VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"GnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

StudentName (1BM23CS155)

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 **StudentName (1BM23CS000),** 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.

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

Index

10		**openend Deadlock and IPC**	

Github Link:
(You should provide your github link which contains all lab programs)

Program 1
Implement Quadratic Equation

Algorithm:

6) ax^2+bx+c : Read a, b, c : If dis b^2-4ac is -ve, display message stating that are no real

```
import java.util.Scanner;
import Static java.long.math.sqrt
Class quadratic{
    int a,b,c;
    double r1,r2,d;

    Void input(){
    Scanner sc=new Scanner(System.in);
    System.out.print("enter value of a:");
     a = sc.nextInt();
     While (a==0){
          System.out.println("enter a non
                              -zero number
                               for a:");
          a = sc.nextInt();
    }
    System.out.print("enter value of b:");
     b = sc.nextInt();
     System.out.print("enter value of c:");
     c = sc.nextInt();
     d = b*b-4*a*c;
    }

    Void display(){
      if (d==0){
          r1 = -b/(2.0*a);
      System.out.println("Roots are Real
```

```
r1 = (-b+Sqrt(d))/(2.0*a);
r2 = (-b-Sqrt(d))/(2.0*a);
System.out.println("Roots are real and
                    distinct");
System.out.println("r1 = "+r1+" , "r2 = "+r2);
}else{
r1 = -b/(2.0*a);
r2 = Sqrt(-d)/(2.0*a);
System.out.println("Roots are imaginary");
}
}
public static void main(String[] args){
quadratic qe = new quadratic();
qe.input();
qe.display();
}
}
```

output:-

```
a=0
printf
"enter a>0"
enter a:2
enter b:3
enter c:4
r1 = -3/4
r2 = i√23/4
enter a:1
enter b:-5
enter c:6
r1 = 2
```

Code:

```
import static java.lang.Math.sqrt;
import java.util.Scanner;

class quadratic {
    int a, b, c;
    double r1, r2, d;

    void input() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter value of a: ");
        a = sc.nextInt();

        while (a == 0) {
            System.out.println("Enter a non-zero number for a:");
            a = sc.nextInt();
        }

        System.out.print("Enter value of b: ");
        b = sc.nextInt();
        System.out.print("Enter value of c: ");
        c = sc.nextInt();

        d = b * b - 4 * a * c;
    }

    void display() {
        if (d == 0) {
            r1 = -b / (2.0 * a);
            System.out.println("Roots are real and equal");
            System.out.println("Root: " + r1);
        } else if (d > 0) {
            r1 = (-b + sqrt(d)) / (2.0 * a);
            r2 = (-b - sqrt(d)) / (2.0 * a);
            System.out.println("Roots are real and different");
            System.out.println("r1 = " + r1 + ", r2 = " + r2);
        } else {
            r1 = -b / (2.0 * a);
            r2 = sqrt(-d) / (2.0 * a);
            System.out.println("Roots are imaginary");
            System.out.println("r1 = " + r1 + " + " + r2 + "i");
            System.out.println("r2 = " + r1 + " - " + r2 + "i");
        }
    }

    public static void main(String[] args) {
        quadratic qe = new quadratic();
        qe.input();
```

```
        qe.display();
    }
}
```

```
D:\>cd 1BM23CS155

D:\1BM23CS155>javac quadratic.java

D:\1BM23CS155>java quadratic.java
Enter value of a: 0
Enter a non-zero number for a:
2
Enter value of b: 3
Enter value of c: 4
Roots are imaginary
r1 = -0.75 + 1.1989578808281798i
r2 = -0.75 - 1.1989578808281798i

D:\1BM23CS155>java quadratic.java
Enter value of a: 1
Enter value of b: -5
Enter value of c: 6
Roots are real and different
r1 = 3.0, r2 = 2.0
```

program2:
Student SGPA

Algorithm:

Develop a Java program to create a Class Student with members usn, name, an array credits and an array marks. Include methods to accept & display details and a method to calculate SGPA of a Student.

Array of

```
import java.util.Scanner;
  Class Subject
  {
    int subMarks;
    int credits;
    int grade;
  };
  Class Student
  {
    String name;
    String usn;
    double sgpa;
    Subject subject[];
    Scanner s;
    Student()
    {
      int i;
      subject = new Subject[8];
      for(i=0; i<8; i++)
      {
        Subject[i] = new subject();
```

```
public void getStudentdetails()
{
    System.out.println("enter Student name");
    name = s.nextLine();
    System.out.println("enter usn:")
    usn = s.nextLine();
}
public void getmarks()
{
    for(int i=0; i<8; i++)
    {
        System.out.println("enter marks
        for subject"+(i+1)+":");
        Subject[i].subjectmarks = s.nextInt();
        System.out.println("enter credits
        for subject"+(i+1)+":");
        Subject[i].credits = s.nextInt();
        Subject[i].Calculate Grade();
        if (Subject[i].SubjectMarks > 100)
        {
            System.out.println("Invaid marks");
            Subject[i].subject marks=0;
        }
        else if (subject[i].subjectMarks<0)
        {
            System.out.println("marks
            Should not be negative");
        }
    }
}
```

```
public void Sgpa()
{
    double total_grade=0;
    int credits=0;
    for(Subject subj :Subject)
    {
        to@grade += (Subj.grade *Subj.credi
        credit += Subj.credits;
    }
    if if
    if (total cre (redits>0)
    {
        Sgpa= grade/credits;
    }
    else
    {
        Sgpa= 0.0;
    }
}
public void displayres( )
{
    System.out.println("name:"+name);
    System.out.println("usn:"+usn);
    System.out.println("Sgpa:"Sgpa);
}
public void calgrade( )
{
    if (marks >=90)
```

```
else if (marks >=70)
  {
    grade=8;
  }
else if (marks >=60)
  {
    grade=7;
  }
else if (marks >=50)
  {
    grade=6;
  }
else if (marks>=40)
  {
    grade=5;
  }
else
  {
    grade=0;
  }
public static void main(String[] args)
  {
    Student S1 = new Student();
    S1.getstudentdetails();
    S1.getmarks();
    S1.computeSgpa();
    S1.displayresult();
  }
}
```

```
Enter n: 1
enter name: K-Saketh
enter USN: 1BM23CS155
enter 1 Subj mark: 78
enter 2 Subj mark: 73
enter Credits for 2: 4
enter 3 Subj marks: 65
enter Credits 3: 3
enter Subj 4 marks: 61
enter Credits of 4: 3
SGPA: 8.014
```

Code:

```
import java.util.Scanner;

public class Student {
   String name, usn;
   double SGPA;

   int[] marks = new int[4];
   int[] credits = new int[4];
   double[] grade points = new double[4];
   double total = 0, credit total = 0;

   Scanner sc = new Scanner(System.in);

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

   void getMarks() {
      for (int j = 0; j < 4; j++) {
         System.out.println("Enter " + (j + 1) + " subject
marks:");
         marks[j] = sc.nextInt();
         System.out.println("Enter credits for subject " + (j + 1) +
":");
         credits[j] = sc.nextInt();

         grade points[j] = (marks[j] / 10.0) + 1;
```

```
        if (grade points[j] > 10) {
            grade points[j] = 10;
        }
    }
    sc.nextLine();
}

void computeSGPA() {
    total = 0;
    credit total = 0;
    for (int j = 0; j < 4; j++) {
        total += grade points[j] * credits[j];
        credit total += credits[j];
    }
    SGPA = total / credit total;
}

void display() {
    System.out.println("Name: " + name);
    System.out.println("USN: " + usn);
    System.out.println("SGPA: " + SGPA);
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter number of students:");
    int numberOfStudents = sc.nextInt();
    sc.nextLine();

    Student[] students = new Student[numberOfStudents];

    for (int i = 0; i < numberOfStudents; i++) {
```

```
            students[i] = new Student();
            students[i].getStudentDetails();
            students[i].getMarks();
            students[i].computeSGPA();
            students[i].display();
        }
            System.out.println("1BM23CS155");
}
```

program 3:
Book details
algorithm:

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

```
import java.util.Scanner;
Class book
{
    String name, author;
    int price, num_pages;
    book(String name, String author, int
        price, int numpages);
        {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numpages = numpages;
        }
    Public String toString()
        {
        return "Book name is: " + this.name
            + "\n Author name: " + this.au
            thor + "price: " + this.pri
            "no of pages: " + this.pages
            "\n";
        }
}
```

```
Class Main
{
public static void main (String[] args)
{
int n,i;
System.out.println("enter n value:");
n=
int n, i;
Scanner in = new Scanner(System.in);
System.out.println("enter n value:");
n= in.nextInt();
for(i=0;i<n
Book[] b = new Book[n];
for(i=0;i<n;i++)
{
System.out.println("enter name
                    of "+(i+1)+"
                    "book:");
name = in.nextLine()
System.out.println("enter
                    author of"
                    +(i+1)+"book:");
author= in.next();
System.out.println("enter price
                    of "+(i+1)+
                    "book:");
price = in.nextInt();
System.out.println("enter no
                    of pages in
                    "+(i+1)+"book:")
b[i]= new Book(name, author
                    ,price,
                    numpages)
b[i].toString();
```

