VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Ashwini L(24BECS430)

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 **ASHWINI L (24BECS430),** who is a bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

Dr. Prasad G R Professor Ph.D. Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

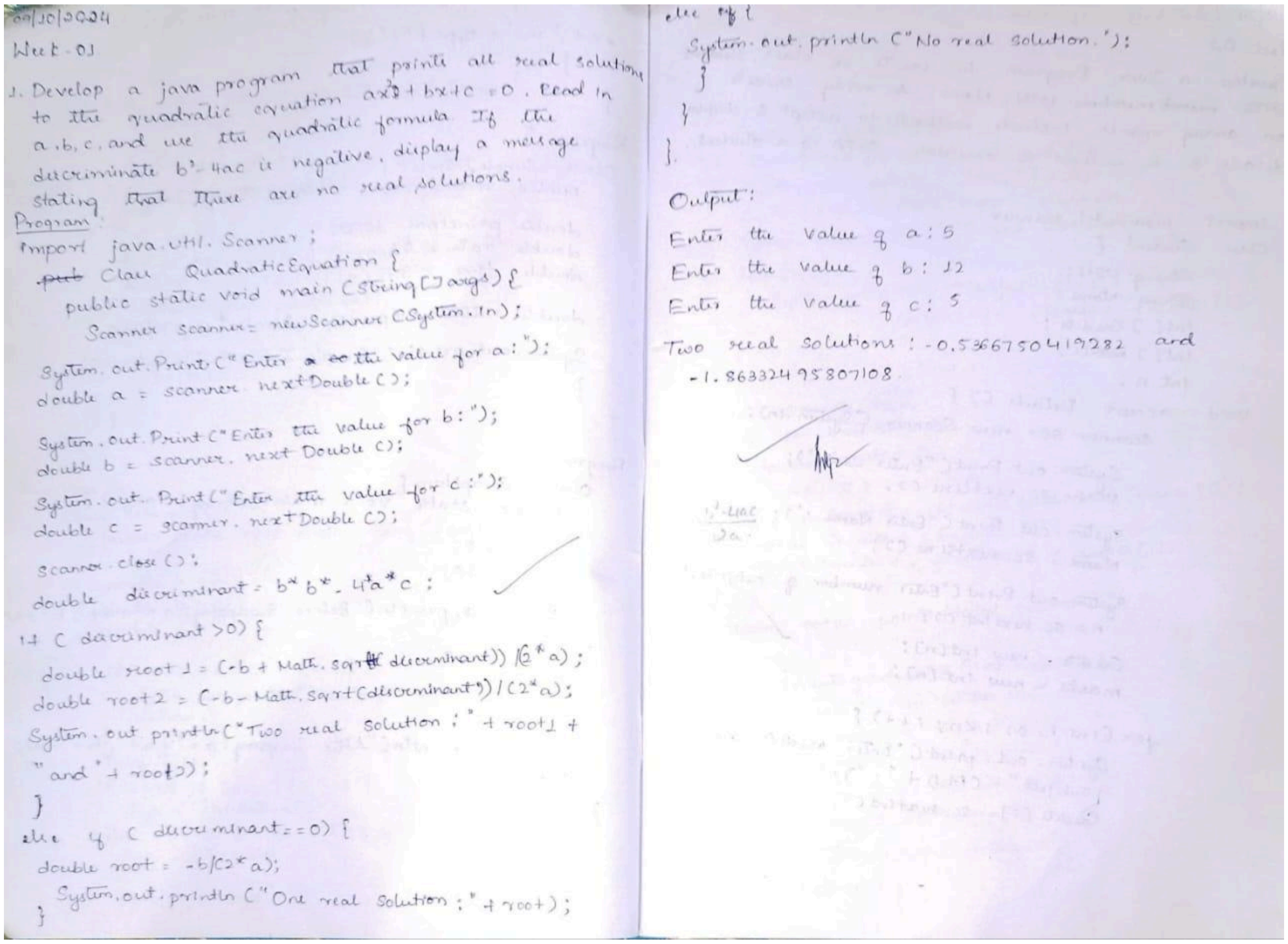
Index

Github Link:
https://github.com/Ashvinigowda/Java-programs-

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 discriminant b2-4ac is negative, display a message stating that there are no real solutions.

Observation :



09/10/2024

Week-01

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

Program:

```
import java.util.Scanner;
 ~~pub~~ Class QuadraticEquation {
  public static void main (String[] args) {
    Scanner scanner= new Scanner (System.in);
 System.out.Print ("Enter ~~a~~ ~~ee~~ the value for a:");
 double a = scanner nextDouble ();

 System.out.Print ("Enter the value for b:");
 double b = scanner. next Double ();

 System.out.Print ("Enter the value for c:");
 double c = scanner. nextDouble ();

 scanner.close ();
 double discriminant = b*b*-4*a*c ;
if ( dacriminant >0) {
 double root1 = (-b + Math.sqrt(discriminant)) /(2*a) ;
 double root2 = (-b-Math.sqrt(discriminant)) / (2*a);
System.out println ("Two real solution :" + root1 +
 " and " + root2));
 }
else if ( discriminant==0) {
 double root = -b/(2*a);
  System.out.println ("One real solution :" + root);
 }
else if {
  System.out.println ("No real solution.");
   }
  }
}.
```

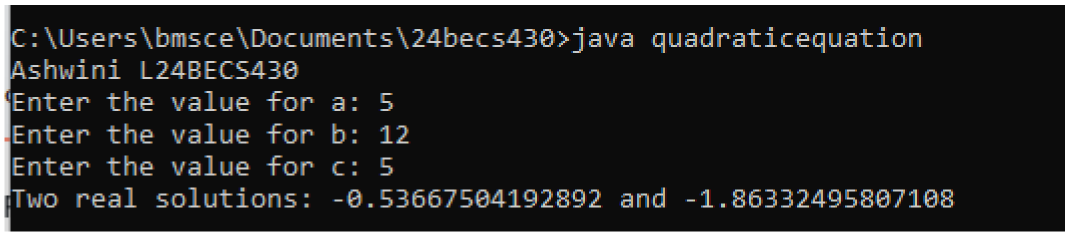
Output:

```
Enter the value of a: 5
Enter the value of b: 12
Enter the value of c: 5
Two real solutions : -0.5366750419282 and
 -1.863324958071O8
```

Code:

```
import java.util.Scanner;
class quadratic equation {
    public static void main(String[] args) {
        System.out.println("Ashwini L" +"24BECS430");
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the value for a: ");
        double a = scanner.nextDouble();
        System.out.print("Enter the value for b: ");
        double b = scanner.nextDouble();
        System.out.print("Enter the value for c: ");
        double c = scanner.nextDouble();

        scanner.close();
        double discriminant = b * b - 4 * a * c;

        if (discriminant > 0) {
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
            System.out.println("Two real solutions: " + root1 + " and " + root2);
        } else if (discriminant == 0) {
            double root = -b / (2 * a);
            System.out.println("One real solution: " + root);
        } else {
            System.out.println("No real solution.");
        }
    }
}
```

Output:

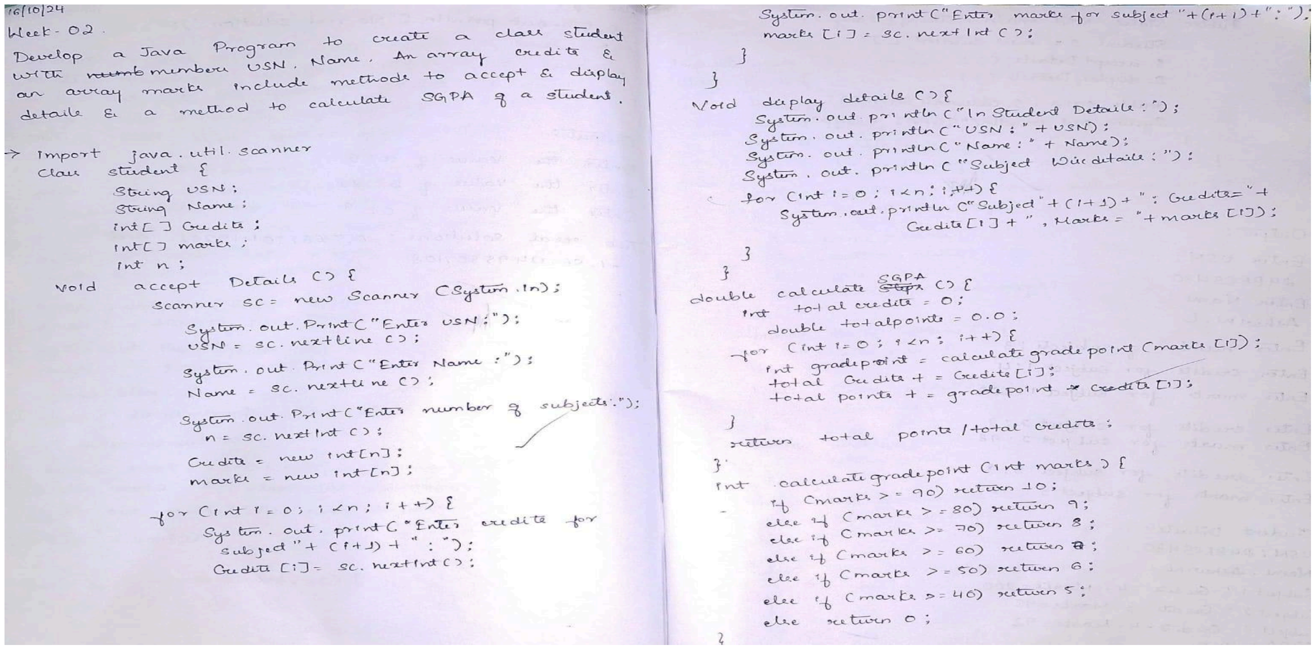


```
C:\Users\bmsce\Documents\24becs430>java quadraticequation
Ashwini L24BECS430
Enter the value for a: 5
Enter the value for b: 12
Enter the value for c: 5
Two real solutions: -0.53667504192892 and -1.86332495807108
```

Program 2

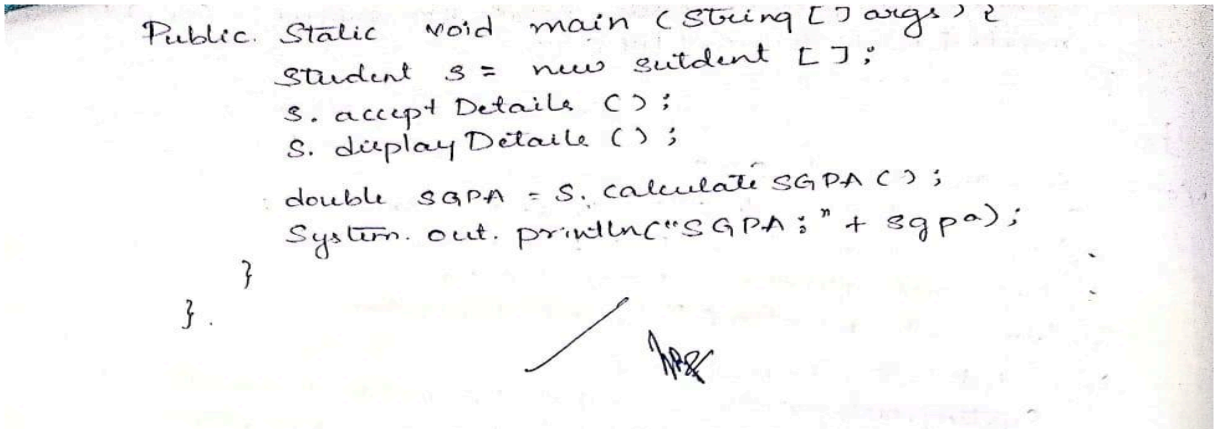
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks.Include methods to accept and display details and a method to calculate SGPA of a student.

Observation :



16/10/24
Week - 02 .
Develop a Java Program to create a class student with ~~numb~~ members USN , Name , An array credits & an array marks include methods to accept & display details & a method to calculate SGPA of a student .

```
→ Import  java . util . scanner
  Class  student {
          String USN ;
          String Name ;
          int[ ] Credits ;
          int[ ] marks ;
          int n ;
     void  accept  Details () {
             Scanner  SC =  new Scanner (System . In) ;
               System. out. Print (" Enter USN :") ;
               USN =  sc. nextline () ;
               System . out . Print (" Enter Name :") ;
               Name =  sc. nextline () ;
               System . out. Print (" Enter number of subjects :") ;
                 n = sc. nextInt () ;
               Credits = new int[n] ;
               marks = new int[n] ;
           for (int i = 0 ; i < n ; i ++) {
                 System . out . print (" Enter credits for
                   subject " + (i+1) + " : ") ;
                 Credits [i] = sc. nextInt () ;
            System. out. print (" Enter marks for subject " + (i+1) + " : ") ;
            marks [ i ] = sc. next Int () ;
         }
     }
   Void  display details () {
         System. out. println ("In Student Details :") ;
        System. out. println (" USN : " + USN) ;
        System. out. println (" Name : " + Name) ;
        System . out. println (" Subject Wise details : ") ;
        for (int i = 0 ; i < n ; i++) {
             System. out. println (" Subject " + (i+1) + " : Credits = " +
                                Credits [i] + " , Marks = " + marks [i]) ;
        }
     }
   double  calculate ~~Stgpa~~ SGPA () {
        int  total credits = 0 ;
          double totalpoints = 0.0 ;
       for  (int i = 0 ; i < n ; i++) {
          int gradepoint = calculate gradepoint (marks [i]) ;
          total  Credits + = Credits [i] ;
          total points + = gradepoint * Credits [i] ;
      }
      return  total  points / total credits ;
   }
   int   calculate gradepoint (int marks ) {
          if (marks >= 90) return 10 ;
          else if (marks >= 80) return 9 ;
          else if (marks >= 70) return 8 ;
          else if (marks >= 60) return 7 ;
          else if (marks >= 50) return 6 ;
          else if (marks >= 40) return 5 ;
          else  return 0 ;
   }
```

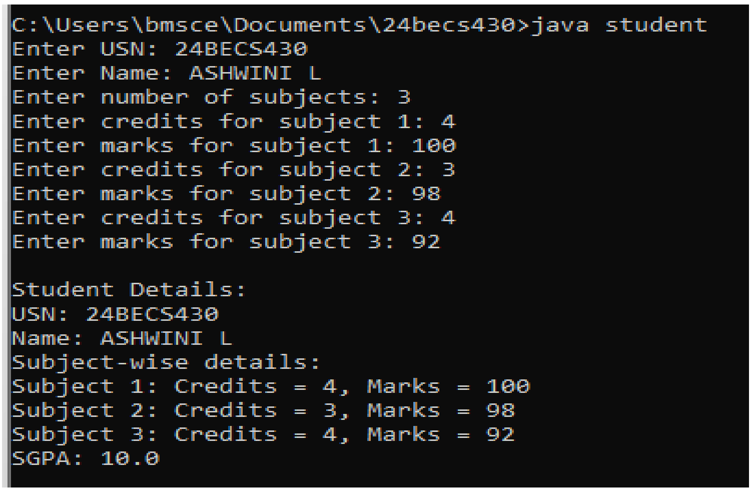


```
Public Static void main (String [ ] args ) {
        Student s = new Student [ ] ;
        s. accept Details () ;
        S. display Details () ;
        double SGPA = S. calculate SGPA () ;
        System. out. println ("SGPA : " + sgpa) ;
    }
}.
```

