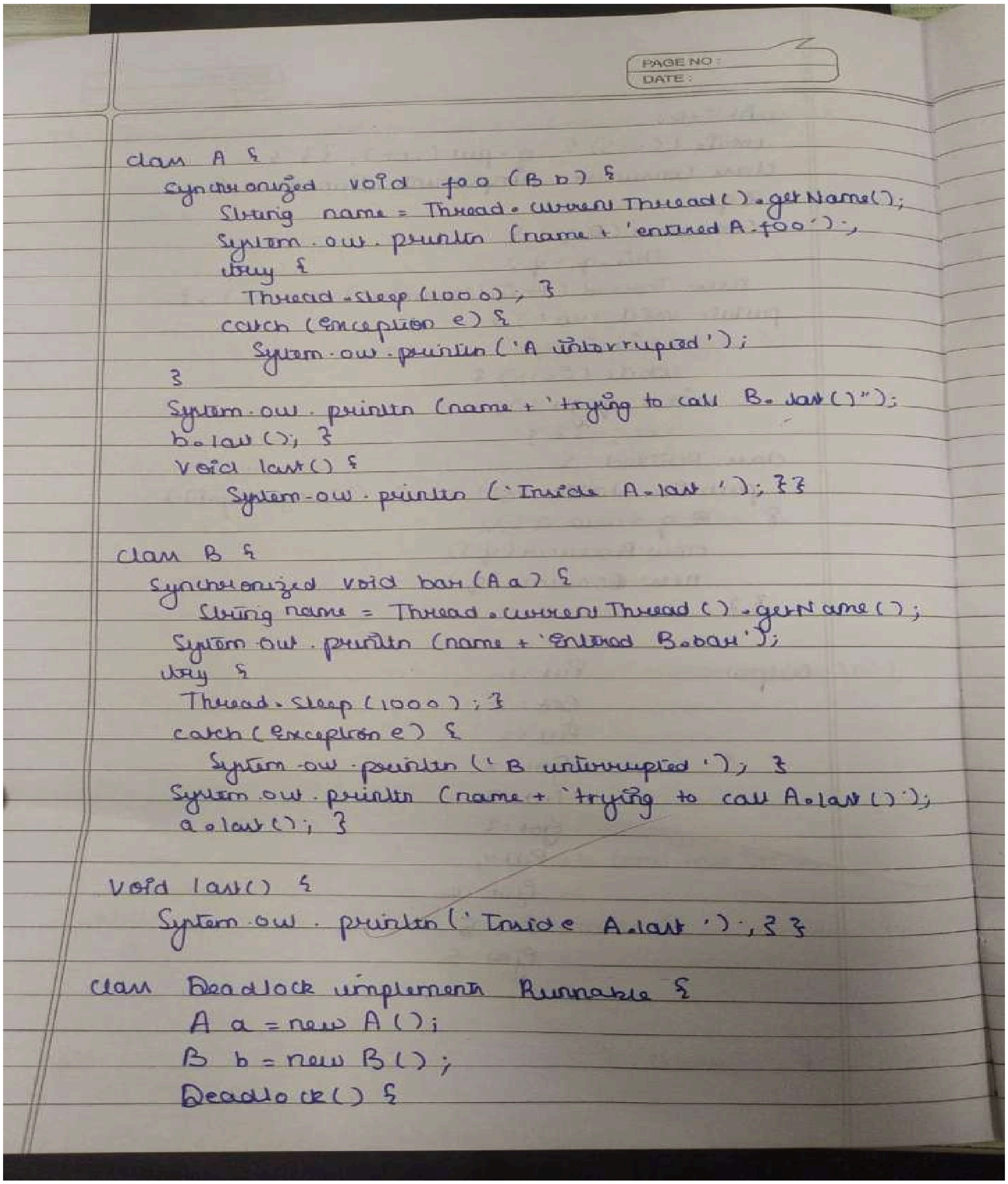
```
class Q {
   int n;
   boolean valueSet = false;
   synchronized int get() {
      while (!valueSet) { // Wait if no value is available
         try {
            System.out.println("\nConsumer waiting\n");
            wait();
         } catch (InterruptedException e) {
            System.out.println("InterruptedException caught");
         }
      }

      System.out.println("Got: " + n);
      valueSet = false; // Value has been consumed
      System.out.println("\nIntimate Producer\n");
      notify(); // Notify producer to produce the next value
      return n;
   }
   synchronized void put(int n) {
      while (valueSet) { // Wait if the previous value hasn't been consumed
         try {
            System.out.println("\nProducer waiting\n");
            wait();
         } catch (InterruptedException e) {
            System.out.println("InterruptedException caught");
         }
      }

      this.n = n; // Store the produced value
      valueSet = true; // Mark the value as produced
      System.out.println("Put: " + n);
      System.out.println("\nIntimate Consumer\n");
      notify(); // Notify consumer to consume the value
   }
}

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++); // Produce values from 0 to 14
       }
   }
}

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(); // Consume values
          System.out.println("Consumed: " + r);
          i++;
       }
   }
}

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



