VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHREYA RAJ (1BM23CS317)

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

Sep2024-Jan 2025

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

CERTIFICATE

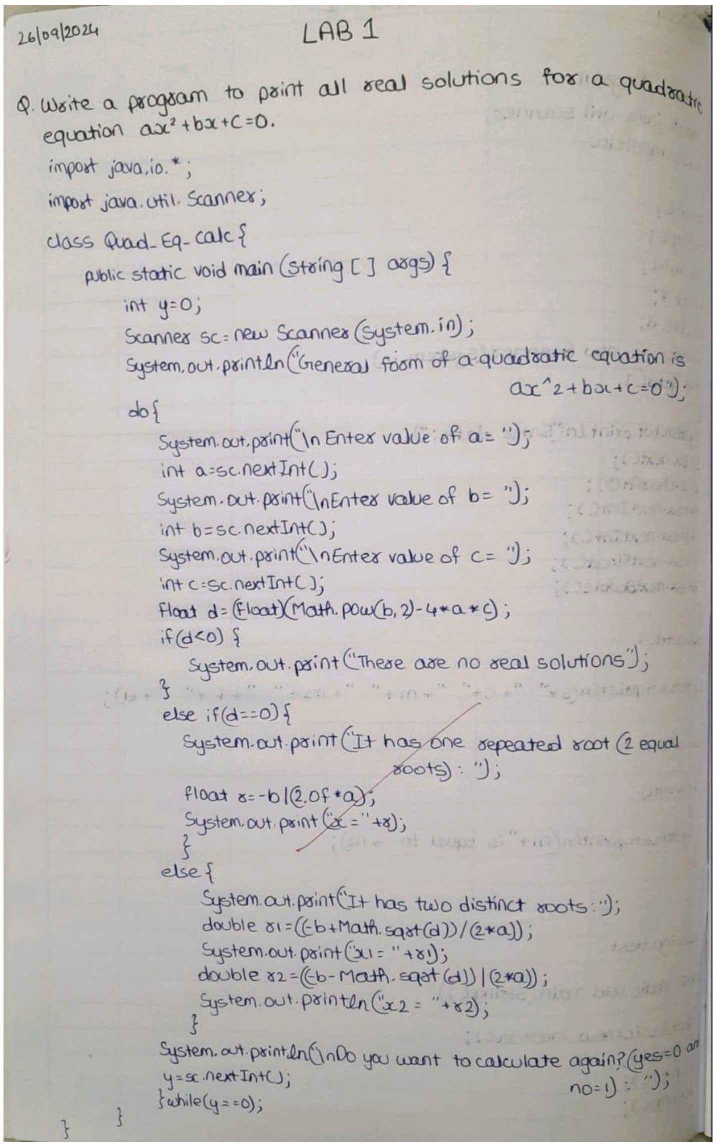
This is to certify that the Lab work entitled "**OBJECT ORIENTED JAVA PROGRAMMING**" carried out by **SHREYA RAJ(1BM23CS317),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

Dr. Nandhini Vineeth	**Dr. Kavitha Sooda**
Associate Professor, Department of CSE, BMSCE, Bengaluru	Professor and Head, Department of CSE BMSCE, Bengaluru

INDEX

Program 1

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

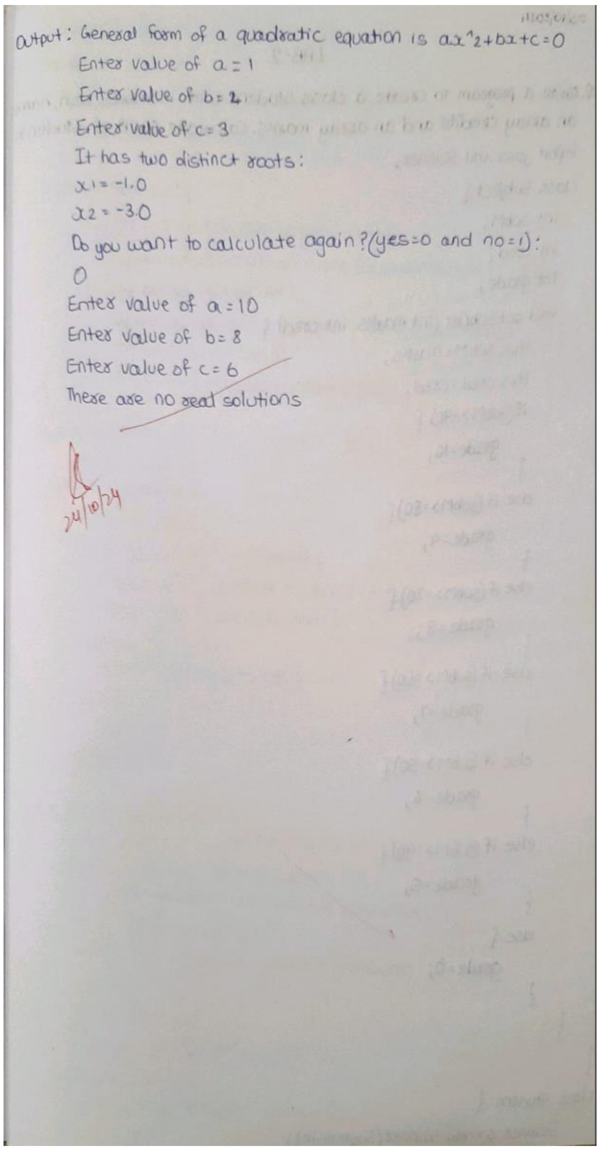


26/09/2024

LAB 1

Q. Write a program to print all real solutions for a quadratic equation $ax^2+bx+C=0$.

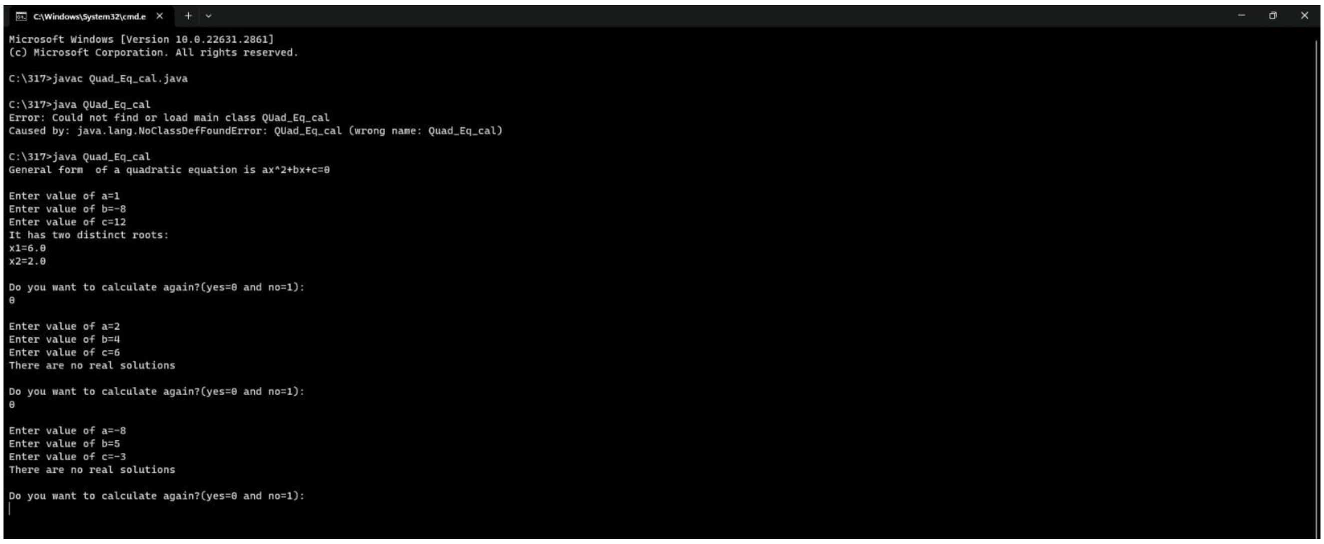
```
import java.io.*;
import java.util.Scanner;
class Quad_Eq_calc{
    public static void main (String [] args) {
        int y=0;
        Scanner sc=new Scanner(System.in);
        System.out.println("General form of a quadratic equation is
                                              ax^2+bx+c=0");
        do{
            System.out.print("\n Enter value of a= ");
            int a=sc.nextInt();
            System.out.print("\nEnter value of b= ");
            int b=sc.nextInt();
            System.out.print("\nEnter value of c= ");
            int c=sc.nextInt();
            Float d=(Float)(Math.pow(b,2)-4*a*c);
            if(d<0) {
               System.out.print("These are no real solutions");
            }
            else if(d==0){
              System.out.print("It has one repeated root (2 equal
                                         roots): ");
              float r=-b/(2.0f*a);
              System.out.print("x= "+r);
              }
            else{
                System.out.print("It has two distinct roots:");
                double r1=((-b+Math.sqrt(d))/(2*a));
                System.out.print("x1= "+r1);
                double r2=((-b-Math.sqrt(d))/(2*a));
                System.out.println("x2= "+r2);
              }
            System.out.println("\nDo you want to calculate again?(yes=0 and
            y=sc.nextInt();                                 no=1): ");
            }while(y==0);
        }
}
```



```
output: General form of a quadratic equation is ax^2+bx+c=0
        Enter value of a=1
        Enter value of b=2
        Enter value of c=3
        It has two distinct roots:
        x1=-1.0
        x2=-3.0
        Do you want to calculate again?(yes=0 and no=1):
        0
        Enter value of a=10
        Enter value of b=8
        Enter value of c=6
        There are no real solutions
```

[illegible]
24/10/24

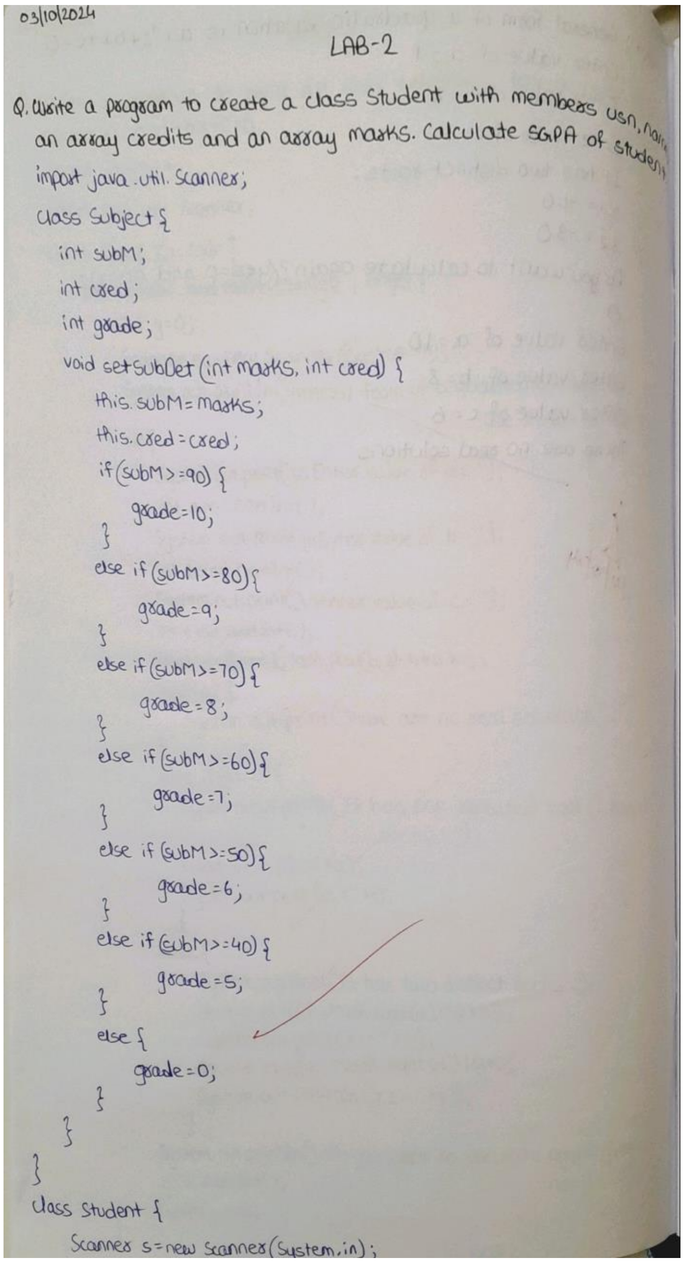
```
import java.util.Scanner;

class Quad_Eq_cal{
    public static void main(String [] args){
        int y=0;
        Scanner sc=new Scanner(System.in);
        System.out.println("General form  of a quadratic equation is ax^2+bx+c=0");
        do{
            System.out.print("\nEnter value of a=");
            int a=sc.nextInt();
            System.out.print("Enter value of b=");
            int b=sc.nextInt();
            System.out.print("Enter value of c=");
            int c=sc.nextInt();
            float d=(float)(Math.pow(b,2)-4*a*c);
            if(d<0){
                System.out.println("There are no real solutions");
            }
            else if(d==0){
                System.out.println("It has one repeated root(2 equal roots):");
                float r=-b/(2.0f*a);
                System.out.println("x="+r);
            }
            else{
                System.out.println("It has two distinct roots:");
                double r1=((-b+Math.sqrt(d))/(2*a));
                System.out.println("x1="+r1);
                double r2=((-b-Math.sqrt(d))/(2*a));
                System.out.println("x2="+r2);
            }
            System.out.println("\nDo you want to calculate again?(yes=0 and no=1): ");
            y=sc.nextInt();
        }while(y==0);
    }
}
```



```
Microsoft Windows [Version 10.0.22631.2861]
(c) Microsoft Corporation. All rights reserved.

C:\317>javac Quad_Eq_cal.java

C:\317>java QUad_Eq_cal
Error: Could not find or load main class QUad_Eq_cal
Caused by: java.lang.NoClassDefFoundError: QUad_Eq_cal (wrong name: Quad_Eq_cal)

C:\317>java Quad_Eq_cal
General form  of a quadratic equation is ax^2+bx+c=0

Enter value of a=1
Enter value of b=-8
Enter value of c=12
It has two distinct roots:
x1=6.0
x2=2.0

Do you want to calculate again?(yes=0 and no=1):
0

Enter value of a=2
Enter value of b=4
Enter value of c=6
There are no real solutions

Do you want to calculate again?(yes=0 and no=1):
0

Enter value of a=-8
Enter value of b=5
Enter value of c=-3
There are no real solutions

Do you want to calculate again?(yes=0 and no=1):
```

Program 2

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

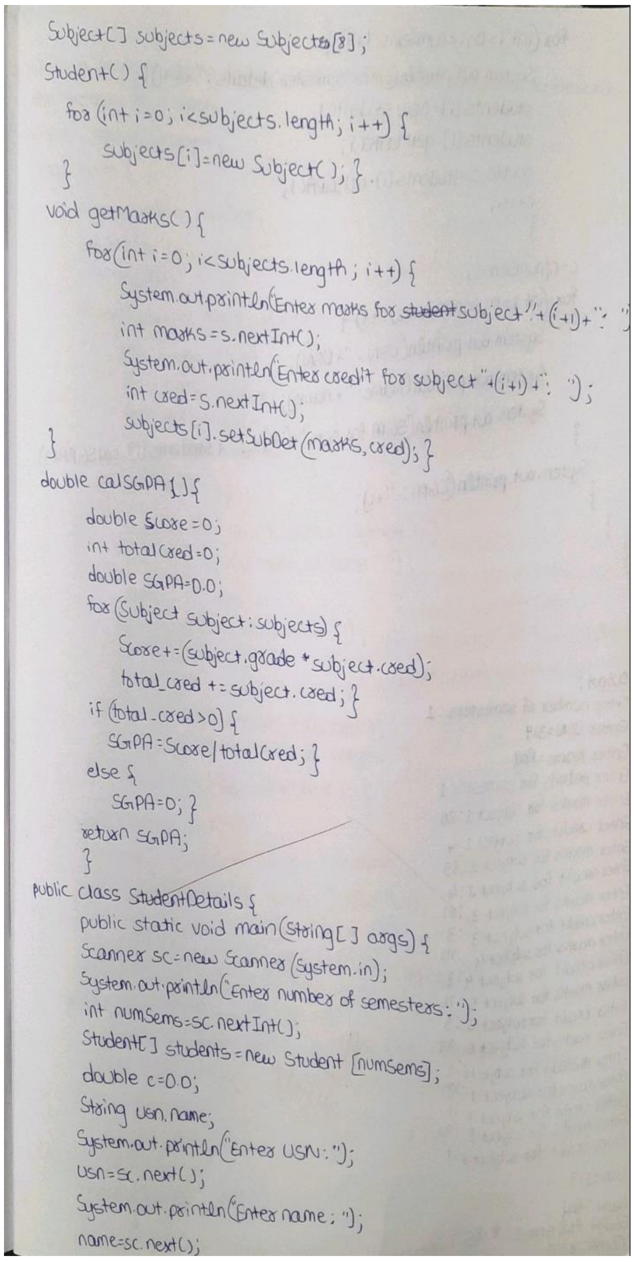


03/10/2024

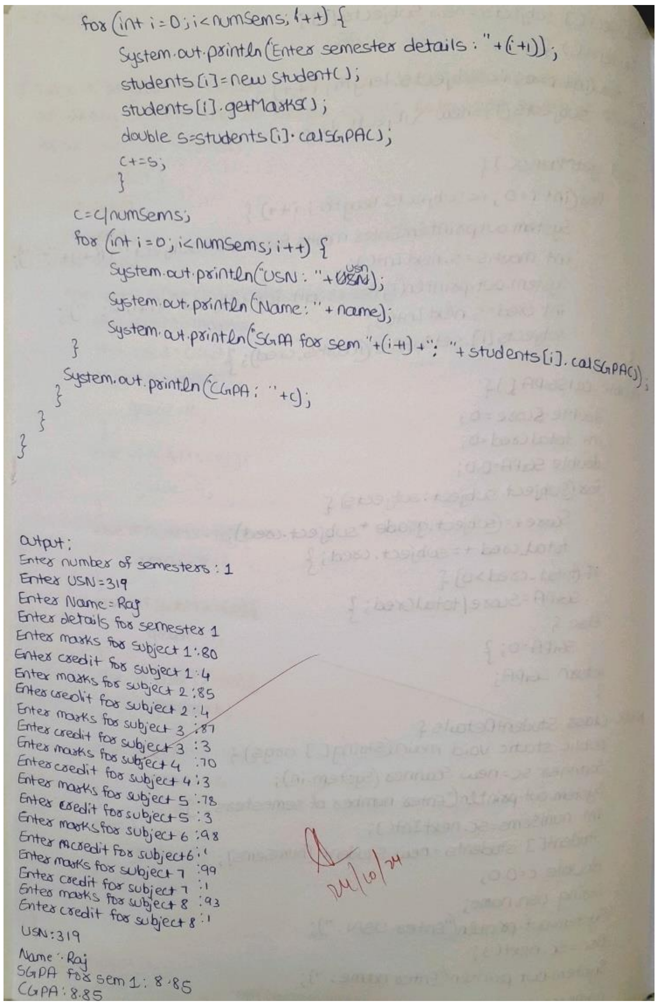
LAB-2

Q. Write a program to create a class Student with members usn, name, an array credits and an array marks. Calculate SGPA of student.

```
import java.util.Scanner;
class Subject {
  int subM;
  int cred;
  int grade;
  void setSubDet (int marks, int cred) {
    this.subM=marks;
    this.cred=cred;
    if(subM>=90) {
      grade=10;
    }
    else if(subM>=80){
      grade=9;
    }
    else if(subM>=70){
      grade=8;
    }
    else if(subM>=60){
      grade=7;
    }
    else if(subM>=50){
      grade=6;
    }
    else if(subM>=40){
      grade=5;
    }
    else {
      grade=0;
    }
  }
}
class Student {
  Scanner s=new Scanner(System.in);
```



```
Subject[] subjects=new Subjects[8];
Student() {
  for (int i=0; i<subjects.length; i++) {
     subjects[i]=new Subject(); }
  }
void getMarks(){
   for(int i=0; i<subjects.length; i++) {
      System.out.println("Enter marks for ~~student~~ subject "+(i+1)+": ");
      int marks=s.nextInt();
      System.out.println("Enter credit for subject "+(i+1)+": ");
      int cred=s.nextInt();
      subjects[i].setSubDet(marks, cred); }
}
double calSGPA(){
   double Score=0;
   int totalCred=0;
   double SGPA=0.0;
   for (Subject subject: subjects) {
      Score+=(subject.grade * subject.cred);
      total_cred += subject.cred; }
   if (total_cred>0) {
     SGPA=Score/totalCred; }
   else {
     SGPA=0; }
  return SGPA;
  }
public class StudentDetails {
   public static void main(String[] args) {
   Scanner sc=new Scanner(System.in);
   System.out.println("Enter number of semesters: ");
   int numSems=sc.nextInt();
   Student[] students=new Student [numSems];
   double c=0.0;
   String usn,name;
   System.out.println("Enter USN: ");
   usn=sc.next();
   System.out.println("Enter name: ");
   name=sc.next();
```