Code:

```
import java.util.Scanner;
class student {
    String usn;
    String name;
    int[] credits;
    int[] marks;
    int n;

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

        System.out.print("Enter USN: ");
        usn = sc.nextLine();
        System.out.print("Enter Name: ");
        name = sc.nextLine();
        System.out.print("Enter number of subjects: ");
        n = sc.nextInt();
        credits = new int[n];
        marks = new int[n];
        for (int i = 0; i < n; i++) {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = sc.nextInt();
       System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = sc.nextInt();
        }
    }
    void displayDetails() {
        System.out.println("\nStudent Details:");
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Subject-wise details:");

        for (int i = 0; i < n; i++) {
            System.out.println("Subject " + (i + 1) + ": Credits = " + credits[i] + ", Marks = " + marks[i]);
        }
    }

    double calculateSGPA() {
        int totalCredits = 0;
        double totalPoints = 0.0;

        for (int i = 0; i < n; i++) {
            int gradePoint = calculateGradePoint(marks[i]);
            totalCredits += credits[i];
            totalPoints += gradePoint * credits[i];
        }

        return totalPoints / totalCredits;
    }

    int calculateGradePoint(int marks) {
        if (marks >= 90) return 10;
        else if (marks >= 80) return 9;
        else if (marks >= 70) return 8;
        else if (marks >= 60) return 7;
        else if (marks >= 50) return 6;
        else if (marks >= 40) return 5;
        else return 0; // Fail
    }
    public static void main(String[] args) {
        student s = new student();
        s.acceptDetails();
        s.displayDetails();
            double sgpa = s.calculateSGPA();
        System.out.println("SGPA: " + sgpa);
    }
}
```

Output:

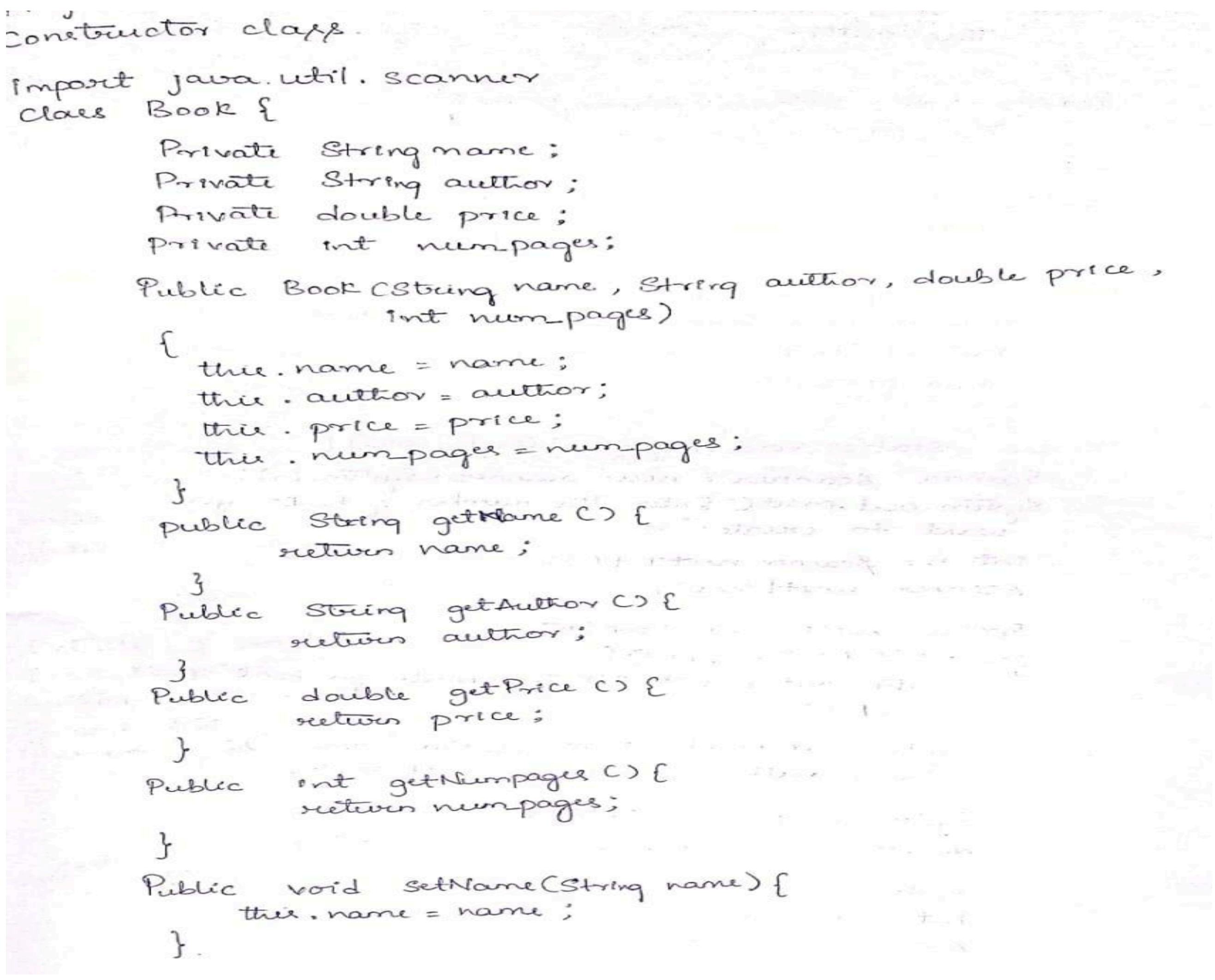


```
C:\Users\bmsce\Documents\24becs430>java student
Enter USN: 24BECS430
Enter Name: ASHWINI L
Enter number of subjects: 3
Enter credits for subject 1: 4
Enter marks for subject 1: 100
Enter credits for subject 2: 3
Enter marks for subject 2: 98
Enter credits for subject 3: 4
Enter marks for subject 3: 92

Student Details:
USN: 24BECS430
Name: ASHWINI L
Subject-wise details:
Subject 1: Credits = 4, Marks = 100
Subject 2: Credits = 3, Marks = 98
Subject 3: Credits = 4, Marks = 92
SGPA: 10.0
```

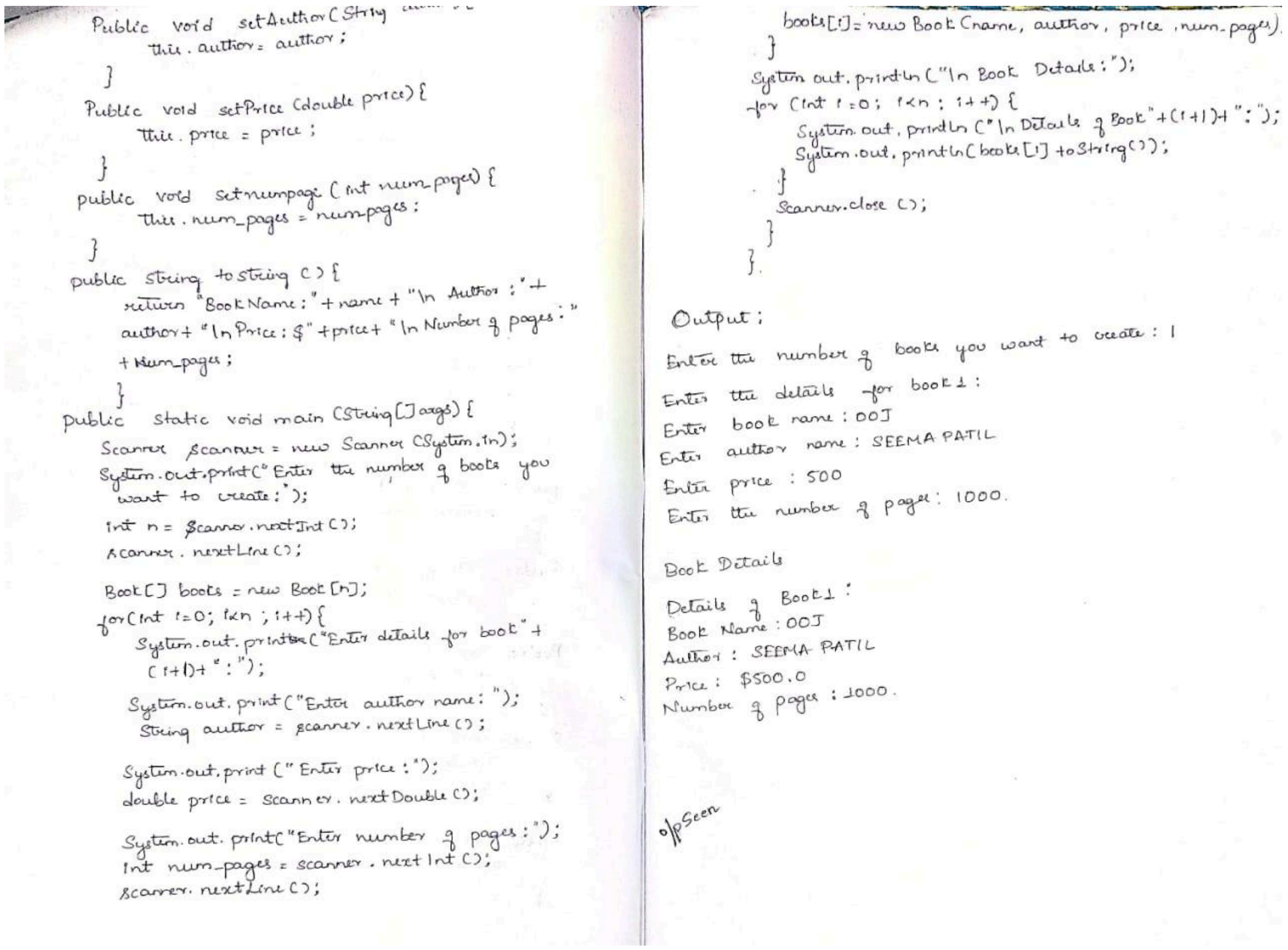
Program 3

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.

Observation:



```
Constructor class.
Import java.util.scanner
Class Book {
        Private String name ;
        Private String author ;
        Private double price ;
        Private int num_pages;
       Public Book (String name , String author, double price ,
                    int num_pages)
        {
          this.name = name ;
          this . author = author;
          this . price = price ;
          this . num_pages = num_pages ;
        }
        public String getName () {
               return name ;
         }
        Public String getAuthor () {
               return author ;
         }
        Public double getPrice () {
               return price ;
         }
        Public int getNumpages () {
               return num_pages;
        }
       Public void setName(String name) {
             this.name = name ;
        }.
```



```
Public void setAuthor(String
        this.author= author;
    }
  Public void setPrice (double price){
        this. price = price;
    }
  public void setnumpage ( int num_pages) {
        this. num_pages = num_pages;
    }
  public String tostring ( ) {
       return "Book Name:"+ name + "\n Author :"+
       author+ "\n Price : $"+price+ "\n Number of pages:"
       + Num_pages;
     }
public  static void main (String[] args) {
    Scanner scanner = new Scanner (System.in);
    System.out.print(" Enter the number of books you
      want to create:");
    int n = scanner.nextInt ();
    scanner. nextLine ();

    Book[] books = new Book [n];
   for(int i=0; i<n ; i++){
       System.out. println("Enter details for book" +
        (i+1)+ ":");
      System.out. print ("Enter author name: ");
       String author = scanner. nextLine ();

     System.out. print (" Enter price :");
     double price = Scanner. nextDouble ();

     System. out. print("Enter number of pages:");
     Int num_pages = scanner . next Int ();
     scanner. nextLine ();
         books[i]= new Book (name, author, price ,num_pages).
       }
    System out. println ("\n Book Details:");
   for (int i =0; i<n ; i++) {
        System. out. println ("\n Details of Book"+(i+1)+":");
        System.out. println(books[i] toString());
     }
     Scanner.close ();
    }
  }.
```

Output:

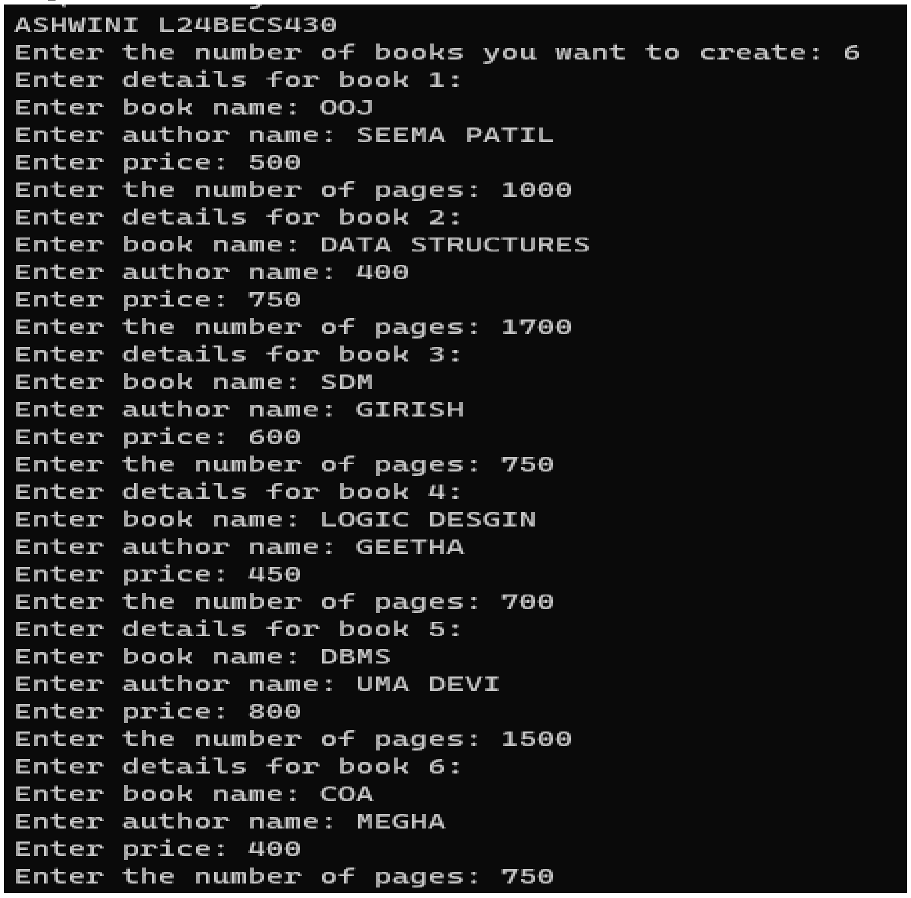
```
Enter the number of books you want to create : 1
Enter the details for book 1 :
Enter book name : 00J
Enter author name : SEEMA PATIL
Enter price : 500
Enter the number of pages : 1000.

Book Details

Details of Book 1 :
Book Name : 00J
Author : SEEMA PATIL
Price : $500.0
Number of pages : 1000.
```

o/p seen

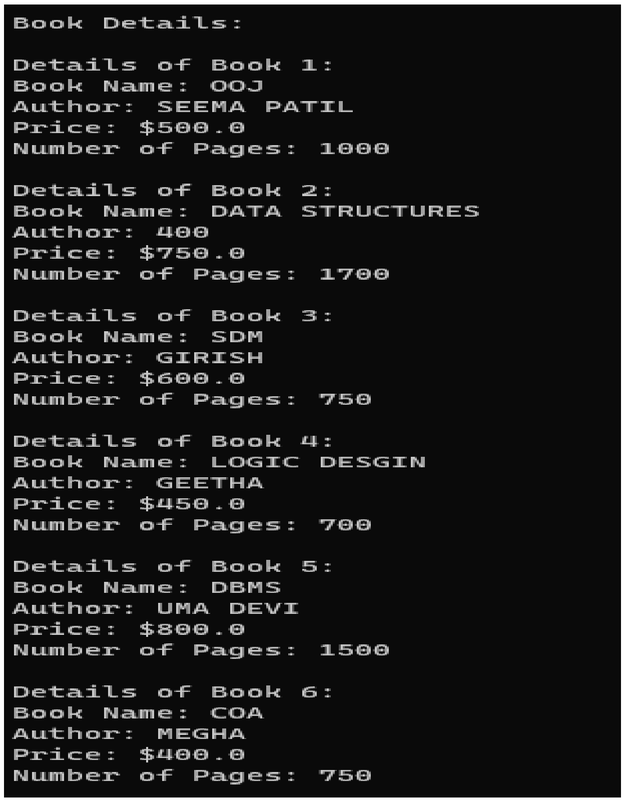
Code:

```
import java.util.Scanner;

class Book {
    private String name;
    private String author;
    private double price;
    private int numPages;

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

    public String getAuthor() {
        return author;
    }

    public double getPrice() {
        return price;
    }

    public int getNumPages() {
        return numPages;
    }

    public void setName(String name) {
        this.name = name;
    }

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

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

    public void getNumPages(int numPages) {
        this.numPages = numPages;
    }

    public String toString() {
        return "Book Name: " + name + "\nAuthor: " + author + "\nPrice: $" + price + "\nNumber of
Pages: " + numPages;
    }

    public static void main(String[] args) {
        System.out.println("ASHWINI L"+"24BECS430");
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of books you want to create: ");
        int n = scanner.nextInt();
        scanner.nextLine();

        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for book " + (i + 1) + ":");
            System.out.print("Enter book name: ");
            String name = scanner.nextLine();

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

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

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

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

        System.out.println("\nBook Details:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nDetails of Book " + (i + 1) + ":");
            System.out.println(books[i].toString());

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

Output:



