VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Gayathri S (24BECE417)

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 **Gayathri S (24BECS417),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a Database Management Systems (23CS3PCDBM) work prescribed for the said degree.

Lab faculty Swathi Sridharan Assistant Professor Department of CSE, BMSCE	Dr. Joythi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link:

“https://github.com/GayathriS-CSE/OOJ”

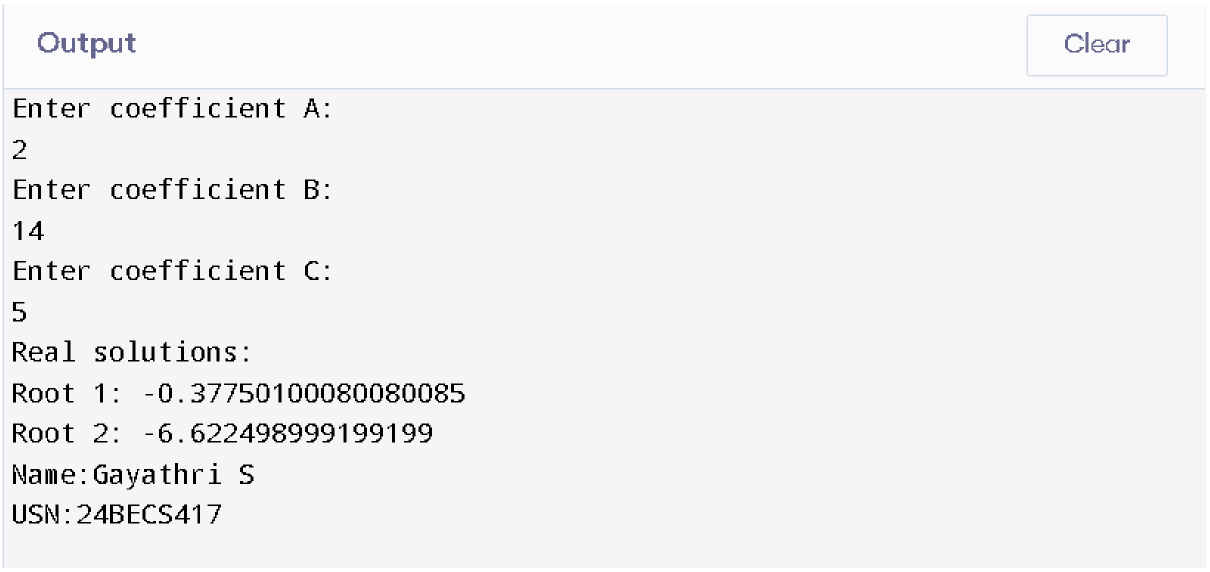
Lab program-1

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.

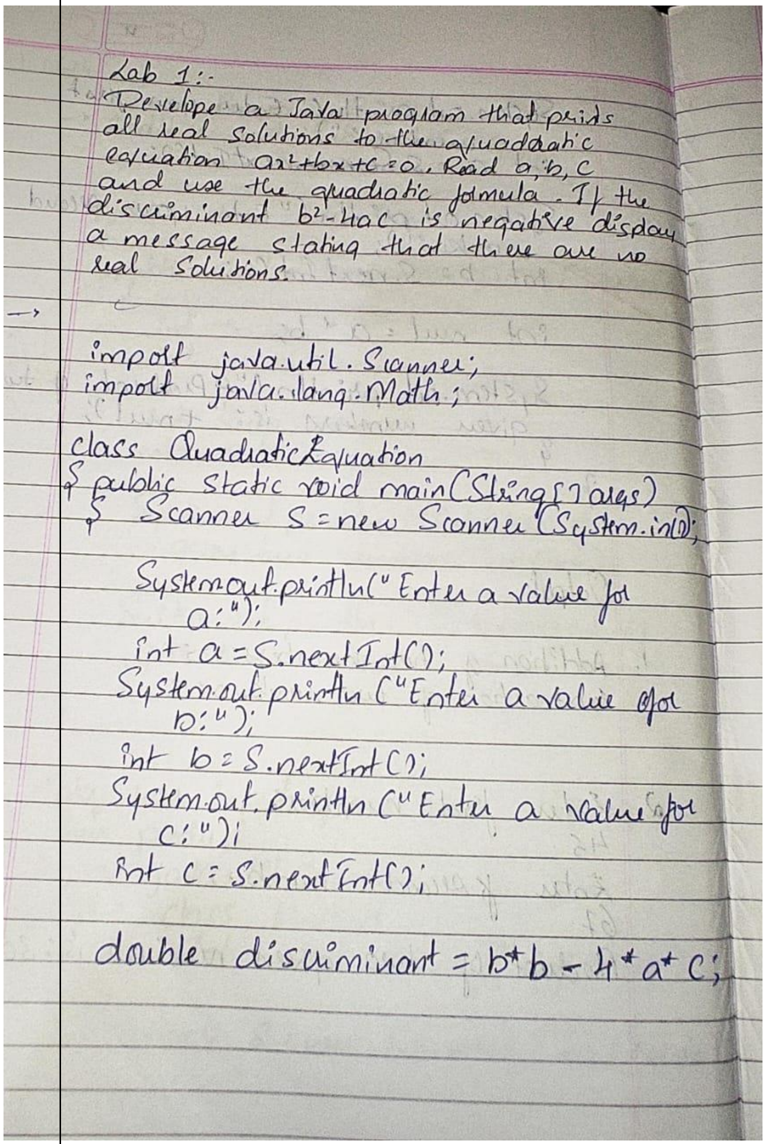
CODE:

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

public class QuadraticEquationSolver {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter coefficient A:");
        double a = scanner.nextDouble();
        System.out.println("Enter coefficient B:");
        double b = scanner.nextDouble();
        System.out.println("Enter coefficient C:");
        double c = scanner.nextDouble();

        double discriminant = b * b - 4 * a * c;

        if (discriminant >= 0) {
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);

            if (discriminant == 0) {
                System.out.println("One real solution: " + root1);
            } else {
                System.out.println("Real solutions:");
                System.out.println("Root 1: " + root1);
                System.out.println("Root 2: " + root2);
            }
        } else {
            System.out.println("No real solutions.");
        }
        scanner.close();
    }
}
```

OUTPUT:



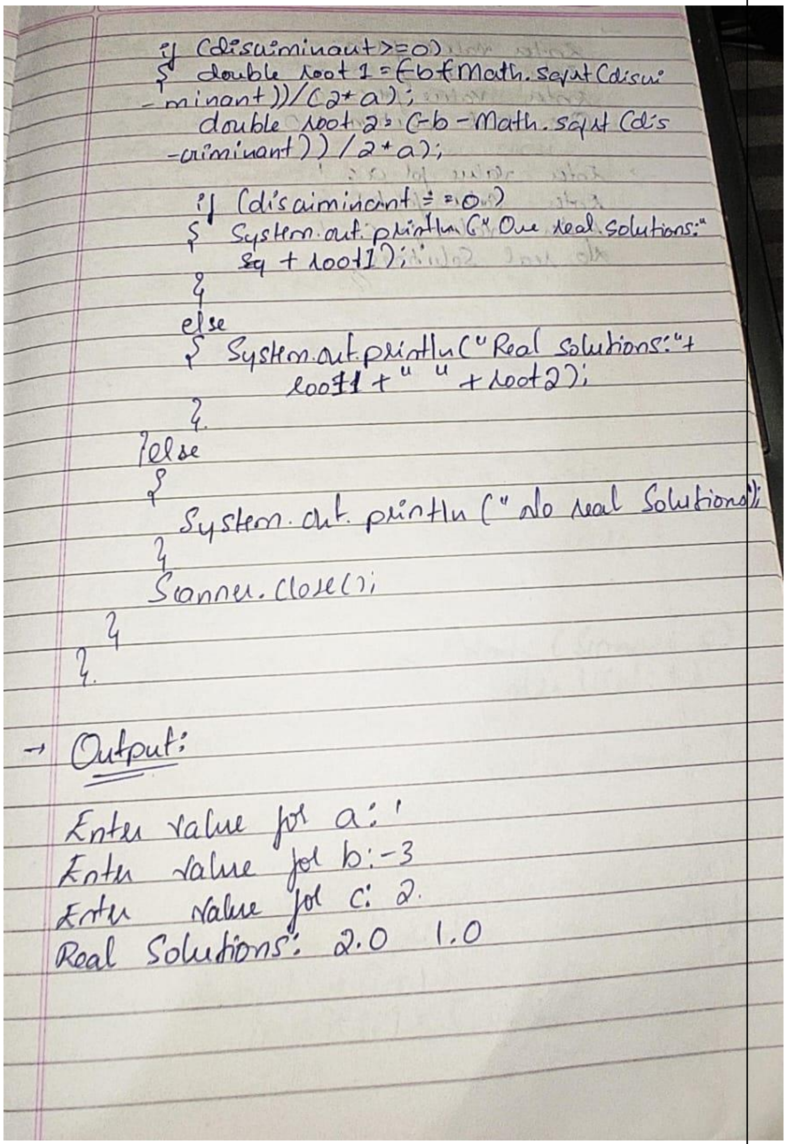
```
Enter coefficient A:
2
Enter coefficient B:
14
Enter coefficient C:
5
Real solutions:
Root 1: -0.37750100080080085
Root 2: -6.622498999199199
Name:Gayathri S
USN:24BECS417
```



Lab 1:-

Develope a Java program that prints all real solutions to the quadadratic equation $ax^2+bx+c=0$. Read a, b, c and use the quadriatic formula. If the discriminant b^2-4ac is negative display a message stating that there are no real solutions.

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

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

     System.out.println("Enter a value for
        a:");
     int a = S.nextInt();
    System.out.println("Enter a value for
        b:");
    int b = S.nextInt();
    System.out.println("Enter a value for
       c:");
    int c = S.nextInt();

  double discriminant = b*b - 4*a*c;
```



```
      if (discriminant>=0)
      { double root 1 = (-b+Math.sqrt(discri
      -minant))/(2*a);
         double root 2 = (-b - Math.sqrt(dis
      -criminant))/(2*a);

         if (discriminant = = 0)
         { System.out.println("One real solutions:"
              sq + root1);
         }
        else
        { System.out.println("Real solutions:"+
                  root1 + "  " + root2);
        }
     }else
     {
        System.out.println("No real Solutions");
     }
     Scanner.close();
  }
}.
```