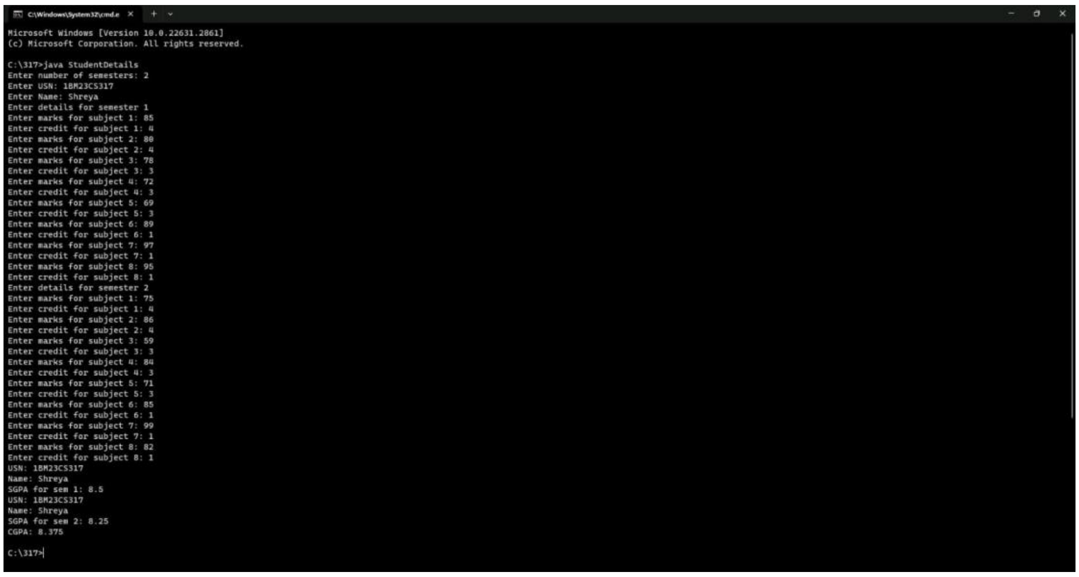


```
        for (int i=0;i<numSems;i++) {
            System.out.println("Enter semester details: "+(i+1));
            students[i]=new Student();
            students[i].getMarks();
            double s=students[i].calSGPA();
            c+=s;
            }
        c=c/numSems;
        for (int i=0;i<numSems;i++) {
            System.out.println("USN: "+~~USN~~usn);
            System.out.println("Name: "+name);
            System.out.println("SGPA for sem "+(i+1)+": "+students[i].calSGPA());
        }
        System.out.println("CGPA: "+c);
      }
   }
}
```

```
output:
Enter number of semesters : 1
Enter USN=319
Enter Name=Raj
Enter details for semester 1
Enter marks for subject 1:80
Enter credit for subject 1:4
Enter marks for subject 2:85
Enter credit for subject 2:4
Enter marks for subject 3 :87
Enter credit for subject 3 :3
Enter marks for subject 4  :70
Enter credit for subject 4:3
Enter marks for subject 5 :78
Enter credit for subject 5 :3
Enter marks for subject 6 :98
Enter ~~m~~credit for subject6 :1
Enter marks for subject 7  :99
Enter credit for subject 7  :1
Enter marks for subject 8  :93
Enter credit for subject 8 :1
USN:319
Name:Raj
SGPA for sem 1: 8.85
CGPA:8.85
```

[illegible] 24/10/24

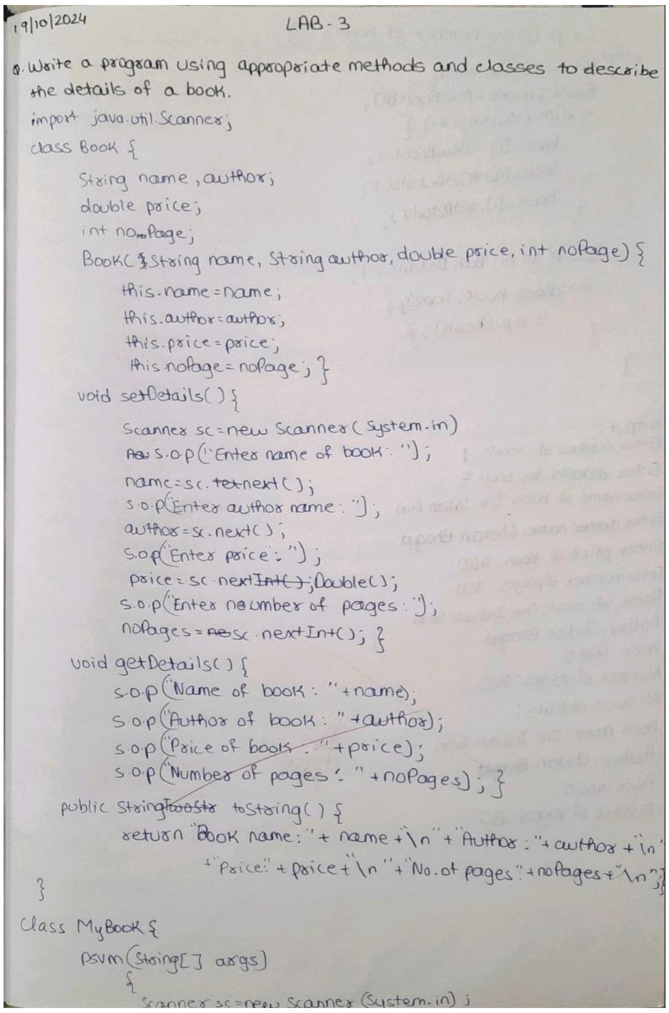
```
import java.util.Scanner;

class Subject {
    int subM;
    int cred;
    int grade;

    void setSubDet(int marks, int cred) {
        this.subM = marks;
        this.cred = cred;

        if (subM >= 90) {
            grade = 10;
        } else if (subM >= 80) {
            grade = 9;
        } else if (subM >= 70) {
            grade = 8;
        } else if (subM >= 60) {
            grade = 7;
        } else if (subM >= 50) {
            grade = 6;
        } else if (subM >= 40) {
            grade = 5;
        } else {
            grade = 0;
        }
    }
}

class Student {

    Scanner s = new Scanner(System.in);
    Subject[] subjects = new Subject[8];

    Student() {
        for (int i = 0; i < subjects.length; i++) {
            subjects[i] = new Subject();
        }
    }

    void getMarks() {
        for (int i = 0; i < subjects.length; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            int marks = s.nextInt();
            System.out.print("Enter credit for subject " + (i + 1) + ": ");
            int cred = s.nextInt();
            subjects[i].setSubDet(marks, cred);
        }
    }

    double calSGPA() {
        double Score = 0;
        int totalCred = 0;
        double SGPA = 0.0;

        for (Subject subject : subjects) {
            Score += (subject.grade * subject.cred);
            totalCred += subject.cred;
        }

        if (totalCred > 0) {
            SGPA = Score / totalCred;
        } else {
            SGPA = 0;
        }
        return SGPA;
    }
}

public class StudentDetails {

    public static void main(String[] arg) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of semesters: ");
        int numSems = sc.nextInt();

        Student[] students = new Student[numSems];
        double cumulativeSGPA = 0.0;


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

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


        for (int i = 0; i < numSems; i++) {
            System.out.println("Enter details for semester " + (i + 1));
            students[i] = new Student();
            students[i].getMarks();
            double semSGPA = students[i].calSGPA();
            cumulativeSGPA += semSGPA;
        }


        for (int i = 0; i < numSems; i++) {
            System.out.println("USN: " + usn);
            System.out.println("Name: " + name);
            System.out.println("SGPA for sem " + (i + 1) + ": " + students[i].calSGPA());
        }


        double CGPA = cumulativeSGPA / numSems;
        System.out.println("CGPA: " + CGPA);
    }
```



```
Microsoft Windows [Version 10.0.22631.2861]
(c) Microsoft Corporation. All rights reserved.

C:\317>java StudentDetails
Enter number of semesters: 2
Enter USN: 1BM23CS317
Enter Name: Shreya
Enter details for semester 1
Enter marks for subject 1: 85
Enter credit for subject 1: 4
Enter marks for subject 2: 80
Enter credit for subject 2: 4
Enter marks for subject 3: 78
Enter credit for subject 3: 3
Enter marks for subject 4: 72
Enter credit for subject 4: 3
Enter marks for subject 5: 69
Enter credit for subject 5: 3
Enter marks for subject 6: 89
Enter credit for subject 6: 1
Enter marks for subject 7: 97
Enter credit for subject 7: 1
Enter marks for subject 8: 95
Enter credit for subject 8: 1
Enter details for semester 2
Enter marks for subject 1: 75
Enter credit for subject 1: 4
Enter marks for subject 2: 86
Enter credit for subject 2: 4
Enter marks for subject 3: 59
Enter credit for subject 3: 3
Enter marks for subject 4: 84
Enter credit for subject 4: 3
Enter marks for subject 5: 71
Enter credit for subject 5: 3
Enter marks for subject 6: 85
Enter credit for subject 6: 1
Enter marks for subject 7: 99
Enter credit for subject 7: 1
Enter marks for subject 8: 82
Enter credit for subject 8: 1
USN: 1BM23CS317
Name: Shreya
SGPA for sem 1: 8.5
USN: 1BM23CS317
Name: Shreya
SGPA for sem 2: 8.25
CGPA: 8.375

C:\317>
```

Program 3

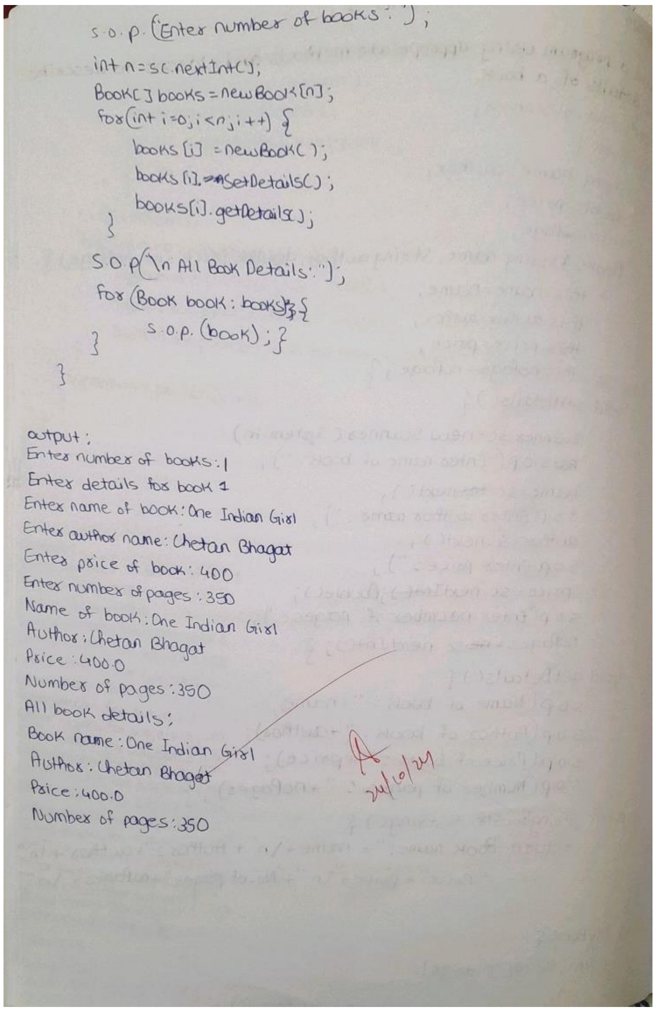
Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program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



19/10/2024 LAB-3

Q. Write a program using appropriate methods and classes to describe the details of a book.

```
import java.util.Scanner;
class Book {
    String name, author;
    double price;
    int noPage;
    Book(~~{~~String name, String author, double price, int noPage) {
        this.name=name;
        this.author=author;
        this.price=price;
        this.noPage=noPage; }
    void setDetails() {
        Scanner sc=new Scanner(System.in)
        ~~Aw~~ s.o.p("Enter name of book: ");
        name=sc.~~tex~~next();
        s.o.p("Enter author name: ");
        author=sc.next();
        s.o.p("Enter price: ");
        price=sc.next~~Int();~~Double();
        s.o.p("Enter noumber of pages: ");
        noPages=~~ne~~sc.nextInt(); }
    void getDetails() {
        s.o.p("Name of book: "+name);
        s.o.p("Author of book: "+author);
        s.o.p("Price of book: "+price);
        s.o.p("Number of pages: "+noPages); }
    public String~~toostr~~ toString() {
        return "Book name: "+ name +"\n"+ "Author: "+ author + "\n"
               +"Price:" + price+"\n"+"No.of pages"+noPages+"\n";}
}
Class MyBook {
    psvm(String[] args)
    {
     Scanner sc=new Scanner (System.in);
```

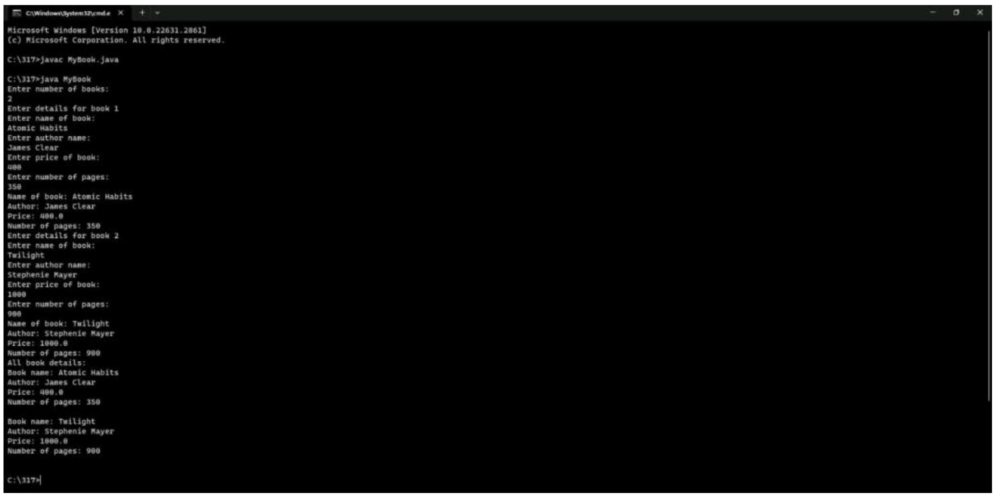


```
     s.o.p.("Enter number of books: ");
     int n=sc.nextInt();
     Book[] books=new Book[n];
     for(int i=0;i<n;i++) {
         books[i] =new Book();
         books[i].setDetails();
         books[i].getDetails();
     }
     S.O.P("\n All Book Details:");
     for (Book book: books){
         s.o.p.(book);}
     }
}
```

```
output:
Enter number of books:1
Enter details for book 1
Enter name of book:One Indian Girl
Enter author name:Chetan Bhagat
Enter price of book:400
Enter number of pages:350
Name of book:One Indian Girl
Author:Chetan Bhagat
Price:400.0
Number of pages:350
All book details:
Book name:One Indian Girl
Author:Chetan Bhagat
Price:400.0
Number of pages:350
```

[illegible] 24/10/24

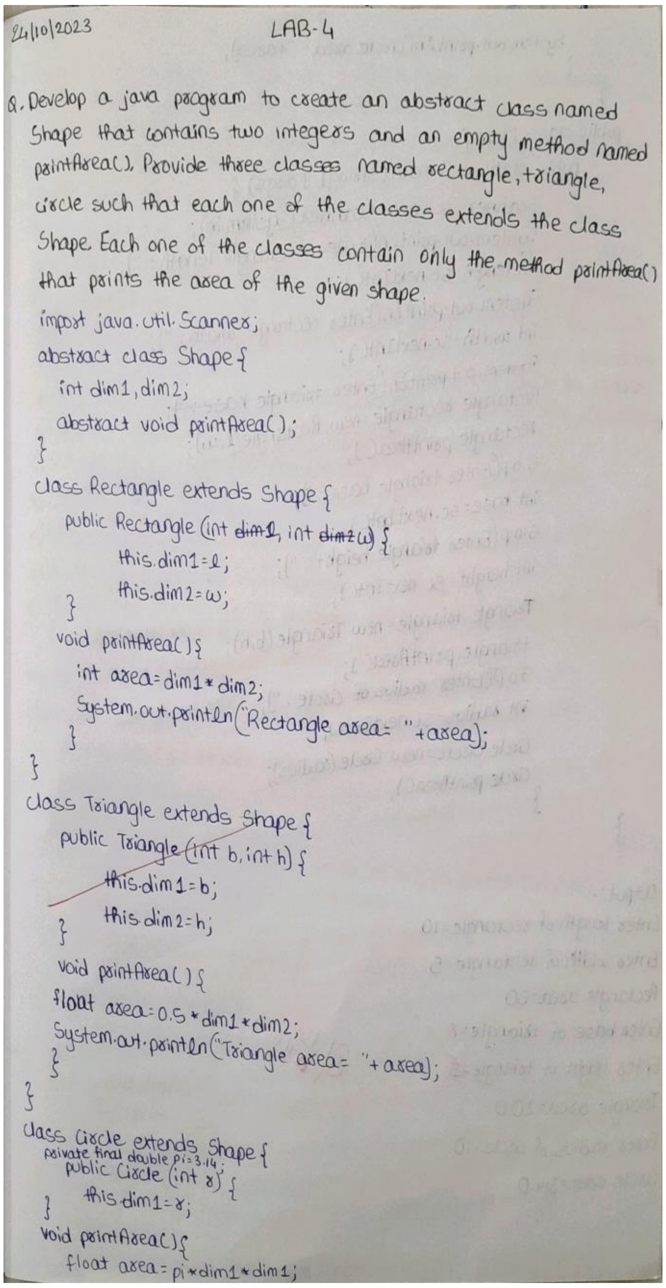
```
import java.util.Scanner;
class Book {
    String name, author;
    double price;
    int noPage;
    Book() {}
    Book(String name, String author, double price, int noPage) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.noPage = noPage;
    }
    void setDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter name of book: ");
        name = sc.nextLine();
        System.out.println("Enter author name: ");
        author = sc.nextLine();
        System.out.println("Enter price of book: ");
        price = sc.nextDouble();
        System.out.println("Enter number of pages: ");
        noPage = sc.nextInt();
    }
    void getDetails() {
        System.out.println("Name of book: " + name);
        System.out.println("Author: " + author);
        System.out.println("Price: " + price);
        System.out.println("Number of pages: " + noPage);
    }
    public String toString() {
        return "Book name: " + name + "\n" + "Author: " + author + "\n" + "Price: " + price + "\n" +
"Number of pages: " + noPage + "\n";
    }
}
class MyBook {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of books: ");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            books[i] = new Book();
            System.out.println("Enter details for book " + (i + 1));
            books[i].setDetails();
            books[i].getDetails();
        }
        System.out.println("All book details: ");
        for (Book book : books) {
            System.out.println(book);
        }
    }
}
```



```
Microsoft Windows [Version 10.0.22631.2861]
(c) Microsoft Corporation. All rights reserved.

C:\317>javac MyBook.java

C:\317>java MyBook
Enter number of books:
2
Enter details for book 1
Enter name of book:
Atomic Habits
Enter author name:
James Clear
Enter price of book:
400
Enter number of pages:
350
Name of book: Atomic Habits
Author: James Clear
Price: 400.0
Number of pages: 350
Enter details for book 2
Enter name of book:
Twilight
Enter author name:
Stephenie Mayer
Enter price of book:
1000
Enter number of pages:
900
Name of book: Twilight
Author: Stephenie Mayer
Price: 1000.0
Number of pages: 900
All book details:
Book name: Atomic Habits
Author: James Clear
Price: 400.0
Number of pages: 350

Book name: Twilight
Author: Stephenie Mayer
Price: 1000.0
Number of pages: 900


C:\317>
```