Date
Page

Output:

no of books: 2
name of book 1: geeta
author name book1: ravi
price of book 1: 240
no of pages in book1: 375

Book name: geeta
Aathor name: ravi
price: 240
no of pages: 375

~~Boo~~ name of book2: Anjali
author name book2: Mani
price of book2 : 400
no of pages in book2: 780

Aug. 14/10/24

Code:

```
import java.util.Scanner;

 class Book {
    String name, author;
    int price, numPages;

    Book(String name, String author, int price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    public String toString() {
        return "Book name: " + this.name + "\n" +
               "Author name: " + this.author + "\n" +
               "Price: " + this.price + "\n" +
               "Number of pages: " + this.numPages + "\n";
    }
}

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

        Book[] books = new Book[n];

        for (int i = 0; i < n; i++) {
```

```
            System.out.print("Enter the name of book " + (i + 1) + ":
");
            String name = in.nextLine();
            System.out.print("Enter the author of book " + (i + 1) +
": ");
            String author = in.nextLine();
            System.out.print("Enter the price of book " + (i + 1) + ":
");
            int price = in.nextInt();
            System.out.print("Enter the number of pages in book " +
(i + 1) + ": ");
            int numPages = in.nextInt();
            in.nextLine();

            books[i] = new Book(name, author, price, numPages);
            System.out.println(books[i]);
        }


    }
}
```

```
C:\Users>cd ..

C:\>D:

D:\>cd 1BM23CS155

D:\1BM23CS155>javac Main.java

D:\1BM23CS155>java Main.java
Enter the number of books: 3
Enter the name of book 1: geeta
Enter the author of book 1: ravi
Enter the price of book 1: 240
Enter the number of pages in book 1: 375
Book name: geeta
Author name: ravi
Price: 240
Number of pages: 375

Enter the name of book 2: anjali
Enter the author of book 2: mani
Enter the price of book 2: 300
Enter the number of pages in book 2: 780
Book name: anjali
Author name: mani
Price: 300
Number of pages: 780
```

program 4:
Implementing abstract class
algorithm:

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 & 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 given Shape.

```
import java.util.Scanner;
abstract Class Shape
{
    double dim1;
    double dim2;
    Shape(double dim1, double dim2)
    {
        this.dim1 = dim1;
        this.dim2 = dim2;
    }
    abstract void printarea();
}

class Rectangle extends Shape
{
    Rectangle(double dim1, double dim2)
    {super(double a, double b)
    }
    public void printarea()
    {
        double area = dim1 * dim2;
    }
}
```

```
class triangle extends Shape
{
public triangle (double base, double heigh
{
super (base, height);
}
public void printarea()
{
double area= 0.5*base*height;
System.out.println("Area of triangle."
tarea);
}
}
Class circle extends Shape
{
public circle (int radius)
{
super (radius, 0)
}
public void printArea()
{
double area = 3.14* dim1 * dim2;
}

public class main
{
public static void main(String args[]
{
Scanner in = new Scanner(System.
System.out.println("enter length
of rectangle:");
double l = in.nextDouble()
System.out.println("enter breadth:");
double b= in.nextDouble();
```

```
System.out.println("enter area of triangle");
double base = in.nextDouble();
System.out.println("enter height of
                    triangle:");
double height= in.nextDouble();
System.out.println("enter radius:");
double radius =in.nextDouble();
rectangle r= new rectangle(l,b);
Triangle t = new triangle(base,height);
Circle c = new Circle(radius);
r.printarea();
t.printarea();
c.printarea();
}
}
```

Output:-

enter length: 3
enter breadth: 2
area of rectangle = 6
enter base: 4
enter height: 8
area of triangle : 16
enter radius: 4
area of circle: 50.24

Code:

```
import java.util.Scanner;


abstract class Shape
{
   int dimension1;
   int dimension2;

   public Shape(int dimension1, int
dimension2)
   {
      this.dimension1 = dimension1;
      this.dimension 2 = dimension2;
   }

   public abstract void printArea();
}


class Rectangle extends Shape
{
   public Rectangle(int length, int width)
   {
      super(length, width);
   }


   public void printArea()
   {
```

```
        int area = dimension1 * dimension2;
        System.out.println("Area of
Rectangle: " + area);
    }
}


class Triangle extends Shape
{
    public Triangle(int base, int height)
    {
        super(base, height);
    }


    public void printArea()
    {
        double area = 0.5 * dimension1 *
dimension2;
        System.out.println("Area of Triangle:
" + area);
    }
}


class Circle extends Shape
{
    public Circle(int radius)
    {
        super(radius, 0);
```

```
    }


    public void printArea()
    {
        double area = Math.PI * dimension1 *
dimension1;
        System.out.println("Area of Circle: "
+ area);
    }
}


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


        System.out.print("Enter the length of
the rectangle: ");
        int length = scanner.nextInt();
        System.out.print("Enter the width of
the rectangle: ");
        int width = scanner.nextInt();
        Shape rectangle = new
Rectangle(length, width);
        rectangle.printArea();
```

```
        System.out.print("Enter the base of the triangle: ");
        int base = scanner.nextInt();
        System.out.print("Enter the height of the triangle: ");
        int height = scanner.nextInt();
        Shape triangle = new Triangle(base, height);
        triangle.printArea();


        System.out.print("Enter the radius of the circle: ");
        int radius = scanner.nextInt();
        Shape circle = new Circle(radius);
        circle.printArea();


        scanner.close();
    }
}
```

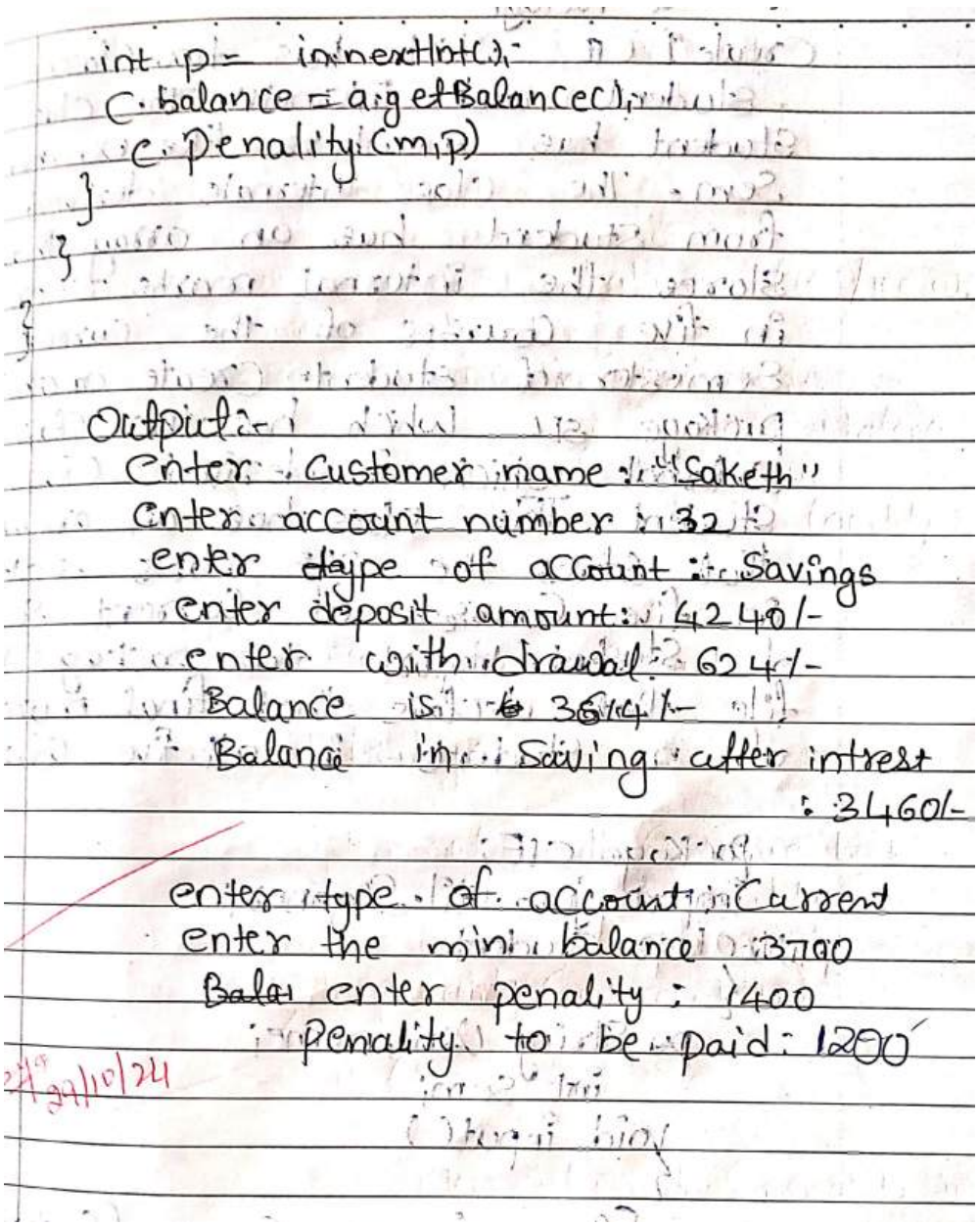
```
D:\1BM23CS155>java Main.java
Area of Rectangle: 15
Area of Triangle: 12.0
Area of Circle: 153.938040025
```