→ Output:

```
Enter value for a: 1
Enter value for b: -3
Enter value for c: 2
Real Solutions: 2.0  1.0
```

Lab program-2

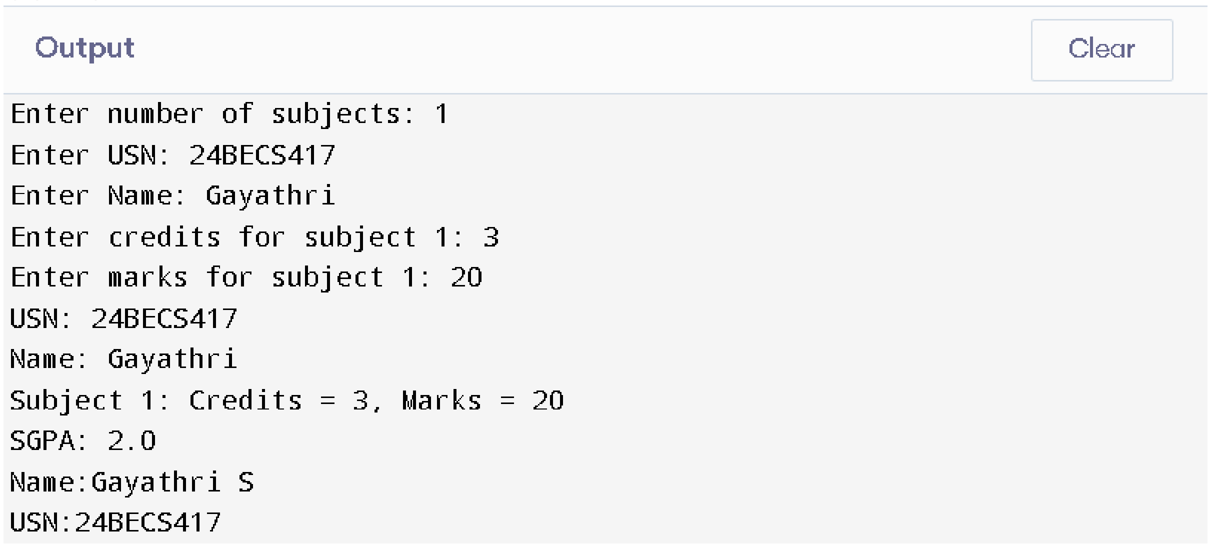
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

CODE:

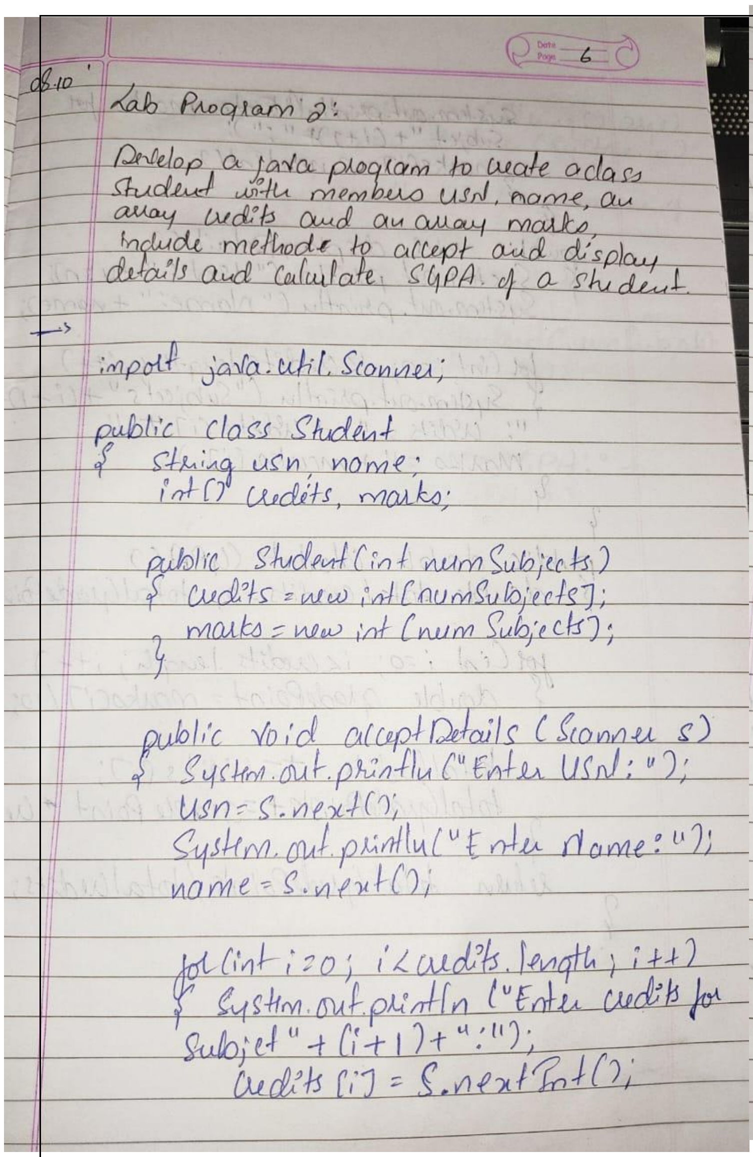
```
import java.util.Scanner;

public class Student {
   String usn, name;
   int[] credits, marks;

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

   public void acceptDetails(Scanner scanner) {
      System.out.print("Enter USN: ");
      usn = scanner.next();
      System.out.print("Enter Name: ");
      name = scanner.next();

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

   public void displayDetails() {
      System.out.println("USN: " + usn);
      System.out.println("Name: " + name);

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


   public double calculateSGPA() {
      double totalCredits = 0, totalGradePoints = 0;

      for (int i = 0; i < credits.length; i++) {
         double gradePoint = marks[i] / 10;

         totalCredits += credits[i];
         totalGradePoints += gradePoint * credits[i];
```

```
        }

        return totalGradePoints / totalCredits;
    }

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

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

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

        System.out.println("SGPA: " + student.calculateSGPA());
    }
}
```

OUTPUT:



```
Enter number of subjects: 1
Enter USN: 24BECS417
Enter Name: Gayathri
Enter credits for subject 1: 3
Enter marks for subject 1: 20
USN: 24BECS417
Name: Gayathri
Subject 1: Credits = 3, Marks = 20
SGPA: 2.0
Name:Gayathri S
USN:24BECS417
```



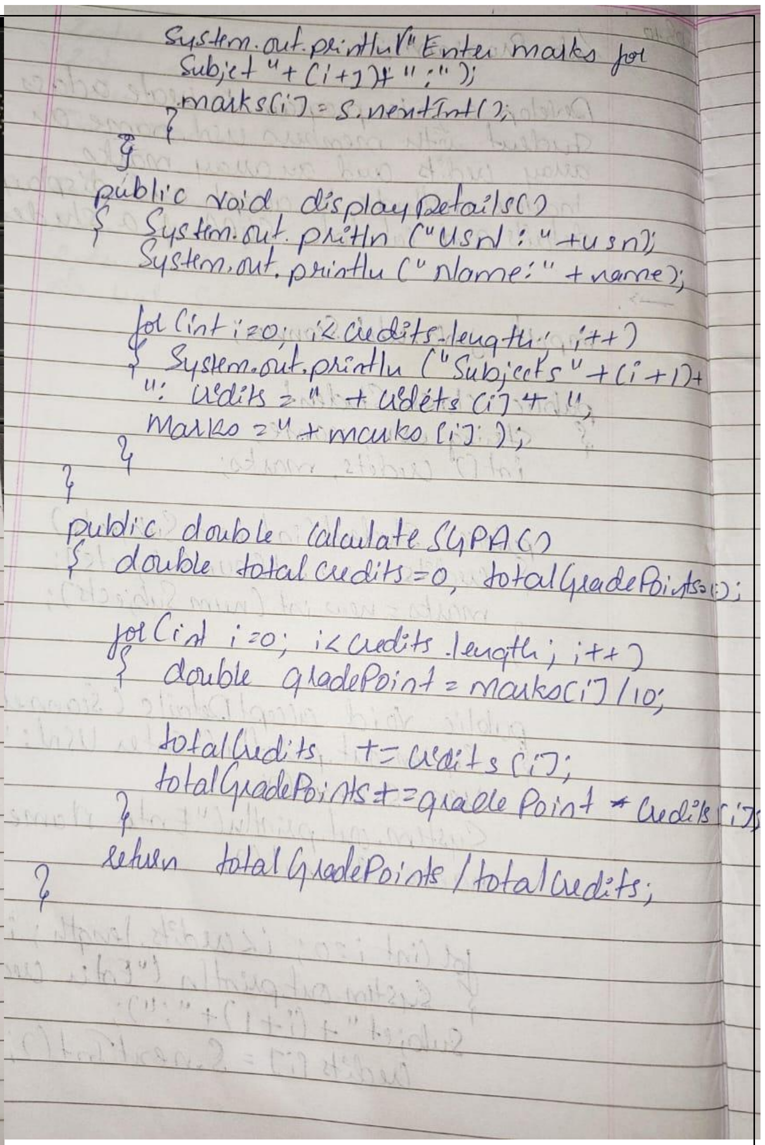
08.10

Lab Program 2:

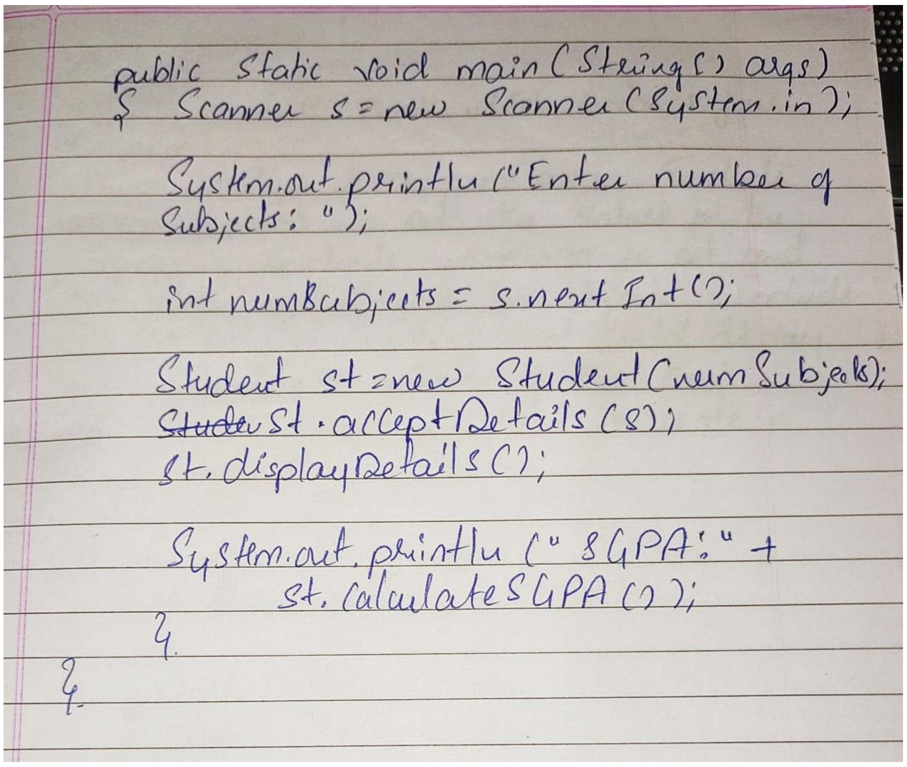
Develop a java program to create aclass Student with members usn, name, an array credits and an array marks, include methods to accept and display details and calculate SGPA of a student.

→

```
import java.util.Scanner;

public class Student
{ String usn name;
  int[] credits, marks;

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

  public void acceptDetails (Scanner s)
  { System.out.println("Enter USN: ");
    usn= s.next();
    System.out.println("Enter Name: ");
    name = s.next();

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



```
        System.out.println("Enter marks for
        Subject "+(i+1)+":");
        marks[i] = s.nextInt();
      }
  }
  public void displayDetails()
  { System.out.pritln ("USN : "+usn);
    System.out.println ("Name: " + name);

    for(int i=0; i<credits.length; i++)
    { System.out.println ("Subjects"+(i+1)+
    ": credits = " + credits [i] + ",
    Marks =" + marks [i] );
    }
  }
  public double calculate SGPA()
  { double totalcredits=0, totalGradePoints=0;

    for(int i=0; i<credits.length; i++)
    { double gradePoint = marks[i]/10;

      totalcredits += credits [i];
      totalGradePoints+=gradePoint * credits[i];
    }
    return totalGradePoints / totalcredits;
  }
```



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

  System.out.println("Enter number of
  Subjects: ");

  int numSubjects = s.next Int();

  Student st=new Student(numSubjects);
  ~~Student~~ st.acceptDetails (s);
  st.displayDetails();

  System.out.println (" SGPA:" +
          st.calculateSGPA());
  }
}
```

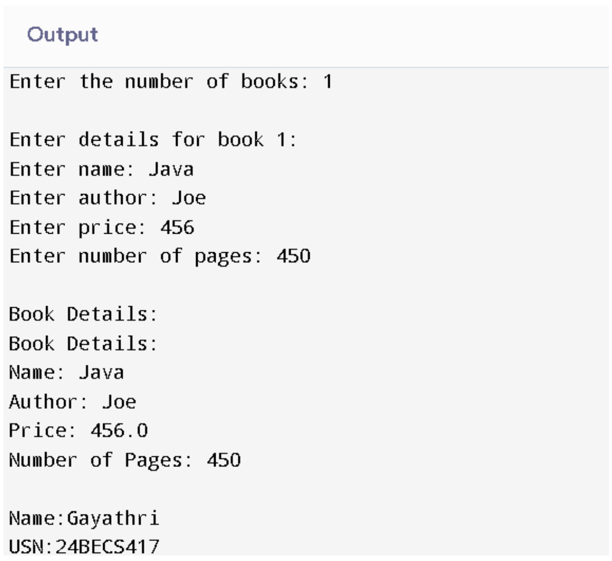
Lab program-3

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.

CODE:

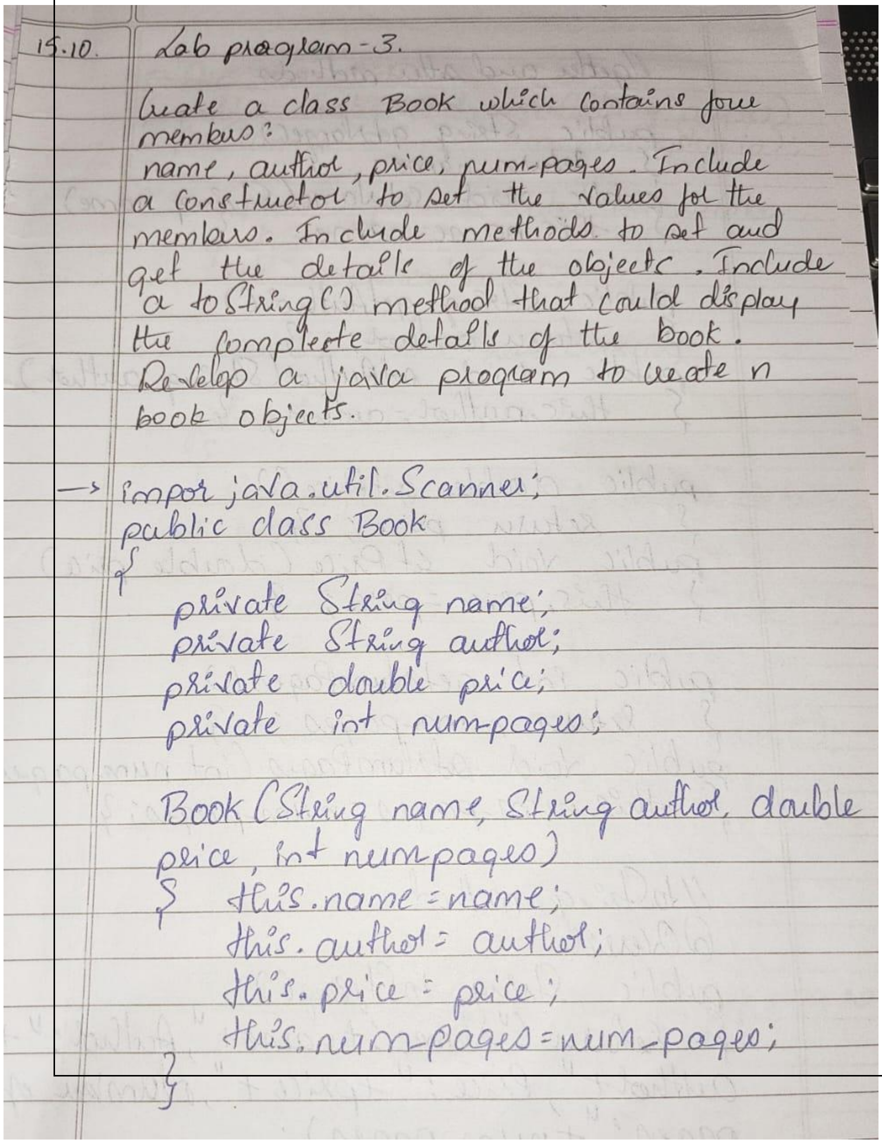
```
import java.util.Scanner;

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 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 setNumPages(int numPages) {
        this.numPages = numPages;
    }

    public String getName() {
        return name;
    }

    public String getAuthor() {
        return author;
    }

    public double getPrice() {
        return price;
    }

    public int getNumPages() {
        return numPages;
    }

    @Override
    public String toString() {
        return "Book Details:\n" +
                "Name: " + name + "\n" +
                "Author: " + author + "\n" +
                "Price: " + price + "\n" +
                "Number of Pages: " + numPages + "\n";
    }
}

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

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

        Book[] books = new Book[n];

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

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

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

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

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

            books[i] = new Book(name, author, price, numPages);
        }
        System.out.println("\nBook Details:");
        for (Book book : books) {
            System.out.println(book);
        }
    }
}
```

OUTPUT:



Output

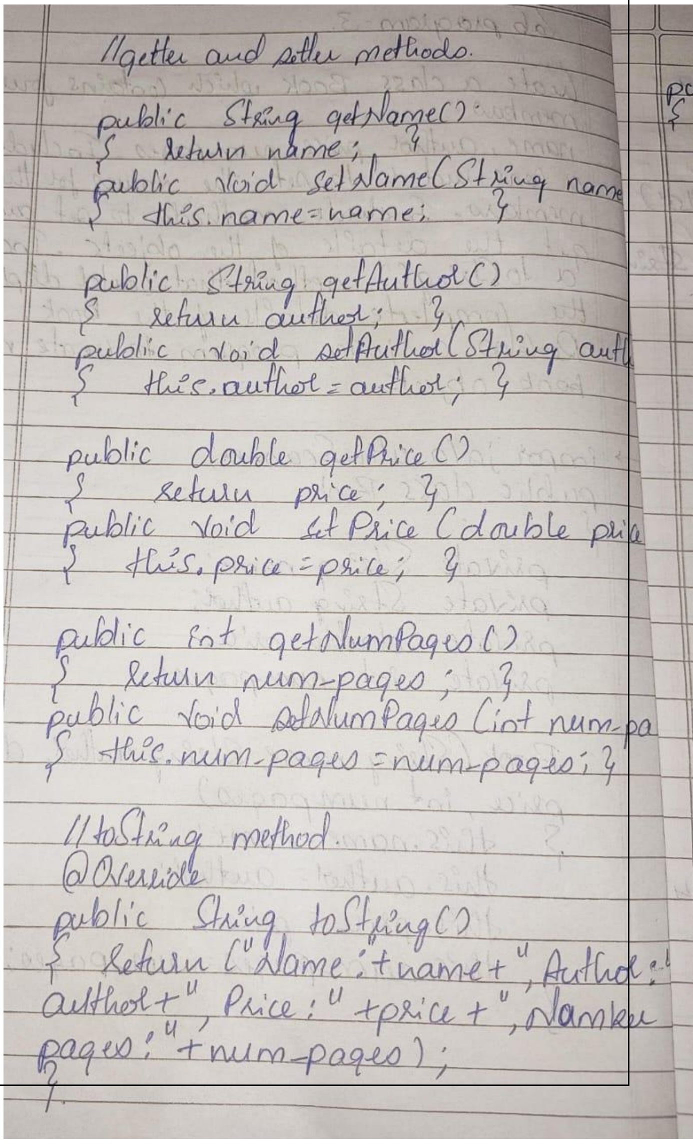
```
Enter the number of books: 1

Enter details for book 1:
Enter name: Java
Enter author: Joe
Enter price: 456
Enter number of pages: 450

Book Details:
Book Details:
Name: Java
Author: Joe
Price: 456.0
Number of Pages: 450

Name:Gayathri
USN:24BECS417
```



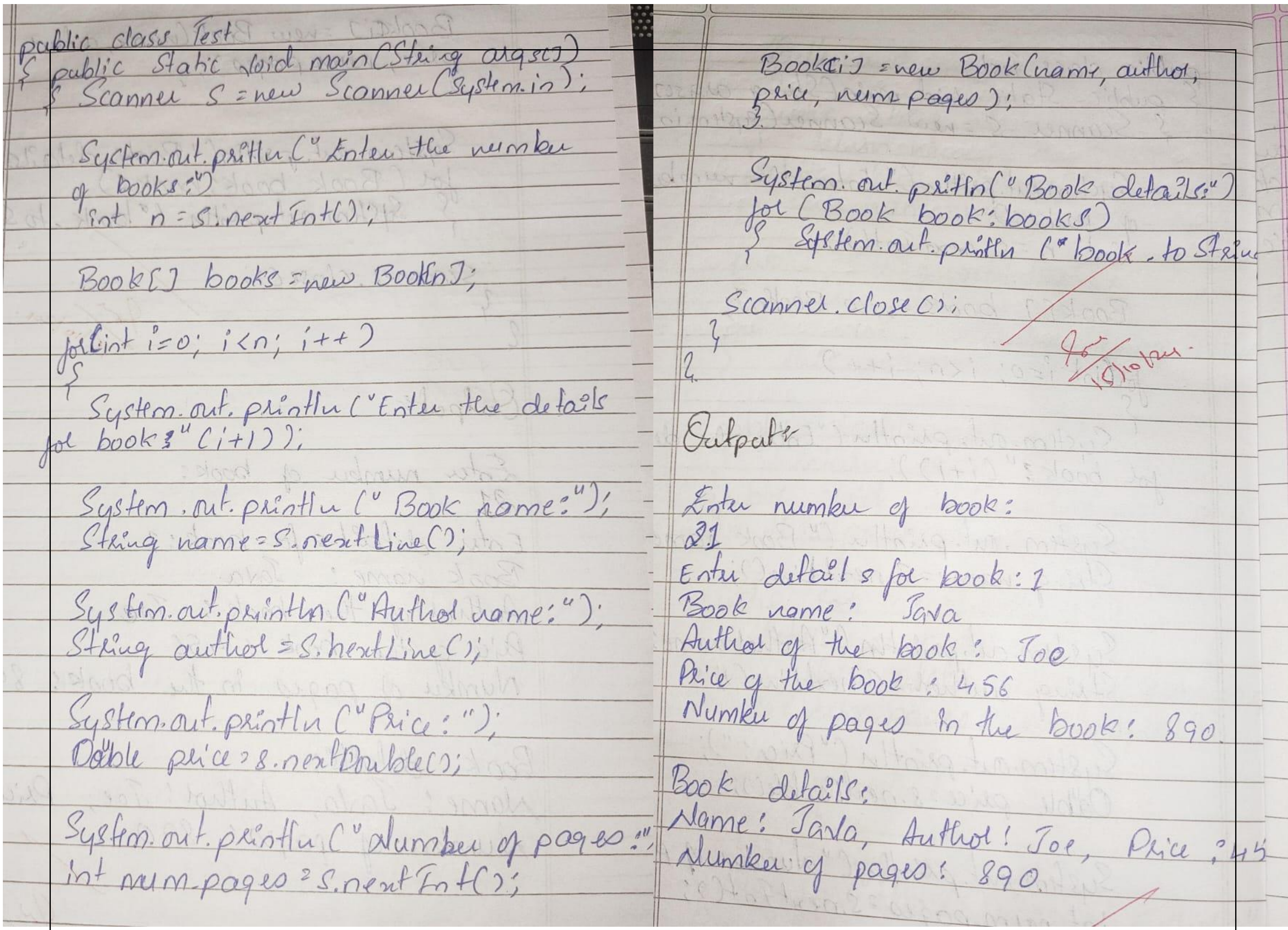
15.10. Lab program-3.

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.

```
→ impor java.util.Scanner;
public class Book
{
    private String name;
    private String author;
    private double price;
    private int num-pages;

    Book(String name, String author, double
    price, int numpages)
    {   this.name=name;
        this.author= author;
        this.price = price;
        this.num-pages=num-pages;
    }
```



```
//getter and setter methods.

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

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

public double getPrice()
{   return price;   }
public void set Price (double pric
{   this.price=price;   }

public int getNumPages()
{   Return num-pages;   }
public void setNumPages (int num-pa
{   this.num-pages=num-pages;   }

//toString method.
@Override
public String toString()
{   Return ("Name:"+name+", Author:"
author+", Price:"+price+", Number
pages:"+num-pages);
}
```



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

System.out.println("Enter the number of books:");
int n = s.nextInt();

Book[] books = new Book[n];

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

System.out.println("Book name:");
String name=s.nextLine();

System.out.println("Author name:");
String author=s.nextLine();

System.out.println("Price:");
Double price=s.nextDouble();

System.out.println("Number of pages:");
int num_pages=s.nextInt();

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

System.out.println("Book details:")
for (Book book:books)
{ System.out.println ("book.toStri
}
Scanner.close();
}
}
```

[illegible]

Output

```
Enter number of book:
1
Enter details for book: 1
Book name:  Java
Author of the book: Joe
Price of the book: 456
Number of pages in the book: 890.

Book details:
Name: Java, Author: Joe, Price: 45
Number of pages: 890.
```

Lab program-4

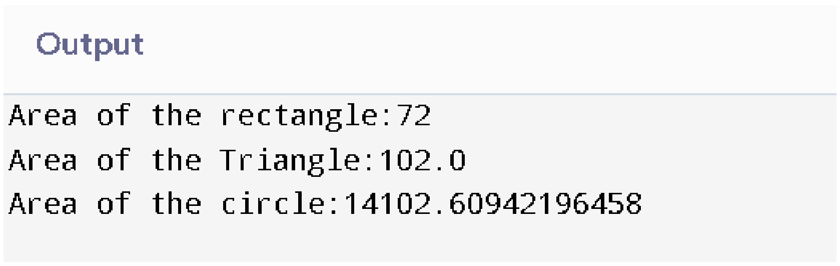
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea(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

CODE:

```
import java.lang.Math;

abstract class Shape
{
   protected int val1;
   protected int val2;