Program 4

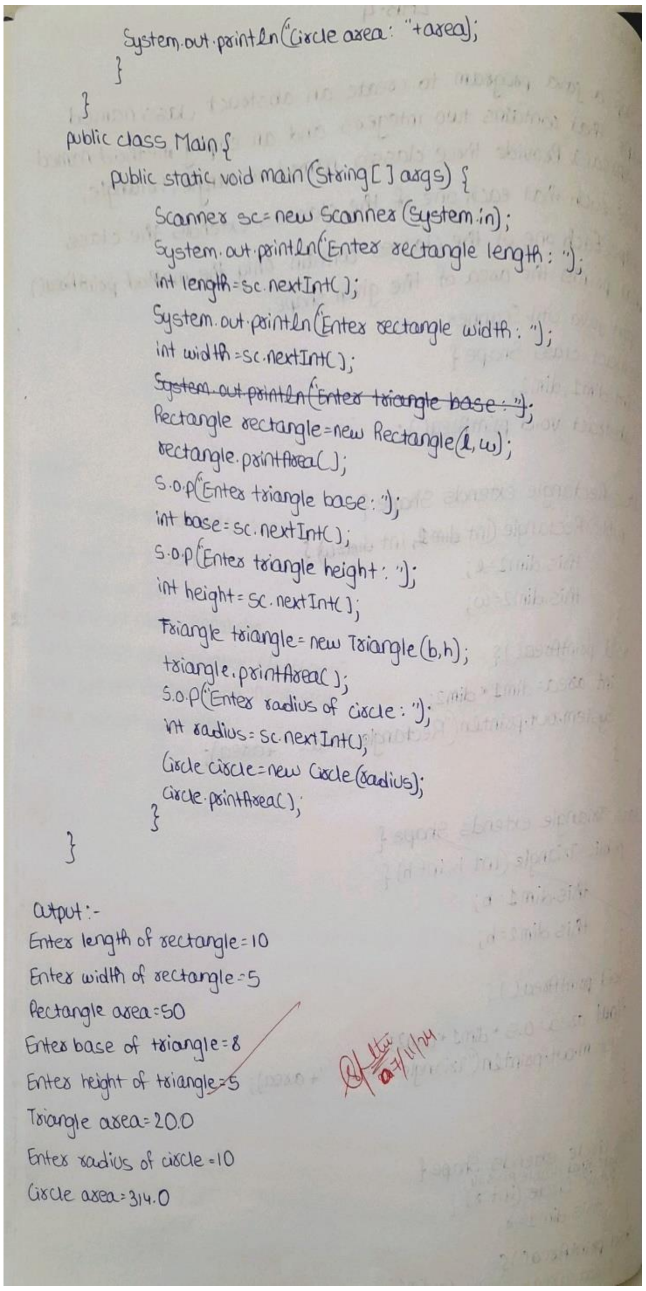
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea()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



24/10/2023 LAB-4

Q. 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 the given shape.

```
import java.util.Scanner;
abstract class Shape{
  int dim1,dim2;
  abstract void printArea();
}
class Rectangle extends Shape{
   public Rectangle(int ~~dim1~~l, int ~~dim2~~w) {
         this.dim1=l;
         this.dim2=w;
   }
  void printArea(){
    int area=dim1*dim2;
    System.out.println("Rectangle area= "+area);
  }
}
class Triangle extends Shape{
   public Triangle(int b,int h){
        this.dim1=b;
        this.dim2=h;
   }
   void printArea(){
   float area=0.5*dim1*dim2;
   System.out.println("Triangle area= "+area);
   }
}
class Circle extends Shape{
  private final double pi=3.14;
    public Circle (int r) {
      this.dim1=r;
  }
  void printArea(){
     float area=pi*dim1*dim1;
```



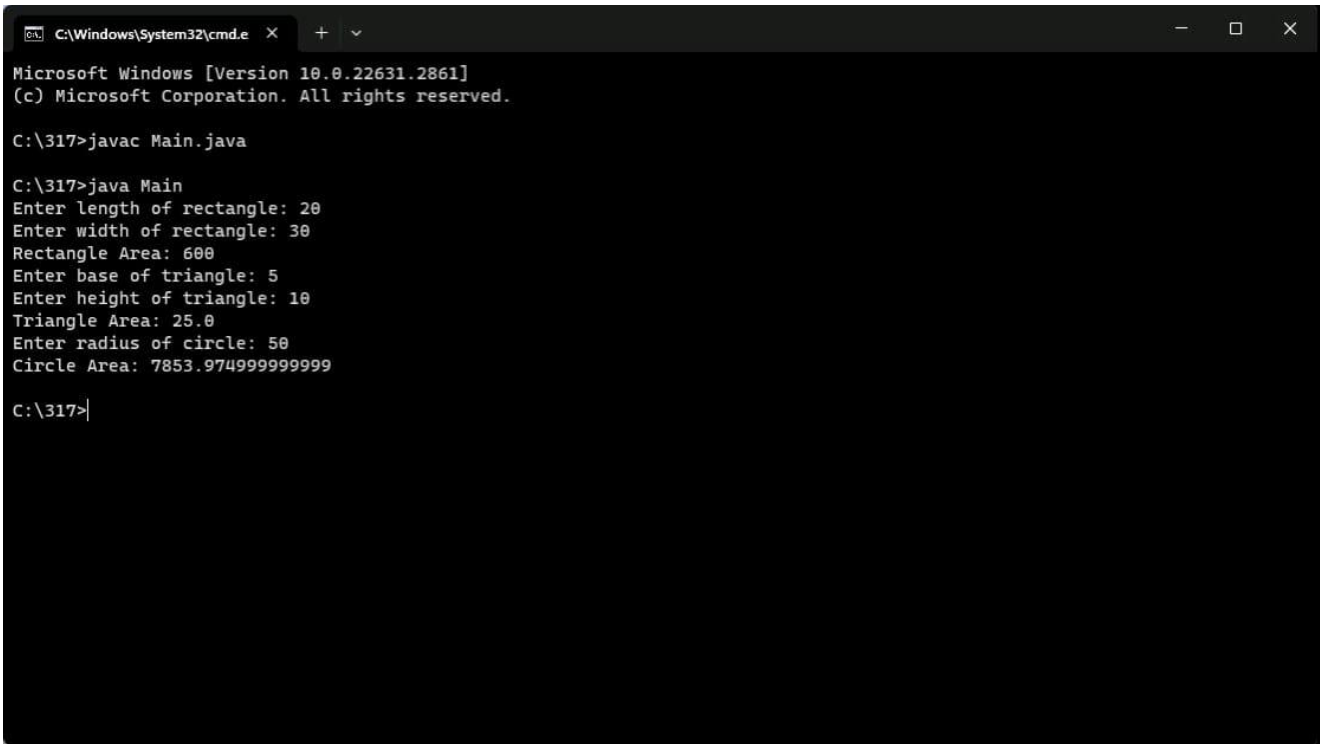
```
        System.out.println("Circle area: "+area);
      }
  }
  public class Main{
     public static void main(String[] args) {
         Scanner sc=new Scanner(System.in);
         System.out.println("Enter rectangle length: ");
         int length=sc.nextInt();
         System.out.println("Enter rectangle width: ");
         int width=sc.nextInt();
         ~~System.out.println("Enter triangle base: ");~~
         Rectangle rectangle=new Rectangle(l,w);
         rectangle.printArea();
         S.o.p("Enter triangle base: ");
         int base=sc.nextInt();
         S.o.p("Enter triangle height: ");
         int height=sc.nextInt();
         Triangle triangle=new Triangle(b,h);
         triangle.printArea();
         S.o.p("Enter radius of circle: ");
         int radius=sc.nextInt();
         Circle circle=new Circle(radius);
         circle.printArea();
       }
  }
```

Output:-

```
Enter length of rectangle=10
Enter width of rectangle=5
Rectangle area=50
Enter base of triangle=8
Enter height of triangle=5
Triangle area=20.0
Enter radius of circle=10
Circle area=314.0
```

[illegible] 7/11/24

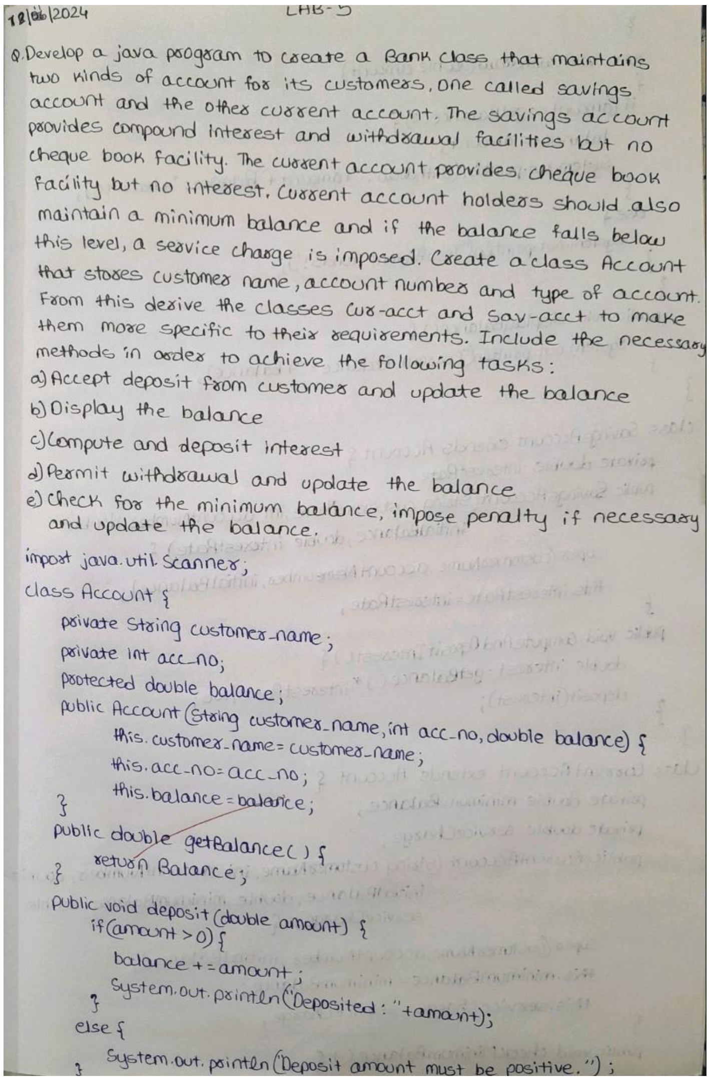
```
import java.util.Scanner;


abstract class Shape {
    int dimension1;
    int dimension2;
    abstract void printArea();
}

class Rectangle extends Shape {


    public Rectangle(int length, int width) {
        this.dimension1 = length;
        this.dimension2 = width;
    }


    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {


    public Triangle(int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
    }



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

class Circle extends Shape {
    private final double pi = 3.14159;


    public Circle(int radius) {
        this.dimension1 = radius;
        this.dimension2 = 0;
    }


    void printArea() {
        double area = pi * dimension1 * dimension1;
        System.out.println("Circle Area: " + area);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);



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


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

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

        scanner.close();
    }
}
```



```
Microsoft Windows [Version 10.0.22631.2861]
(c) Microsoft Corporation. All rights reserved.

C:\317>javac Main.java

C:\317>java Main
Enter length of rectangle: 20
Enter width of rectangle: 30
Rectangle Area: 600
Enter base of triangle: 5
Enter height of triangle: 10
Triangle Area: 25.0
Enter radius of circle: 50
Circle Area: 7853.974999999999

C:\317>
```

Program 5

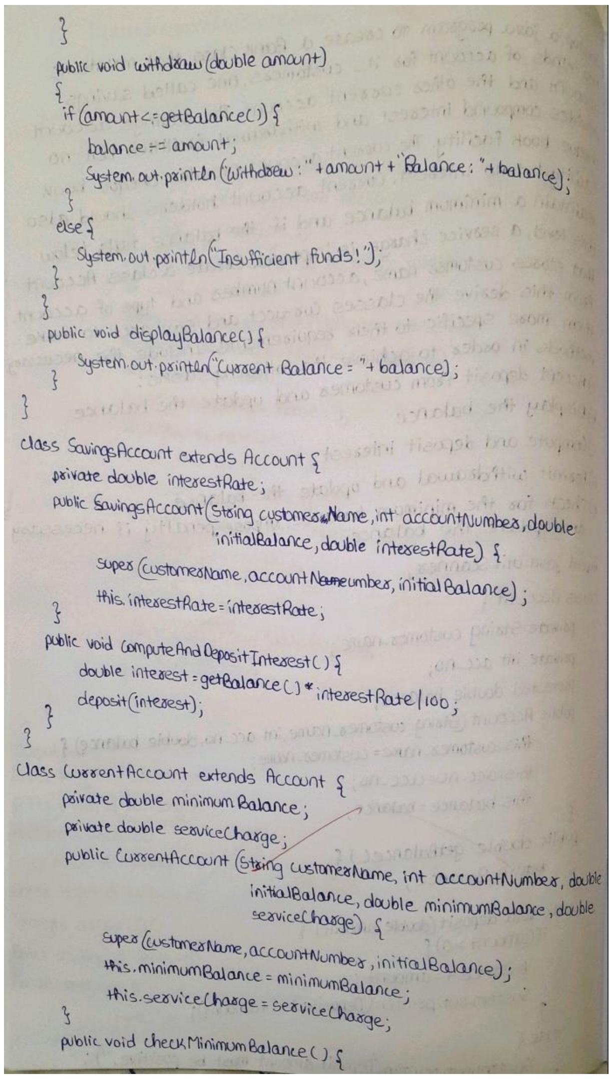
Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book 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 Check for the minimum balance, impose penalty if necessary and update the balance.



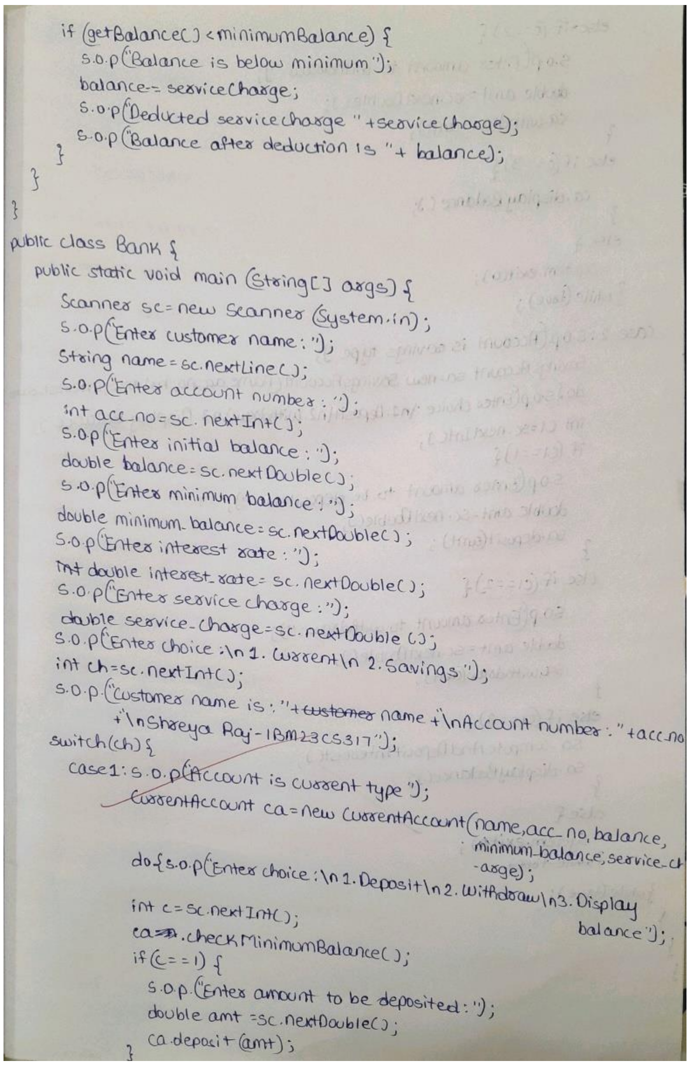
18/06/2024 LAB-5

Q. Develop a java program to create a Bank class 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.

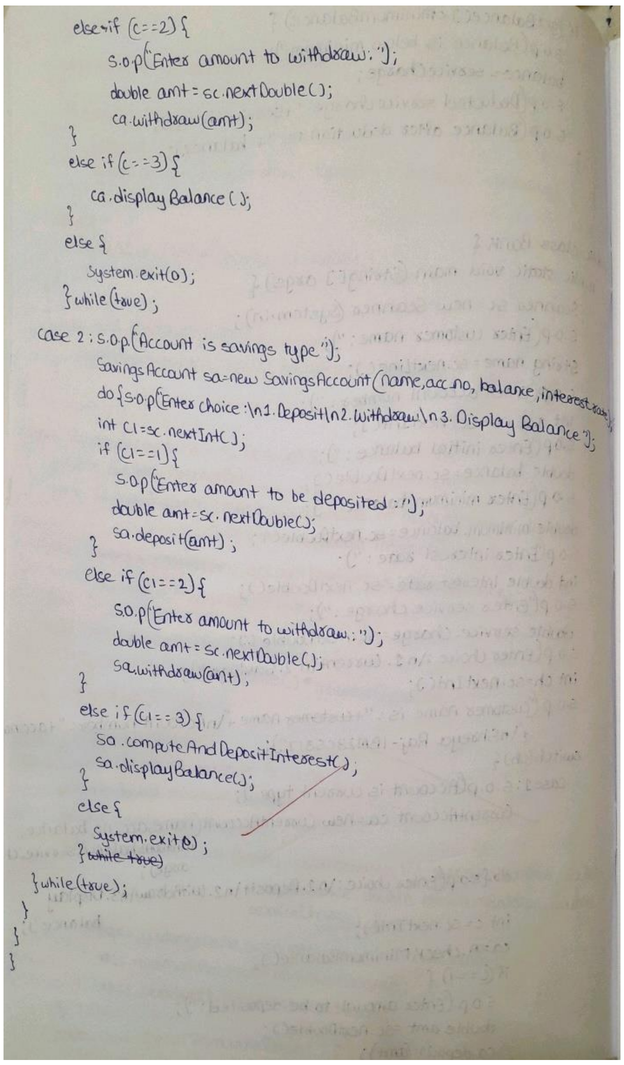
```
import java.util.Scanner;
class Account {
    private String customer_name;
    private int acc_no;
    protected double balance;
    public Account (String customer_name, int acc_no, double balance) {
            this.customer_name = customer_name;
            this.acc_no = acc_no;
            this.balance = balance;
    }
    public double getBalance() {
        return Balance;
    }
    public void deposit (double amount) {
        if(amount > 0) {
            balance += amount;
            System.out.println("Deposited : "+amount);
        }
        else {
            System.out.println("Deposit amount must be positive.");
        }
```