program 5:
Bank program
Algorithm:

```
Class SavACC extends ACCount
{
    int roi=5;
    Void getInterest()
    {
        double interest = balance * (roi/100.0);
        balance+= interest;
        System.out.println("Balace in
                            Savings:"+balance);
        System.out.println("No Cheque
                            book facility");
    }
}
Class CurrACC extends ACCount
{
    Void facility()
    {
        System.out.println("Cheque book
                            facility");
    }
    void penality(int m, int p)
    {
        if(balance<m)
        {
            System.out.println("penality:"+p);
        else
        {
            System.out.println("No penality");
        }
    }
```

```
class account
{
import java.util.Scanner;
Class account
{
String Cname;
int accno;
String type;
int balance=0;
Void getdeposite(int depamo)
{
balance+=depamo;
}
Void withdrawal(int wamo)
{
if(balance<=0)
{
System.out.println("account is
                    empty:");
}
else
{
balance-=wamo;
}
}
Void display()
{
System.out.println("balance is:"+
                    balance);
```

```
Class Bank
{
  public static void main(String args[])
  {
    ACCount a = new ACCount();
    SaVACC S = new SaVACC();
    CurrACC c = new CurrACC();
    Scanner in = new Scanner(System.
    System.out.println("enter name:");
    a.cname = in.nextInt();
    System.out.println("ACCno is:"+a.accn
    in.nextLine();
    if (a.type == "Savings"))
    {
      System.out.println("enter depam
      int A = in.nextInt();
      a.deposit(A);
      System.out.println("enter withd
                          -wal amoun
      int B = in.nextInt();
      a.withdrawal(B)
      a.display();
      S.balance = a.getbalance();
      S.getInterest();
    }
    else
    {
      C.facility();
      System.out.println("enter
                          min bal
```


int p = in.nextInt();
c.balance = a.getBalance();
c.penality(m,p)
}
}
Output:-
Enter Customer name : "Saketh"
enter account number : 32
enter type of account : Savings
enter deposit amount: 4240/-
enter withdrawal: 624/-
Balance is 3616/-
Balance in Saving after intrest
: 3460/-
enter type of account : Current
enter the min balance 3700
Bala enter penality : 1400
penality to be paid: 1200
24 24/10/24

Code:

```
import java.util.Scanner;

class Account {
    String cname;
```

```
    int accno;
    String type;
    int balance = 0;

    void deposit(int depamo) {
        balance += depamo;
    }

    void withdrawal(int wamo) {
        if (balance <= 0) {
            System.out.println("Account is
empty.");
        } else if (wamo > balance) {
            System.out.println("Insufficient
balance for withdrawal.");
        } else {
            balance -= wamo;
        }
    }

    void display() {
        System.out.println("Balance is: " +
balance);
    }

    int getBalance() {
        return balance;
    }
}

class SavAcc extends Account {
    int roi = 5;
```

```
    void getInterest() {
        double interest = balance * (roi / 100.0);
        System.out.println("interest
is:"+interest);
        balance += interest;
        System.out.println("Balance in savings
account after interest is: " + balance);
        System.out.println("No cheque book
facility.");
    }
}

class CurrAcc extends Account {
    void facility() {
        System.out.println("Cheque book
facility provided but no interest facility.");
    }

    void penalty(int m, int p) {
        if (balance < m) {
            System.out.println("Penalty to be
paid: " + p);
        } else {
            System.out.println("No penalty
applicable.");
        }
    }
}

class Bank {
    public static void main(String args[]) {
        Account a = new Account();
```

```
        SavAcc s = new SavAcc();
        CurrAcc c = new CurrAcc();
        Scanner in = new Scanner(System.in);

        System.out.println("Enter the customer
name:");
        a.cname = in.nextLine();
        System.out.println("Customer name is: "
+ a.cname);

        System.out.println("Enter the account
number:");
        a.accno = in.nextInt();
        System.out.println("Account number is:
" + a.accno);

        in.nextLine();
        System.out.println("Enter the type of
account (savings/current):");
        a.type = in.nextLine();

        if (a.type.equalsIgnoreCase("savings"))
{
            System.out.println("Enter the deposit
amount:");
            int A = in.nextInt();
            a.deposit(A);

            System.out.println("Enter the
withdrawal amount:");
            int B = in.nextInt();
            a.withdrawal(B);
```

```
            a.display();

            s.balance = a.getBalance();
            s.getInterest();
        } else if
(a.type.equalsIgnoreCase("current")) {
            c.facility();
            System.out.println("Enter the
minimum balance:");
            int m = in.nextInt();
            System.out.println("Enter the
penalty:");
            int p = in.nextInt();
            c.balance = a.getBalance();
            c.penalty(m, p);
        } else {
            System.out.println("Invalid account
type.");
        }

    }
}
```

```
D:\1BM23CS155>java Bank.java
Enter the customer name:
saketh
Customer name is: saketh
Enter the account number:
23
Account number is: 23
Enter the type of account (savings/current):
savings
Enter the deposit amount:
4500
Enter the withdrawal amount:
2300
Balance is: 2200
interest is:110.0
Balance in savings account after interest is: 2310
No cheque book facility.
```

program 6:
Package program
Algorithm:

Date / /
Page

a Package

Create a 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 student. Create another package SEE which has the class External which is derived Class of Student. This class has an array that stores the SEE marks scored in five Courses of current Semester of Student. Import two packages in a file that declares that final marks of n Students in all five Courses

```
package CIE;
import java.util.Scanner;
public class Student
{
    String usn, name;
    int sem;
    void input()
    {
        Scanner in= new Scanner(System.in)
        usn= in.nextLine();
        System.out.println("enter usn:");
        usn = in.nextLine();
        System.out.println("enter name:");
```

Date __/__/__
Page _____

```
Package SEE;
import CIE.*;
Class Internals extends Student
{  public marksInput(CIE())
  {int marks[ ] = new int[5];
   for(int i=0 ; i<5 ;i++)
   {
     System.out.println("enter marks of"
                        + (i+1)+ "student");
     marks[i]=in.nextLine();
   }
  }
```

```
package SEE;
 import CIE.*;
Class Externals extends Internals
{   int fmarks[ ] = new int[5];
    int emarks[ ] = new int[5];
  {public marks inputSee()
   for(i=0 ; i<5 ; i++)
    {  Scanner in = new Scanner(System.in)
       System.out.println("enter see
                  marks of"+(i+1)+
                  "subject");
       emarks[i] = in.nextInt();
       fmarks[i] = (marks[i]
                  + emarks[i]
                    2)
    }
}
```

```
import CIE.*
import SEE.*
Class finalmarks
{
 public static void main(String args[])
```

```
import java.util.Scanner;
Package CIE
Class Student
{
  String name, usn;
  int sem;
  void input()
  {
    Scanner in = new Sganner(System.in);
    System.out.println("enter name:");
     name= in.nextLine()
    System.out.println("enter usn:");
     usn= in.nextLine();
    System.out.println("enter sem:");
     sem = in.nextInt();
  }
  void display()
  {
    System.out.println("name = "+name);
    System.out.println("usn= "+usn);
    System.out.println("sem= "+sem);
  }
}
import CIE.*
Class internals extends Student
{
  int marks[] = new int[5];
  for(int i=0; i<5, i++)
```

```
package SEE
import CIE.*
Class Externals extends Internals
{   int fmarks[] = new int[5];
    int emarks[] = new int[5];
      void inputsee()
      {
      for(int i=0;i<5 ;i++)
      {
      System.out.println("enter
                  see marks of "+(i+1)+
                  "Student:");
      emarks[i] = in.nextInt();
      }
      }
    void finalmarks()
      {
      for(int i=0; i<5 ,i++)
      {
      System.out.println("
            fmarks[i] = marks[i]
                  +(emarks[i]/2)
      }
}
```

```
System.out.println("enter n value:");
int n = in.nextInt();
for(int i=0 ; i<n ; i++)
{
    external[] e = new external[n];
    for(int i=0 ; i<n ; i++)
    {
        e[i] = new external();
        e[i].input();
        e[i].display();
        e[i].inputIE();
        e[i].inputSee();
        e[i].finalmarks();
    }
}
}
```

Output:-

```
enter n :2
enter name : K.S.R
enter usn : 1BM23CS155
enter sem: 1
enter marks in 1 subject in cie
                                : 36
                    2           : 32
                    3           : 29
                    4           : 27
                    5           : 34
enter marks in 1 subj in see: 78
                    2           : 48
                    3           : 49
```