   Shape(int val1,int val2)
```

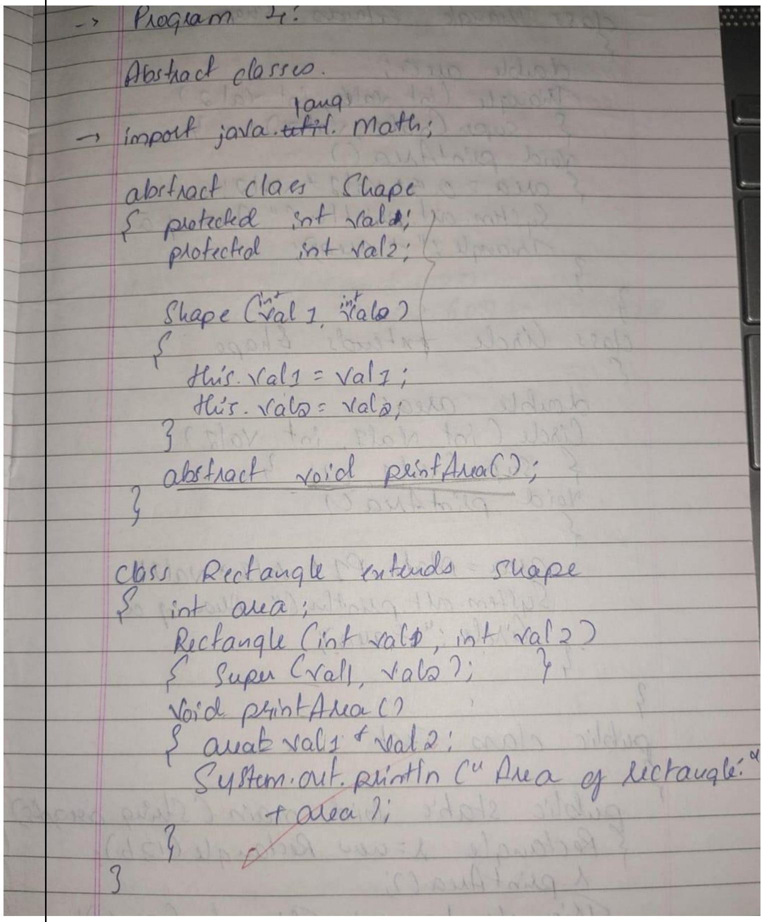
```
    {
        this.val1=val1;
        this.val2=val2;
    }
    abstract void printArea();
}
class Rectangle extends Shape
{
    int area;
    Rectangle(int val1,int val2)
    {
        super(val1,val2);
    }
    void printArea()
    {
        area=val1*val2;
        System.out.println("Area of the rectangle:"+area);
    }
}
class Triangle extends Shape
{
    double area;
    Triangle(int val1,int val2)
    {
        super(val1,val2);
    }
    void printArea()
    {
        area=0.5*val1*val2;
        System.out.println("Area of the Triangle:"+area);
    }
}
class Circle extends Shape
{
    double area;
    Circle(int val1,int val2)
    {
        super(val1,0);
    }
    void printArea()
    {
        area=Math.PI*val1*val1;
        System.out.println("Area of the circle:"+area);
    }
}
public class Test
{
    public static void main(String args[])
    {
        Rectangle r=new Rectangle(12,6);
        r.printArea();
        Triangle t=new Triangle(34,6);
        t.printArea();
        Circle c=new Circle(67,0);
        c.printArea();
    }
}
```

OUTPUT:

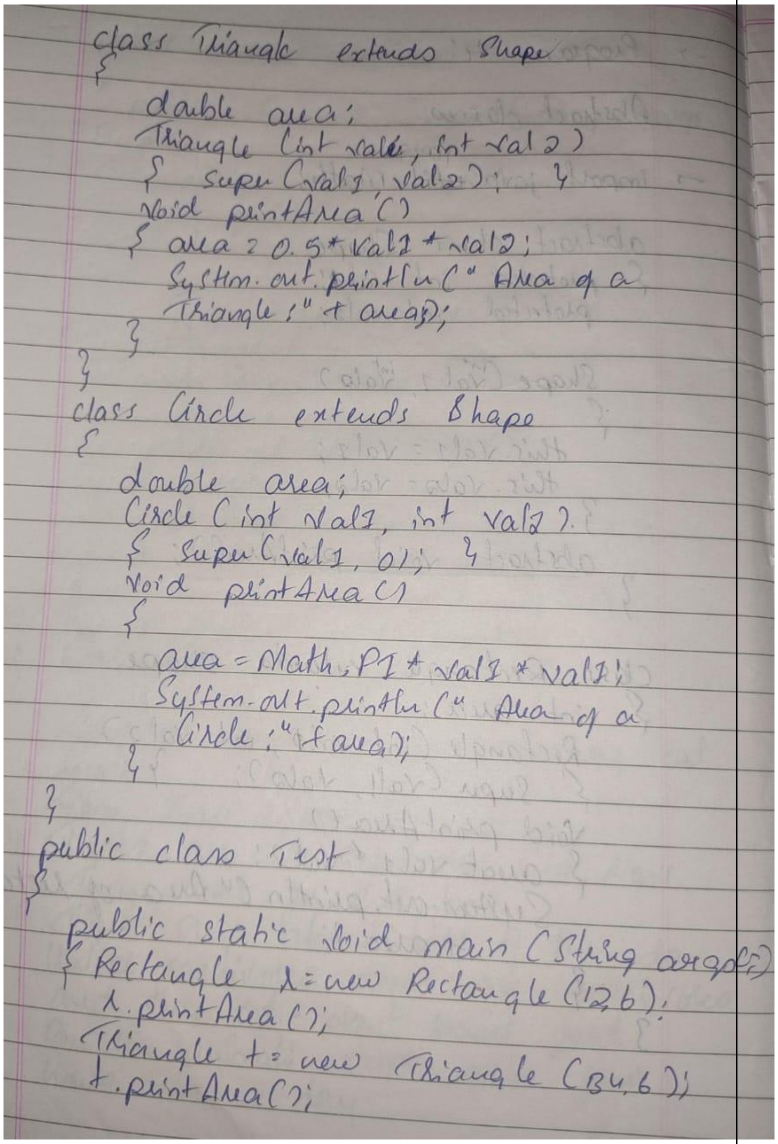


Output

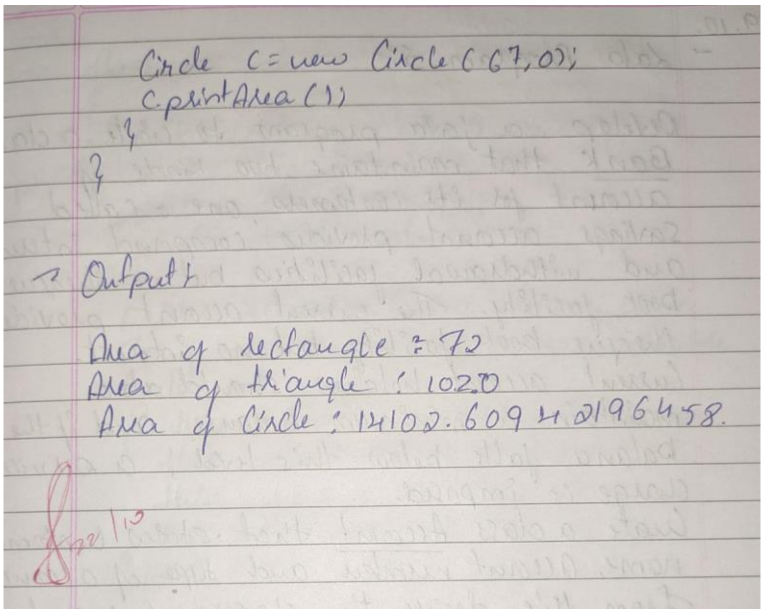
```
Area of the rectangle:72
Area of the Triangle:102.0
Area of the circle:14102.60942196458
```



```
-> Program 4:
Abstract classes.
                lang
-> import java.~~util~~. Math;
abstract class Shape
{ protected int val1;
  protected int val2;

  Shape (int val1, int val2)
  {
     this.val1 = val1;
     this.val2 = val2;
  }
  abstract void printArea();
}

class Rectangle extends Shape
{ int area;
  Rectangle (int val1, int val2)
  { Super (val1, val2);     }
  Void printArea ()
  { area= val1 * val2;
    System.out.println ("Area of rectangle:"
        + area );
  }
}
```



```
class Triangle extends Shape
{
   double area;
   Triangle (int val1, int val2)
   { super (val1, val2);   }
   Void printArea ()
   { area = 0.5 * val1 * val2;
     System.out.println (" Area of a
     Triangle :" + area);
   }
}
class Circle extends Shape
{
  double area;
  Circle ( int val1, int val2 ).
  { super (val1, 0);  }
  Void printArea ()
  {
     area = Math.PI * val1 * val1;
     System.out.println (" Area of a
      Circle :" + area);
  }
}
public class Test
{
  public static void main (String args[])
  { Rectangle r= new Rectangle (12,6);
    r.printArea();
   Triangle t= new Triangle (34,6);
   t.printArea();
```



```
Circle c=new Circle(67,0);
c.printArea();
}
}
```

Output:

```
Area of rectangle : 72
Area of triangle : 102.0
Area of Circle : 14102.6094 2196458.
```

[illegible]

Lab 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:

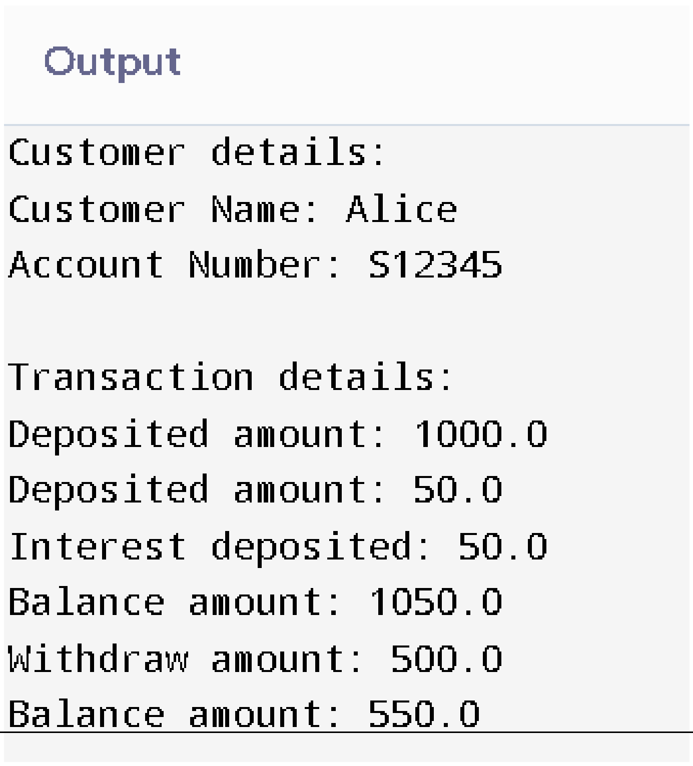
a) Accept deposit from customer and update the balance.
b) Display the balance.
c) Compute and deposit interest
d) Permit withdrawal and update the balance

Check for the minimum balance, impose penalty if necessary and update the balance.

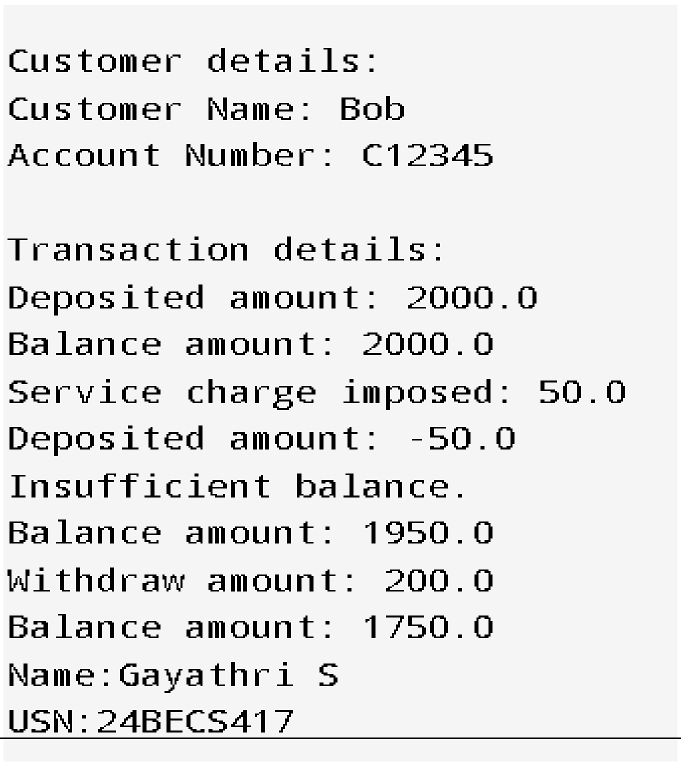
CODE:

```
class Account {
    public String customerName;
    public 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);
    }

    public String toString()
    {   return "Customer Name: "+customerName+"\nAccount Number: "+accountNumber;    }
}

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 String toString()
    {   return "Customer Name: "+customerName+"\nAccount Number: "+accountNumber;    }
}

public class Bank {
    public static void main(String[] args) {
        SavAcct savingsAccount = new SavAcct("Alice", "S12345", 5.0);
        System.out.println("Customer details:\n"+savingsAccount.toString());
        System.out.println("\nTransaction details:");
        savingsAccount.deposit(1000);
        savingsAccount.computeAndDepositInterest();
        savingsAccount.displayBalance();
        savingsAccount.withdraw(500);
        savingsAccount.displayBalance();

        System.out.println();

        CurAcct currentAccount = new CurAcct("Bob", "C12345", 1000, 50);
        System.out.println("Customer details:\n"+currentAccount.toString());
        System.out.println("\nTransaction details:");
        currentAccount.deposit(2000);
        currentAccount.displayBalance();
        currentAccount.withdraw(1900);
        currentAccount.displayBalance();
        currentAccount.withdraw(200);
        currentAccount.displayBalance();
    }
}
```

OUTPUT:


Output

```
Customer details:
Customer Name: Alice
Account Number: S12345

Transaction details:
Deposited amount: 1000.0
Deposited amount: 50.0
Interest deposited: 50.0
Balance amount: 1050.0
Withdraw amount: 500.0
Balance amount: 550.0
```


```
Customer details:
Customer Name: Bob
Account Number: C12345

Transaction details:
Deposited amount: 2000.0
Balance amount: 2000.0
Service charge imposed: 50.0
Deposited amount: -50.0
Insufficient balance.
Balance amount: 1950.0
Withdraw amount: 200.0
Balance amount: 1750.0
Name:Gayathri S
USN:24BECS417
```

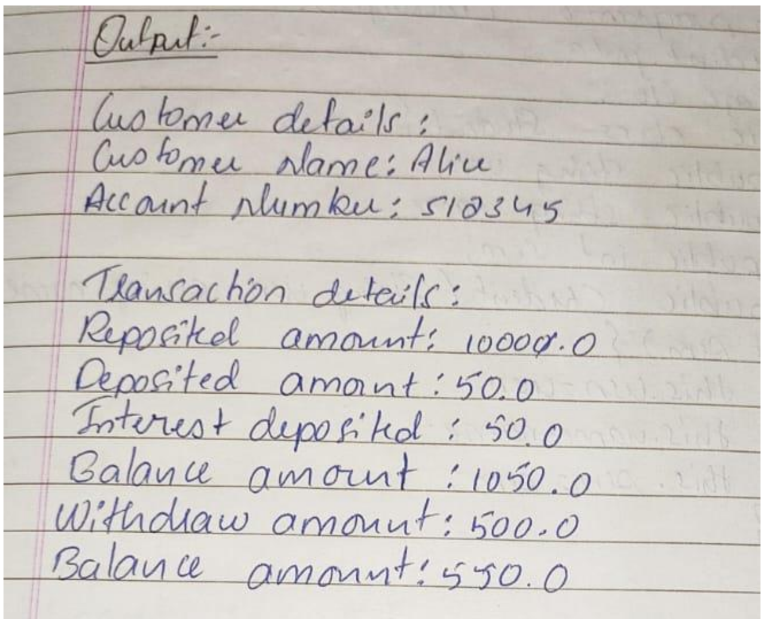
```
->.
class Accounts
{
private { public String customer_name;
        { public String account_no;
        { public String acc_type;
  ~~private~~ double balance;
protected.
  public Accounts(String customer name,
  String accountno, String acctype, double
  balance)
  { this. customer_name = customer_name;
    this. account_no = account_no;
    this. acc_type = acc_type;
    this. balance = 0.0;
  }

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

  public void withdraw (double amount)
  { if (~~amount~~ balance > amount)
     { balance -= amount;
       System.out.println("Withdraw
    amount: "+ amount );
     }.
    else
     { System.out.println ("Insufficient
           balance"); }
  }.
```

```
        SavAct
class ~~CurAct~~ extends Accounts.
{.  private double interest_rate,
    public SavAct (String customer_name,
    String account_no, String acc_type,
    double interest_rate ).
    { Super(customer_name, account_no,
    "Savings")
       this. interest_rate = 0.05;
    }.

    public void getBalance ()
    { return balance ; }

    public void compute And Deposit Interest()
    { double currentbalance = getBalance();
      double interest = currentbalance *
      interest_rate /100 ;
      deposit (interest ~~rate~~);
      System.out.println (" Interest deposited:"+
      interest);
    }.
}.

class CurAct extends Accounts.
{

  private double minimumbalance;
  private double servicecharges
```

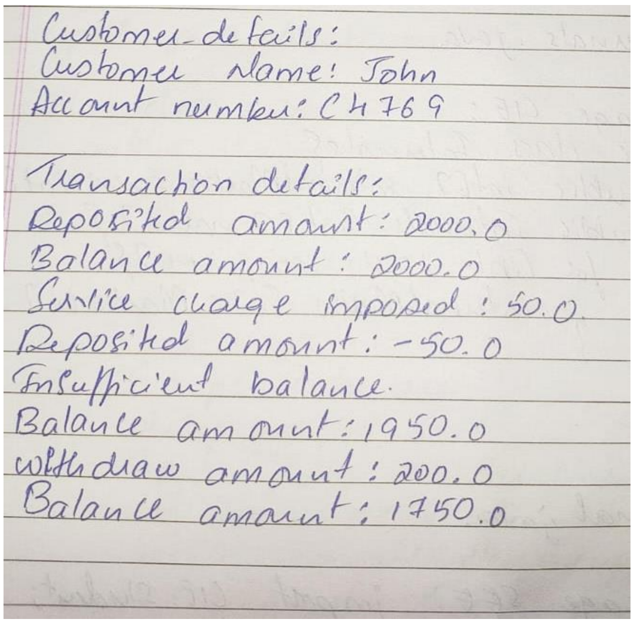
```
  public AcCurAct (String customer_name,
  String account_no String acc_type,
  double minimumbalance, double
  service charge)
  { Super ( customer name, account_no.
    "Current")
     this. minimumbalance = 1000;
     this. service charge = 50;
  }.

public withdraw( double amount)
{ double currentBalance = getBalanceFilter();
  if (currentbalance - amount < minimumBalance
  { deposit(servicecharge);
     S.O.P(" Service Charge:"+servicecharge);
     S.o.p("Insufficient balance");
  }
  else
  } Super. withdraw (amount); }.
}

private double get Balance ()
{ return balance; }
```

```
public ~~Bank~~ class Bank
{
  public static void main (String [] args)
  {
   SavAct Savingaccount = new
      SavAct ("Alice", "S12345"), ~~550~~);
   Savingaccount. deposit (1000;)
   Savingaccount.compute And Deposit Interest();
   Savingaccount. displaybalance();
   Savingaccount . withdraw (500);
   Savingaccount. displayBalance ();

   S.O.p("");

  CurAct currentAccount = new ~~A~~ CurAct ("Joh,
   "C4769");
  currentAccount. deposit (2000 );
  CurrentAccount . display Balance ();
  currentAccount . withdraw (1 200);
  CurrentAccount. display Balance ();
  CurrentAccount. withdraw (200 );
  CurrentAccount . display Balance();
}
```

Proceed



```
Output:-

Customer details:
Customer Name: Alice
Account Number: 512345

Transaction details:
Deposited amount: 1000.0
Deposited amount: 50.0
Interest deposited : 50.0
Balance amount : 1050.0
Withdraw amount: 500.0
Balance amount: 550.0
```