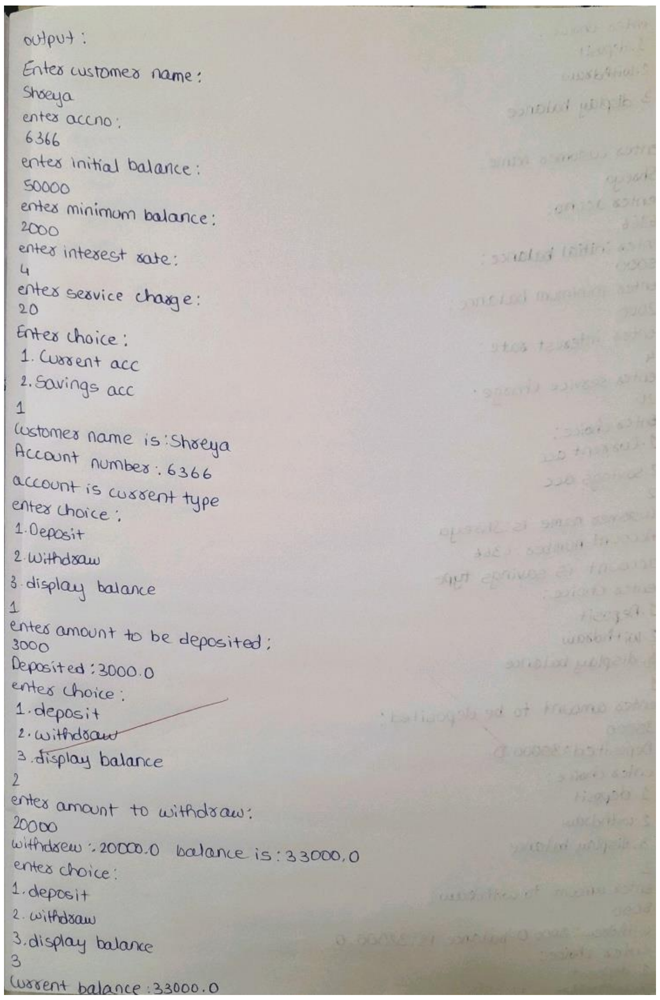
```
    }
    public void withdraw (double amount)
    {
     if (amount <= getBalance()) {
        balance -= amount;
        System.out.println("Withdrew : "+amount + "Balance : "+ balance);
     }
    else {
       System.out.println("Insufficient funds!");
    }
  }
  public void displayBalance() {
       System.out.println("Current Balance = "+ balance);
    }
}
class SavingsAccount extends Account {
    private double interestRate;
    public SavingsAccount(String customerName, int accountNumber, double
                          initialBalance, double interestRate) {
        super (customerName, account~~Name~~umber, initialBalance);
        this.interestRate = interestRate;
    }
    public void computeAndDepositInterest() {
        double interest = getBalance() * interestRate/100;
        deposit(interest);
    }
}
Class CurrentAccount extends Account {
    private double minimumBalance;
    private double serviceCharge;
    public CurrentAccount (String customerName, int accountNumber, double
                           initialBalance, double minimumBalance, double
                           serviceCharge) {
        super (customerName, accountNumber, initialBalance);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }
    public void checkMinimumBalance() {
```



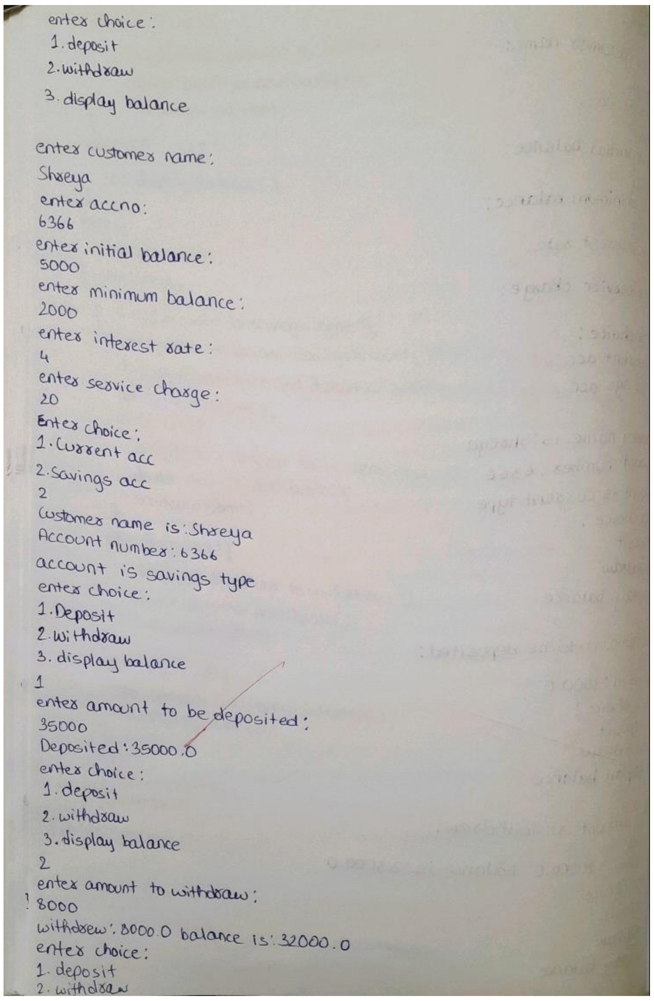
```
    if (getBalance() <minimumBalance) {
      s.o.p("Balance is below minimum");
      balance-= serviceCharge;
      s.o.p("Deducted service charge " +serviceCharge);
      s.o.p("Balance after deduction is "+ balance);
    }
  }
}
public class Bank {
  public static void main (String[] args) {
    Scanner sc=new Scanner (System.in);
    s.o.p("Enter customer name:");
    String name=sc.nextLine();
    s.o.p("Enter account number:");
    int acc_no=sc.nextInt();
    s.o.p("Enter initial balance:");
    double balance=sc.nextDouble();
    s.o.p("Enter minimum balance:");
    double minimum_balance=sc.nextDouble();
    s.o.p("Enter interest rate:");
    ~~int~~ double interest_rate= sc.nextDouble();
    s.o.p("Enter service charge:");
    double service_charge=sc.nextDouble();
    s.o.p("Enter choice:\n 1. Current\n 2.Savings");
    int ch=sc.nextInt();
    s.o.p("Customer name is:"+ ~~customer~~ name +"\nAccount number:"+acc_no
        +"\nShreya Raj-1BM23CS317");
    switch(ch){
     case1: s.o.p("Account is current type");
       CurrentAccount ca=new CurrentAccount(name,acc_no,balance,
                                            minimum_balance,service_ch
                                            -arge);
       do{s.o.p("Enter choice:\n1.Deposit\n2.Withdraw\n3.Display
                                                        balance");
       int c=sc.nextInt();
       ca~~=a~~.checkMinimumBalance();
       if(c==1) {
         s.o.p.("Enter amount to be deposited:");
         double amt =sc.nextDouble();
         ca.deposit(amt);
       }
```



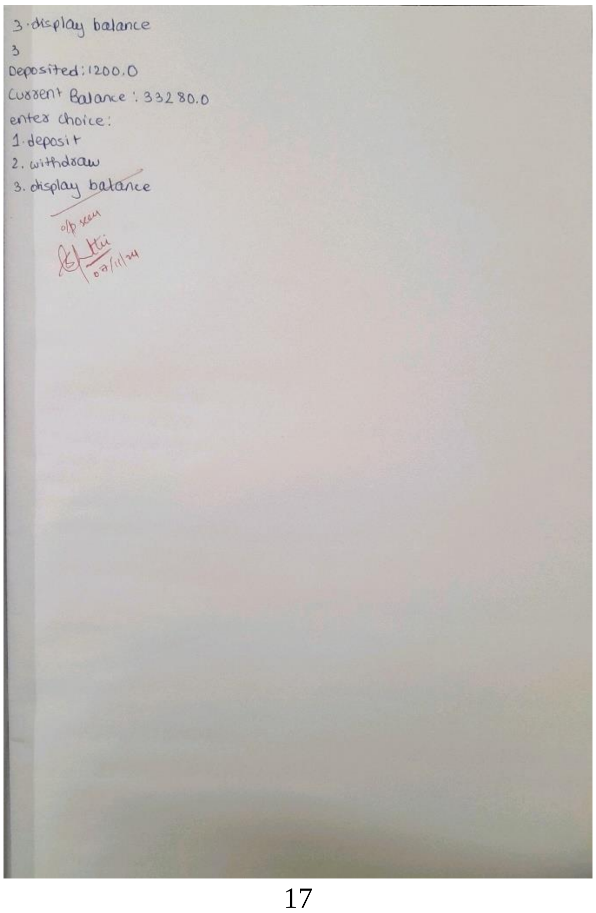
```
       else if (c==2) {
          s.o.p("Enter amount to withdraw:");
          double amt=sc.nextDouble();
          ca.withdraw(amt);
       }
       else if (c==3) {
         ca.displayBalance();
       }
       else {
         System.exit(0);
       }while(true);
     case 2: s.o.p("Account is savings type");
         SavingsAccount sa=new SavingsAccount(name,acc_no,balance,interest_rate);
         do{s.o.p("Enter choice:\n1.Deposit\n2.Withdraw\n3.Display Balance");
         int c1=sc.nextInt();
         if (c1==1){
           s.o.p("Enter amount to be deposited:");
           double amt=sc.nextDouble();
           sa.deposit(amt);
         }
         else if (c1==2){
           s.o.p("Enter amount to withdraw:");
           double amt=sc.nextDouble();
           sa.withdraw(amt);
         }
         else if (c1==3){
           sa.computeAndDepositInterest();
           sa.displayBalance();
         }
         else{
           System.exit(0);
         }~~while true)~~
     }while(true);
    }
  }
}
```



```
output:
Enter customer name:
Shreya
enter accno:
6366
enter initial balance:
50000
enter minimum balance:
2000
enter interest rate:
4
enter service charge:
20
Enter choice:
1. Current acc
2. Savings acc
1
Customer name is: Shreya
Account number: 6366
account is current type
enter choice:
1. Deposit
2. Withdraw
3. display balance
1
enter amount to be deposited:
3000
Deposited: 3000.0
enter choice:
1. deposit
2. withdraw
3. display balance
2
enter amount to withdraw:
20000
withdrew: 20000.0  balance is: 33000.0
enter choice:
1. deposit
2. withdraw
3. display balance
3
Current balance: 33000.0
```



```
enter choice:
1. deposit
2. withdraw
3. display balance

enter customer name:
Shreya
enter accno:
6366
enter initial balance:
5000
enter minimum balance:
2000
enter interest rate:
4
enter service charge:
20
Enter choice:
1. Current acc
2. Savings acc
2
Customer name is: Shreya
Account number: 6366
account is savings type
enter choice:
1. Deposit
2. Withdraw
3. display balance
1
enter amount to be deposited:
35000
Deposited: 35000.0
enter choice:
1. deposit
2. withdraw
3. display balance
2
enter amount to withdraw:
8000
withdrew: 8000.0 balance is: 32000.0
enter choice:
1. deposit
2. withdraw
```

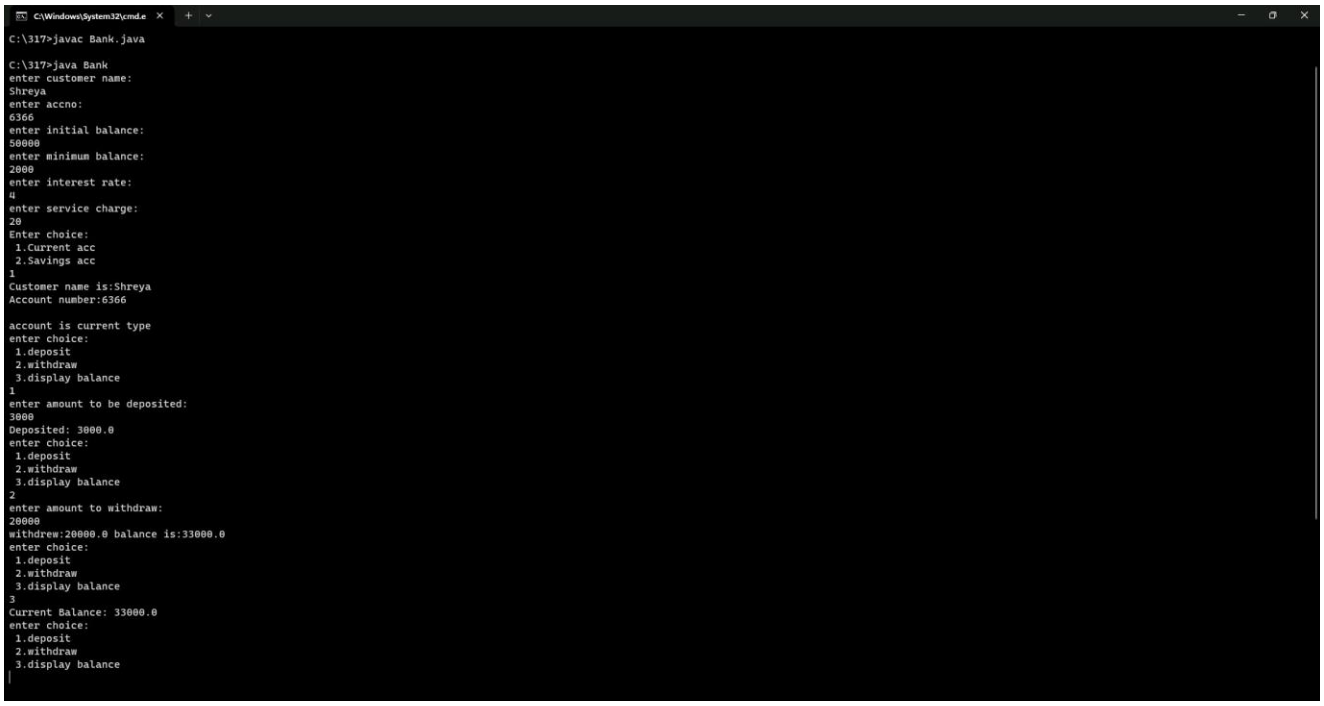


```
3. display balance
3
Deposited: 1200.0
Current Balance: 33280.0
enter choice:
1. deposit
2. withdraw
3. display balance
```

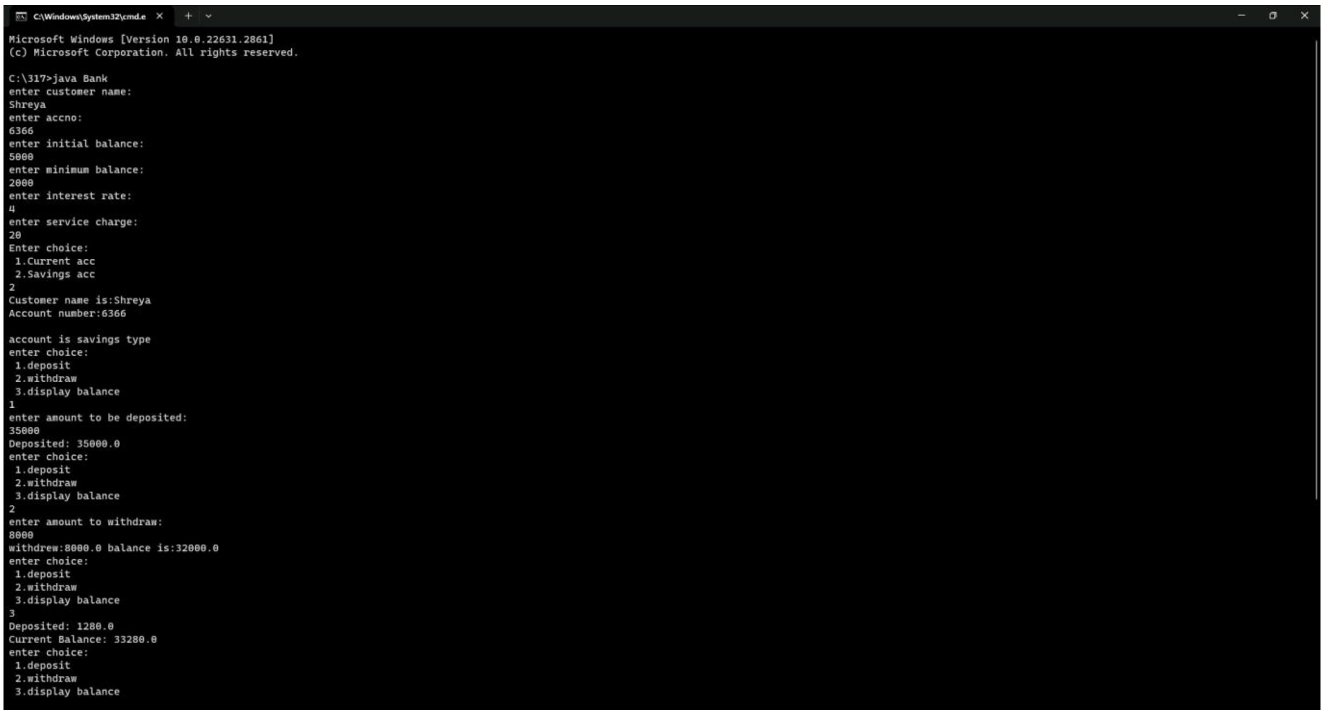
o/p seen

[illegible] 07/11/24

```
import java.util.Scanner;

class Account {
    private String customer_name;
    private int acc_no;
    protected double balance;

    public Account(String customer_name, int acc_no, double balance) {
        this.customer_name = customer_name;
        this.acc_no = acc_no;
        this.balance = balance;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount);
        } else {
            System.out.println("Deposit amount must be positive.");
        }
    }
   public void withdraw(double amount)
     {
       if(amount<=getBalance()){
          balance-=amount;
          System.out.println("withdrew:"+amount + " balance is:"+ balance);
          }
       else
         System.out.println("Insufficient funds!!");
     }
    public void displayBalance(){
        System.out.println("Current Balance: " + balance);
    }
}

class SavingsAccount extends Account {
    private double interestRate;

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

    public void computeAndDepositInterest() {
        double interest = getBalance() * interestRate / 100;
        deposit(interest);
    }
}
class CurrentAccount extends Account {
    private double minimumBalance;
    private double serviceCharge;

    public CurrentAccount(String customerName, int accountNumber, double initialBalance, double
minimumBalance, double serviceCharge) {
        super(customerName, accountNumber, initialBalance);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }
    public void checkMinimumBalance() {
        if (getBalance() < minimumBalance) {
            System.out.println("Balance is below minimum");
            balance-=serviceCharge;
            System.out.println("Deducted service charge:" +serviceCharge);
            System.out.println("Balance after deduction is:"+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.nextLine();
        System.out.println("enter accno:");
        int acc_no=sc.nextInt();
        System.out.println("enter initial balance:");
        double balance=sc.nextDouble();
        System.out.println("enter minimum balance:");
        double minimum_balance=sc.nextDouble();
        System.out.println("enter interest rate:");
        double interest_rate=sc.nextDouble();
        System.out.println("enter service charge:");
        double service_charge=sc.nextDouble();
        System.out.println("Enter choice:\n 1.Current acc\n 2.Savings acc");
        int ch=sc.nextInt();
        System.out.println("Customer name is:"+ name+"\nAccount number:"+acc_no+"\n");

        switch(ch){
            case(1):
                System.out.println("account is current type");
                CurrentAccount ca = new
CurrentAccount(name,acc_no,balance,minimum_balance,service_charge);
               do{ System.out.println("enter choice:\n 1.deposit\n 2.withdraw\n 3.display balance");
                int c=sc.nextInt();
                ca.checkMinimumBalance();
                if(c==1){
                   System.out.println("enter amount to be deposited:");
                   double amt=sc.nextDouble();
                     ca.deposit(amt);}
                else if(c==2){
                   System.out.println("enter amount to withdraw:");
                   double amt=sc.nextDouble();
                   ca.withdraw(amt);}
                else if(c==3){
                   ca.displayBalance();}
                else
                  System.exit(0);
                 }while(true);

            case(2):
                 System.out.println("account is savings type");
                SavingsAccount sa=new SavingsAccount(name,acc_no,balance,interest_rate);
                do{ System.out.println("enter choice:\n 1.deposit\n 2.withdraw\n 3.display balance");
                int c1=sc.nextInt();
                if(c1==1){
                   System.out.println("enter amount to be deposited:");
                   double amt=sc.nextDouble();
                     sa.deposit(amt);}
                else if(c1==2){
                   System.out.println("enter amount to withdraw:");
                   double amt=sc.nextDouble();
                   sa.withdraw(amt);}
                else if(c1==3){
                 sa.computeAndDepositInterest();
                   sa.displayBalance();}
                else{
                  System.exit(0);
                      }
                }while(true);
    }
}
}
```