4 : 7
5 : 8

final marks in 1 subj : 75
2 : 56
3 : 54
4 : 67
5 : 76

enter name : Ram
enter usn : 164
enter sem : 3
name = Ram
usn = 164
sem = 3
enter marks in 1 CIE : 27
2 : 41
3 : 32
4 : 47
5 : 50

enter marks in 1 see : 67
: 64
: 68
: 94
: 88

final marks in 1 se : 61
: 73
: 66
: 94
: 94

13/11/24

8

Code:

```
package cie;

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

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

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

package cie;

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

    public Internals(String usn, String name,
int sem, int[] internalMarks) {
        super(usn, name, sem);
```

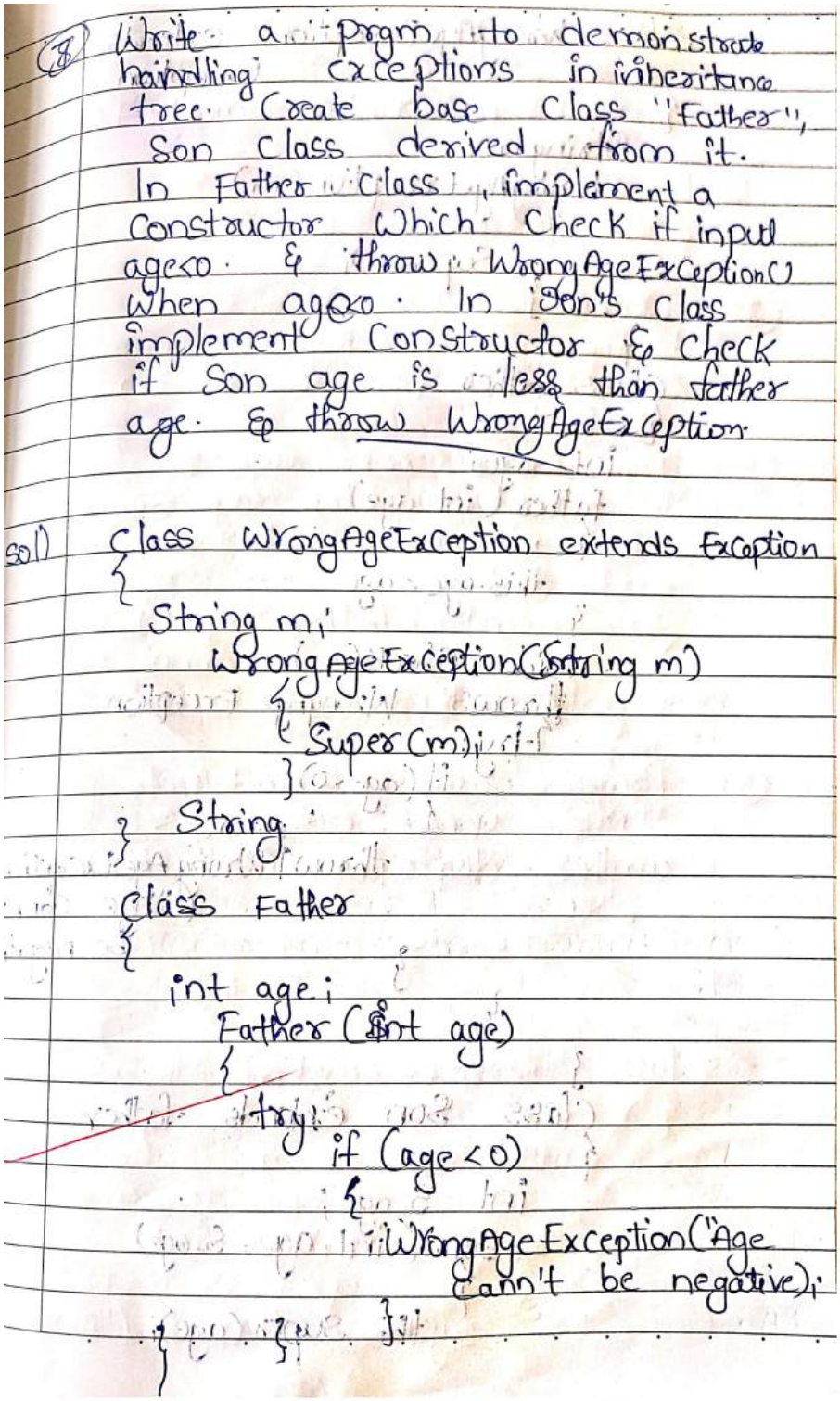
```
        this.internalMarks = internalMarks;
    }
}

import cie.*;
import see.*;
import java.util.Scanner;

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

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

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


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

            System.out.print("USN: ");
            String usn = scanner.next();
            System.out.print("Name: ");
            String name = scanner.next();
            System.out.print("Semester: ");
```

```
            int sem = scanner.nextInt();

            int[] internalMarks = new int[5];
            System.out.println("Enter 5 internal
marks: ");
            for (int j = 0; j < 5; j++) {
                internalMarks[j] =
scanner.nextInt();
            }
            internalStudents[i] = new
Internals(usn, name, sem, internalMarks);

            int[] seeMarks = new int[5];
            System.out.println("Enter 5 SEE
marks: ");
            for (int j = 0; j < 5; j++) {
                seeMarks[j] = scanner.nextInt();
            }
            externalStudents[i] = new
External(usn, name, sem, seeMarks);
        }


        System.out.println("\nFinal Marks of
Students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nDetails of
Student " + (i + 1) + ":");
            internalStudents[i].display();

            int[] finalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
```

```
                finalMarks[j] =
internalStudents[i].internalMarks[j] +
(externalStudents[i].seeMarks[j] / 2);
            }

            System.out.println("Final Marks in 5
Courses:");
            for (int mark : finalMarks) {
                System.out.print(mark + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}
```

```
D:\package>java FinalMarksCalculator
Enter number of students: 3
Enter details for Student 1:
USN: 1BM23CS148
Name: Rohan
Semester: 3
Enter 5 internal marks:
40
45
44
47
50
Enter 5 SEE marks:
78
88
90
94
96
Enter details for Student 2:
USN: 1BM23CS203
Name: Soham
Semester: 3
Enter 5 internal marks:
47
48
43
44
47
Enter 5 SEE marks:
98
97
91
88
89
Enter details for Student 3:
USN: 1BM23CS258
Name: Praveen
Semester: 3
Enter 5 internal marks:
45
43
40
47
50
Enter 5 SEE marks:
89
83
90
79
92
```

program 7:
Algorithm:



(8) Write a prgm to demonstrate handling exceptions in inheritance tree. Create base class "Father", Son class derived from it. In Father class, implement a constructor which check if input age<0. & throw WrongAgeException() when age<0. In Son's class implement constructor & check if Son age is less than father age. & throw WrongAgeException.

sol)

```
class WrongAgeException extends Exception
{
  String m;
    WrongAgeException(String m)
      { super(m); }
}
class Father
{
  int age;
    Father(int age)
    {
      try{
        if (age<0)
        {
          throw new WrongAgeException("Age
                  can't be negative");
        }
```

```
Class WrongAgeException extends
                          Exception
{
   String m;
   WrongAgeException(m)
   {
      Super(m);
   }
}
Class father
{
   int age;
   father (int age)
   {
      this.age=age
   }
   void check(  )
    throws WrongAge Exception
    try
       if(age<0)
       {
          throw new WrongAgeException
                    ("Age cannot
                     be negative
       }
    }
}
Class Son extends father
{
   int S_age;
   Son (int age, S_age)
   {
      Super(age);
```

```
this.S_age = S_age;
Void checkage() throws WAE
{
    if(S
    try:
        if (S_age >= age)
        {
        throw new WrongAge
            Exception("Son
            age Cann't
            be more than
            fathers age);
        }
    }
Void
class main
{
  public static void main(String args[])
  {
    System.out.println("enter age of
        son:");
    int x =
    Scanner in = Scanner (System.in)
    intx - in.nextInt();
    System.out.println("enter fathers
                        age:");
    int y = in.nextInt();
    father f = new father(y);
    Son S = new Son(x,y);
    try:{ f.Check()
        S.Checkage()
```

```
catch(WrongAgeException e);
{
    System.out.println("Caught :"+e);
}
}
}
```

(8) Demonstrate two threads, one thread displaying "BMS college of Engineering" once every ten seconds an

output:-

enter son age: 21
enter father age:32
age Cannot be negative
enter son age: 18
enter father age: 9
Son's age cannot exceed fathers age.