```
Customer details:
Customer Name: John
Account number: C4769

Transaction details:
Deposited amount: 2000.0
Balance amount: 2000.0
Service charge imposed : 50.0
Deposited amount: -50.0
Insufficient balance.
Balance amount: 1950.0
Withdraw amount: 200.0
Balance amount: 1750.0
```

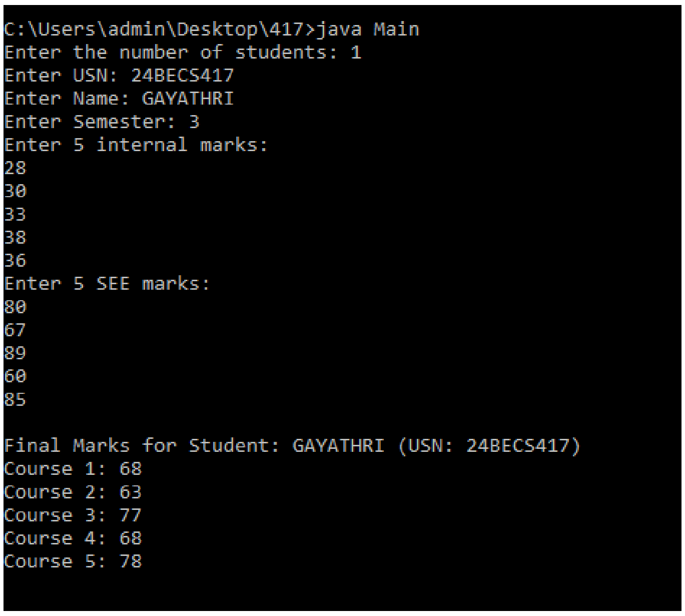
Lab program-6

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.

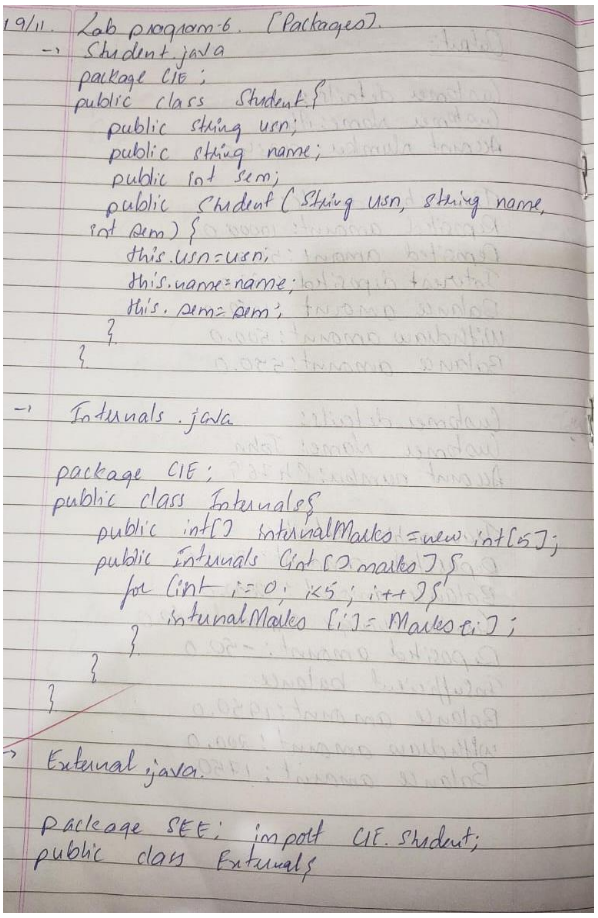
CODE:

```
package CIE;

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

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

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

    public Internals(int[] marks) {
        if (marks.length == 5) {
            System.arraycopy(marks, 0, internalMarks, 0, 5);
        } else {
            throw new IllegalArgumentException("Exactly 5 marks are required for internal
marks.");
        }
    }
}

// Package SEE: Contains External class
package SEE;

import CIE.Student;

public class External extends Student {
    public int[] externalMarks = new int[5];

    public External(String usn, String name, int sem, int[] marks) {
        super(usn, name, sem);
        if (marks.length == 5) {
            System.arraycopy(marks, 0, externalMarks, 0, 5);
        } else {
            throw new IllegalArgumentException("Exactly 5 marks are required for SEE marks.");
        }
    }
}

// Main Program: Computes final marks
import CIE.*;
import SEE.*;
import java.util.Scanner;

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

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

        External[] students = new External[n];

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

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

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

            System.out.print("Enter Semester: ");
            int sem = scanner.nextInt();

            System.out.println("Enter Internal Marks (5 courses):");
            int[] internalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                internalMarks[j] = scanner.nextInt();
            }

            System.out.println("Enter SEE Marks (5 courses):");
            int[] externalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                externalMarks[j] = scanner.nextInt();
            }
            scanner.nextLine();

            Internals internals = new Internals(internalMarks);
            students[i] = new External(usn, name, sem, externalMarks);

            System.out.println("\nCalculating final marks for student...");
        }

        System.out.println("\nFinal Marks for all students:");
        for (External student : students) {
            System.out.println("\nUSN: " + student.usn);
            System.out.println("Name: " + student.name);
            System.out.println("Semester: " + student.sem);
            System.out.println("Final Marks:");
            for (int j = 0; j < 5; j++) {
                int finalMark = (student.externalMarks[j] / 2) + student.internalMarks[j];
                System.out.println("Course " + (j + 1) + ": " + finalMark);
            }
        }

        scanner.close();
    }
}
```

OUTPUT:



```
C:\Users\admin\Desktop\417>java Main
Enter the number of students: 1
Enter USN: 24BECS417
Enter Name: GAYATHRI
Enter Semester: 3
Enter 5 internal marks:
28
30
33
38
36
Enter 5 SEE marks:
80
67
89
60
85

Final Marks for Student: GAYATHRI (USN: 24BECS417)
Course 1: 68
Course 2: 63
Course 3: 77
Course 4: 68
Course 5: 78
```



19/11. Lab program-6. [Packages].

→ Student.java

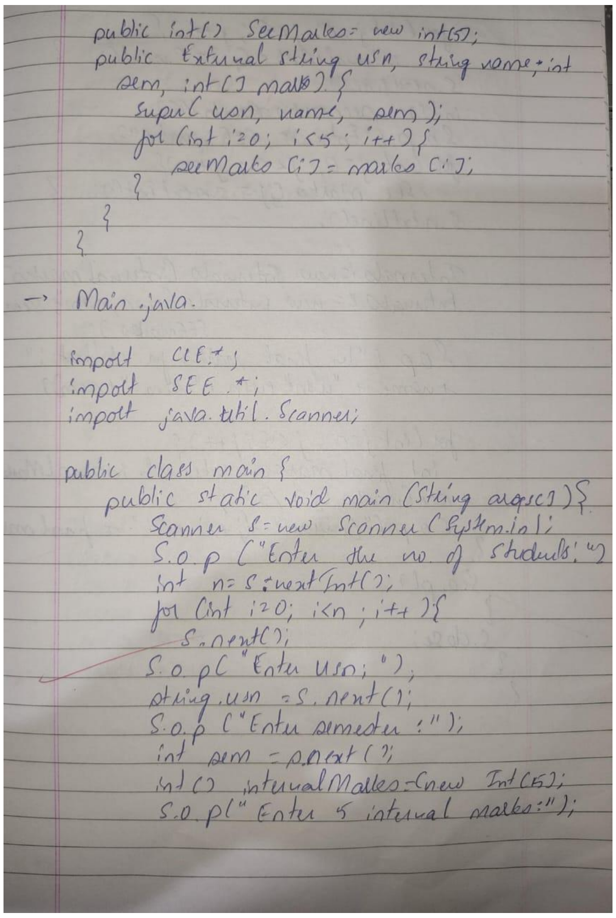
```
package CIE ;
public class Student{
   public string usn;
   public string name;
   public int sem;
   public Student ( String usn, string name,
  int sem) {
      this.usn=usn;
      this.name=name;
      this. sem= sem;
   }
}
```

→ Internals . java