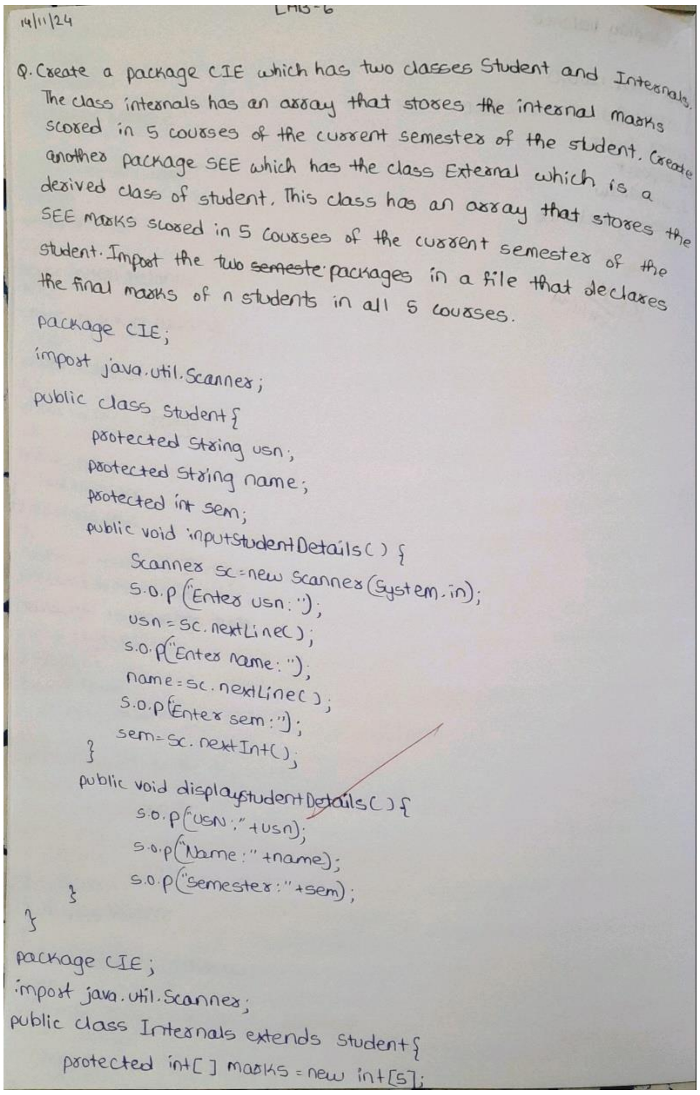
```
C:\317>javac Bank.java

C:\317>java Bank
enter customer name:
Shreya
enter accno:
6366
enter initial balance:
50000
enter minimum balance:
2000
enter interest rate:
4
enter service charge:
20
Enter choice:
 1.Current acc
 2.Savings acc
1
Customer name is:Shreya
Account number:6366

account is current type
enter choice:
 1.deposit
 2.withdraw
 3.display balance
1
enter amount to be deposited:
3000
Deposited: 3000.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
2
enter amount to withdraw:
20000
withdrew:20000.0 balance is:33000.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
3
Current Balance: 33000.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
```



```
Microsoft Windows [Version 10.0.22631.2861]
(c) Microsoft Corporation. All rights reserved.

C:\317>java Bank
enter customer name:
Shreya
enter accno:
6366
enter initial balance:
5000
enter minimum balance:
2000
enter interest rate:
4
enter service charge:
20
Enter choice:
 1.Current acc
 2.Savings acc
2
Customer name is:Shreya
Account number:6366

account is savings type
enter choice:
 1.deposit
 2.withdraw
 3.display balance
1
enter amount to be deposited:
35000
Deposited: 35000.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
2
enter amount to withdraw:
8000
withdrew:8000.0 balance is:32000.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
3
Deposited: 1280.0
Current Balance: 33280.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
```

Program 6

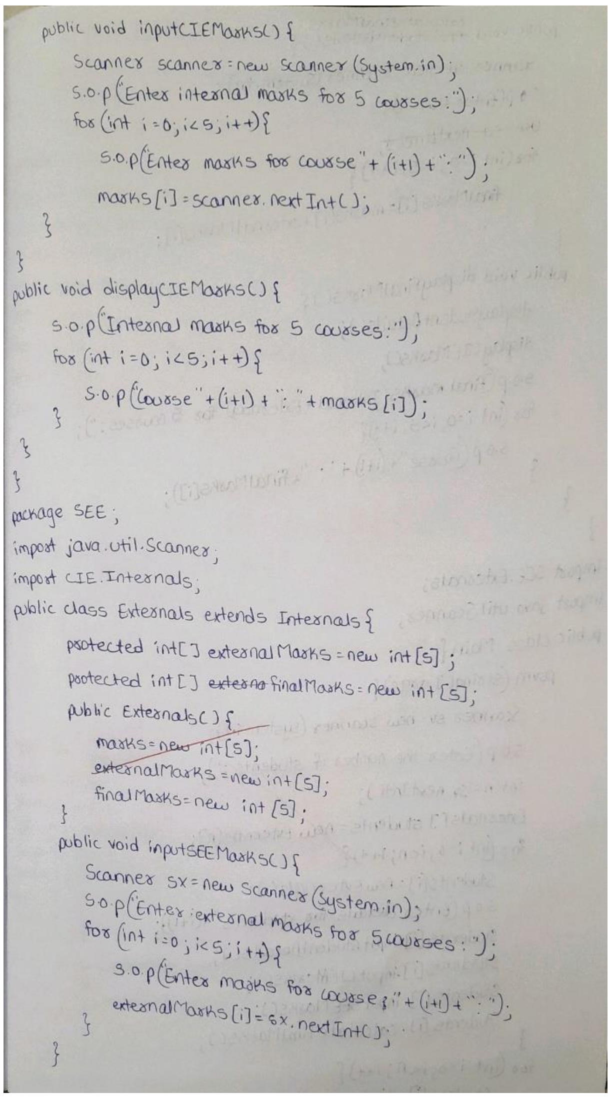
Create a package CIE which has two classes - Personal and Internals. The class Personal has members like usn, name, sem. The class Internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External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



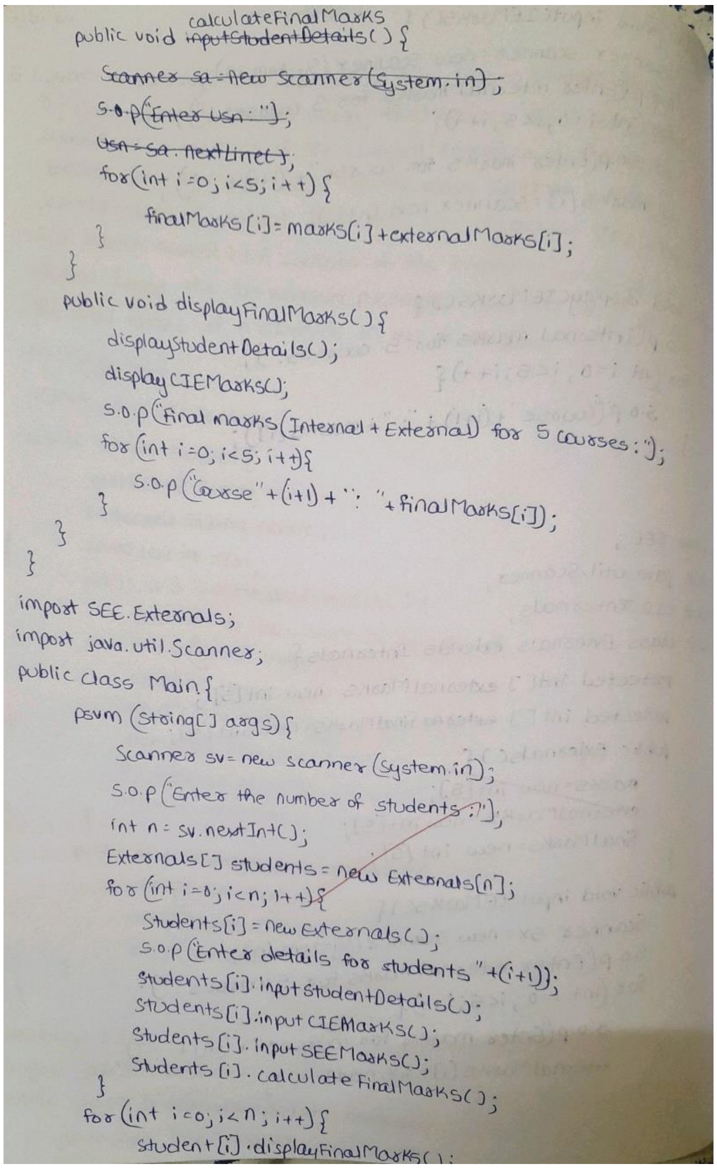
14/11/24 LAB-6

Q. Create a package CIE which has two classes Student and Internals. The class internals has an array that stores the internal marks scored in 5 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 5 courses of the current semester of the student. Import the two ~~semeste~~ packages in a file that declares the final marks of n students in all 5 courses.

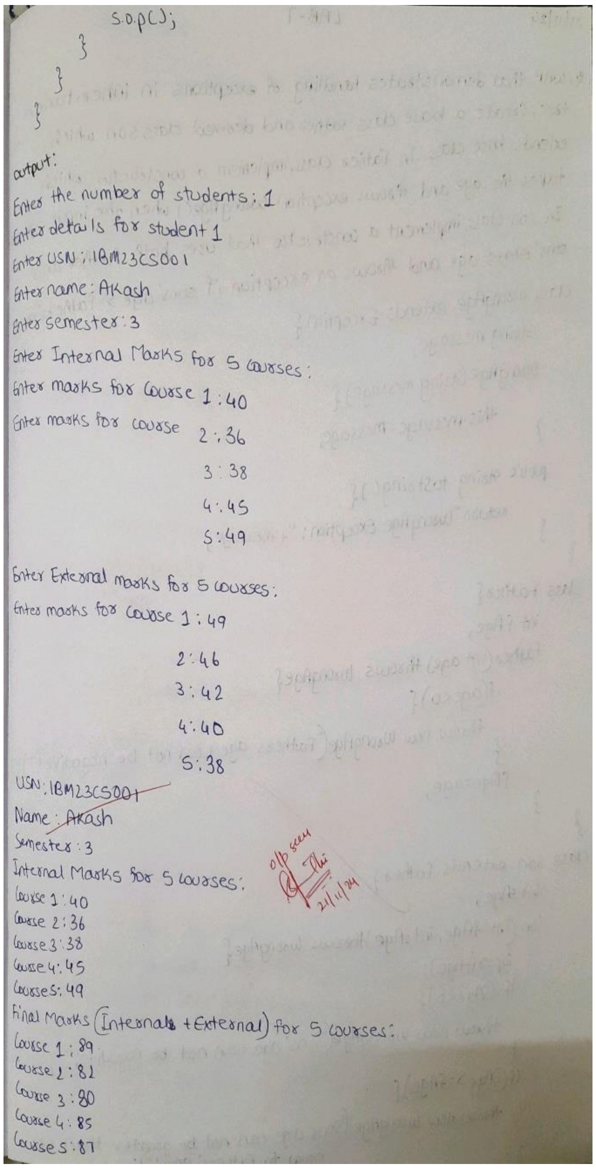
```
package CIE;
import java.util.Scanner;
public class Student{
    protected String usn;
    protected String name;
    protected int sem;
    public void inputStudentDetails() {
        Scanner sc=new Scanner(System.in);
        s.o.p("Enter usn:");
        usn=sc.nextLine();
        s.o.p("Enter name:");
        name=sc.nextLine();
        s.o.p("Enter sem:");
        sem=sc.nextInt();
    }
    public void displayStudentDetails() {
        s.o.p("USN:"+usn);
        s.o.p("Name:"+name);
        s.o.p("Semester:"+sem);
    }
}
package CIE;
import java.util.Scanner;
public class Internals extends Student{
    protected int[] marks=new int[5];
```



```
public void inputCIEMarks() {
    Scanner scanner=new Scanner(System.in);
    s.o.p("Enter internal marks for 5 courses:");
    for (int i=0;i<5;i++){
        s.o.p("Enter marks for course"+(i+1)+":");
        marks[i]=scanner.nextInt();
    }
}
public void displayCIEMarks() {
    s.o.p("Internal marks for 5 courses:");
    for (int i=0;i<5;i++){
        s.o.p("Course"+(i+1)+": "+marks[i]);
    }
}
}
package SEE;
import java.util.Scanner;
import CIE.Internals;
public class Externals extends Internals{
    protected int[] externalMarks=new int[5];
    protected int[] ~~externa~~ finalMarks=new int[5];
    public Externals() {
        marks=new int[5];
        externalMarks=new int[5];
        finalMarks=new int[5];
    }
    public void inputSEEMarks() {
        Scanner sx=new Scanner(System.in);
        s.o.p("Enter external marks for 5 courses:");
        for (int i=0;i<5;i++){
            s.o.p("Enter marks for course:"+(i+1)+":");
            externalMarks[i]=sx.nextInt();
        }
    }
```



```
                    calculateFinalMarks
public void ~~inputStudentDetails~~(){
    ~~Scanner sa=new Scanner(System.in)~~;
    s.o.p(~~"Enter usn:"~~);
    ~~usn=sa.nextLine()~~;
    for(int i=0;i<5;i++){
        finalMarks[i]=marks[i]+externalMarks[i];
    }
}
public void displayFinalMarks(){
    displayStudentDetails();
    displayCIEMarks();
    s.o.p("Final marks(Internal+External) for 5 courses:");
    for(int i=0;i<5;i++){
        s.o.p("Course"+(i+1)+": "+finalMarks[i]);
    }
}
}
import SEE.Externals;
import java.util.Scanner;
public class Main{
    psvm(String[] args){
        Scanner sv=new scanner(system.in);
        s.o.p("Enter the number of students:");
        int n=sv.nextInt();
        Externals[] students=new Externals[n];
        for(int i=0;i<n;i++){
            Students[i]=new Externals();
            s.o.p("Enter details for students"+(i+1));
            Students[i].inputStudentDetails();
            Students[i].input CIEMarks();
            Students[i].input SEEMarks();
            Students[i].calculate FinalMarks();
        }
    for(int i=0;i<n;i++){
        Student[i].displayFinalMarks();
```



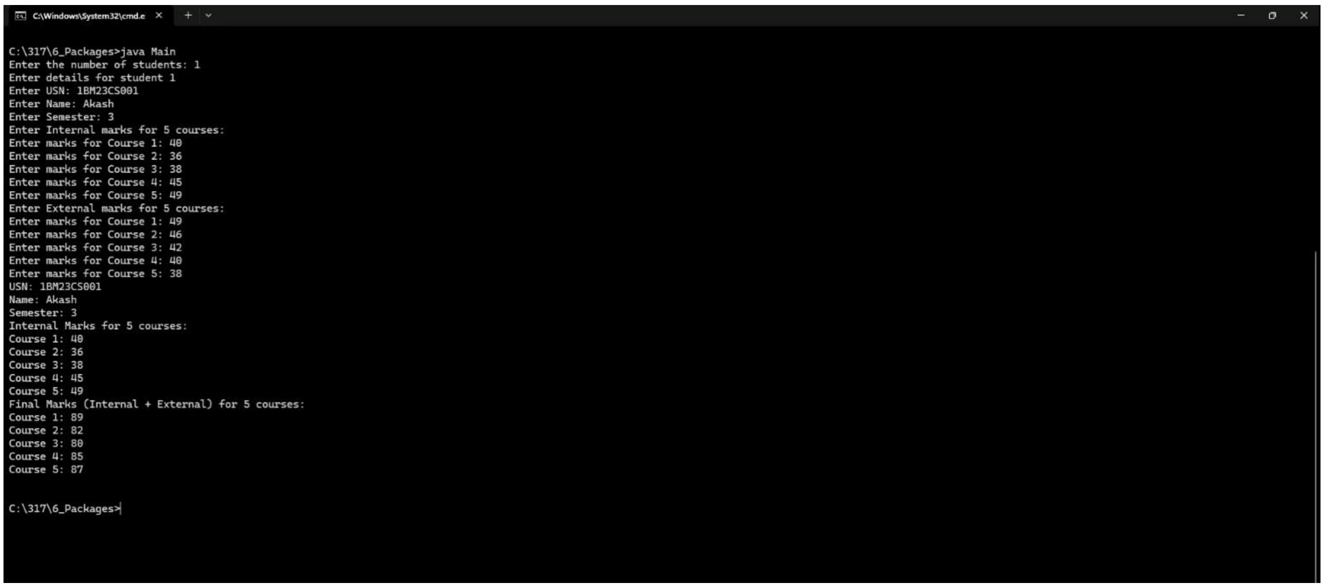
```
            s.o.p();
        }
    }
}
```

output:

```
Enter the number of students: 1
Enter details for student 1
Enter USN: 1BM23CS001
Enter name: Akash
Enter semester: 3
Enter Internal Marks for 5 courses:
Enter marks for Course 1:40
Enter marks for course 2:36
                       3:38
                       4:45
                       5:49

Enter External marks for 5 courses:
Enter marks for Course 1:49
                      2:46
                      3:42
                      4:40
                      5:38
USN: 1BM23CS001
Name: Akash
Semester: 3
Internal Marks for 5 courses:
Course 1:40
Course 2:36
Course3:38
Course4:45
Course5:49
Final Marks (Internals + External) for 5 courses:
Course 1:89
Course 2:82
Course 3:80
Course 4:85
Course 5:87
```