```
C:\Users\Natar\Downloads>javac PCFixed.java

C:\Users\Natar\Downloads>java PCFixed
Ikshitha P
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


Producer waiting

Got: 4

Intimate Producer

Consumed: 4
Put: 5

Intimate Consumer


Producer waiting
```

Deadlock:

Algorithm:



PAGE NO :
DATE :

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

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

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

class Deadlock implements Runnable {
      A a = new A();
      B b = new B();
      Deadlock() {
```

PAGE NO :
DATE :

```
Thread . currentThread () . set Name ('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(); }}.
```

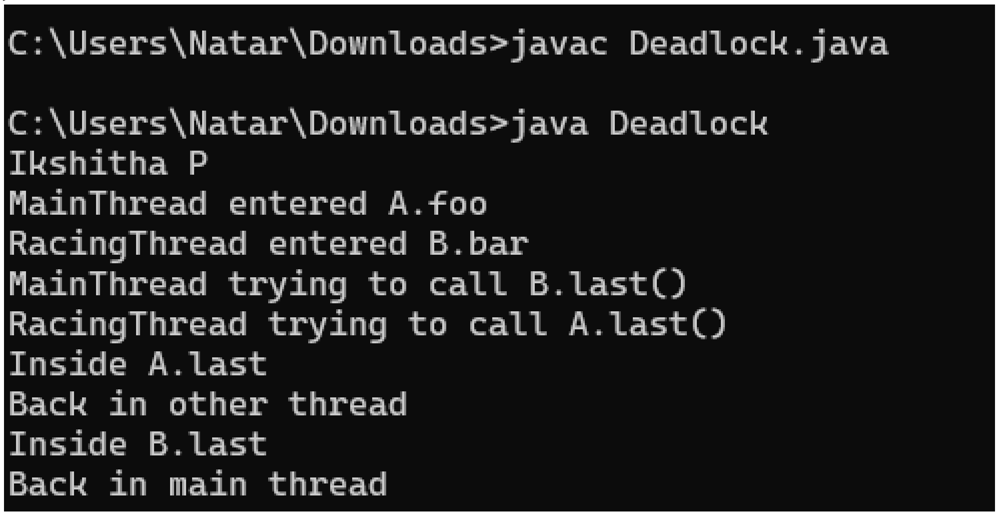
Output :

```
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
Inside A.last
Back in main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread .
```

[illegible]
28/11/21

Code:

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

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

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

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

class B {
   synchronized void bar(A a) {
      String name = Thread.currentThread().getName();
      System.out.println(name + " entered B.bar");

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

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

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

class Deadlock implements Runnable {
   A a = new A();
   B b = new B();

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

        // Create a new thread to simulate deadlock
        Thread t = new Thread(this, "RacingThread");
        t.start();

        // Main thread acquires lock on A and tries to call B.last()
        a.foo(b);
        System.out.println("Back in main thread");
    }

    public void run() {
        // This thread acquires lock on B and tries to call A.last()
        b.bar(a);
        System.out.println("Back in other thread");
    }

    public static void main(String args[]) {
        System.out.println("Ikshitha P");
        new Deadlock();
    }
}
```



```
C:\Users\Natar\Downloads>javac Deadlock.java

C:\Users\Natar\Downloads>java Deadlock
Ikshitha P
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
RacingThread trying to call A.last()
Inside A.last
Back in other thread
Inside B.last
Back in main thread
```