```
ASHWINI L24BECS430
Enter the number of books you want to create: 6
Enter details for book 1:
Enter book name: OOJ
Enter author name: SEEMA PATIL
Enter price: 500
Enter the number of pages: 1000
Enter details for book 2:
Enter book name: DATA STRUCTURES
Enter author name: 400
Enter price: 750
Enter the number of pages: 1700
Enter details for book 3:
Enter book name: SDM
Enter author name: GIRISH
Enter price: 600
Enter the number of pages: 750
Enter details for book 4:
Enter book name: LOGIC DESGIN
Enter author name: GEETHA
Enter price: 450
Enter the number of pages: 700
Enter details for book 5:
Enter book name: DBMS
Enter author name: UMA DEVI
Enter price: 800
Enter the number of pages: 1500
Enter details for book 6:
Enter book name: COA
Enter author name: MEGHA
Enter price: 400
Enter the number of pages: 750
```

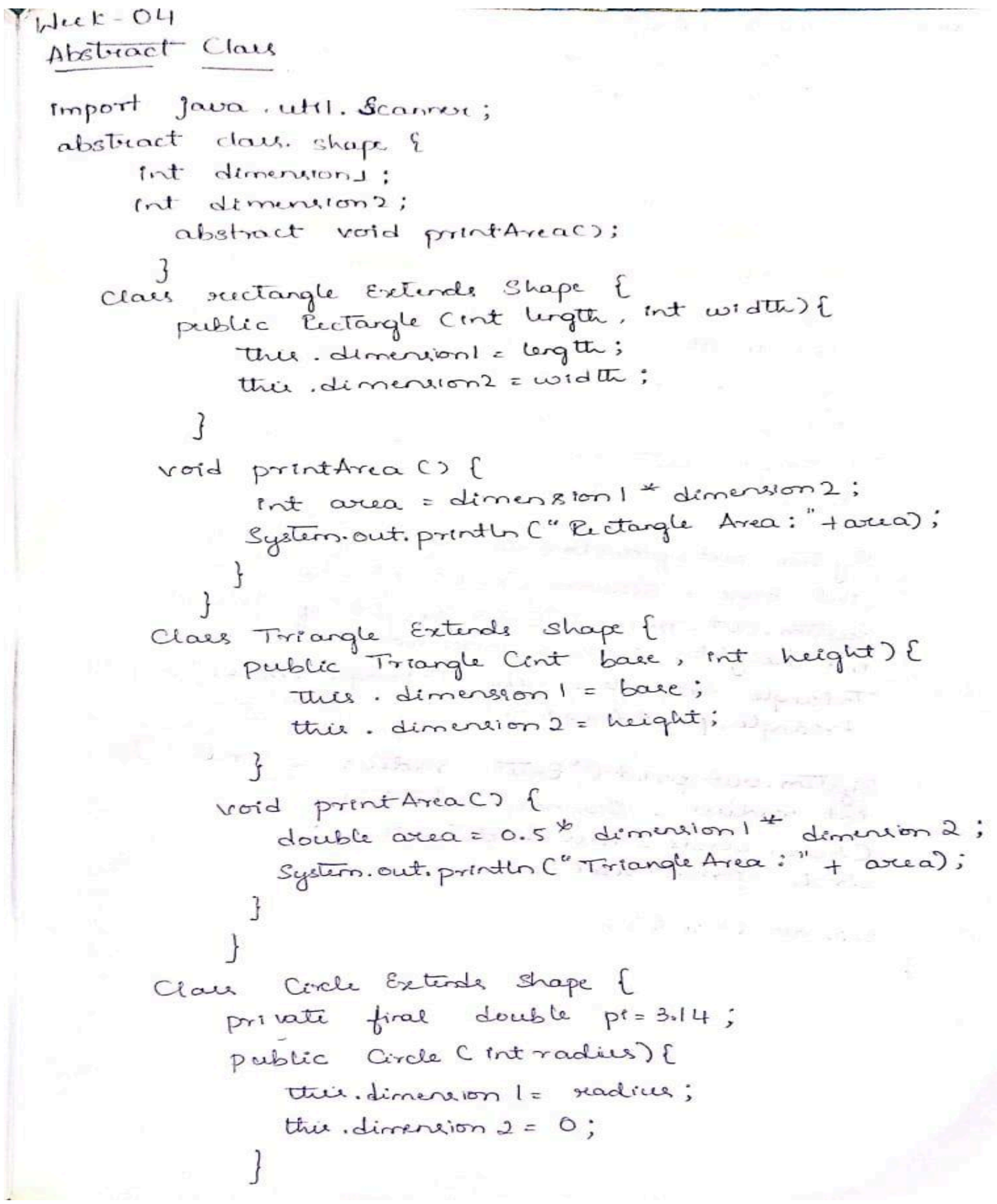


```
Book Details:

Details of Book 1:
Book Name: OOJ
Author: SEEMA PATIL
Price: $500.0
Number of Pages: 1000

Details of Book 2:
Book Name: DATA STRUCTURES
Author: 400
Price: $750.0
Number of Pages: 1700

Details of Book 3:
Book Name: SDM
Author: GIRISH
Price: $600.0
Number of Pages: 750

Details of Book 4:
Book Name: LOGIC DESGIN
Author: GEETHA
Price: $450.0
Number of Pages: 700

Details of Book 5:
Book Name: DBMS
Author: UMA DEVI
Price: $800.0
Number of Pages: 1500

Details of Book 6:
Book Name: COA
Author: MEGHA
Price: $400.0
Number of Pages: 750
```

Program 4

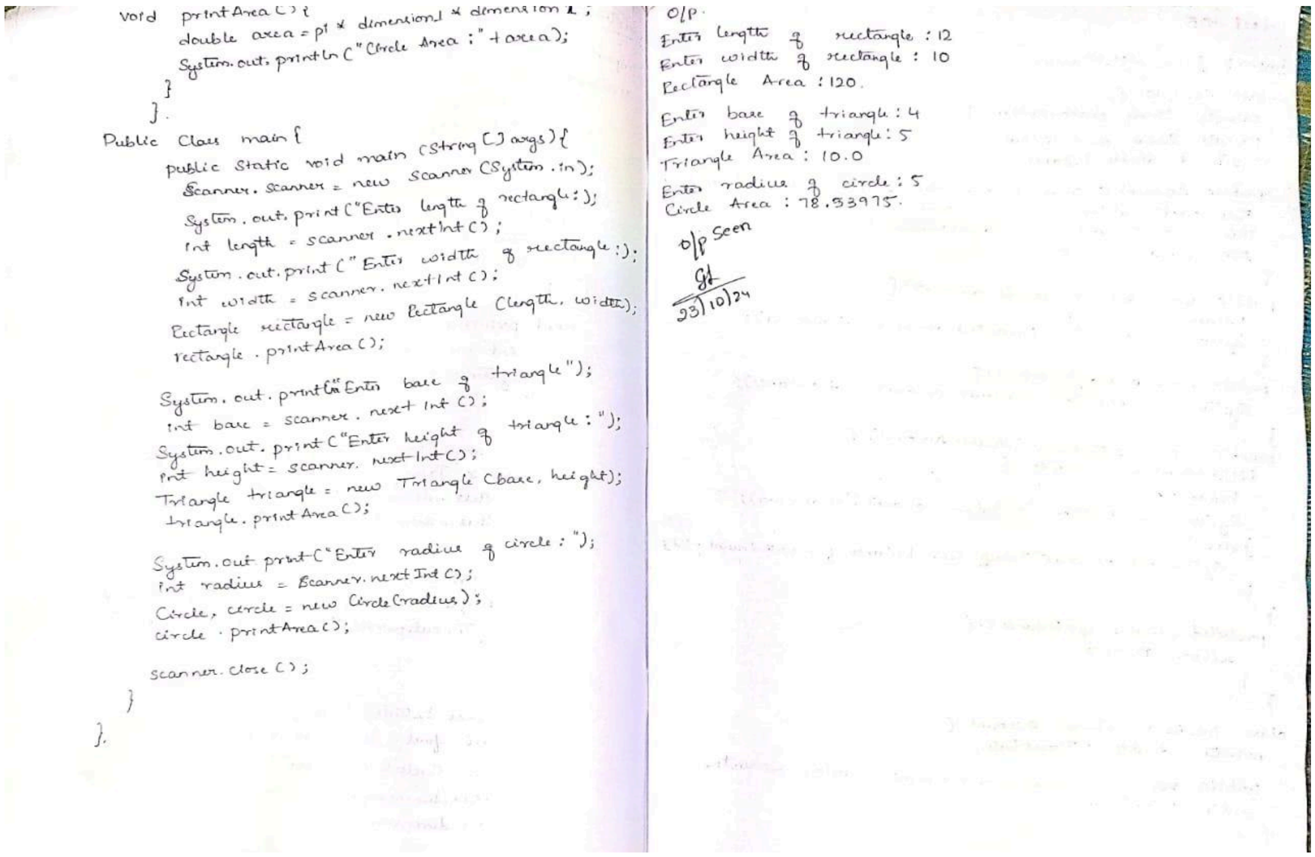
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method printArea(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

Observation:



Week - 04
Abstract Class

```
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 = 3.14;
       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.println("Enter base of triangle");
    int base = scanner.next Int();
   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();
  }
 }.
```

O/P.

```
Enter length of rectangle : 12
Enter width of rectangle : 10
Rectangle Area : 120.

Enter base of triangle : 4
Enter height of triangle : 5
Triangle Area : 10.0

Enter radius of circle : 5
Circle Area : 78.53975.
```

O/P Seen

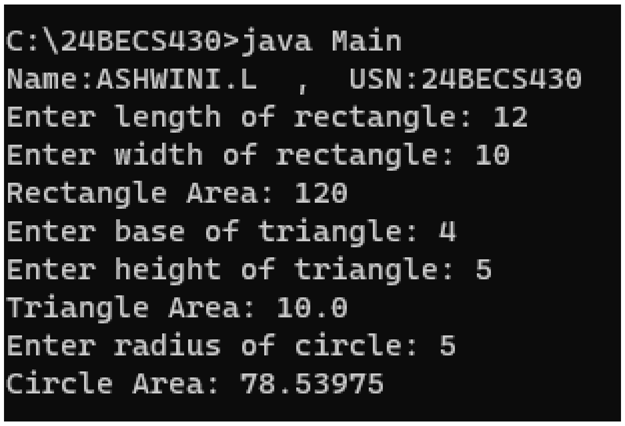
[illegible]
23/10/24

Code:

```
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.println("Name:ASHWINI.L  , USN:24BECS430");
        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();
    }
}
```

Output:

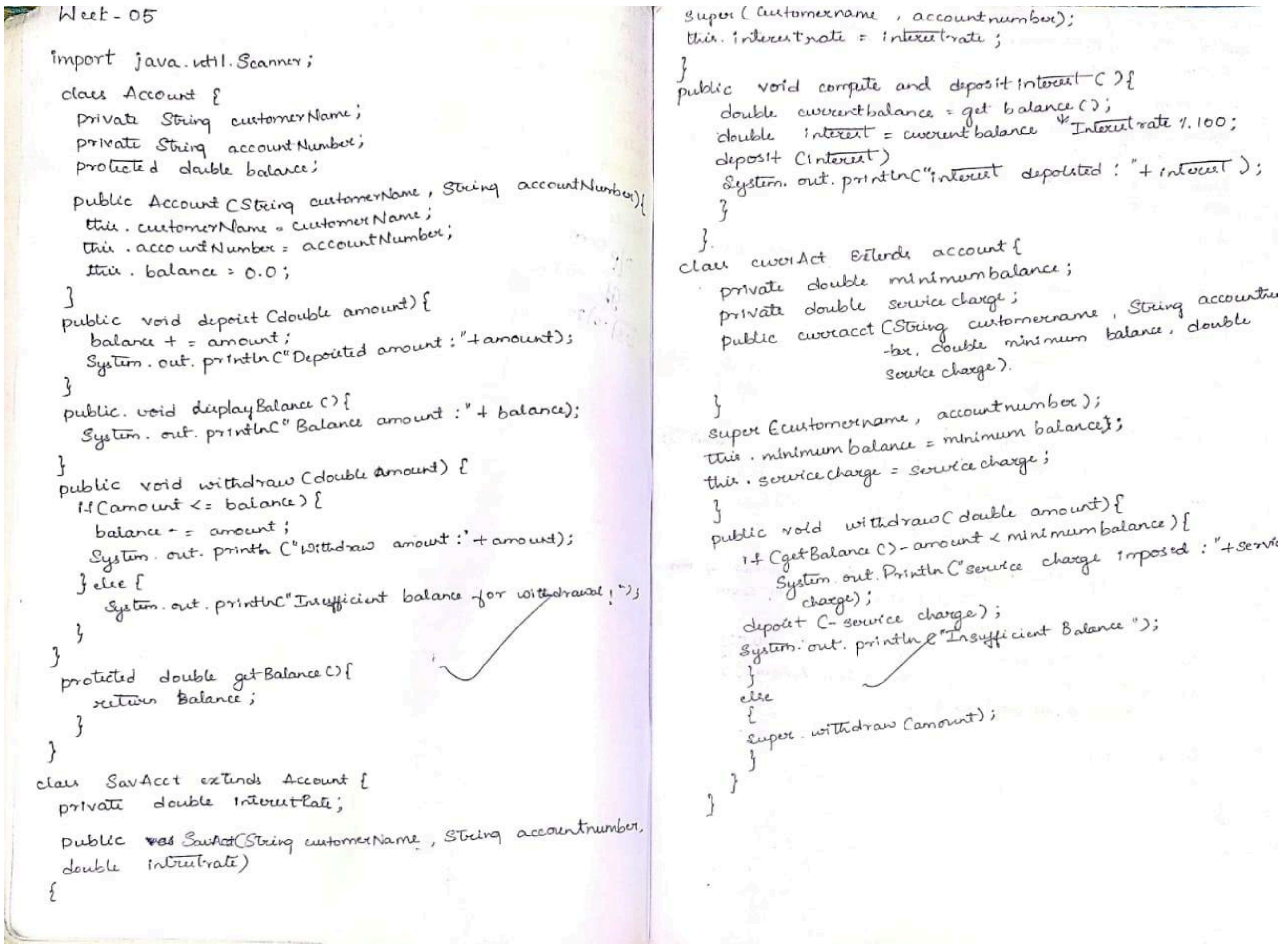