o/p seen
[illegible]
21/11/24

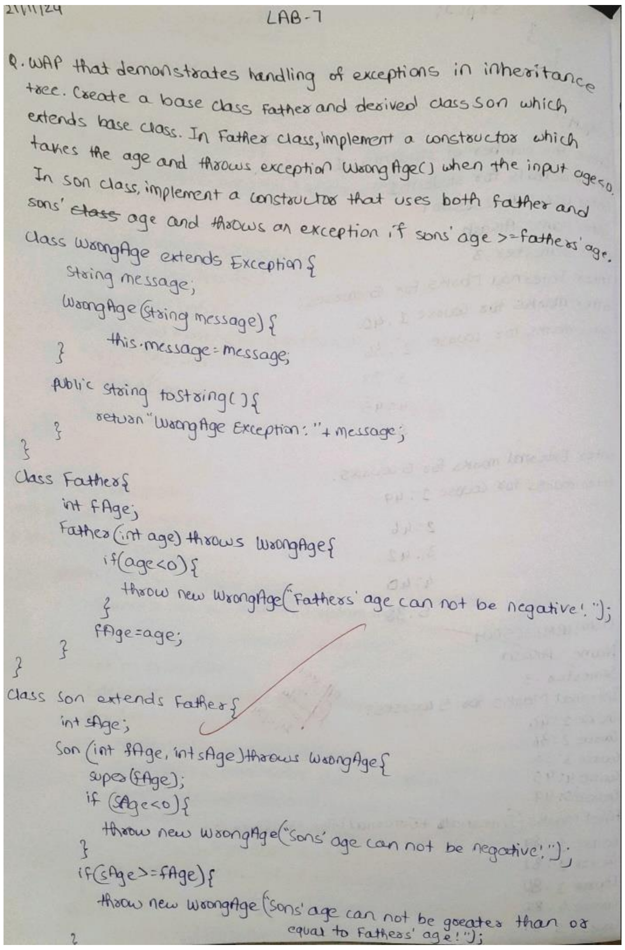
```
package CIE;

import java.util.Scanner;

public class Internals extends Student {
    protected int[] marks = new int[5];  // Marks for 5 courses

    // Method to input internal marks
    public void inputCIEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter Internal marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for Course " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }
    // Method to display internal marks
    public void displayCIEmarks() {
        System.out.println("Internal Marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + marks[i]);
        }
    }
}

package CIE;

import java.util.Scanner;

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



    public void inputStudentDetails() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = scanner.nextLine();
        System.out.print("Enter Name: ");
        name = scanner.nextLine();
        System.out.print("Enter Semester: ");
        sem = scanner.nextInt();
    }

    // Method to display student details
    public void displayStudentDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}

package SEE;

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

public class Externals extends Internals {
    protected int[] externalMarks = new int[5];  // External marks for 5 courses
    protected int[] finalMarks = new int[5];  // Final marks (internal + external)

    // Constructor
    public Externals() {
        marks = new int[5];  // Initialize internal marks
        externalMarks = new int[5];  // Initialize external marks
        finalMarks = new int[5];  // Initialize final marks
    }

    // Method to input external marks
    public void inputSEEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter External marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for Course " + (i + 1) + ": ");
            externalMarks[i] = scanner.nextInt();
        }
    }

    // Method to calculate final marks
    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = marks[i] + externalMarks[i];  // Simple addition for final marks
        }
    }

    // Method to display final marks
    public void displayFinalMarks() {
        displayStudentDetails();
        displayCIEmarks();
        System.out.println("Final Marks (Internal + External) for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}

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

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

        Externals[] students = new Externals[n];


        for (int i = 0; i < n; i++) {
            students[i] = new Externals();

            System.out.println("Enter details for student " + (i + 1));
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
            students[i].calculateFinalMarks();
        }
        for(int i=0; i<n; i++){
            students[i].displayFinalMarks();
            System.out.println();
        }
    }
}
```



```
C:\317\6_Packages>java Main
Enter the number of students: 1
Enter details for student 1
Enter USN: 1BM23CS001
Enter Name: Akash
Enter Semester: 3
Enter Internal marks for 5 courses:
Enter marks for Course 1: 40
Enter marks for Course 2: 36
Enter marks for Course 3: 38
Enter marks for Course 4: 45
Enter marks for Course 5: 49
Enter External marks for 5 courses:
Enter marks for Course 1: 49
Enter marks for Course 2: 46
Enter marks for Course 3: 42
Enter marks for Course 4: 40
Enter marks for Course 5: 38
USN: 1BM23CS001
Name: Akash
Semester: 3
Internal Marks for 5 courses:
Course 1: 40
Course 2: 36
Course 3: 38
Course 4: 45
Course 5: 49
Final Marks (Internal + External) for 5 courses:
Course 1: 89
Course 2: 82
Course 3: 80
Course 4: 85
Course 5: 87


C:\317\6_Packages>
```

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=father's age.

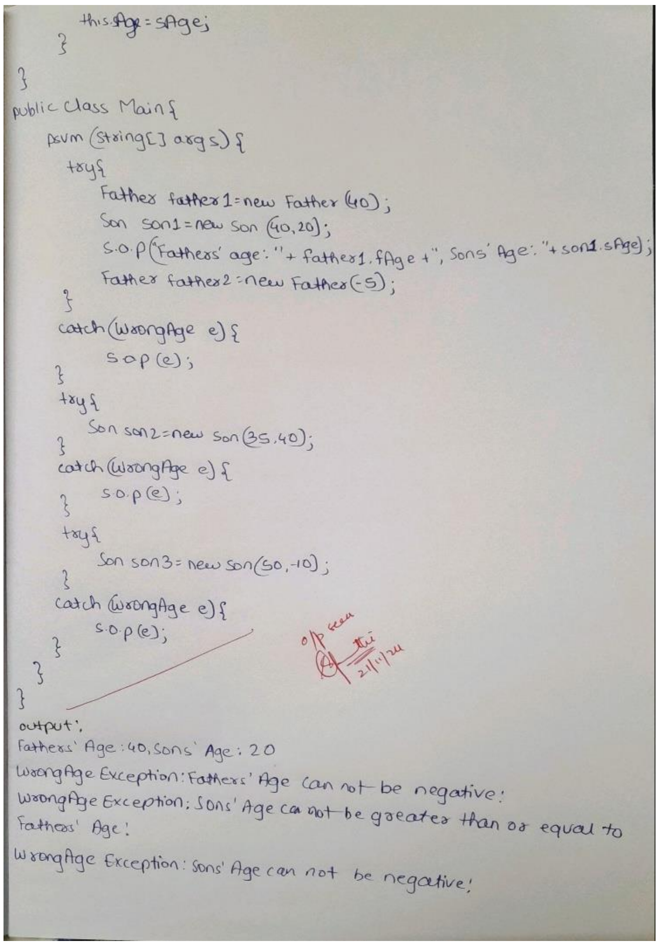


21/11/24

LAB-7

Q. WAP that demonstrates handling of exceptions in inheritance tree. Create a base class Father and derived class Son which extends base class. In Father class, implement a constructor which takes the age and throws exception WrongAge() when the input age<0. In son class, implement a constructor that uses both father and sons' ~~class~~ age and throws an exception if sons' age >=fathers' age.

```
Class WrongAge extends Exception{
    String message;
    WrongAge(String message){
        this.message=message;
    }
    public String toString(){
        return "WrongAge Exception: "+message;
    }
}
Class Father{
    int fAge;
    Father(int age) throws WrongAge{
        if(age<0){
            throw new WrongAge("Fathers' age can not be negative!");
        }
        fAge=age;
    }
}
Class Son extends Father{
    int sAge;
    Son (int fAge, int sAge)throws WrongAge{
        super(fAge);
        if (sAge<0){
            throw new WrongAge("Sons' age can not be negative!");
        }
        if(sAge>=fAge){
            throw new WrongAge("Sons' age can not be greater than or
                                    equal to Fathers' age!");
        }
```



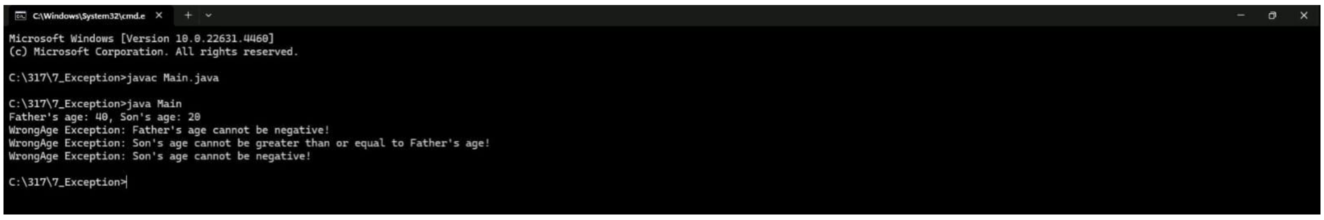
```
        this.sAge=sAge;
    }
}
public Class Main{
    psvm (String[] args){
        try{
            Father father1=new Father(40);
            Son son1=new Son(40,20);
            s.o.p("Fathers' age: "+ father1.fAge +", Sons' Age: "+son1.sAge);
            Father father2=new Father(-5);
        }
        catch(WrongAge e){
            s.o.p(e);
        }
        try{
            Son son2=new Son(35,40);
        }
        catch(WrongAge e){
            s.o.p(e);
        }
        try{
            Son son3= new Son(50,-10);
        }
        catch(WrongAge e){
            s.o.p(e);
        }
    }
}
```

o/p seen [illegible] 21/11/24

output:

```
Fathers' Age:40, Sons' Age: 20
WrongAge Exception: Fathers' Age can not be negative!
WrongAge Exception: Sons' Age can not be greater than or equal to
Fathers' Age!
WrongAge Exception: Sons' Age can not be negative!
```

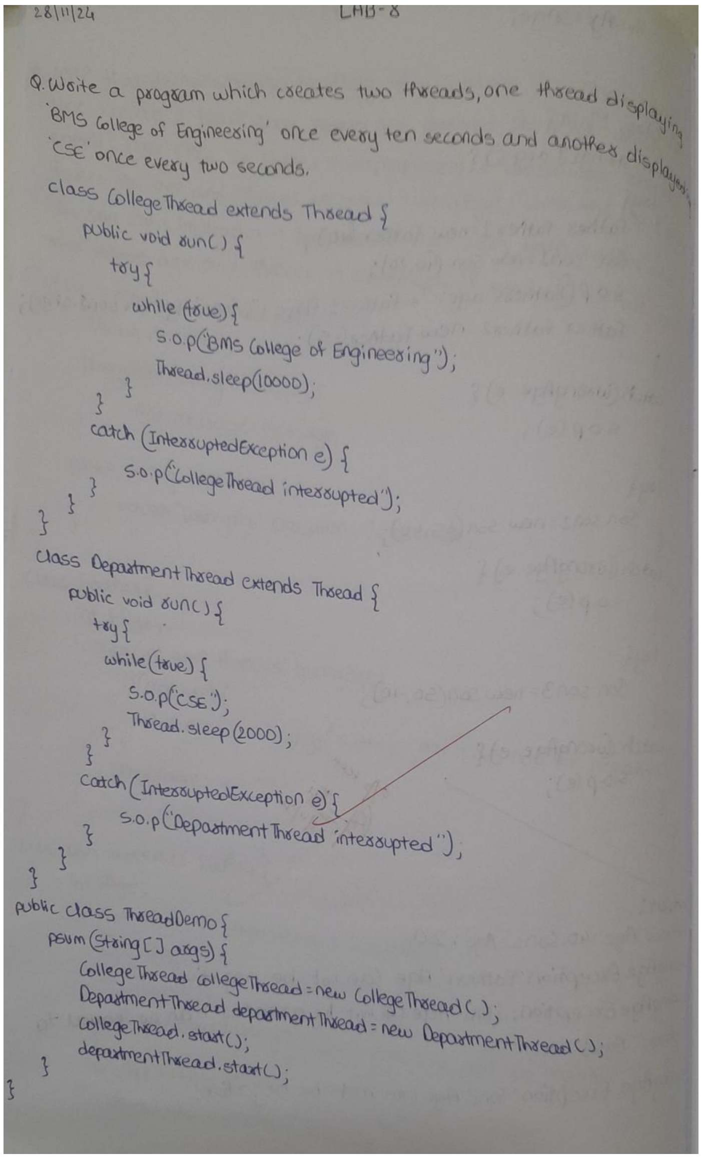
```
Class WrongAge extends Exception {
String message;
WrongAge(String message) {
this.message = message;
}
public String toString() {
return "Wrong Age Exception: " + message;
}
}
Class Father {
int fAge;
Father(int age) throws WrongAge {
if (age < 0) {
throw new WrongAge("Father's age cannot be negative!");
}
fAge = age;
}
}

Class Son extends Father {
int sAge;
Son(int fAge, int sAge) throws WrongAge {
super(fAge);
if(sAge>0){
throw new WrongAge("Sons' age can not be negative!");
}
if (sAge > =fAge) {
throw new WrongAge("Son's age cannot be greater than or equal to father's age!");
}
this.sAge = sAge;
}
}
public class Main {
public static void main(String[] args) {
try {
Father father1 = new Father(40);
Son son1 = new Son(40, 20);
System.out.println("Fathers' age:"+father1.fAge+",Sons' Age:"+son1.sAge);
Father father2 = new Father(-5);
}
catch (WrongAge e) {
System.out.println(e);
}
try{
Son son2 = new son(35, 40);
}
catch(WrongAge e) {
System.out.prinltn(e);
}
try{
Son son3 = new son(50, -10);
}
catch(WrongAge e){
System.out.prinltn(e);
}
}
}
```



```
Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

C:\317\7_Exception>javac Main.java

C:\317\7_Exception>java Main
Father's age: 40, Son's age: 20
WrongAge Exception: Father's age cannot be negative!
WrongAge Exception: Son's age cannot be greater than or equal to Father's age!
WrongAge Exception: Son's age cannot be negative!

C:\317\7_Exception>
```

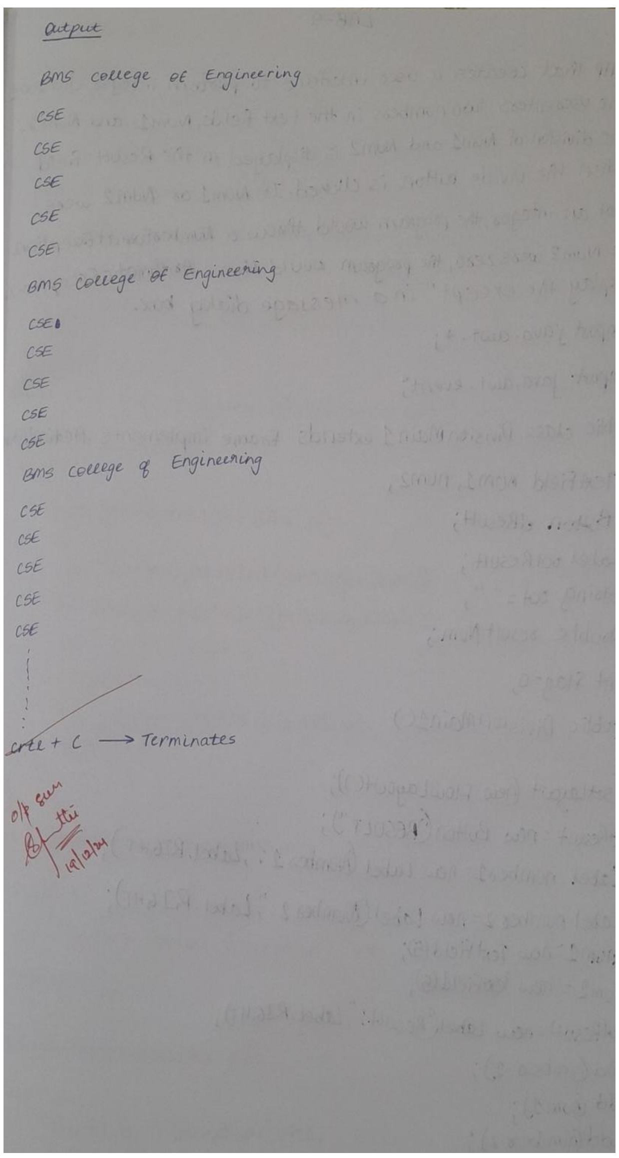
Program 8

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.


28/11/24 LAB-8

Q. 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.

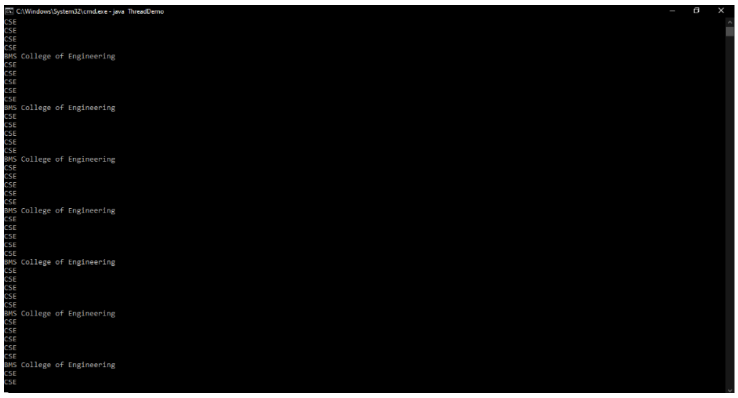
```
class CollegeThread extends Thread {
    public void run() {
        try {
            while (true) {
                s.o.p("BMS College of Engineering");
                Thread.sleep(10000);
            }
        }
        catch (InterruptedException e) {
            s.o.p("CollegeThread interrupted");
        }
    }
}
class DepartmentThread extends Thread {
    public void run() {
        try {
            while (true) {
                s.o.p("CSE");
                Thread.sleep(2000);
            }
        }
        catch (InterruptedException e) {
            s.o.p("DepartmentThread interrupted");
        }
    }
}
public class ThreadDemo {
    psvm (String[] args) {
        CollegeThread collegeThread = new CollegeThread();
        DepartmentThread departmentThread = new DepartmentThread();
        collegeThread.start();
        departmentThread.start();
    }
}
```


Output

```
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
:
:
ctrl + C ⟶ Terminates
```

o/p seen
[illegible]
19/12/24

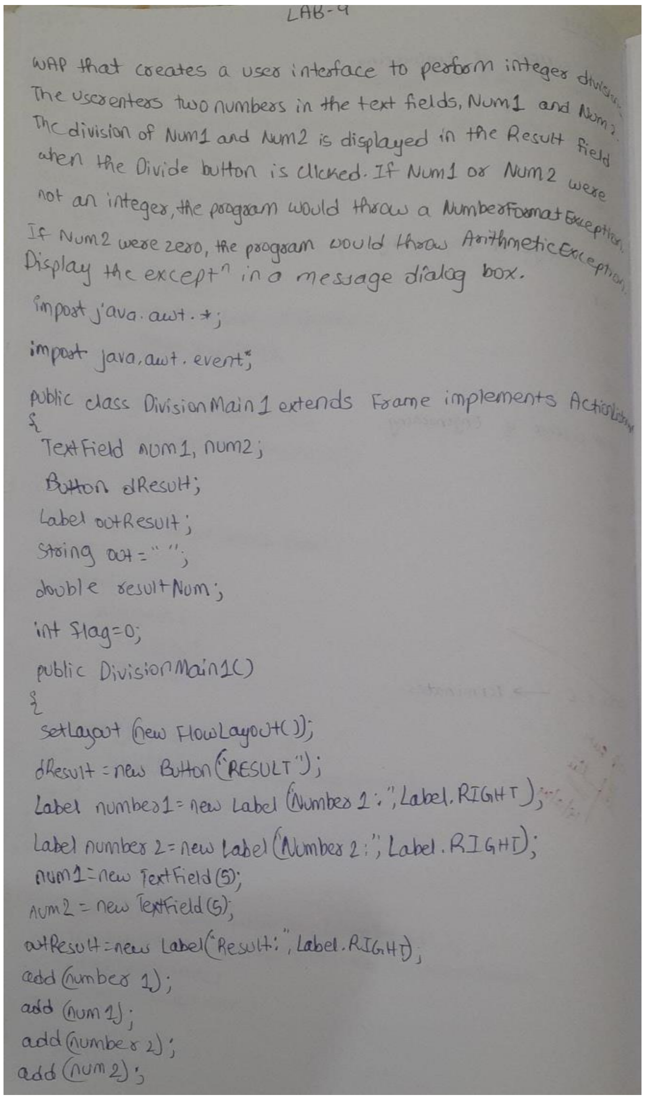
```
class CollegeThread extends Thread {
    boolean execution=true;
    public void end() {
    execution=false;
    }
    public void run(){
        try {
            while (execution) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000); // 10 seconds
            }
        } catch (InterruptedException e) {
            System.out.println("CollegeThread interrupted");
        }
    }
}

class DepartmentThread extends Thread {
    boolean execution=true;
    public void terminate(){
        execution=false;
    }
    public void run() {
        try {
            while (execution) {
                System.out.println("CSE");
                Thread.sleep(2000); // 2 seconds
            }
        } catch (InterruptedException e) {
            System.out.println("DepartmentThread interrupted");
        }
    }
}


public class Mainn {
    public static void main(String[] args) {
        // Create and start threads
        CollegeThread collegeThread = new CollegeThread();
        DepartmentThread departmentThread = new DepartmentThread();

        collegeThread.start();
        departmentThread.start();
    try{
        Thread.sleep(30000);
    }
    catch(InterruptedException e){
        System.out.println("Program terminated");
    }
}
}
```



```
C:\Windows\System32\cmd.exe - java  ThreadDemo
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
```