(8) Program on threads, to print "BMSE" once every 10 sec & another displaying "CSE" for 2 sec

```
class BMSCE extends Thread
{
    public void run()
    {
        for(int i=0; i<5; i++)
        {
            System.out.println("BMSCE");
```

Code:
Exception Handling

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

class Father {
    int age;

    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Father's age cannot
be negative!");
        }
        this.age = age;
        System.out.println("Father's age is set to:
" + this.age);
    }
}

class Son extends Father {
    int sonAge;

    public Son(int fatherAge, int sonAge) throws
WrongAge {
        super(fatherAge);
        if (sonAge >= fatherAge) {
```

```
            throw new WrongAge("Son's age cannot be
greater than or equal to Father's age!");
        }
        this.sonAge = sonAge;
        System.out.println("Son's age is set to: " +
this.sonAge);
    }
}

public class Main {
    public static void main(String[] args) {
        try {
            Father father = new Father(40);
            Son son = new Son(40, 25);
        } catch (WrongAge e) {
            System.out.println("Exception: " +
e.getMessage());
        }

        try {
            Father father = new Father(40);
            Son son = new Son(40, 45);
        } catch (WrongAge e) {
            System.out.println("Exception: " +
e.getMessage());
        }

        try {
            Father father = new Father(-1);
        } catch (WrongAge e) {
            System.out.println("Exception: " +
e.getMessage());
        }
    }
}

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

```
class Father {
    int age;

    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Father's age cannot
be negative!");
        }
        this.age = age;
        System.out.println("Father's age is set to:
" + this.age);
    }
}

class Son extends Father {
    int sonAge;

    public Son(int fatherAge, int sonAge) throws
WrongAge {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be
greater than or equal to Father's age!");
        }
        this.sonAge = sonAge;
        System.out.println("Son's age is set to: " +
this.sonAge);
    }
}

public class Main {
    public static void main(String[] args) {
        try {
            Father father = new Father(40);
            Son son = new Son(40, 25);
        } catch (WrongAge e) {
            System.out.println("Exception: " +
e.getMessage());
        }
```

```
        try {
            Father father = new Father(40);
            Son son = new Son(40, 45);
        } catch (WrongAge e) {
            System.out.println("Exception: " +
e.getMessage());
        }

        try {
            Father father = new Father(-1);
        } catch (WrongAge e) {
            System.out.println("Exception: " +
e.getMessage());
        }
    }
}
```

```
D:\1BM23CS155>java Main.java
Father's age is set to: 40
Father's age is set to: 40
Son's age is set to: 25
Father's age is set to: 40
Father's age is set to: 40
Exception: Son's age cannot be greater than or equal to F
Exception: Father's age cannot be negative!
```

program 8:
Threads implementation
Algorithm:

```
try:
{
    Thread.Sleep(10000);
}
Catch (InterruptedException e)
{
}
}
Class CSE extends Thread
{
    public void run()
    {
        for(int i=0; i<5; i++)
        {
            System.out.println("CSE
            try
            {
                Thread.Sleep(2000);
            }
            Catch(InterruptedExceptio
            {
            }
        }
    Class mainprgm
    {
```

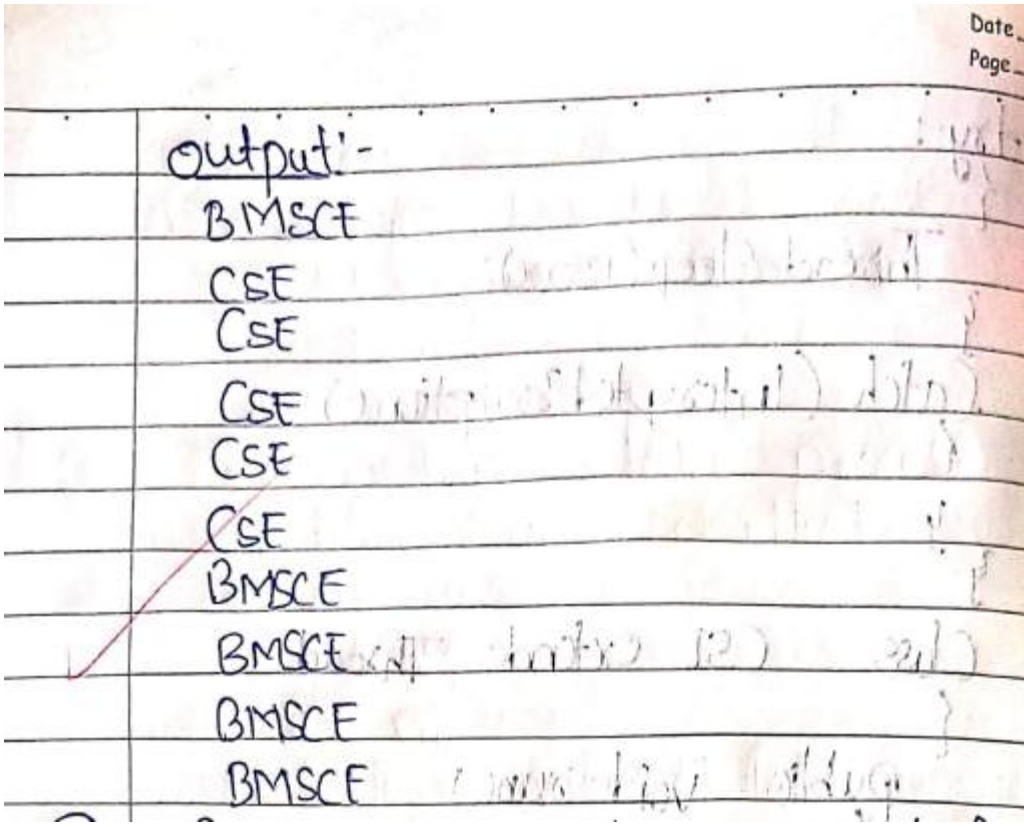
```
Cath(WrongAgeException e);
    {
        System.out.println("Caught:"+e);
    }
}
}
```

(8) Demonstrate two threads, one thread displaying "BMS College of Engineering" once every ten seconds an output:-

enter son age: 21
enter father age: 32
age Cann't be negative
enter son age: 18
enter father age: 9
Son's age cannot exceed fathers age.

(8) Program "one threads to print "BMSE" once every 10 sec & another displaying "CSE" for 2 sec

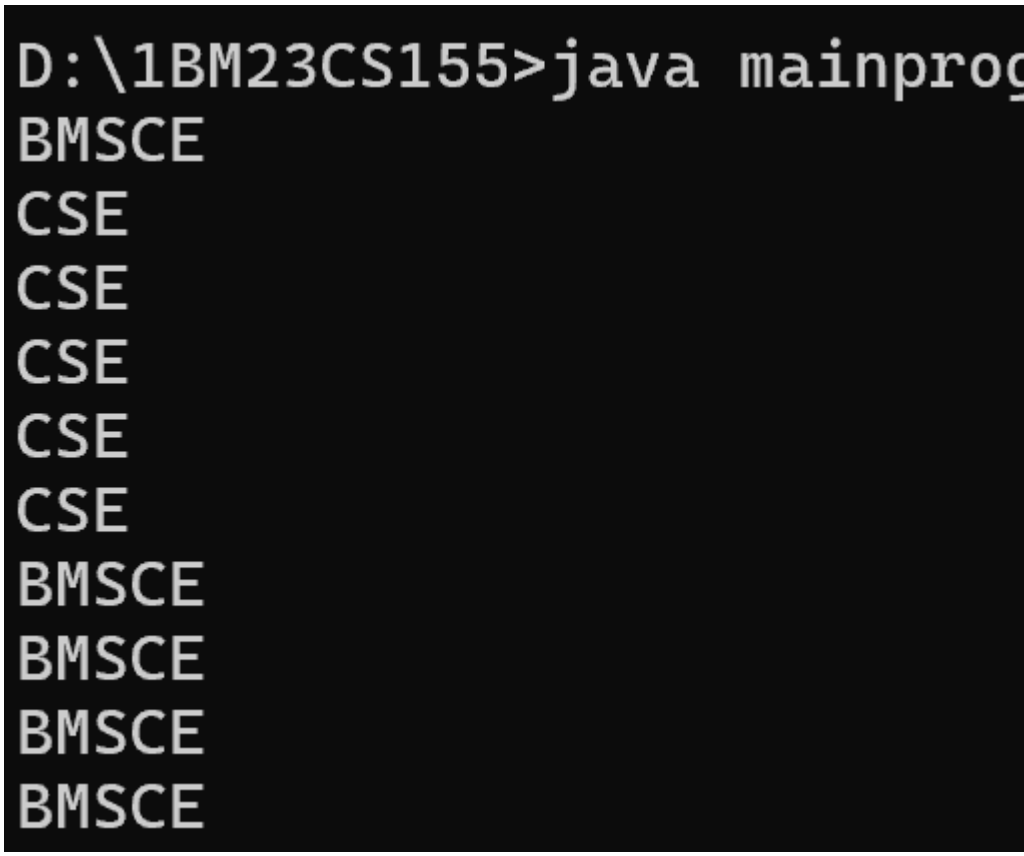
```
class BMSCE extends Thread
{
    public void run()
    {
        for(int i=0; i<5; i++)
        {
            System.out.println("BMSCE");
```



Code:

```
class BMSCE extends Thread
{
  public void run()
  {
        for(int i=0;i<5;i++)
        {
              System.out.println("BMSCE");
              try{Thread.sleep(10000);}
              catch(InterruptedException e){}
        }
  }
```