```
C:\24BECS430>java Main
Name:ASHWINI.L  ,  USN:24BECS430
Enter length of rectangle: 12
Enter width of rectangle: 10
Rectangle Area: 120
Enter base of triangle: 4
Enter height of triangle: 5
Triangle Area: 10.0
Enter radius of circle: 5
Circle Area: 78.53975
```

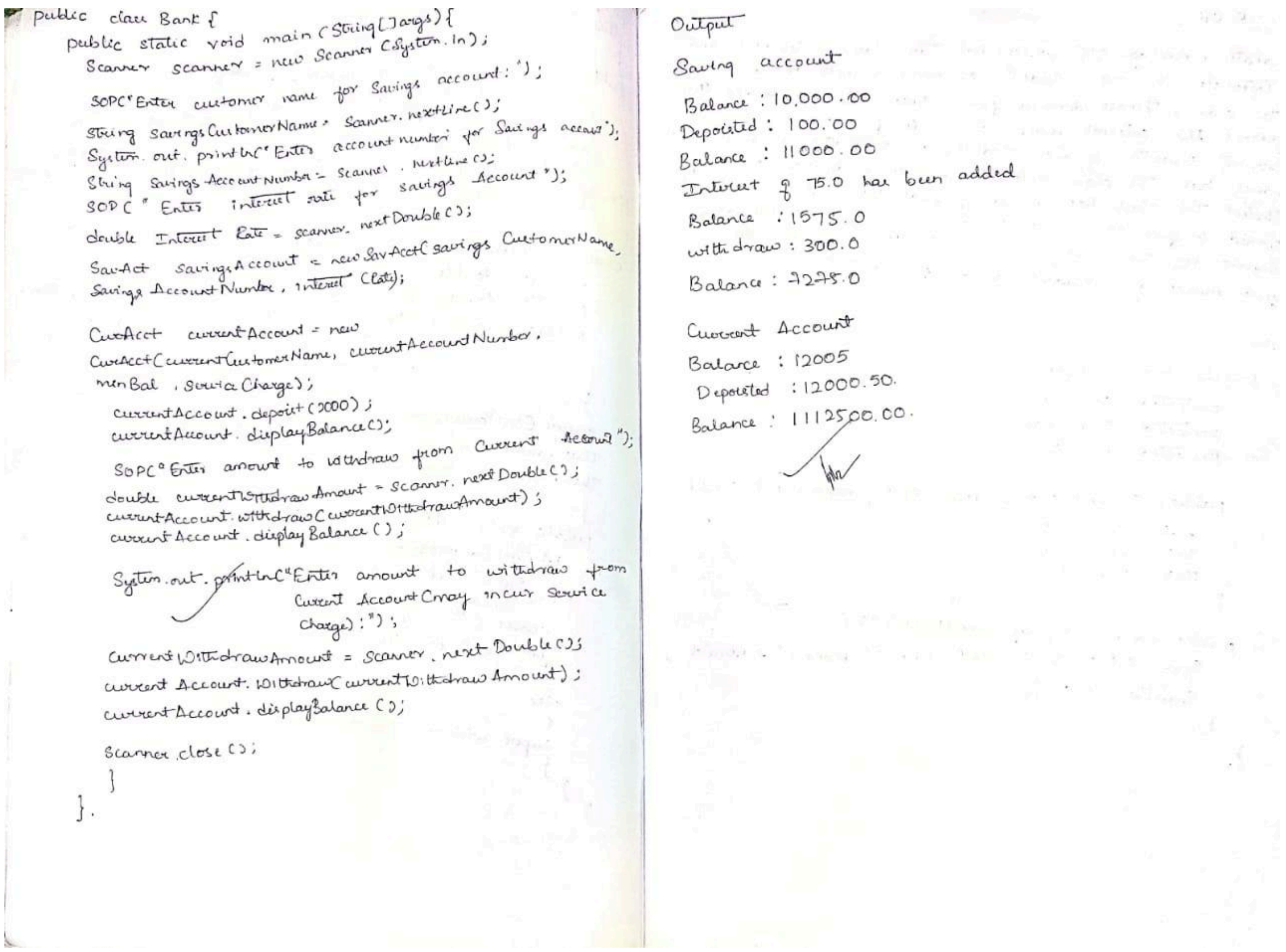
Program 5

Develop a Java program to create a class Bank that maintains two kinds of accounts for its customers, one called savings account and the other a 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.

Observation:



```
Week-05

import java.util.Scanner;
 class Account {
  private String customerName;
  private String accountNumber;
  protected double balance;

  public Account (String customerName, String accountNumber){
   this.customerName = customerName;
   this.accountNumber = accountNumber;
   this. balance = 0.0;
 }
 public void deposit (double amount) {
   balance + = amount;
   System.out.println("Deposited amount :"+amount);
 }
 public. void displayBalance () {
   System. out. println(" Balance amount :" + balance);
 }
 public void withdraw (double Amount) {
  If(amount <= balance) {
   balance -= amount;
   System. out. println ("Withdraw amout :'+amount);
  }else {
   System.out.println("Insufficient balance for withdrawal !");
  }
 }
 protected double getBalance(){
    return Balance;
  }
}
class SavAcct extends Account {
 private double InterestRate;

 public ~~was~~ SavAct(String customerName, String accountnumber,
 double interestrate)
 {
 super( customername , accountnumber);
 this. interestrate = interestrate;
}
public void compute and deposit interest ( ){
   double currentbalance = get balance ();
   double interest = current balance * Interestrate % 100;
   deposit (interest)
   System. out. println("interest depolited : "+ interest );
   }
 }.
class currAct Extends account {
  private double minimumbalance;
  private double service charge;
  public curracct (String customername , String accountnu
              -ber, double minimum balance, double
              service charge).

  }
 super (customername, accountnumber);
 this . minimum balance = minimum balance};
 this . service charge = service charge;
  }
 public void withdraw( double amount){
   if (getBalance ()- amount < minimumbalance ){
      System.out.Println("service charge imposed : "+servi
        charge);
   deposit (- service charge);
   System.out. println ("Insufficient Balance ");
   }
   else
   {
   super . withdraw (amount);
   }
  }
}
```



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

    SOP("Enter customer name for Savings account:");
    String savingsCustomerName = Scanner.nextLine();
    System.out.println("Enter account number for Savings account");
    String SavingsAccountNumber = Scanner.nextLine();
    SOP(" Enter interest rate for savings Account");
    double InterestRate = scanner.nextDouble();
    SavAct savingsAccount = new SavAcct(savingsCustomerName,
    SavingsAccountNumber, interest(Rate);

    CurAcct currentAccount = new
    CurAcct(currentCustomerName, currentAccountNumber,
    minBal , Service Charge);
     currentAccount.deposit(2000);
     currentAccount.displayBalance();
     SOP("Enter amount to withdraw from Current Account");
    double currentWithdrawAmount = Scanner.nextDouble();
    currentAccount.withdraw(currentWithdrawAmount);
    current Account.display Balance ();

     System.out.println("Enter amount to withdraw from
                 Current Account(may incur Service
                 Charge):");
    CurrentWithdrawAmount = Scanner.next Double();
    current Account.Withdraw(currentWithdraw Amount);
    currentAccount.displayBalance ();

    Scanner.close();
    }
  }.
```

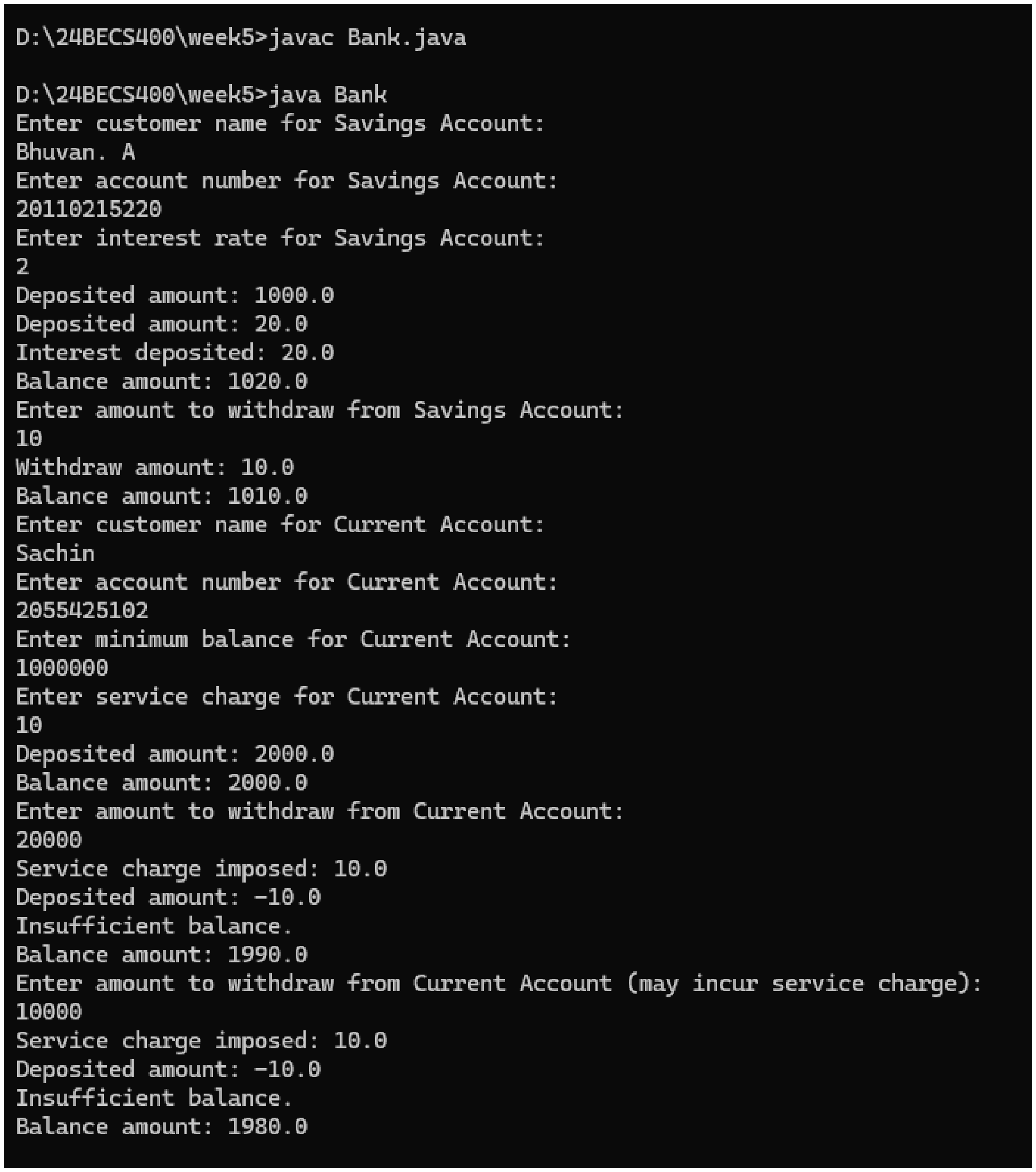
```
Output
Saving account
 Balance : 10,000.00
 Deposited : 100.00
 Balance : 11000.00
 Interest of 75.0 has been added
  Balance : 1575.0
  withdraw : 300.0
  Balance : 7275.0

  Current Account
  Balance : 12005
   Deposited : 12000.50.
  Balance : 1112500.00.
```

Code:

```
import java.util.Scanner;
class Account {
   private String customerName;
   private String accountNumber;
   protected double balance;

   public Account(String customerName, String accountNumber) {
      this.customerName = customerName;
      this.accountNumber = accountNumber;
      this.balance = 0.0;
   }
   public void deposit(double amount) {
      balance += amount;
      System.out.println("Deposited amount: " + amount);
   }

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

```
    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdraw amount: " + amount);
        } else {
            System.out.println("Insufficient balance for withdrawal!");
        }
    }
    protected double getBalance() {
        return balance;
    }
}
class SavAcct extends Account {
    private double interestRate;
    public SavAcct(String customerName, String accountNumber, double interestRate) {
        super(customerName, accountNumber);
        this.interestRate = interestRate;
    }
    public void computeAndDepositInterest() {
        double currentBalance = getBalance();
        double interest = currentBalance * interestRate / 100;
        deposit(interest);
        System.out.println("Interest deposited: " + interest);
    }
}
class CurAcct extends Account {
    private double minimumBalance;
    private double serviceCharge;

    public CurAcct(String customerName, String accountNumber, double minimumBalance, double
serviceCharge) {
        super(customerName, accountNumber);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }
    public void withdraw(double amount) {
        if (getBalance() - amount < minimumBalance) {
            System.out.println("Service charge imposed: " + serviceCharge);
            deposit(-serviceCharge);
            System.out.println("Insufficient balance.");
        } else {
            super.withdraw(amount);
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter customer name for Savings Account:");
        String savingsCustomerName = scanner.nextLine();
        System.out.println("Enter account number for Savings Account:");
        String savingsAccountNumber = scanner.nextLine();
        System.out.println("Enter interest rate for Savings Account:");
        double interestRate = scanner.nextDouble();

        SavAcct savingsAccount = new SavAcct(savingsCustomerName, savingsAccountNumber,
interestRate);
        savingsAccount.deposit(1000);
        savingsAccount.computeAndDepositInterest();
        savingsAccount.displayBalance();
        System.out.println("Enter amount to withdraw from Savings Account:");
        double withdrawAmount = scanner.nextDouble();
        savingsAccount.withdraw(withdrawAmount);
        savingsAccount.displayBalance();
        scanner.nextLine();
        System.out.println("Enter customer name for Current Account:");
        String currentCustomerName = scanner.nextLine();
        System.out.println("Enter account number for Current Account:");
        String currentAccountNumber = scanner.nextLine();
        System.out.println("Enter minimum balance for Current Account:");
        double minimumBalance = scanner.nextDouble();
        System.out.println("Enter service charge for Current Account:");
        double serviceCharge = scanner.nextDouble();

        CurAcct currentAccount = new CurAcct(currentCustomerName, currentAccountNumber,
minimumBalance, serviceCharge);
        currentAccount.deposit(2000);
        currentAccount.displayBalance();

        System.out.println("Enter amount to withdraw from Current Account:");
        double currentWithdrawAmount = scanner.nextDouble();
        currentAccount.withdraw(currentWithdrawAmount);
        currentAccount.displayBalance();


        System.out.println("Enter amount to withdraw from Current Account (may incur service
charge):");
        currentWithdrawAmount = scanner.nextDouble();
        currentAccount.withdraw(currentWithdrawAmount);
        currentAccount.displayBalance();
        scanner.close();
    }
}
```

Output:

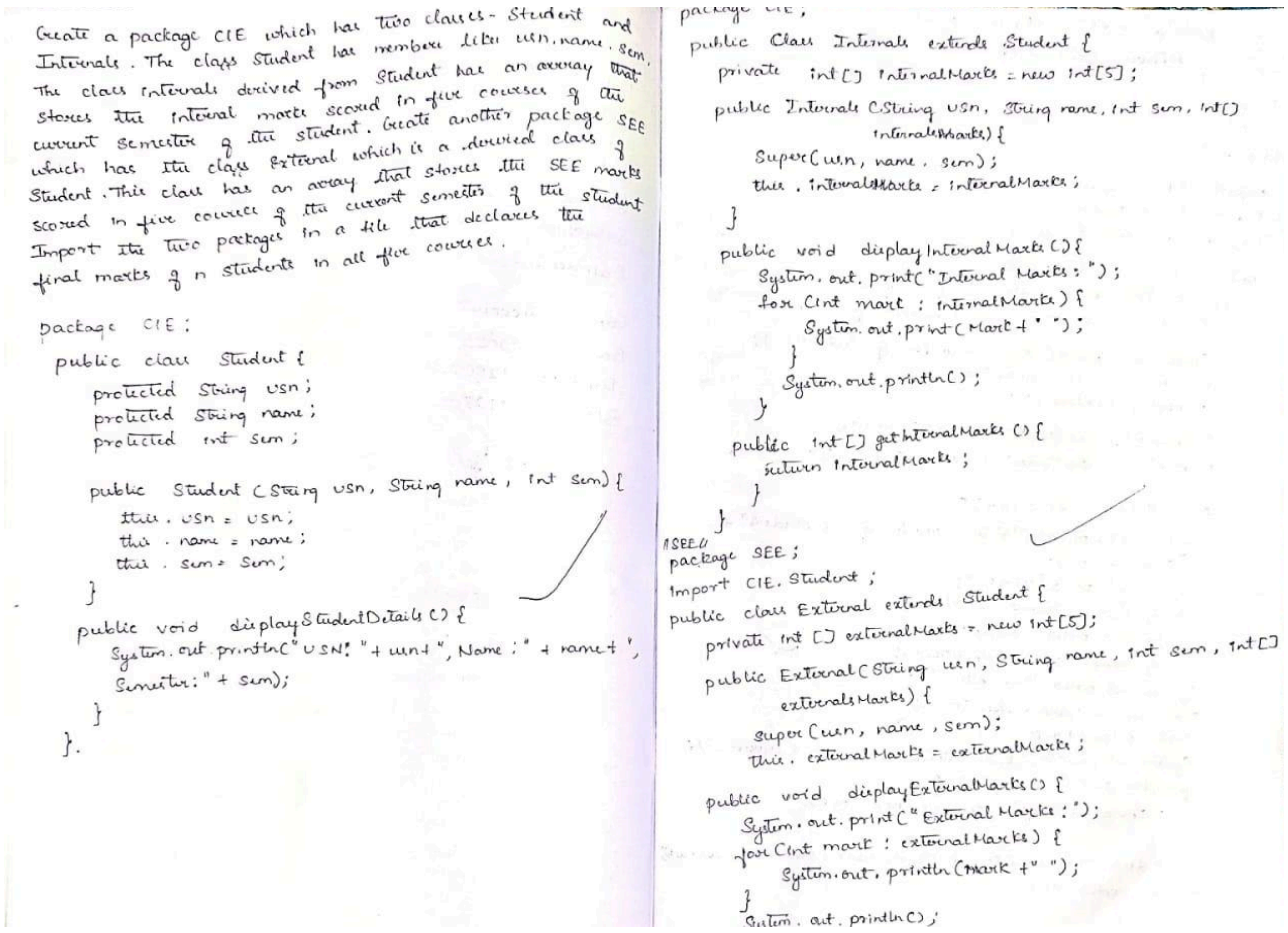