Program 9

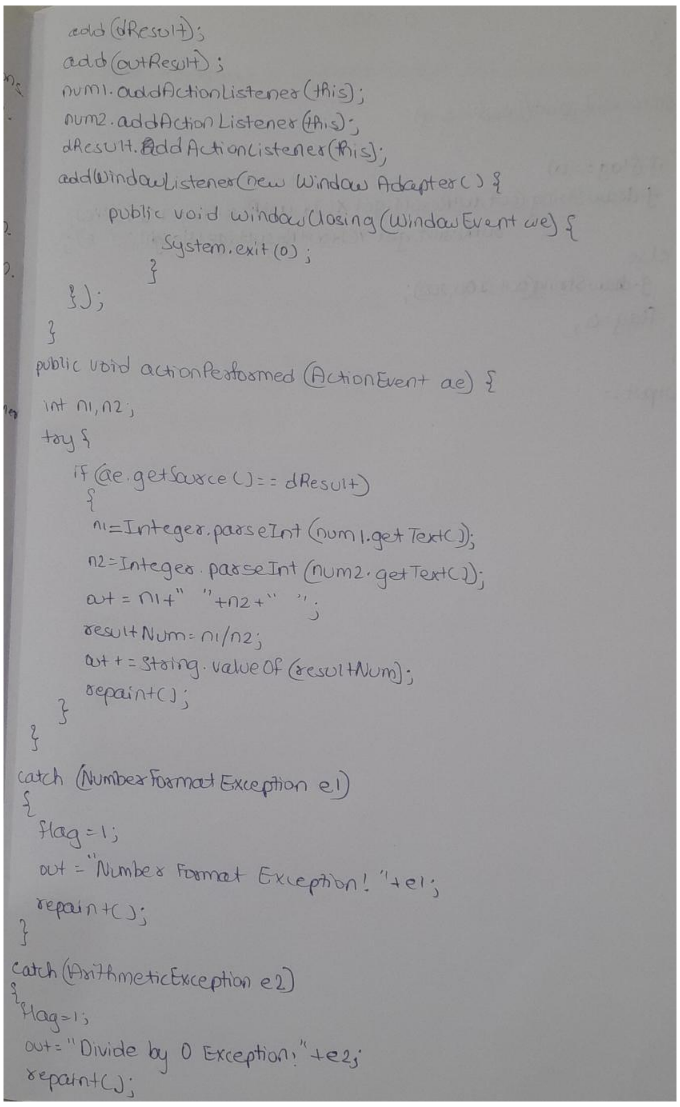
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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



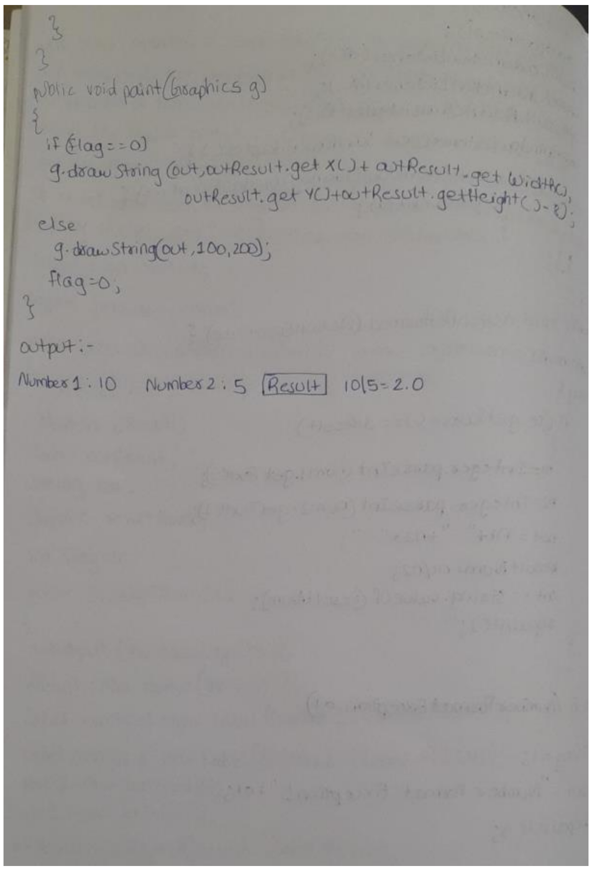
LAB-4

WAP that creates a user interface to perform integer division.
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 ArithmeticException.
Display the except^n in a message dialog box.

```
import java.awt.*;
import java.awt.event*;
public class DivisionMain1 extends Frame implements ActionListener
{
 TextField num1, num2;
 Button dResult;
 Label outResult;
 String out = " ";
 double resultNum;
 int flag=0;
 public DivisionMain1()
{
 setLayout (new FlowLayout());
 dResult = new Button("RESULT");
 Label number1 = new Label ("Number 1:", Label.RIGHT);
 Label number 2 = new Label ("Number 2:", Label.RIGHT);
 num1 = new TextField(5);
 num2 = new TextField(5);
outResult = new Label("Result:", Label.RIGHT);
add (number 1);
add (num 1);
add (number 2);
add (num 2);
```



```
  add (dResult);
  add (outResult);
  num1.addActionListener (this);
  num2.addActionListener (this);
  dResult.AddActionListener(this);
  addWindowListener(new Window Adapter() {
       public void windowClosing(WindowEvent we) {
            System.exit (0);
          }
  });
 }
public void actionPerformed (ActionEvent ae) {
 int n1,n2;
 try {
    if (ae.getSource() == dResult)
     {
      n1=Integer.parseInt (num1.getText());
      n2=Integer.parseInt (num2.getText());
      out = n1+" "+n2+" ";
      resultNum = n1/n2;
      out += String.valueOf (resultNum);
      repaint();
    }
 }
catch (NumberFormatException e1)
{
  flag=1;
  out = "Number Format Exception! "+e1;
  repaint();
}
catch (ArithmeticException e2)
{
 flag=1;
 out="Divide by 0 Exception! "+e2;
 repaint();
```



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

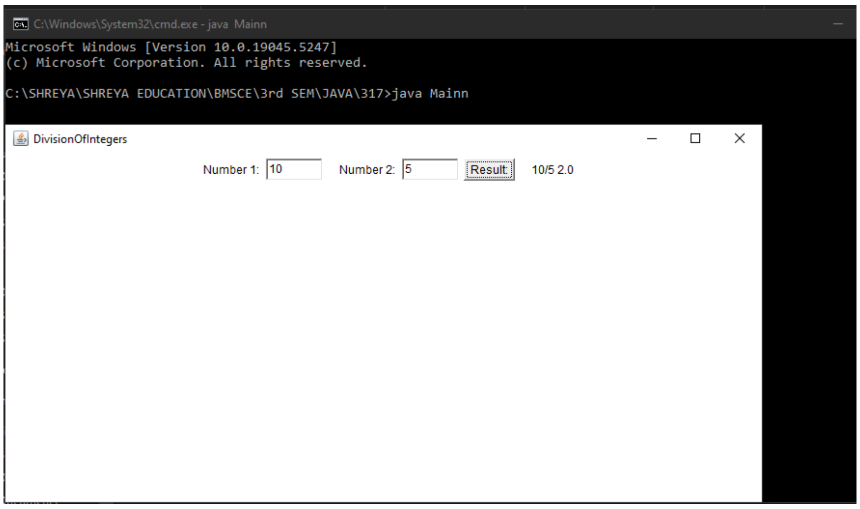
output:-

```
Number 1 : 10   Number 2 : 5  [Result]  10/5=2.0
```

```
import java.awt.*;
import java.awt.event.*;
class DivisionMain1 extends Frame implements ActionListener
{
	TextField num1,num2;
	Button dResult;
	Label outResult;
	String out="";
	double resultNum;
	int flag=0;
	public DivisionMain1()
	{
		setLayout(new FlowLayout());
		dResult = new Button("Result:");
		Label number1 = new Label("Number 1:",Label.RIGHT);
		Label number2 = new Label("Number 2:",Label.RIGHT);
		num1=new TextField(5);
		num2=new TextField(5);
		outResult = new Label("",Label.RIGHT);
		add(number1);
		add(num1);
		add(number2);
		add(num2);
		add(dResult);
		add(outResult);
		num1.addActionListener(this);
```

```
                num2.addActionListener(this);
                dResult.addActionListener(this);
                addWindowListener(new WindowAdapter(){
                        public void windowClosing(WindowEvent e)
                        {
                                System.exit(0);
                        }
                });
        }
        public void actionPerformed(ActionEvent e)
        {
                int n1,n2;
                try
                {
                        if (e.getSource() == dResult)
                        {
                                n1=Integer.parseInt(num1.getText());
                                n2=Integer.parseInt(num2.getText());
                                if(n2==0)
                                {throw new ArithmeticException();}
                                out=n1+"/"+n2+" ";
                                resultNum=n1/n2;
                                out+=resultNum;
                        }
                }
                catch(NumberFormatException e1)
                {
                        flag=1;
                        out="Number Format Exception!"+e1;
                }
                catch(ArithmeticException e1)
                {
                        flag=1;
                        out="Divide by 0 Exception!"+e1;
                }
                outResult.setText(out);
                invalidate();
                validate();
        }
        //public void paint(Graphics g)
        //{
        //      if(flag==0)
        //
        {g.drawString(out,dResult.getX()+dResult.getWidth(),dResult.getY()+outResult.getHeight()-
8);;}
        //      else
        //      {g.drawString(out,100,200); flag=0;}
        //}

}
```

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

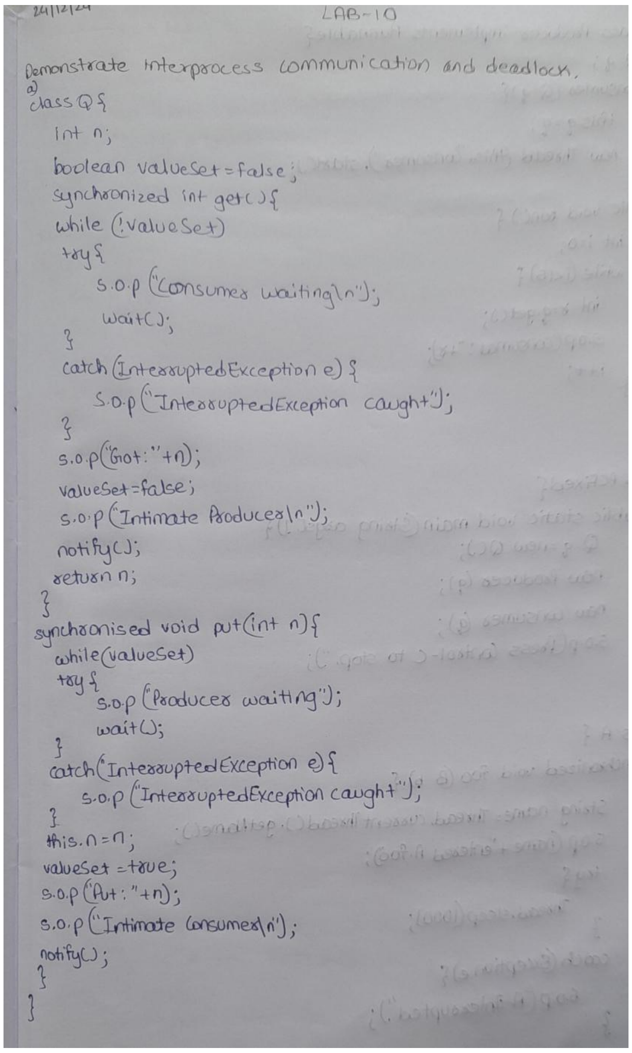


```
C:\Windows\System32\cmd.exe - java  Mainn
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\SHREYA\SHREYA EDUCATION\BMSCE\3rd SEM\JAVA\317>java Mainn
```

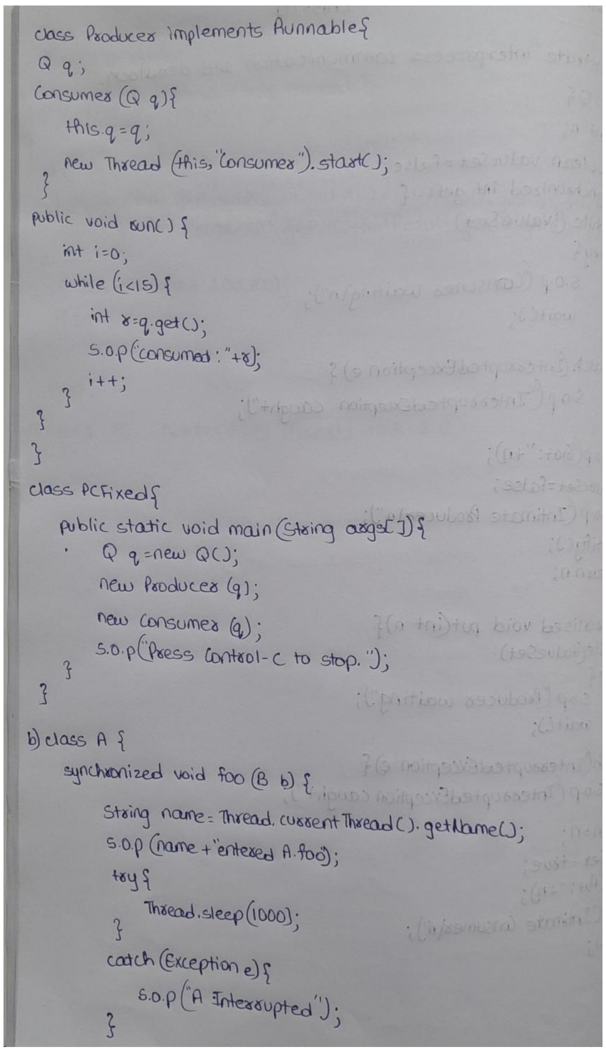
```
DivisionOfIntegers
Number 1: 10    Number 2: 5    Result:    10/5 2.0
```

Program 10

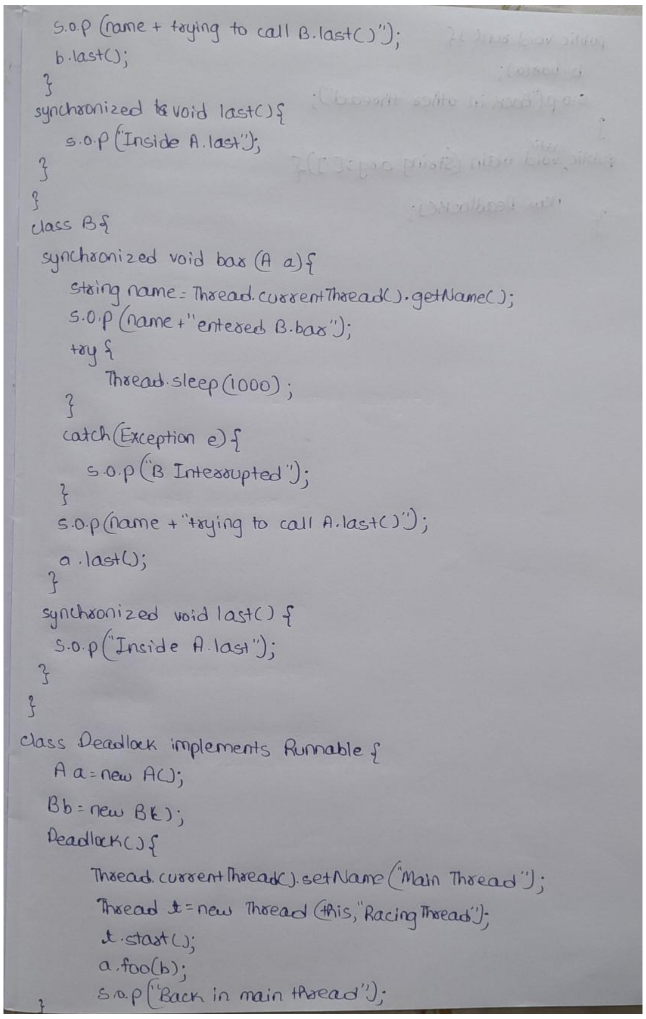
Demonstrate Inter process communication and deadlock.



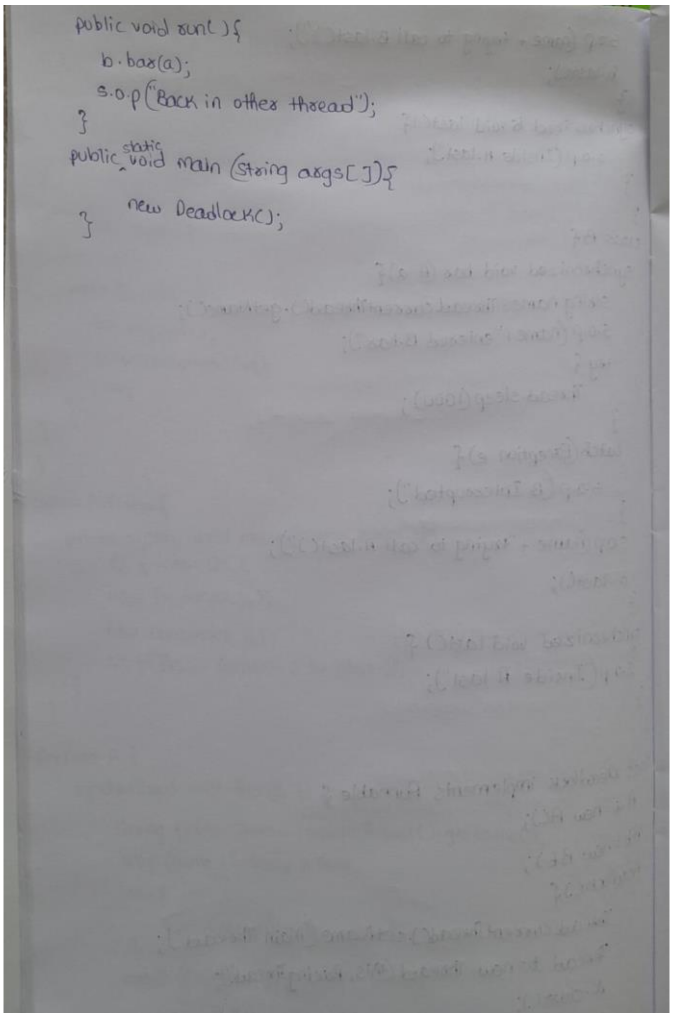
```
24/12/24                        LAB-10

Demonstrate interprocess communication and deadlock.
a)
class Q{
  int n;
  boolean valueSet=false;
  synchronized int get(){
  while (!valueSet)
   try{
      s.o.p ("Consumer waiting\n");
       wait();
   }
   catch(InterruptedException e){
      S.o.p("InterruptedException caught");
   }
   s.o.p("Got:"+n);
   valueSet=false;
   s.o.p("Intimate Producer\n");
   notify();
  return n;
 }
synchronised void put(int n){
  while(valueSet)
  try{
     s.o.p ("Producer waiting");
     wait();
  }
 catch("InterruptedException e){
    s.o.p ("InterruptedException caught");
 }
 this.n=n;
 valueSet =true;
 s.o.p ("Put:"+n);
 s.o.p ("Intimate Consumer\n");
 notify();
 }
}
```



```
class Producer 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();
      s.o.p("consumed:"+r);
      i++;
   }
 }
}
class PCFixed{
   public static void main(String args[]){
   .   Q q=new Q();
       new Producer (q);
       new Consumer (q);
       s.o.p("Press Control-C to stop.");
   }
 }

b) class A {
    synchronized void foo (B b) {
        String name= Thread.currentThread().getName();
        s.o.p (name+"entered A.foo");
        try{
            Thread.sleep(1000);
        }
        catch (Exception e){
            s.o.p("A Interrupted");
        }
```



```
  s.o.p (name + trying to call B.last()");
  b.last();
 }
synchronized ~~ts~~ void last(){
   s.o.p ("Inside A.last");
 }
}
class B{
 synchronized void bar (A a){
   String name = Thread.currentThread().getName();
   s.o.p (name+"entered B.bar");
   try {
      Thread.sleep(1000);
   }
   catch(Exception e){
     s.o.p("B Interrupted");
   }
   s.o.p(name + "trying to call A.last()");
   a.last();
 }
 synchronized void last() {
  s.o.p("Inside 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);
     s.o.p("Back in main thread");
 }
```



```
public void run(){
   b.bar(a);
   s.o.p("Back in other thread");
}
public static void main (String args[]){
    new Deadlock();
}
```