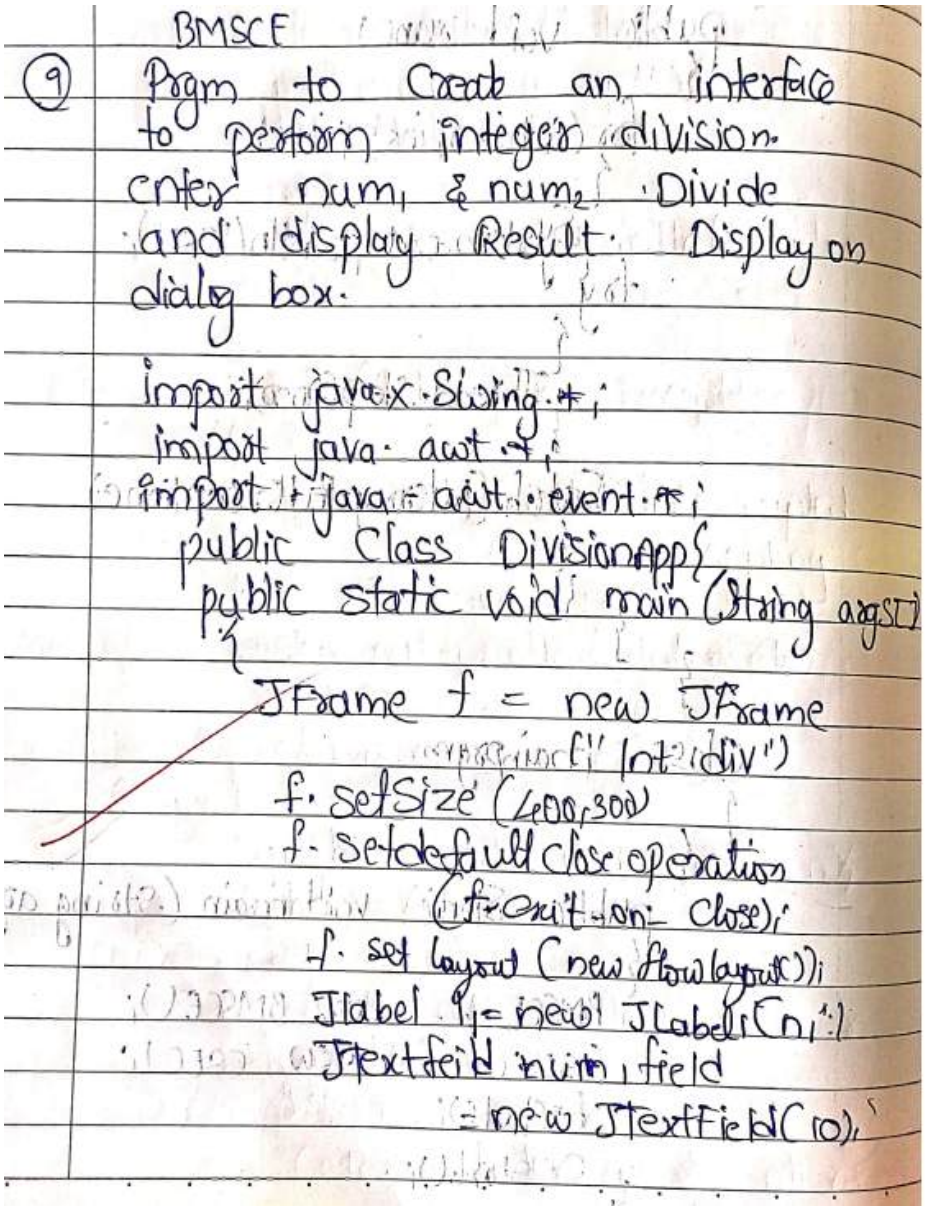
```
}
class CSE extends Thread
{
  public void run()
  {
        for(int i=0;i<5;i++)
        {
              System.out.println("CSE");
              try{Thread.sleep(2000);}
              catch(InterruptedException e){}
        }
  }
}
class main program
{
  public static void main(String args[])
  {
        BMSCE b=new BMSCE();
        CSE c=new CSE();
        b.start();
        c.start();
  }
}
```



program 9:
Opened end
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.

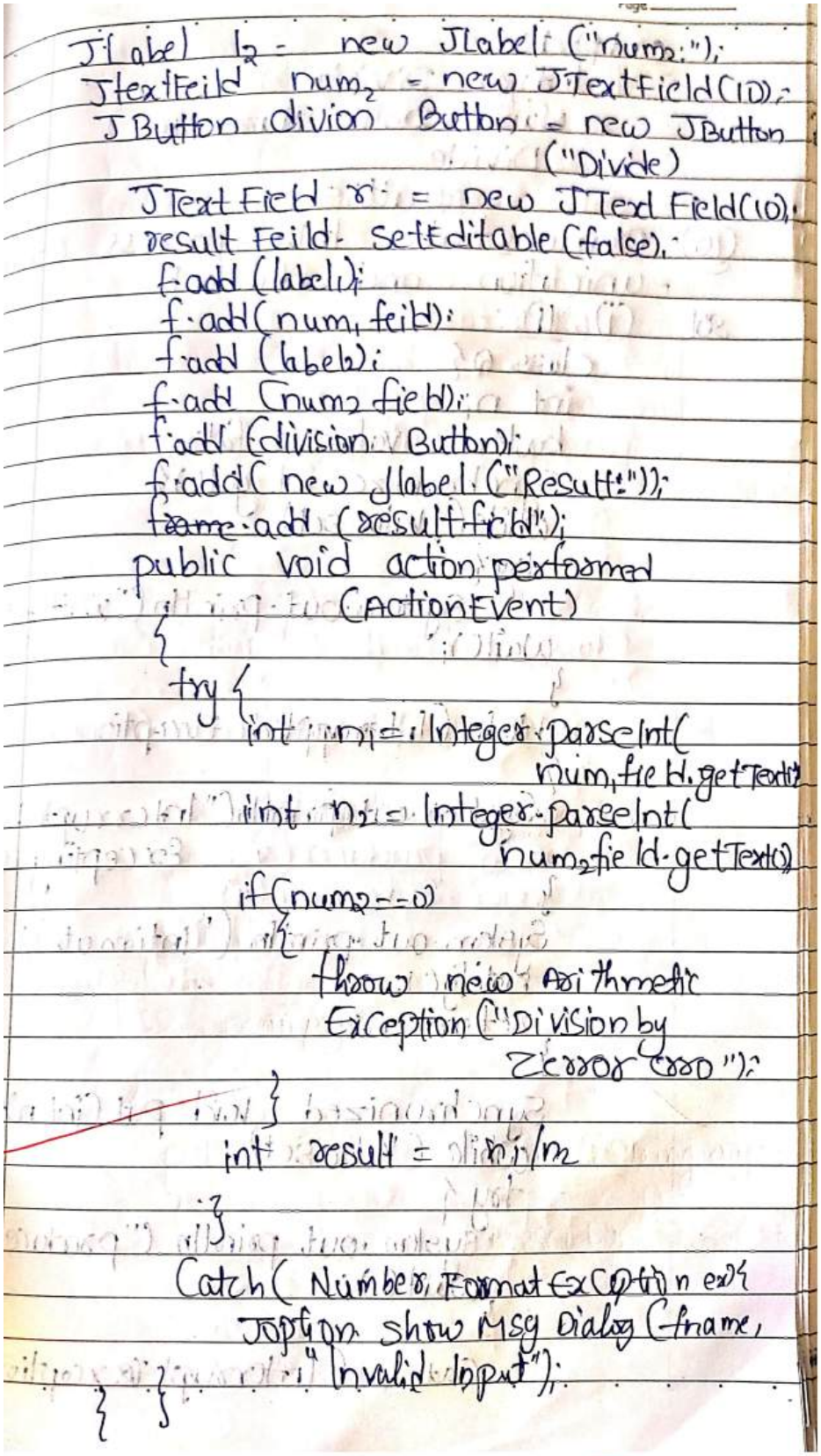
answer:
Algorithm:



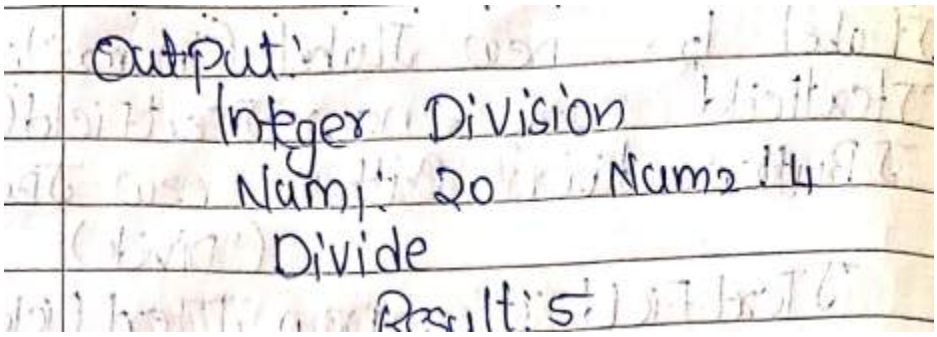
BMSCE

⑨ Pgm to Create an interface to perform integer division. enter num1 & num2 . Divide and display Result. Display on dialog box.

```
import javax.Swing.*;
import java.awt.*;
import java.awt.event.*;
public Class DivisionApp{
public static void main (String args[])
{
    JFrame f = new JFrame
                ("int (div")
    f.setSize (400,300)
    f.setdefault Close operation
        (f.exit_on_ Close);
    f. set layout (new flowlayout());
    JLabel l1= new JLabel("n1:")
    JTextfeild n1im1 field
            = new JTextField(10);
```



```
JLabel l2 = new JLabel("num2:");
JTextFeild num2 = new JTextFicld(10);
JButton division Button = new JButton
                          ("Divide")
JTextField r = new JTextField(10);
result Feild.setEditable(false);
f.add(label1);
f.add(num1 feild);
f.add(label2);
f.add(num2 field);
f.add(division Button);
f.add(new JLabel("Result:"));
frame.add(result field);
public void action performed
          (ActionEvent)
{
 try {
      int n1 = Integer.parseInt(
                num1 field.getText
      int n2 = Integer.parseInt(
                num2 field.getText()
      if(num2==0)
      {
          throw new Arithmetic
          Exception("Division by
                    Zero error ");
      }
      int result = n1/n2
 }
 Catch(NumberFormatException ex{
   JOption show MSg Dialog(frame,
   "Invalid Input");
}}
```