```
D:\24BECS400\week5>javac Bank.java

D:\24BECS400\week5>java Bank
Enter customer name for Savings Account:
Bhuvan. A
Enter account number for Savings Account:
20110215220
Enter interest rate for Savings Account:
2
Deposited amount: 1000.0
Deposited amount: 20.0
Interest deposited: 20.0
Balance amount: 1020.0
Enter amount to withdraw from Savings Account:
10
Withdraw amount: 10.0
Balance amount: 1010.0
Enter customer name for Current Account:
Sachin
Enter account number for Current Account:
2055425102
Enter minimum balance for Current Account:
1000000
Enter service charge for Current Account:
10
Deposited amount: 2000.0
Balance amount: 2000.0
Enter amount to withdraw from Current Account:
20000
Service charge imposed: 10.0
Deposited amount: -10.0
Insufficient balance.
Balance amount: 1990.0
Enter amount to withdraw from Current Account (may incur service charge):
10000
Service charge imposed: 10.0
Deposited amount: -10.0
Insufficient balance.
Balance amount: 1980.0
```

<u>Program 6</u>

Create a package CIE which has two classes- Student 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 Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

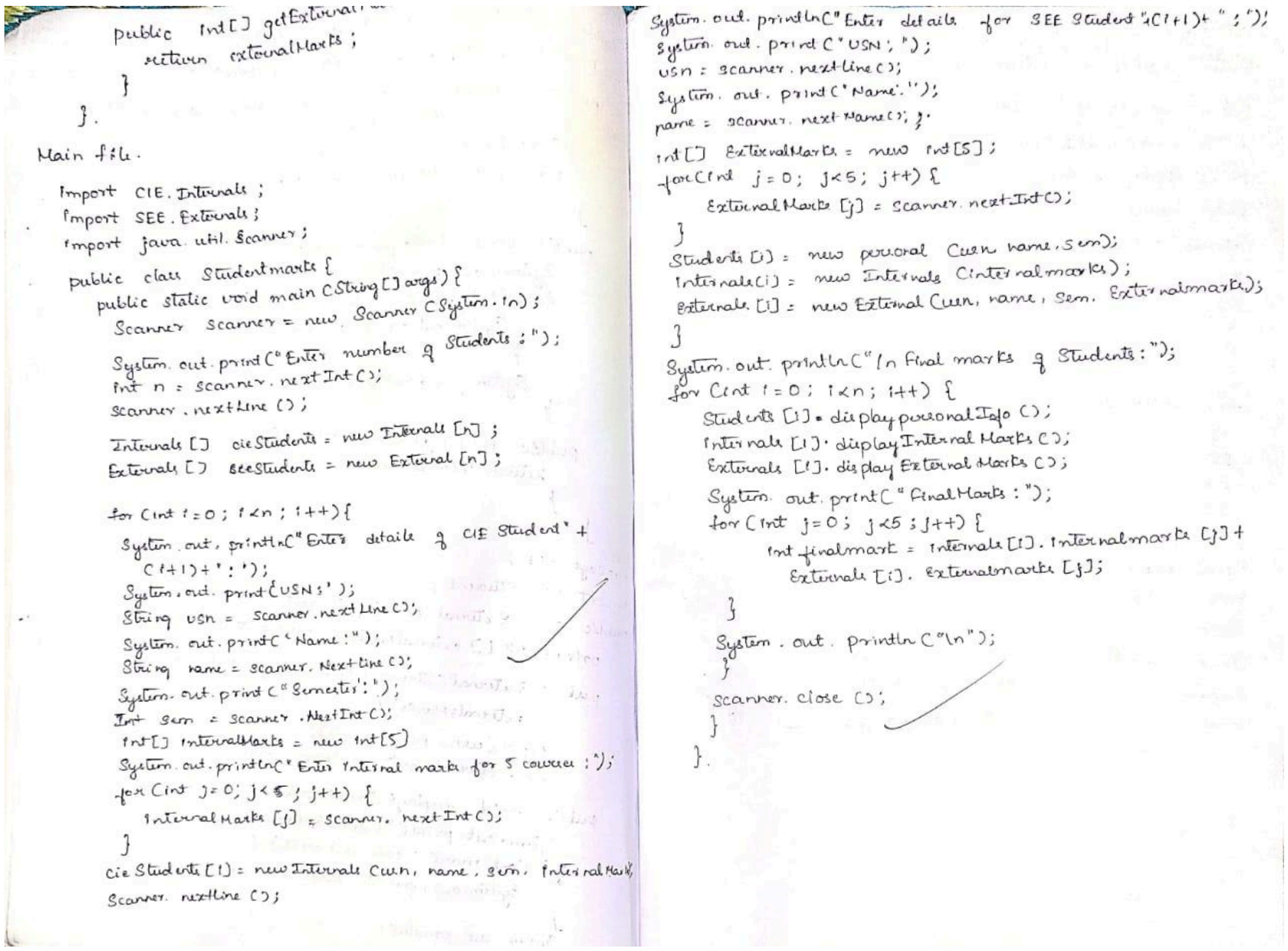
Obesrvation:



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

```
package CIE;
 public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    public Student (String usn, String name, int sem) {
      this. usn = usn;
      this . name = name;
      this . sem = Sem;
    }
  public void displayStudentDetails () {
     System.out.println("USN: "+ usn+ ", Name :" + name+ ",
     Semester:" + sem);
    }
 }.
```

```
package CIE;
 public Class Internals extends Student {
   private int[] internalMarks = new int[5];
   public Internals (String usn, String name, int sem, int[]
                    internalsMarks) {
      Super(usn, name, sem);
      this . internalMarks = internalMarks;
    }
   public void displayInternalMarks (){
      System. out. print("Internal Marks : ");
      for (int mark : internalMarks) {
          System. out. print(Mark + " ");
      }
      System.out.println();
    }
    public int [] getInternalMarks () {
       return internalMarks ;
    }
 }
//SEE//
package SEE;
import CIE. Student ;
public class External extends Student {
   private int [] externalMarks = new int[5];
   public External (String usn, String name, int sem, int[]
          externalsMarks) {
      super (usn, name, sem);
      this. externalMarks = externalMarks ;

   public void displayExternalMarks () {
      System. out. print ("External Marks : ");
     for (int mark : externalMarks) {
          System. out. println (mark +" ");
     }
     Sutem . out. println();
```



```
      public int[] getExternal
          return externalMarks;
      }
  }.

Main file.

  import CIE.Internals;
  import SEE.Externals;
  import java.util.Scanner;
  public class Studentmarks {
    public static void main (String[] args) {
      Scanner scanner = new Scanner (System.in);
      System.out.print ("Enter number of Students:");
      int n = scanner.nextInt();
      scanner.nextLine();
      Internals [] cieStudents = new Internals [n];
      Externals [] seeStudents = new External [n];

      for (int i=0; i<n; i++) {
        System.out.println("Enter details of CIE Student" +
          (i+1)+":");
        System.out.print(USN:');
        String usn = Scanner.nextLine();
        System.out.print('Name:");
        String name = scanner.NextLine();
        System.out.print("Semester':");
        Int sem = scanner.NextInt();
        int[] internalMarks = new int[5]
        System.out.println("Enter internal marks for 5 courses:");
        for (int j=0; j<5; j++) {
          internalMarks [j] = Scanner.nextInt();
        }
      cieStudents[i] = new Internals (usn, name, sem, internalMarks,
      Scanner.nextline();
System.out.println("Enter details for SEE Student"+(i+1)+":");
System.out.print("USN:");
usn = scanner.nextLine();
System.out.print("Name:");
name = scanner.nextName(); }.
int[] ExternalMarks = new int[5];
for(int j=0; j<5; j++) {
     ExternalMarks [j] = Scanner.nextInt();
 }
 Students [i] = new personal (usn name, sem);
 Internals(i) = new Internals (internalmarks);
 Externals [i] = new External (usn, name, Sem. Externalmarks);
 }
 System.out.println("/n Final marks of Students:");
 for (int i=0; i<n; i++) {
    Students [i]. displaypersonalInfo ();
    Internals [i]. displayInternal Marks ();
    Externals [i]. display External Marks ();
    System.out.print("FinalMarks:");
    for (int j=0; j<5; j++) {
          int finalmark = Internals[i].internalmarks [j] +
             Externals [i]. externalmarks [j];
      }
    System.out.println("\n");
    }
   scanner.close();
   }
 }.
```

Code:

CIE
-Internals.java

```
package CIE;
public class Internals extends Student {
    private int[] internalMarks = new int[5];
    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
    public void displayInternalMarks() {
        System.out.print("Internal Marks: ");
        for (int mark : internalMarks) {
            System.out.print(mark + " ");
        }
        System.out.println();
    }

    public int[] getInternalMarks() {
        return internalMarks;
    }
}
```

SEE
-Externals.java

```
package SEE;
import CIE.Student;
public class External extends Student {
private int[] externalMarks = new int[5];
    public External(String usn, String name, int sem, int[] externalMarks) {
        super(usn, name, sem);
        this.externalMarks = externalMarks;
    }
    public void displayExternalMarks() {
        System.out.print("External Marks: ");
        for (int mark : externalMarks) {
            System.out.print(mark + " ");
        }
        System.out.println();
    }
    public int[] getExternalMarks() {
        return externalMarks;
    }
}
```

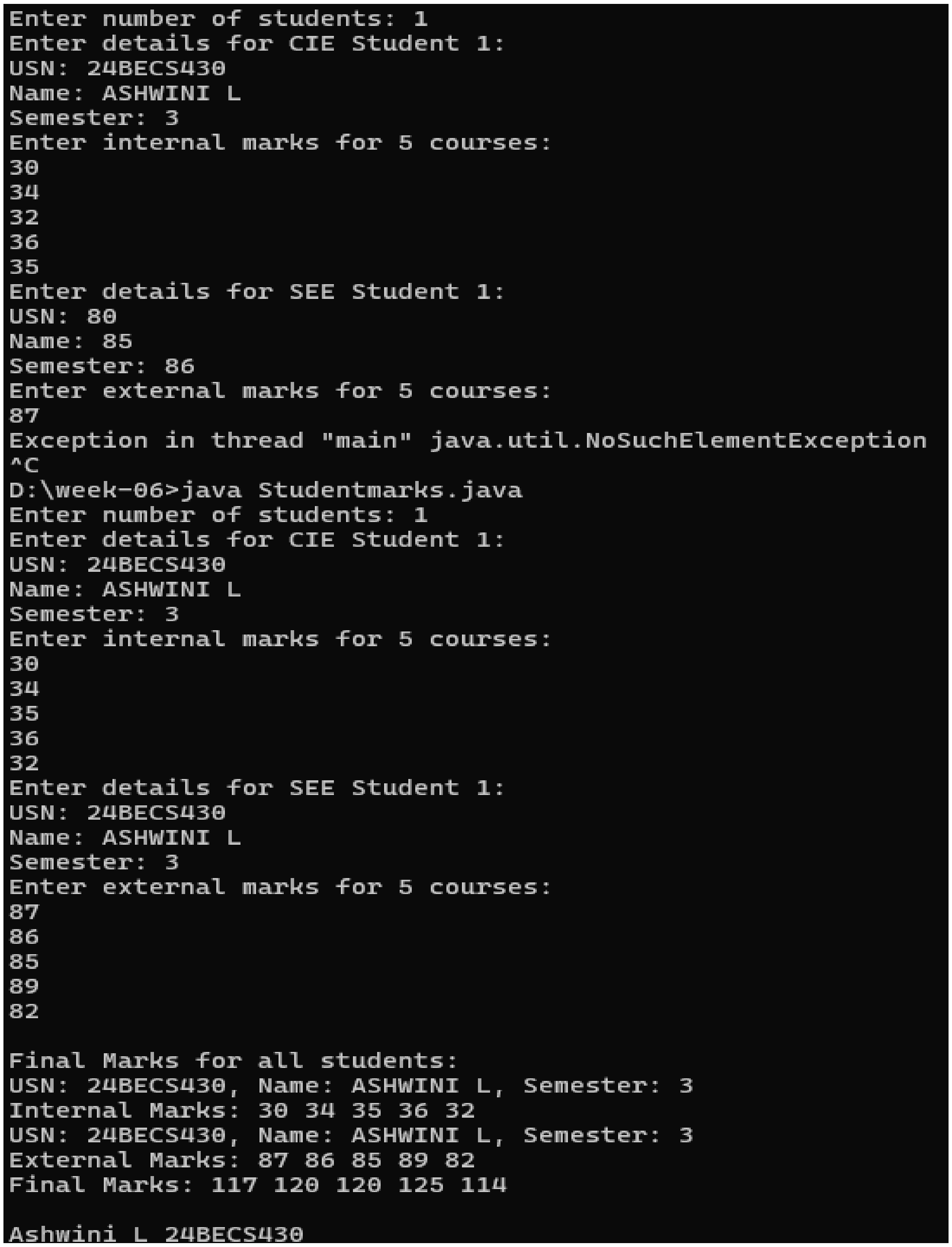
Student.java

```
package CIE;
public class Student {
     protected String usn;
    protected String name;
    protected int sem;