i. Interprocess Communication

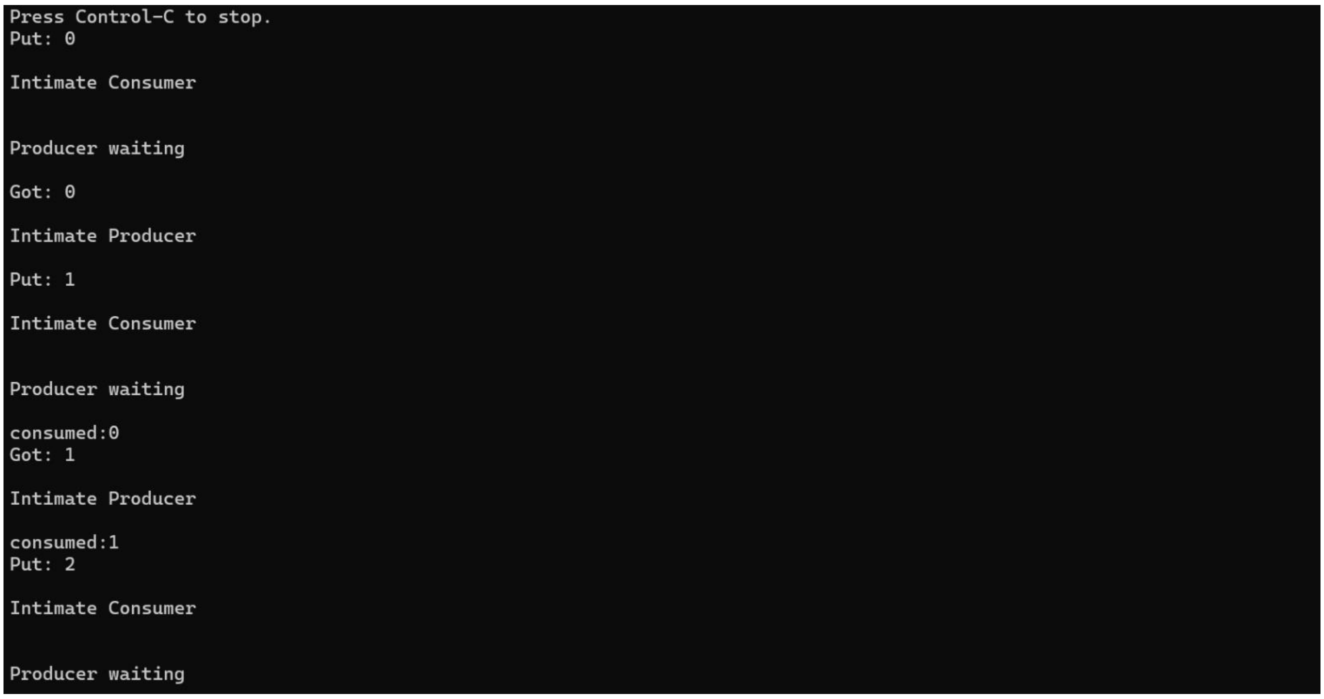
```
class Q {
int n;
boolean valueSet = false;

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

synchronized void put(int n) {
```

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

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

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

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

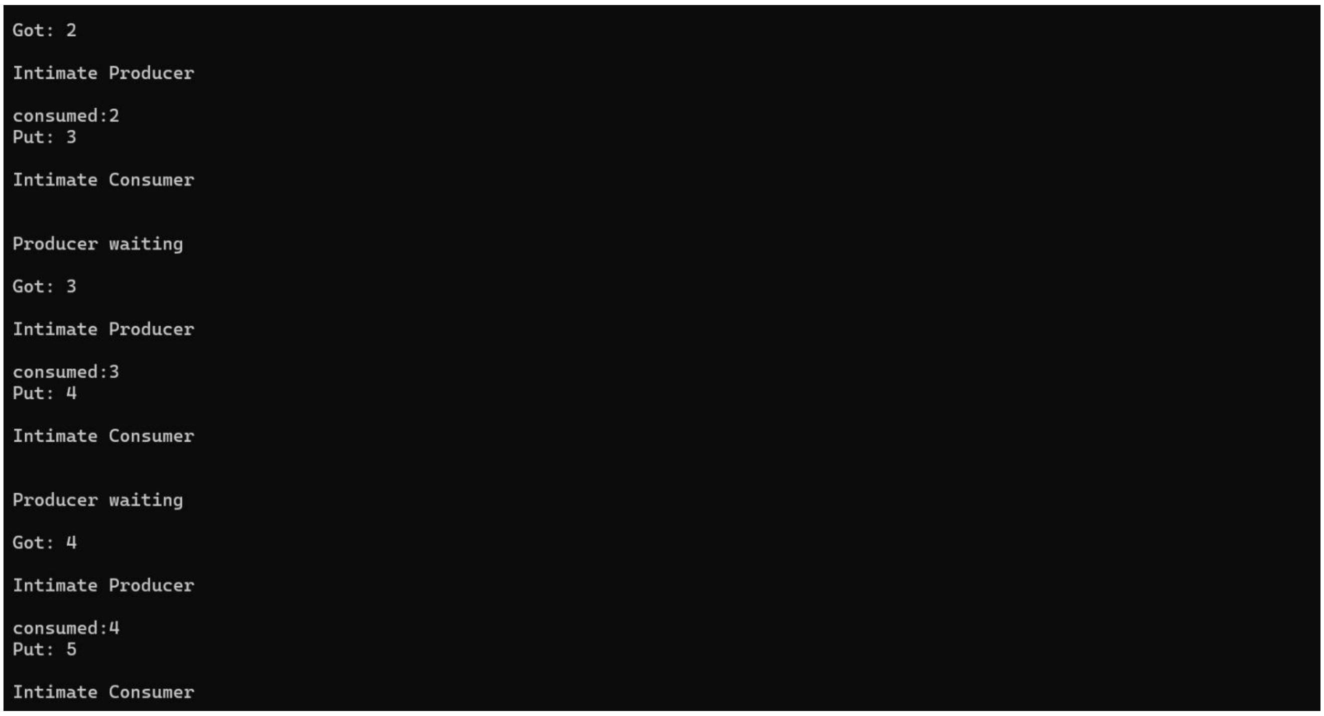
```
}
}
```

ii. Demonstration of deadlock

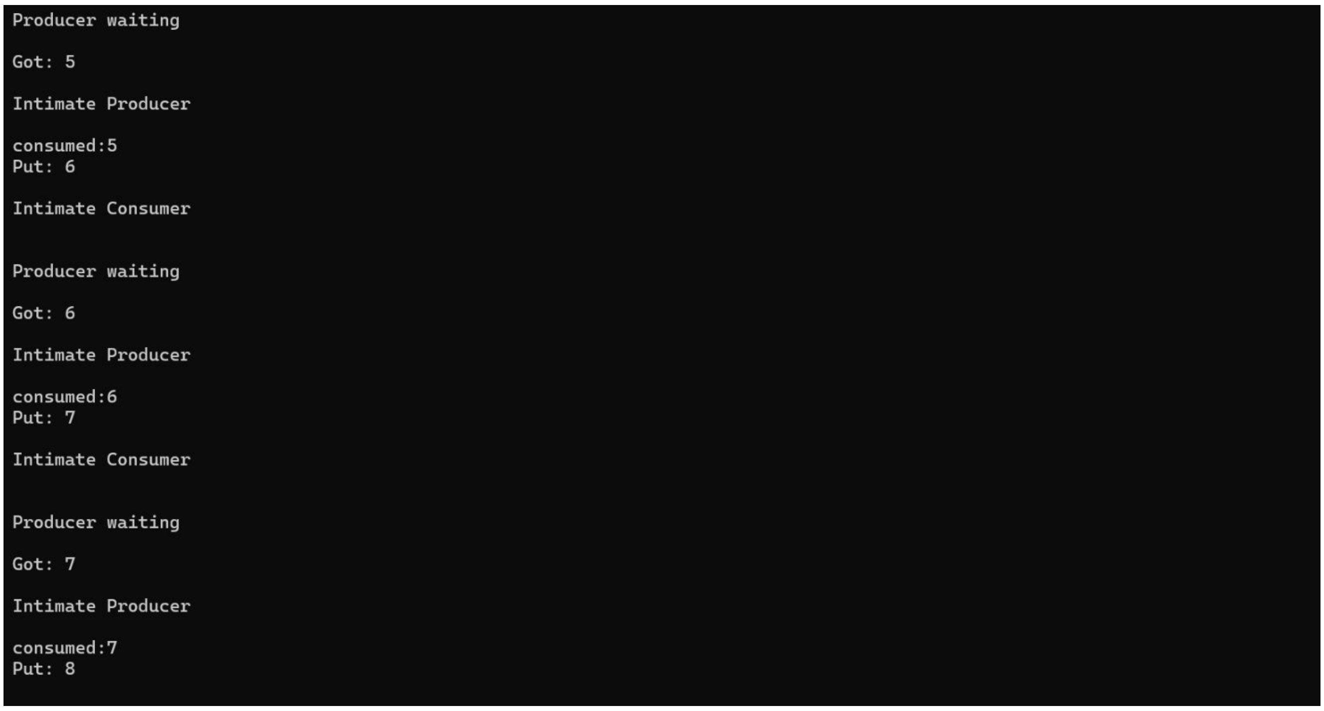
```
class A
{
  synchronized void foo(B b)
   { String name = Thread.currentThread().getName();
      System.out.println(name + " entered A.foo");
      try { Thread.sleep(1000); }
      catch(Exception e) { System.out.println("A Interrupted"); }
      System.out.println(name + " trying to call B.last()");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

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

}


class Deadlock implements Runnable
 {
  A a = new A(); B b = new B();
  Deadlock( ) {
    Thread.currentThread().setName("MainThread");
    Thread t = new Thread(this, "RacingThread");
     t.start(); a.foo(b); // get lock on a in this thread.
     System.out.println("Back in main thread");
   }
 public void run() { b.bar(a); // get lock on b in other thread.
  System.out.println("Back in other thread");
  }
public static void main(String args[]) { new Deadlock(); }
}
```



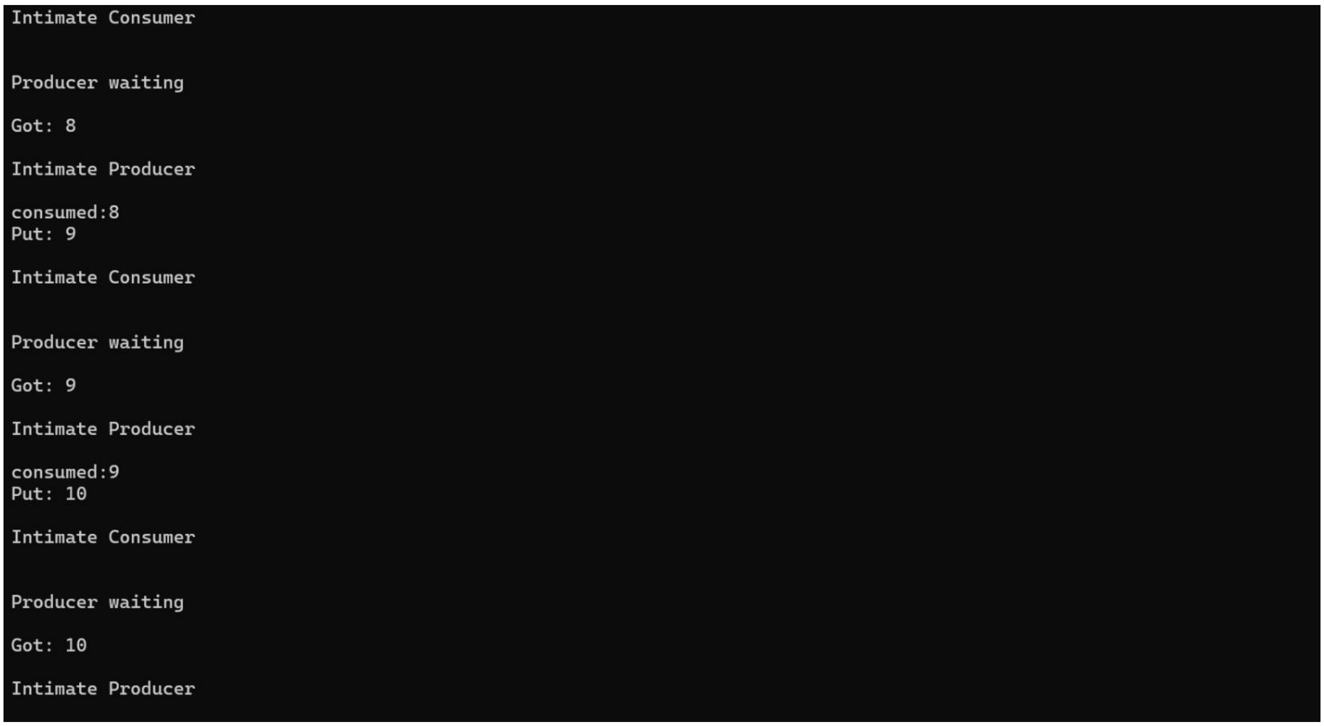
```
Press Control-C to stop.
Put: 0

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
```



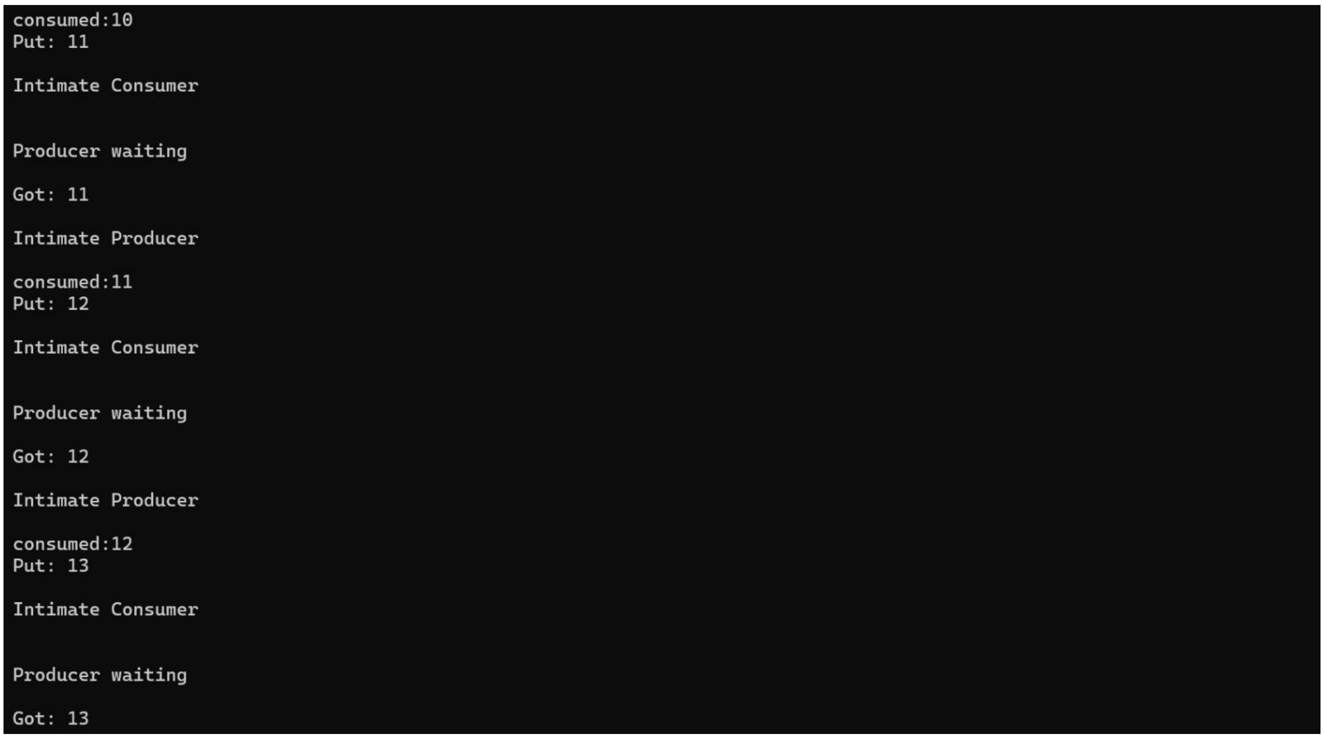
```
Got: 2

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
```



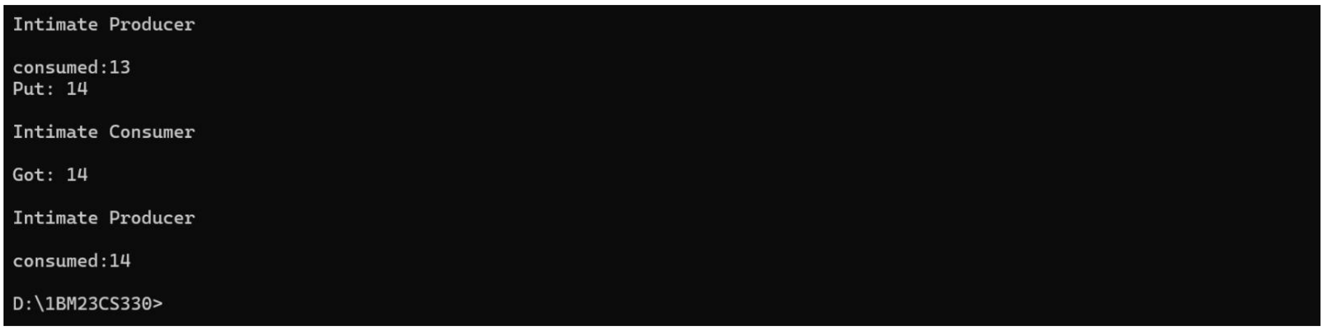
```
Producer waiting

Got: 5

Intimate Producer

consumed:5
Put: 6

Intimate Consumer


Producer waiting

Got: 6

Intimate Producer

consumed:6
Put: 7

Intimate Consumer


Producer waiting

Got: 7

Intimate Producer

consumed:7
Put: 8
```



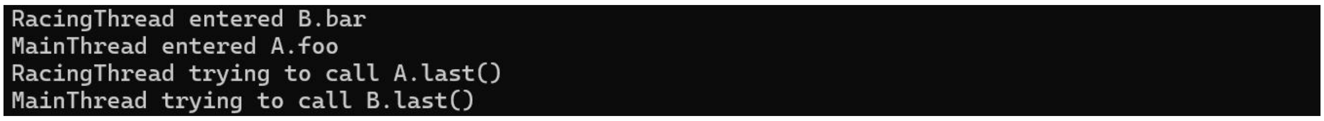
```
Intimate Consumer


Producer waiting

Got: 8

Intimate Producer

consumed:8
Put: 9

Intimate Consumer


Producer waiting

Got: 9

Intimate Producer

consumed:9
Put: 10

Intimate Consumer


Producer waiting

Got: 10

Intimate Producer
```



```
consumed:10
Put: 11

Intimate Consumer


Producer waiting

Got: 11

Intimate Producer

consumed:11
Put: 12

Intimate Consumer


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

D:\1BM23CS330>
```



```
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
MainThread trying to call B.last()
```