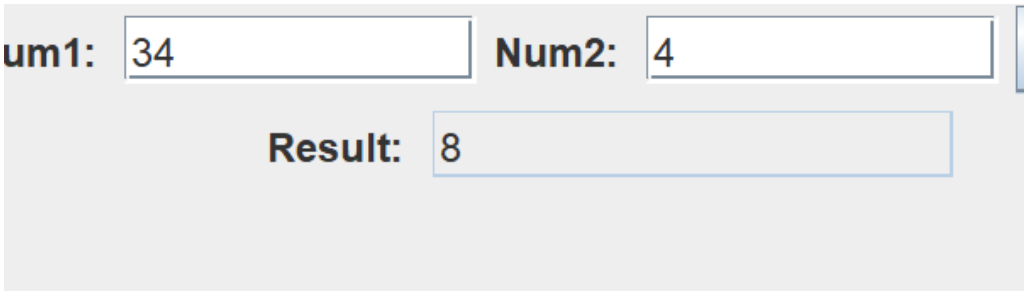
Code:

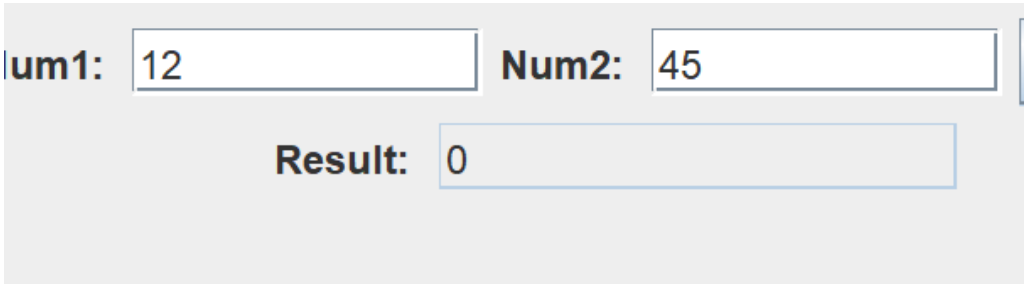
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

```
public class DivisionApp {
   public static void main(String[] args) {
      JFrame frame = new JFrame("Integer
Division");

      JLabel labelNum1 = new
JLabel("Num1:");
      JTextField textFieldNum1 = new
JTextField(10);

      JLabel labelNum2 = new
JLabel("Num2:");
      JTextField textFieldNum2 = new
JTextField(10);

      JLabel labelResult = new
JLabel("Result:");
      JTextField textFieldResult = new
JTextField(10);
      textFieldResult.setEditable(false);

      JButton divideButton = new
JButton("Divide");

      divideButton.addActionListener(new
ActionListener() {
                  public void
actionPerformed(ActionEvent e) {
            try {
               String num1 Text =
textFieldNum1.getText();
```

```
                String num2Text =
textFieldNum2.getText();

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

                if (num2 == 0) {
                    throw new
ArithmeticException("Cannot divide by
zero.");
                }

                int result = num1 / num2;

textFieldResult.setText(String.valueOf(result)
);
            } catch (NumberFormatException
ex) {

JOptionPane.showMessageDialog(frame,
"Invalid input! Please enter integers.",
"Error", JOptionPane.ERROR_MESSAGE);
            } catch (ArithmeticException ex) {

JOptionPane.showMessageDialog(frame,
ex.getMessage(), "Error",
JOptionPane.ERROR_MESSAGE);
            }
        }
    });
```

```
        frame.setLayout(new FlowLayout());

        frame.add(labelNum1);
        frame.add(textFieldNum1);
        frame.add(labelNum2);
        frame.add(textFieldNum2);
        frame.add(divideButton);
        frame.add(labelResult);
        frame.add(textFieldResult);

        frame.setSize(300, 200);

frame.setDefaultCloseOperation(JFrame.EXI
T_ON_CLOSE);
        frame.setVisible(true);
    }
}
```



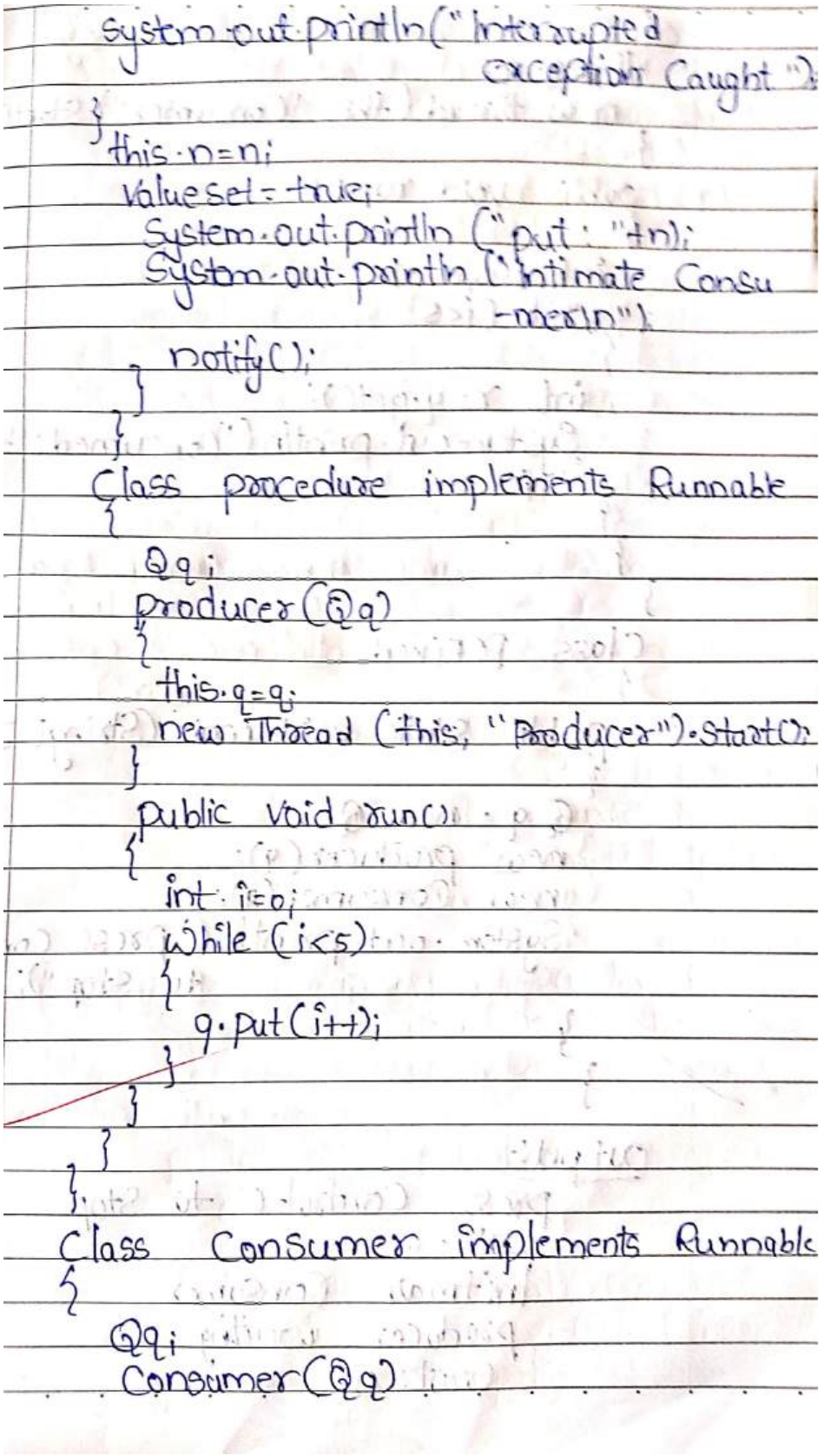


Program 10:
Implementation of Deadlock and IPC algorithm:

(10) Demonstrate Inter process Communication and deadlock

(i) IPC:-

```
class Q{
 int n;
 boolean valueset false;
 Synchronize int get(){
 While(!valueset)
 try{
    System.out.println("Waiting");
 Wait();
 }
 Catch (Interrupted Exception e)
 {
 System.out.println("Interrupted
                    Excepting Caught
 }
 System.out.println("Intimate");
 notify();
 return n;
 }
 Synchronized void put(int n){
 While (Valueset);
  try{
    System.out.println("producer");
    wait();
  }
    Catch (Interrupt Exception e){
```



```
System.out.println("Interrupted
                    exception Caught");
}
this.n=n;
valueset=true;
System.out.println("put: "+n);
System.out.println("Intimate Consu
                    -mer\n");
notify();
}
}
Class procedure implements Runnable
{
Qq;
Producer(Qq)
{
this.q=q;
new Thread(this, "Producer").start();
}
public void run()
{
int i=0;
while(i<5)
{
q.put(i++);
}
}
}
Class Consumer implements Runnable
{
Qq;
Consumer(Qq)
```

IPC:

```
@@ -0,0 +1,80 @@
// Lab_no_10
class Q {
    int n;
    boolean valueSet = false;

    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("\nConsumer
waiting\n");
                wait();
            } catch (InterruptedException e) {

System.out.println("InterruptedException
caught");
            }
        }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\nIntimate
Producer\n");
        notify();
        return n;
    }

    synchronized void put(int n) {
        while (valueSet) {
            try {
                System.out.println("\nProducer
waiting\n");
                wait();
```

```
            } catch (InterruptedException e) {

System.out.println("InterruptedException
caught");
            }
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\nIntimate
Consumer\n");
        notify();
    }
}

class Producer implements Runnable {
    Q q;

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

    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
```

```
    Q q;

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

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

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

```
{
this.q=q;
  new thread (this, "Consumer").start();
}
public void run()
{
  int i=0;
  while (i<5)
  {
    int r=q.get();
    System.out.println("Consumed:"+r);
    i++;
  }
}
}
Class PCFixed
{
public static void main(String[] args)
{
  Q q = new Q();
  new producer(q);
  new Consumer(q);
  System.out.println("press Control-c
                      to stop");
}
}
```

Output:-

press Control-C to stop.
put = 0
Initmate Consumer
producer waiting
Got: 0

Intimate producer
producer waiting
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

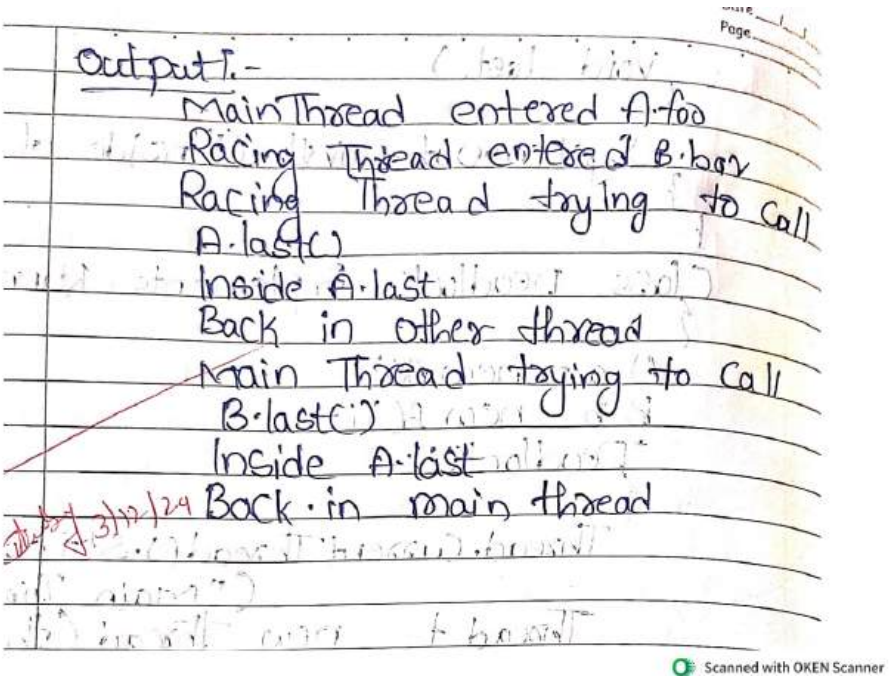
Put: 3
Intimate
Consumer
producer
waiting
Got: 3
Intimate
producer
Consumed: 3
put: 4
Intimate
Consumer
Got: 4
Intimate
producer
Consumed: 4

(ii) Dead Lock

```
Class A
{
 Synchronized void foo(B b)
 {
  String name = Thread.currentThread
  System.out.println("name" + "entered
                      A.foo");
  try
  {
   Thread.sleep(1000);
  }
  Catch (Exception e)
  {
   System.out.println("A Interrupte
                       d");
  }
```

```
System.out.println(name+"Trying to call"+ B.last());
b.last();
}
void last()
{
System.out.println("Inside A.last");
}
}
Class B
{
Synchronized void(Aa)
{
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 of A.last");
}
}
Class Deadlock implements Runnable
{
A a = new A();
B b = new B();
Deadlock()
{
Thread.currentThread().setName
("Main Thread");
Thread t = new Thread(this,
"Racing Thread");
t.start();
a.foo(b);
System.out.println("Back in
main thread");
}
public void run()
{
b.bar(a);
System.out.println("Back in
OtherThread");
}
public static void main(String[]
args)
{
new Deadlock();
}
}
```



code:

```
public class DeadlockExample {

    private static final Object lock1 = new Object();
    private static final Object lock2 = new Object();

    public static void main(String[] args) {
```

```
        Thread thread1 = new Thread(new
Runnable() {
                    public void run() {
                synchronized (lock1) {
                    System.out.println("Thread 1:
Holding lock 1, waiting for lock 2...");
                    try { Thread.sleep(100); } catch
(InterruptedException e) {}
                    synchronized (lock2) {
                        System.out.println("Thread 1:
Acquired lock 2!");
                    }
                }
            }
        });

        Thread thread2 = new Thread(new
Runnable() {
          // Lab_no_10
class Q {
    int n;
    boolean valueSet = false;

    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("\nConsumer
waiting\n");
                wait();
            } catch (InterruptedException e) {

System.out.println("InterruptedException
```

```
caught");
            }
        }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\nIntimate
Producer\n");
        notify();
        return n;
    }

    synchronized void put(int n) {
        while (valueSet) {
            try {
                System.out.println("\nProducer
waiting\n");
                wait();
            } catch (InterruptedException e) {

System.out.println("InterruptedException
caught");
            }
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\nIntimate
Consumer\n");
        notify();
    }
}
```

```
class Producer implements Runnable {
    Q q;

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

    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Q q;

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

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

```
}

public class PCFixed {
   public static void main(String args[]) {
      Q q = new Q();
      new Producer(q);
      new Consumer(q);
      System.out.println("Press Control-C to
stop.");
   }
}
         public void run() {
            synchronized (lock2) {
               System.out.println("Thread 2:
Holding lock 2, waiting for lock 1...");
               try { Thread.sleep(100); } catch
(InterruptedException e) {}
               synchronized (lock1) {
                  System.out.println("Thread 2:
Acquired lock 1!");
               }
            }
         }
      });

      thread1.start();
      thread2.start();
   }
}
```

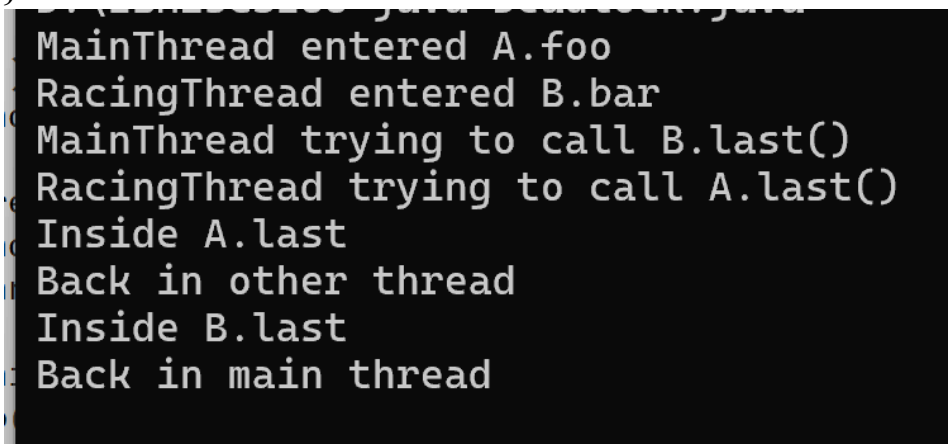
IPC:

```
@@ -0,0 +1,80 @@
// Lab_no_10
class Q {
    int n;
    boolean valueSet = false;

    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("\nConsumer
waiting\n");
                wait();
            } catch (InterruptedException e) {

System.out.println("InterruptedException
caught");
            }
```

```
        }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\nIntimate
Producer\n");
        notify();
        return n;
    }

    synchronized void put(int n) {
        while (valueSet) {
            try {
                System.out.println("\nProducer
waiting\n");
                wait();
            } catch (InterruptedException e) {

System.out.println("InterruptedException
caught");
            }
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\nIntimate
Consumer\n");
        notify();
    }
}

class Producer implements Runnable {
    Q q;
```

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

    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Q q;

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

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

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



```
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

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: 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
```