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

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

main.java

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

public class Studentmarks {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of students: ");
        int n = scanner.nextInt();
        scanner.nextLine();
        Internals[] cieStudents = new Internals[n];
        External[] seeStudents = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for CIE Student " + (i + 1) + ": ");
            System.out.print("USN: ");
            String usn = scanner.nextLine();
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Semester: ");
            int sem = scanner.nextInt();
            int[] internalMarks = new int[5];
            System.out.println("Enter internal marks for 5 courses: ");
            for (int j = 0; j < 5; j++) {
                internalMarks[j] = scanner.nextInt();
            }
            cieStudents[i] = new Internals(usn, name, sem, internalMarks);
            scanner.nextLine();
            System.out.println("Enter details for SEE Student " + (i + 1) + ": ");
            System.out.print("USN: ");
            usn = scanner.nextLine();
            System.out.print("Name: ");
            name = scanner.nextLine();
            System.out.print("Semester: ");
            sem = scanner.nextInt();
            int[] externalMarks = new int[5];
            System.out.println("Enter external marks for 5 courses: ");
            for (int j = 0; j < 5; j++) {
                externalMarks[j] = scanner.nextInt();
            }
            seeStudents[i] = new External(usn, name, sem, externalMarks);
            scanner.nextLine();
        }
        System.out.println("\nFinal Marks for all students:");
        for (int i = 0; i < n; i++) {
            cieStudents[i].displayStudentDetails();
            cieStudents[i].displayInternalMarks();
            seeStudents[i].displayStudentDetails();
            seeStudents[i].displayExternalMarks();

            int[] internalMarks = cieStudents[i].getInternalMarks();
            int[] externalMarks = seeStudents[i].getExternalMarks();
            int[] finalMarks = new int[5];

            for (int j = 0; j < 5; j++) {
                finalMarks[j] = internalMarks[j] + externalMarks[j];
            }
            System.out.print("Final Marks: ");
            for (int mark : finalMarks) {
                System.out.print(mark + " ");
            }
            System.out.println("\n");
            System.out.print("Ashwini L 24BECS430 ");
        }

        scanner.close();
    }
}
```

Output:

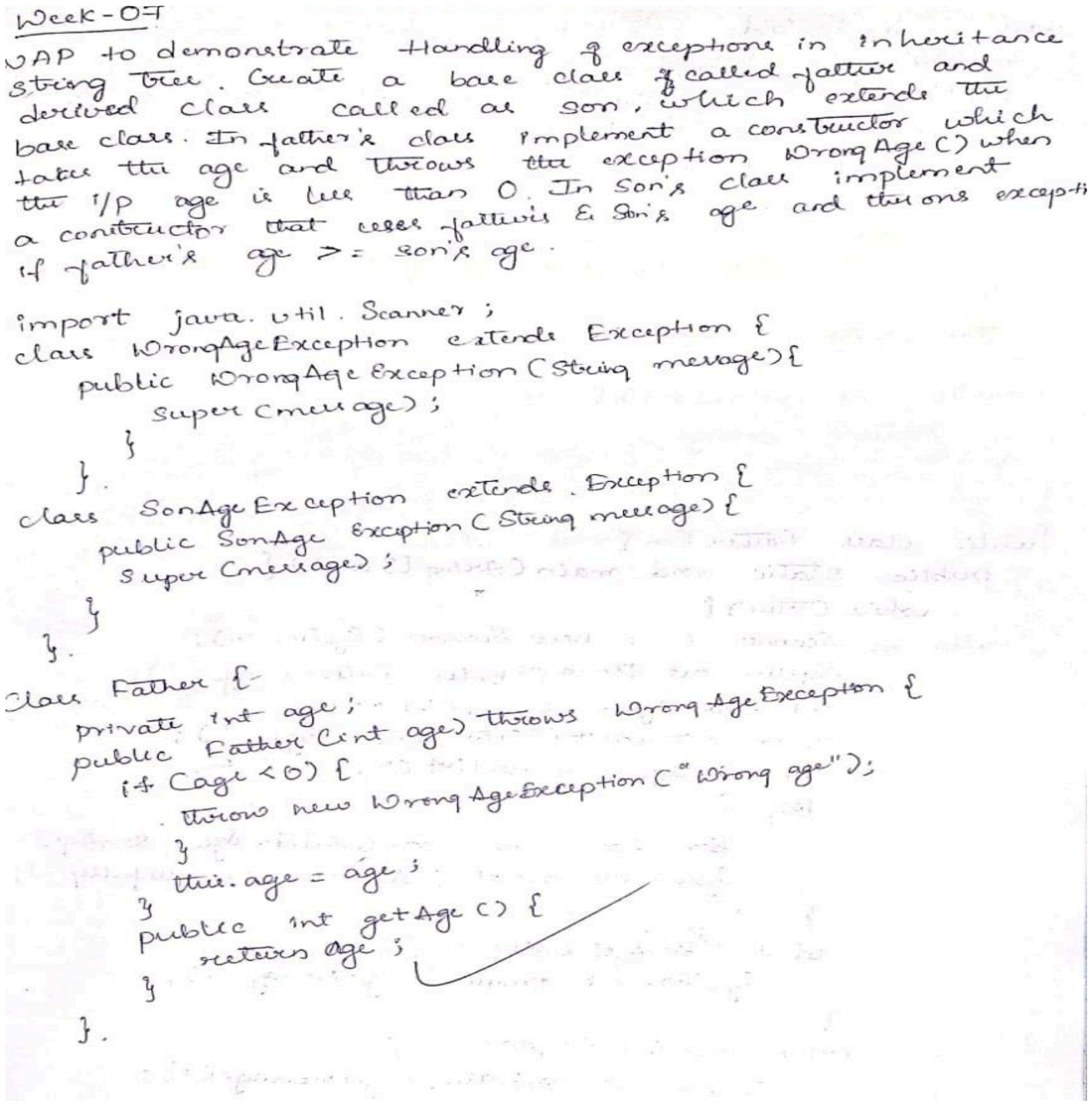


```
Enter number of students: 1
Enter details for CIE Student 1:
USN: 24BECS430
Name: ASHWINI L
Semester: 3
Enter internal marks for 5 courses:
30
34
32
36
35
Enter details for SEE Student 1:
USN: 80
Name: 85
Semester: 86
Enter external marks for 5 courses:
87
Exception in thread "main" java.util.NoSuchElementException
^C
D:\week-06>java Studentmarks.java
Enter number of students: 1
Enter details for CIE Student 1:
USN: 24BECS430
Name: ASHWINI L
Semester: 3
Enter internal marks for 5 courses:
30
34
35
36
32
Enter details for SEE Student 1:
USN: 24BECS430
Name: ASHWINI L
Semester: 3
Enter external marks for 5 courses:
87
86
85
89
82

Final Marks for all students:
USN: 24BECS430, Name: ASHWINI L, Semester: 3
Internal Marks: 30 34 35 36 32
USN: 24BECS430, Name: ASHWINI L, Semester: 3
External Marks: 87 86 85 89 82
Final Marks: 117 120 120 125 114

Ashwini L 24BECS430
```

Program 7

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

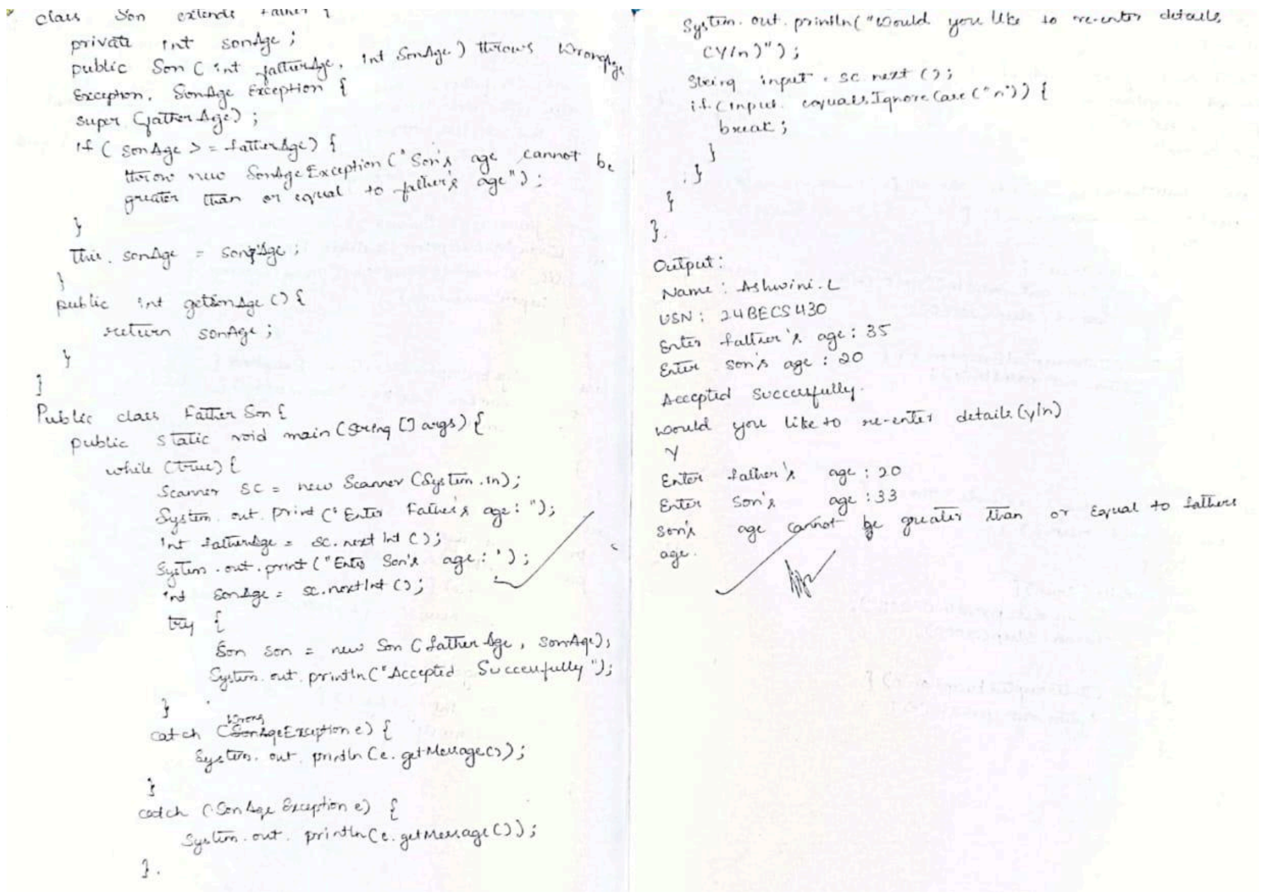
Observation:



Week-07

WAP to demonstrate Handling of exceptions in inheritance string tree. Create a base class of called father and derived class called as son, which extends the base class. In father's class implement a constructor which takes the age and throws the exception WrongAge() when the i/p age is less than 0. In Son's class implement a contructor that uses father's & Son's age and throws exception if father's age >= son's age.

```
import java.util.Scanner;
class WrongAgeException extends Exception {
    public WrongAgeException (String message){
        super (message);
    }
}.
class SonAgeException extends Exception {
    public SonAgeException (String message) {
     Super (message);
    }
}.
class Father {
    private int age;
    public Father (int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException ("Wrong age");
        }
        this.age = age;
    }
    public int getAge () {
        return age;
    }
}.
```



```
class Son extends Father {
  private int sonAge;
  public Son(int fatherAge, int SonAge) throws WrongAge
  Exception, SonAge Exception {
  super (fatherAge);
  if ( sonAge > = fatherAge) {
      throw new SonAgeException ("Son's age cannot be
      greater than or equal to father's age");
  }
  This. sonAge = sonAge;
 }
 public int getSonAge () {
     return sonAge;
  }
}
Public class Father Son {
  public static void main (String [] args) {
     while (true) {
          Scanner SC = new Scanner (System.in);
          System. out. print ("Enter Father's age: ");
          int fatherAge = sc. next Int ();
          System. out. print ("Enter Son's age: ");
          int SonAge = sc.nextInt ();
          try {
              Son son = new Son (father Age, sonAge);
             System. out. println ("Accepted Succeufully");
          }
        catch (~~SonAge~~ Wrong AgeException e) {
            System. out. println (e. getMessage());
        }
       catch (SonAge Exception e) {
          System. out. println (e. getMessage());
       }.
     System. out. println ("Would you like to re-enter details
       (Y/n)");
     String input = sc. next ();
     if (input. equalsIgnoreCase ("n")) {
        break;
      }
   }
  }
}.
```

Output:

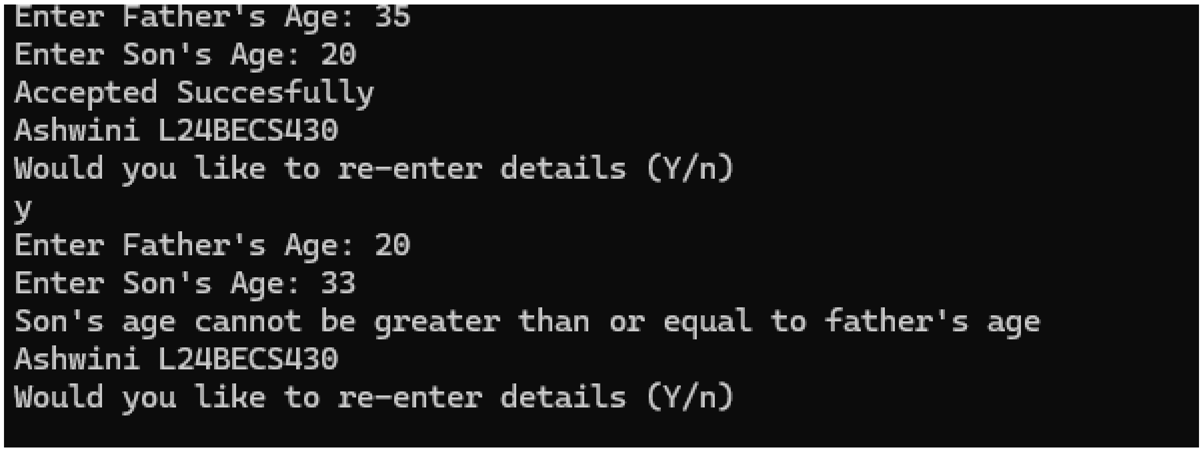
```
Name : Ashwini. L
USN : 24BECS430
Enter father's age : 35
Enter son's age : 20
Accepted Succeufully.
would you like to re-enter details (y/n)
 Y
Enter father's   age : 20
Enter Son's      age : 33
son's  age cannot be greater than or Equal to fathers
age.
```

<u>Code :</u>

```
import java.util.Scanner;
class WrongAgeException extends Exception {
   public WrongAgeException(String message) {
      super(message);
   }
}
class SonAgeException extends Exception {
   public SonAgeException(String message) {
      super(message);
   }
}
class Father {
   private int age;
   public Father(int age) throws WrongAgeException {
      if (age < 0) {
         throw new WrongAgeException("Wrong age");
```

```
        }
        this.age = age;
    }
    public int getAge() {
        return age;
    }
}
class Son extends Father {
    private int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAgeException, SonAgeException {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new SonAgeException("Son's age cannot be greater than or equal to father's age");
        }
        this.sonAge = sonAge;
    }
    public int getSonAge() {
        return sonAge;
    }
}
public class FatherSon{
    public static void main(String[] args) {
        while(true){
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter Father's Age: ");
            int fatherAge = sc.nextInt();
            System.out.print("Enter Son's Age: ");
            int sonAge = sc.nextInt();
            try {
                Son son = new Son(fatherAge, sonAge);
                System.out.println("Accepted Successfully");
            }
            catch (WrongAgeException e) {
                System.out.println(e.getMessage());
            }
            catch (SonAgeException e) {
                System.out.println(e.getMessage());
            }
                System.out.println("Ashwini L"+"24BECS430");
            System.out.println("Would you like to re-enter details (Y/n)");
            String input = sc.next();
            if (input.equalsIgnoreCase("n")) {
                break;
            }
        }
    }
}
```

Output:

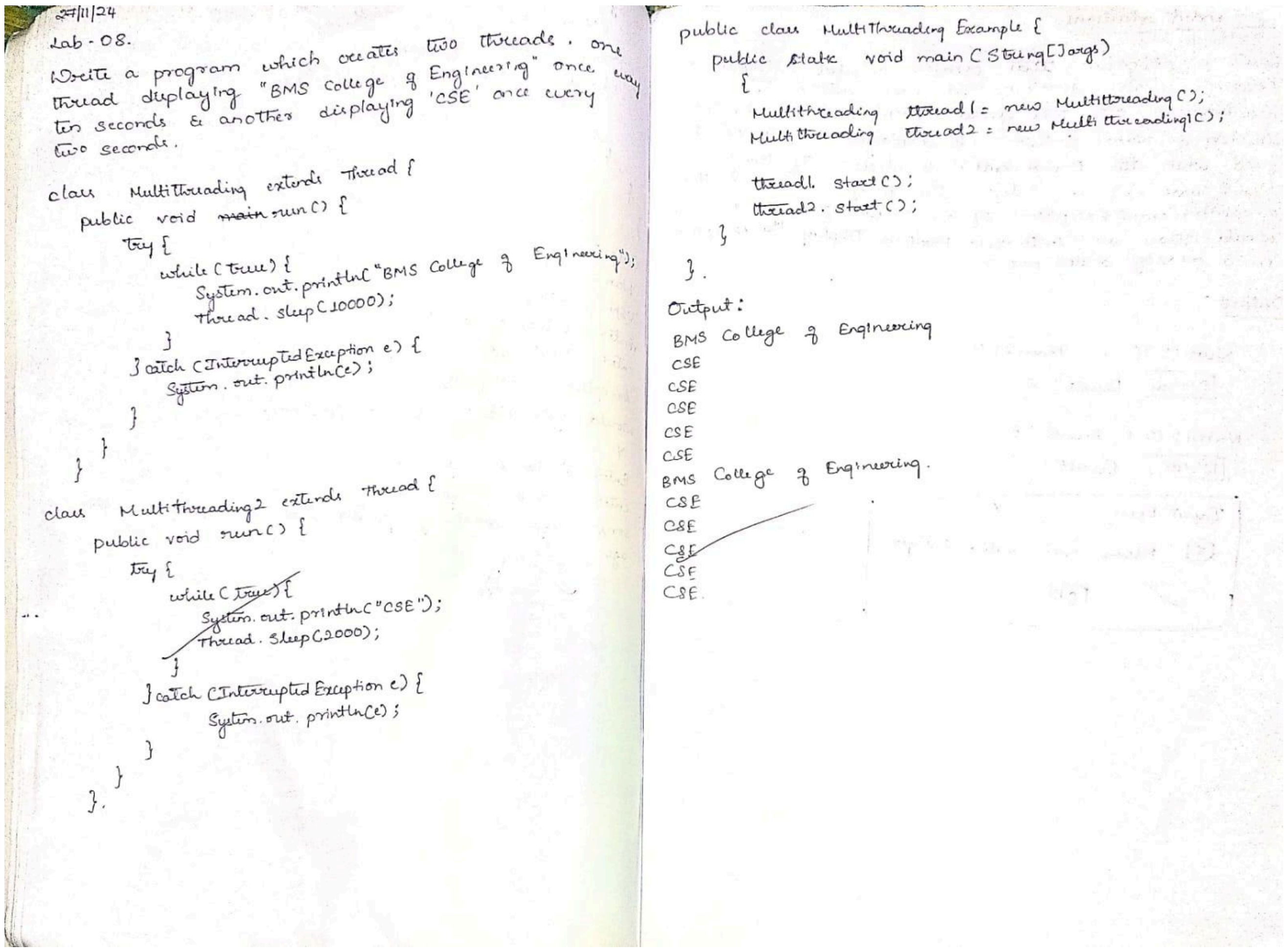


```
Enter Father's Age: 35
Enter Son's Age: 20
Accepted Succesfully
Ashwini L24BECS430
Would you like to re-enter details (Y/n)
y
Enter Father's Age: 20
Enter Son's Age: 33
Son's age cannot be greater than or equal to father's age
Ashwini L24BECS430
Would you like to re-enter details (Y/n)
```

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.

Observation:



27/11/24

Lab - 08.

Write a program which creates two threads. one thread displaying "BMS College of Engineering" once every ten seconds & another displaying 'CSE' once every two seconds.

```
class MultiThreading extends Thread {
   public void ~~main~~ run() {
       try {
          while (true) {
              System.out.println("BMS College of Engineering");
              Thread.sleep(10000);
          }
       } catch (InterruptedException e) {
           System.out.println(e);
       }
    }
 }
class Multithreading2 extends Thread {
    public void run() {
       try {
          while (true) {
              System.out.println("CSE");
              Thread.sleep(2000);
          }
       } catch (InterruptedException e) {
              System.out.println(e);
       }
    }
 }.
```

```
public class MultiThreading Example {
   public static void main (String[]args)
     {
       Multithreading thread1 = new Multithreading();
       Multithreading thread2 = new Multithreading1();

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

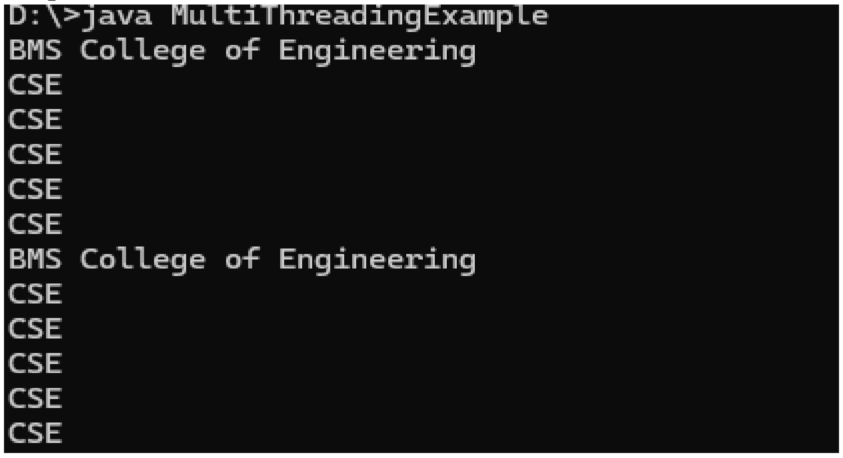
Output:

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

<u>Code:</u>

```
class DisplayMessage1 extends Thread {
   public void run() {
      try {
         while (true) {
            System.out.println("BMS College of Engineering");
            Thread.sleep(10000);
         }
      } catch (InterruptedException e) {
            System.out.println(e);
        }
    }
}
class DisplayMessage2 extends Thread {
    public void run() {
        try {
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }
}
public class MultiThreadingExample {
    public static void main(String[] args) {
        DisplayMessage1 thread1 = new DisplayMessage1();
        DisplayMessage2 thread2 = new DisplayMessage2();
        thread1.start();
        thread2.start();
    }}
```

Output:

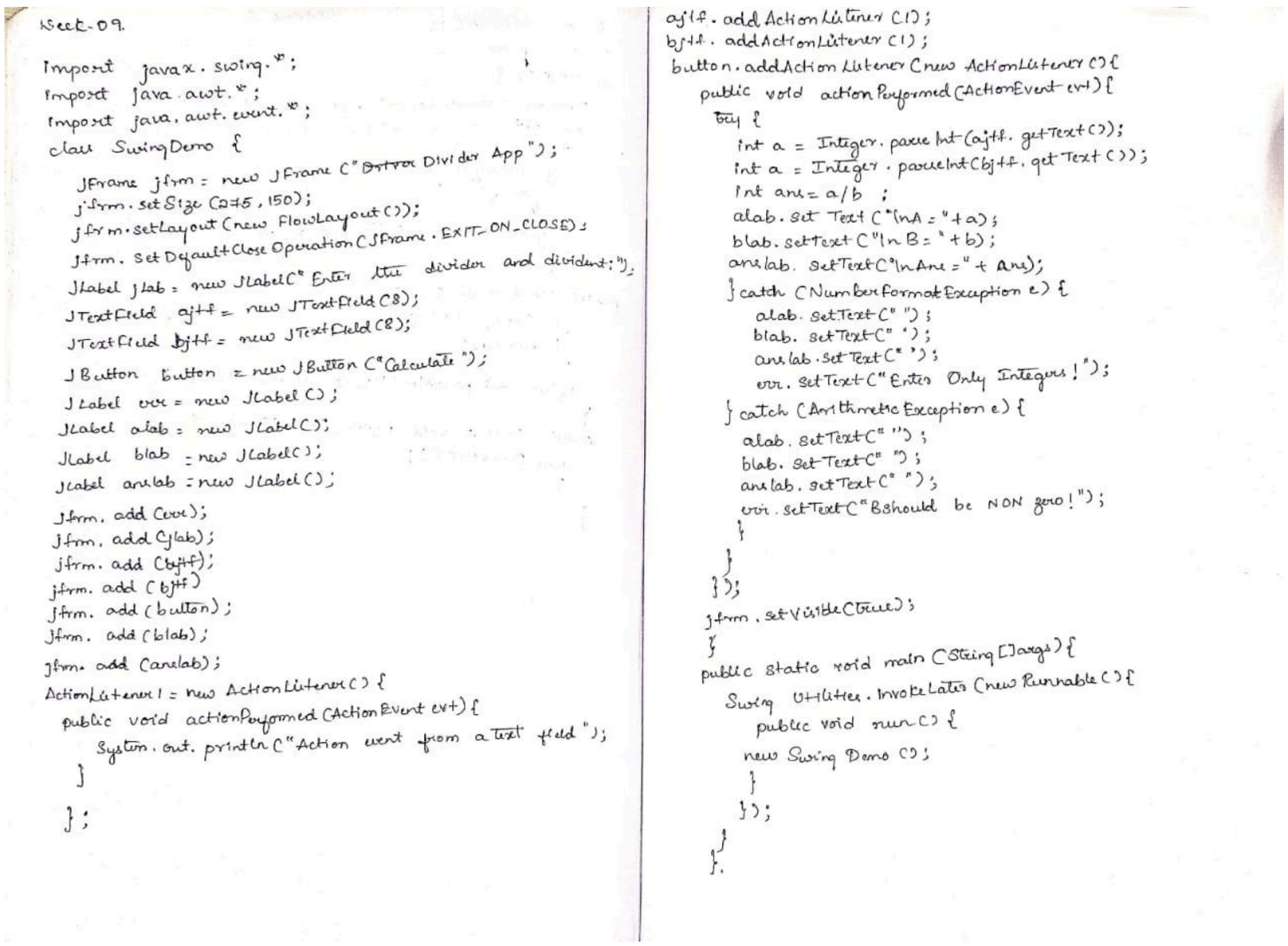


```
D:\>java MultiThreadingExample
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
```

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.

Observation:



```
Week-09.

Import  javax. swing.*;
Import  java.awt.*;
Import  java.awt.event.*;
class  SwingDemo  {
    JFrame jfrm = new JFrame ("~~Error~~ Divider App");
    jfrm. setSize (275, 150);
    jfrm.setLayout (new FlowLayout ());
    jfrm. SetDefaultCloseOperation (JFrame. EXIT_ON_CLOSE);
    JLabel jlab = new JLabel (" Enter the divider and divident:");
    JTextField  ajtf = new JTextField (8);
    JTextField bjtf = new JTextField (8);
    JButton  button  = new JButton ("Calculate");
    JLabel err = new JLabel ();
   JLabel alab = new JLabel();
   JLabel  blab  = new JLabel();
   JLabel  anslab = new JLabel ();
   Jfrm. add (err);
   jfrm. add (jlab);
   jfrm. add (ajtf);
  jfrm. add (bjtf)
  jfrm. add (button);
  jfrm. add (blab);
  jfrm. add (anslab);
  ActionListener l = new ActionListener () {
    public void actionPerformed (ActionEvent evt) {
       System. out. println ("Action event from a text field");
    }
   };
ajtf. addActionListener (l);
bjtf. addActionListener (l);
button. addActionListener (new ActionListener () {
   public void actionPerformed (ActionEvent evt) {
    try {
      int a = Integer. parseInt (ajtf. getText ());
      int a = Integer . parseInt (bjtf. get Text ());
      int ans = a/b  ;
      alab. set Text ("\nA = " + a);
      blab. setText ("\n B = " + b);
     anslab. setText ("\nAns = " + Ans);
     } catch (NumberFormatException e) {
       alab. setText (" ");
       blab. setText (" ");
       anslab. Set Text (" ");
       err. setText ("Enter Only Integers!");
    } catch (ArithmeticException e) {
      alab. setText (" ");
      blab. Set Text (" ");
      anslab. setText (" ");
      err. setText ("Bshould be NON zero!");
      }
    }
  });
jfrm . setVisible (true);
}
public static void main (String []args) {
  Swing Utilities. invokeLater (new Runnable () {
     public void run () {
    new Swing Demo ();
     }
    });
 }
}.
```

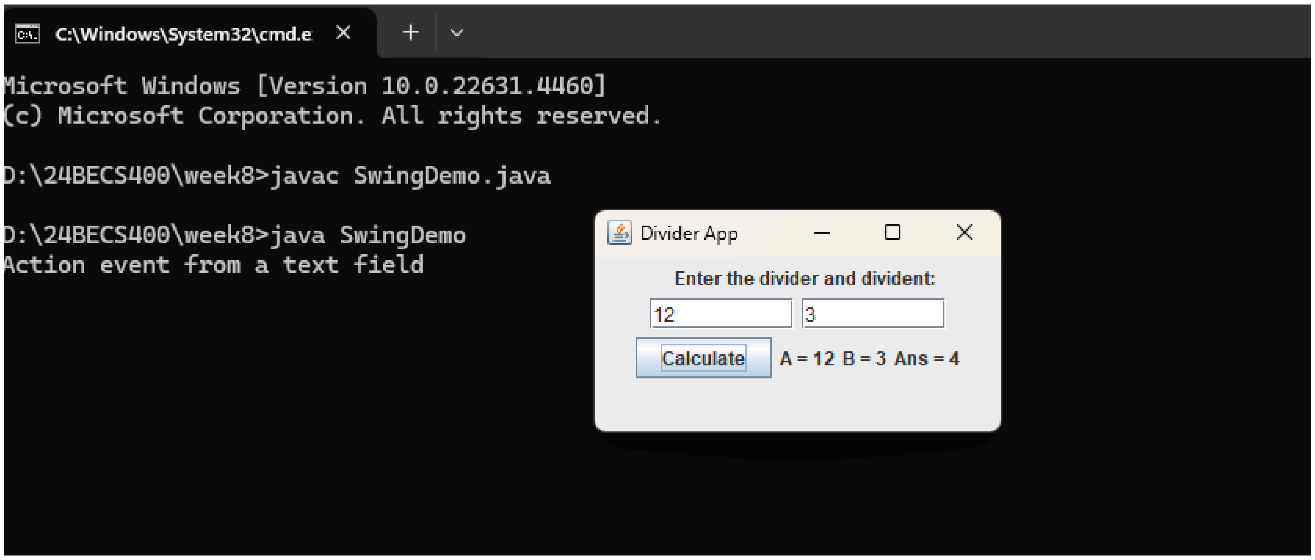
Code:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.*;

class SwingDemo {

  SwingDemo() {

    // create jframe container

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

    jfrm.setSize(275, 150);

    jfrm.setLayout(new FlowLayout());

    // to terminate on close

    jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    // text label

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

    // add text field for both numbers

    JTextField ajtf = new JTextField(8);

    JTextField bjtf = new JTextField(8);

    // calc button

    JButton button = new JButton("Calculate");

    // labels

    JLabel err = new JLabel();

    JLabel alab = new JLabel();

    JLabel blab = new JLabel();

    JLabel anslab = new JLabel();
```

```
// add in order :)
jfrm.add(err); // to display error bois
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);
ActionListener l = new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
      System.out.println("Action event from a text field");
   }
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
      try {
         int a = Integer.parseInt(ajtf.getText());
         int b = Integer.parseInt(bjtf.getText());
         int ans = a / b;
         alab.setText("\nA = " + a);
         blab.setText("\nB = " + b);
         anslab.setText("\nAns = " + ans);
```

```
            } catch (NumberFormatException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("Enter Only Integers!");
            } catch (ArithmeticException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("B should be NON zero!");
            }
        }
    });
    // display frame
    jfrm.setVisible(true);
  }
  public static void main(String args[]) {
    // create frame on event dispatching thread
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            new SwingDemo();
        }
    });
  }
}
```

Output:



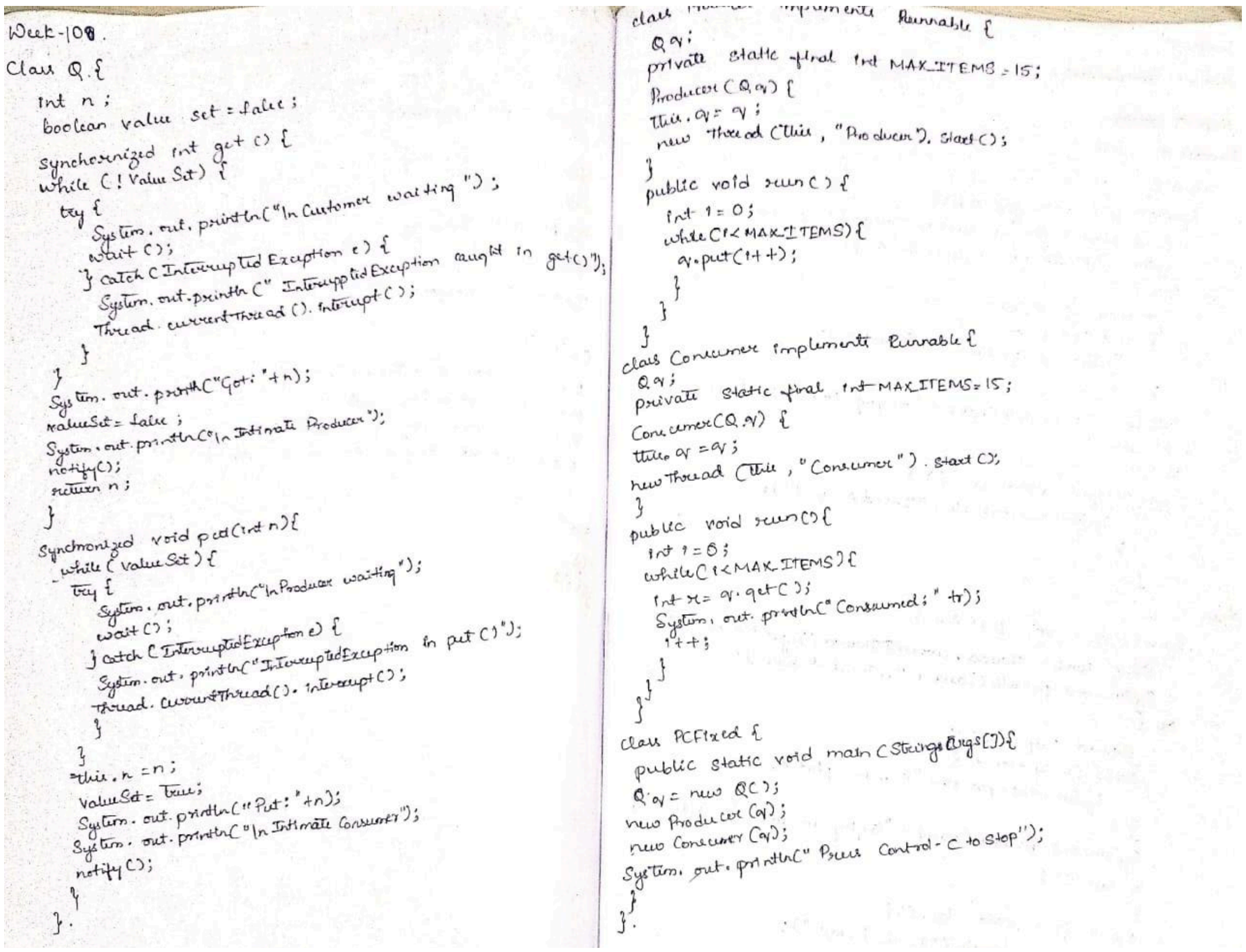
```
Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

D:\24BECS400\week8>javac SwingDemo.java

D:\24BECS400\week8>java SwingDemo
Action event from a text field
```

```
Divider App
Enter the divider and divident:
12    3
Calculate   A = 12  B = 3  Ans = 4
```

Program 10

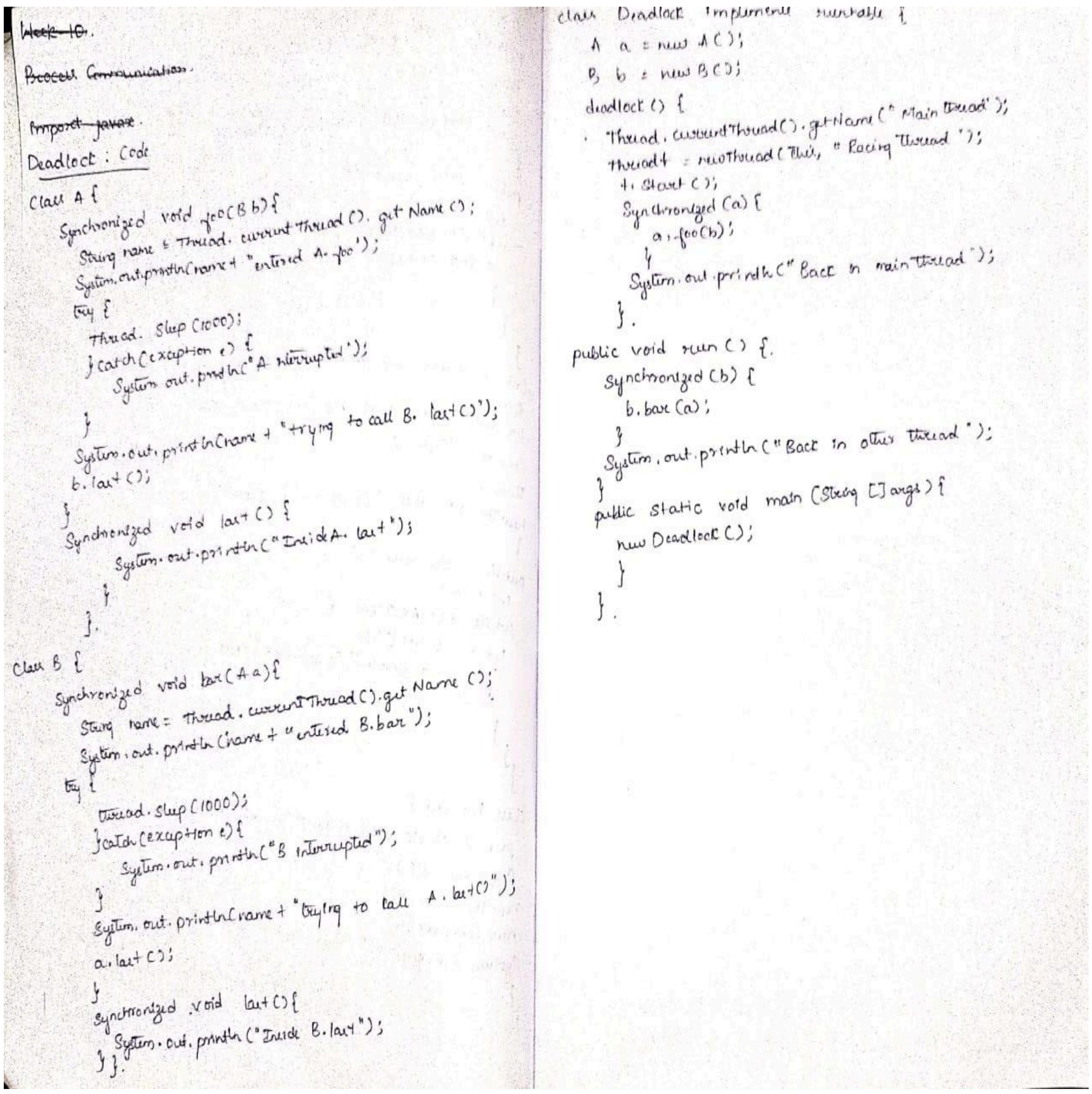
Demonstrate Inter process Communication and deadlock.

Observation:



```
Week-10.
Class Q.{
  int n;
  boolean value set = false;
 synchronized int get () {
 while (! Value Set) {
   try {
     System. out. println("In Customer waiting");
     wait ();
   } catch (InterruptedException e) {
     System. out. println (" InterruptedException caught in get()");
     Thread. currentThread (). interrupt ();
   }
  }
  System. out. println("Got: "+n);
  valueSet = false;
  System. out. println("In Intimate Producer");
  notify();
  return n;
 }
 Synchronized void put(int n){
  while ( value Set ) {
   try {
     System. out. println("In Producer waiting");
     wait ();
   } catch (InterruptedException e) {
     System. out. println("InterruptedException in put ()");
     Thread. currentThread(). interrupt ();
     }
   }
   this. n = n;
   valueSet = True;
   System. out. println("Put: "+n);
   System. out. println("In Intimate Consumer");
   notify ();
  }
 }.
```

```
class [illegible] implements Runnable {
  Q q;
  private static final int MAX_ITEMS = 15;
  Producer (Q q) {
  This. q = q;
  new Thread (this, "Producer"). start ();
  }
  public void run () {
   int i = 0;
   while (i< MAX_ITEMS) {
    q.put(i++);
    }
   }
  }
class Consumer implements Runnable {
  Q q;
 private static final int MAX_ITEMS = 15;
 Consumer(Q q) {
 this. q = q;
 new Thread (this, "Consumer") . start ();
 }
 public void run(){
  int i = 0;
  while(i<MAX_ITEMS){
   int x = q. get ();
   System. out. println(" Consumed: " +x);
    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");
 }
}.
```



~~Week 10~~.

~~Process Communication~~.

~~Import javax~~.

Deadlock : Code

```
Class A {
  Synchronized void foo(B b){
    String name = Thread. current Thread (). get Name ();
    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("InsideA. last");
     }
   }.

Class B {
  Synchronized void bar(A a){
    String name= Thread. currentThread().get Name ();
    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 B.last");
    }}.
```

```
class Deadlock implements runnable {
  A a = new A ();
  B b = new B ();
  deadlock () {
  . Thread. currentThread().getName (" Main thread");
    threadt = newThread (This, " Racing thread ");
     t. Start ();
     Synchronized (a) {
       a.foo(b);
      }
     System.out.println(" Back in main thread ");
    }.
public void run () {.
   synchronized (b) {
     b.bar (a);
   }
  System.out.println (" Back in other thread ");
 }
 public static void main (String [] args) {
   new Deadlock ();
   }
 }.
```

<u>Code:</u>

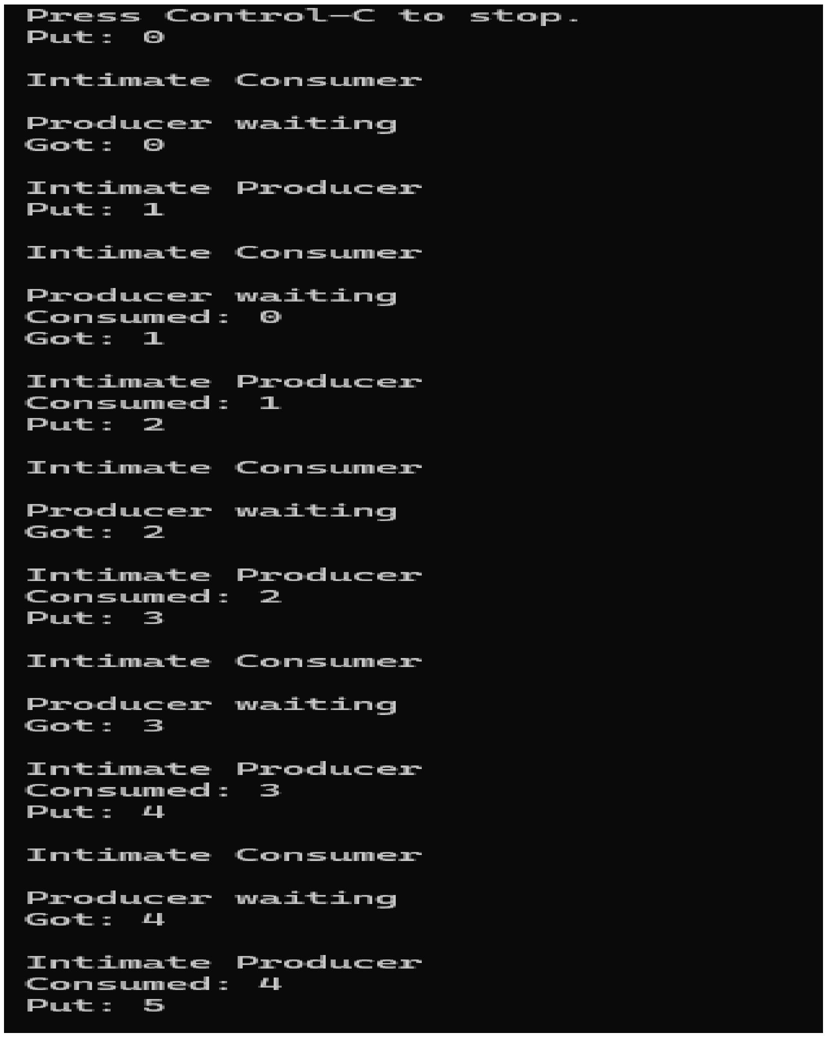
-Process communication

```
class Q {
 int n;
    boolean valueSet = false;
    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("\nConsumer waiting");
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught in get()");
                Thread.currentThread().interrupt();
            }
        }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\nIntimate Producer");
        notify();
        return n;
    }
    synchronized void put(int n) {
        while (valueSet) {
            try {
                System.out.println("\nProducer waiting");
                wait();  // Producer waits if value has already been set
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught in put()");
                Thread.currentThread().interrupt();
            }
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\nIntimate Consumer");
        notify();
    }
}
class Producer implements Runnable {
    Q q;
    private static final int MAX_ITEMS = 15;
    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run() {
        int i = 0;
        while (i < MAX_ITEMS) {
            q.put(i++);
        }
    }
}
class Consumer implements Runnable {
    Q q;
    private static final int MAX_ITEMS = 15;
    Consumer(Q q) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void run() {
        int i = 0;
        while (i < MAX_ITEMS) {  // Consume only up to MAX_ITEMS
            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.");
    }
}
```

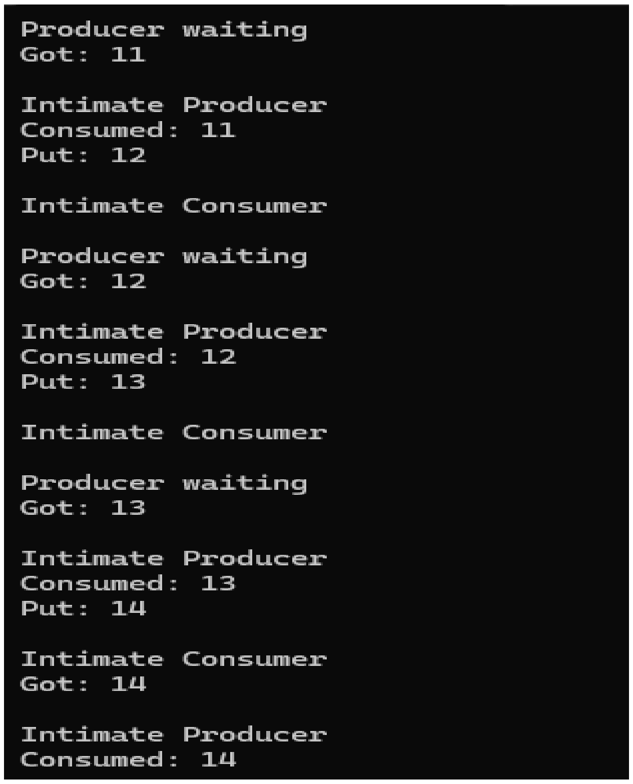
- Deadlock

```
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 B.last");
    }
}
class Deadlock implements Runnable {
    A a = new A();
    B b = new B();
    Deadlock() {
        Thread.currentThread().setName("MainThread");
        Thread t = new Thread(this, "RacingThread");
        t.start();
        // Ensure that main thread always calls a.foo(b)
        synchronized (a) {  // Lock a before b to avoid circular waiting
            a.foo(b); // get lock on a in this
        }
        System.out.println("Back in main thread");
    }
    public void run() {
        // Ensure that the other thread always calls b.bar(a)
        synchronized (b) {  // Lock b before a to avoid circular waiting
            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();
    }
}
```

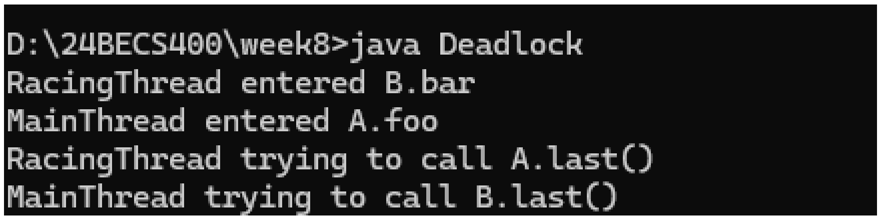
Output:



```
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
```



```
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:\24BECS400\week8>java Deadlock
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
MainThread trying to call B.last()
```