```
package CIE ;
public class Internals{
   public int[] internalMarks =new int[5];
   public internals (int [] marks ] {
      for (int i=0; i<5 ; i++ ]{
         internalMarks [i]= Marks [i] ;
      }
   }
}
```

→ External .java.

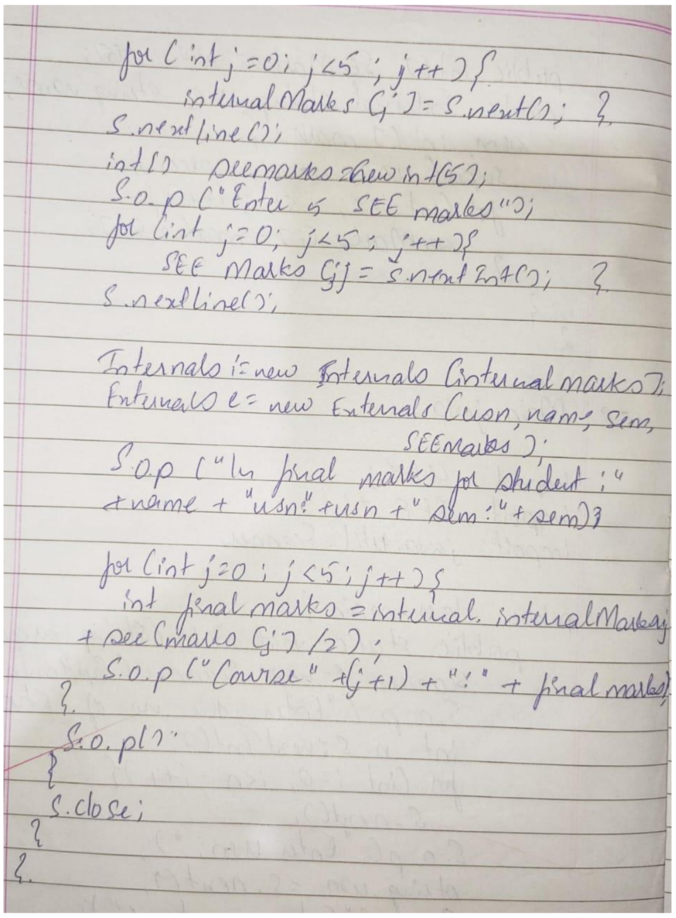
```
package SEE; import CIE. student;
public class Externals{
```



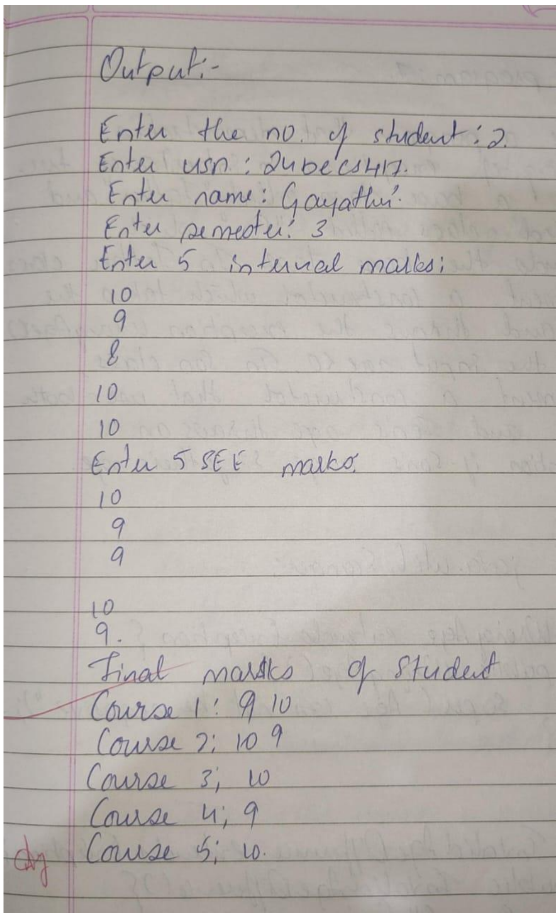
```
   public int[] SeeMarks= new int[5];
   public External string usn, string name, int
     sem, int[] marks) {
      super( usn, name, sem );
      for (int i=0; i<5 ; i++){
         seeMarks [i]= marks [i];
      }
   }
}
```

→ Main .java.

```
import CIE.*;
import SEE .*;
import java. util. Scanner;

public class main {
   public static void main (String args[] ){
      Scanner s= new Scanner (System.in);
      S.o.p ("Enter the no. of students: ")
      int n= s.nextInt();
      for (int i=0; i<n ; i++ ){
         s.next();
      S.o.p( "Enter usn; ");
      string usn =s. next();
      S.o.p ("Enter semester :");
      int sem = s.next ();
      int[] internalMarks=(new Int[5];
      S.o.p(" Enter 5 internal marks:");
```



```
      for ( int j=0; j<5 ; j++ ]{
         internalMarks [j]= s.next(); }
      s.nextline();
      int[] seemarks =new int[5];
      S.o.p ("Enter 5 SEE marks");
      for (int j=0; j<5 ; j++ ){
         SEE Marks [j] = s.nextInt(); }
      s.nextline();

      Internals i= new Internals (internal marks];
      Externals e= new Externals (usn, name, sem,
                           SEEMarks );
      S.o.p ("\n final marks for student :"
      +name + "usn:" +usn + " sem:" + sem)}

      for (int j=0 ; j<5; j++ ){
        int final marks = internal. internalMarks[j]
      + see(marks [j] /2 ) ;
        S.o.p ("Course" +(j+1) + ":" + final marks)
      }.
      S.o.p()
     }
     s.close;
   }
}.
```



Output:-

```
Enter the no. of student : 2.
Enter usn : 24bECS417.
 Enter name : Gayathri.
 Enter semester: 3
 Enter 5 internal marks;
 10
  9
  8
 10
 10
Enter 5 SEE marks.
 10
  9
  9

10
9.
Final marks of Student
Course 1: 9 10
Course 2; 10 9
Course 3; 10
Course 4; 9
Course 5; 10.
```

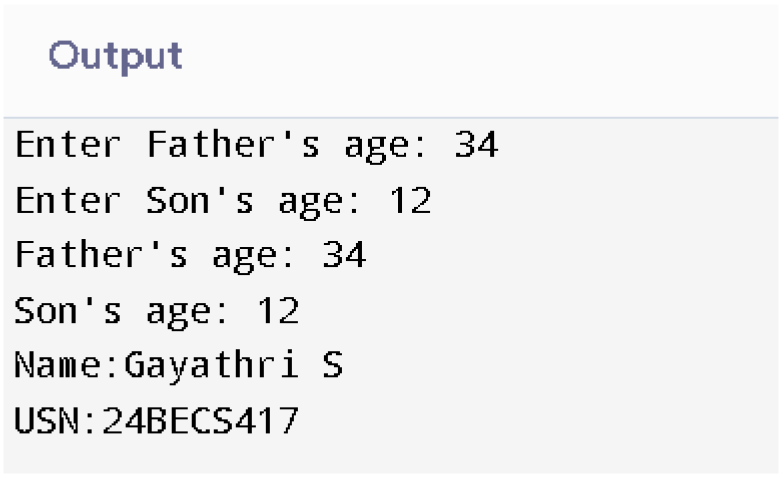
Lab program-7

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 Wrong Age( ) when the input age<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is >=father’s age.

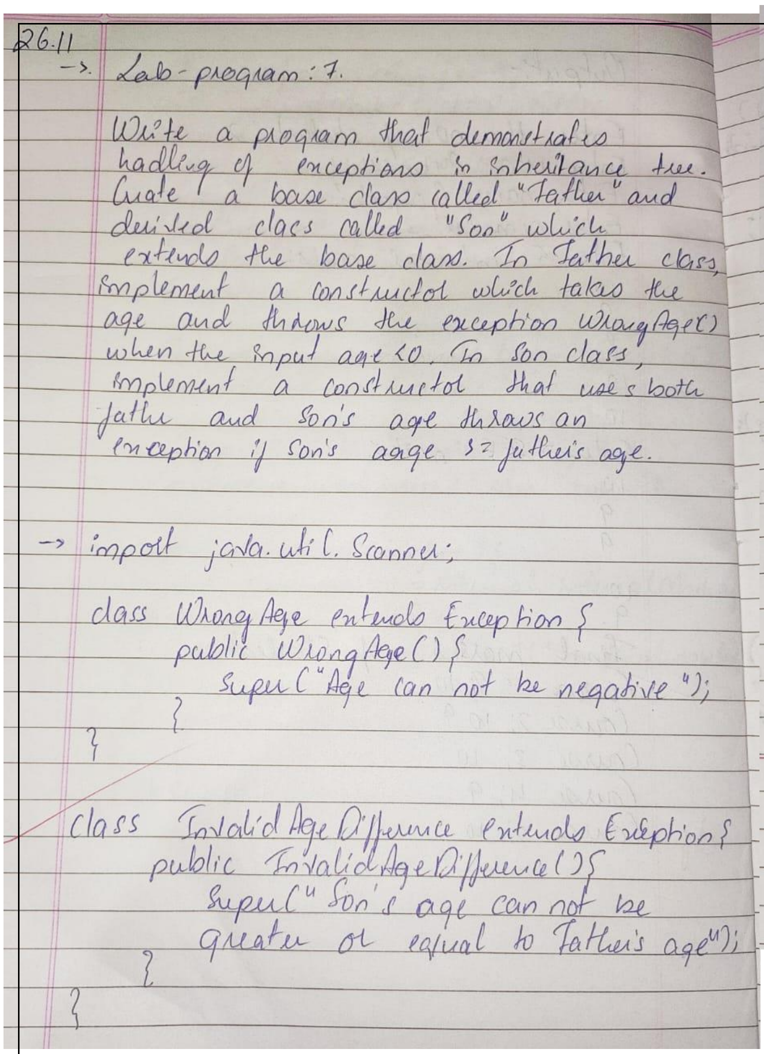
CODE:

```
import java.util.Scanner;


class WrongAge extends Exception {
    public WrongAge() {
        super("Age cannot be negative.");
    }
}

class InvalidAgeDifference extends Exception {
    public InvalidAgeDifference() {
        super("Son's age cannot be greater than or equal to Father's age.");
    }
}

class Father {
    int age;

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

class Son extends Father {
    int sonAge;

    public Son(int fatherAge, int sonAge) throws WrongAge, InvalidAgeDifference {
        super(fatherAge);
        if (sonAge < 0) {
            throw new WrongAge();
        }
        if (sonAge >= fatherAge) {
            throw new InvalidAgeDifference();
        }
        this.sonAge = sonAge;
    }
}

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

        try {
            System.out.print("Enter Father's age: ");
            int fatherAge = scanner.nextInt();

            System.out.print("Enter Son's age: ");
            int sonAge = scanner.nextInt();

            Son son = new Son(fatherAge, sonAge);

            System.out.println("Father's age: " + son.age);
            System.out.println("Son's age: " + son.sonAge);

        } catch (WrongAge | InvalidAgeDifference e) {
            System.out.println("Exception caught: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

OUTPUT:



```
Enter Father's age: 34
Enter Son's age: 12
Father's age: 34
Son's age: 12
Name:Gayathri S
USN:24BECS417
```

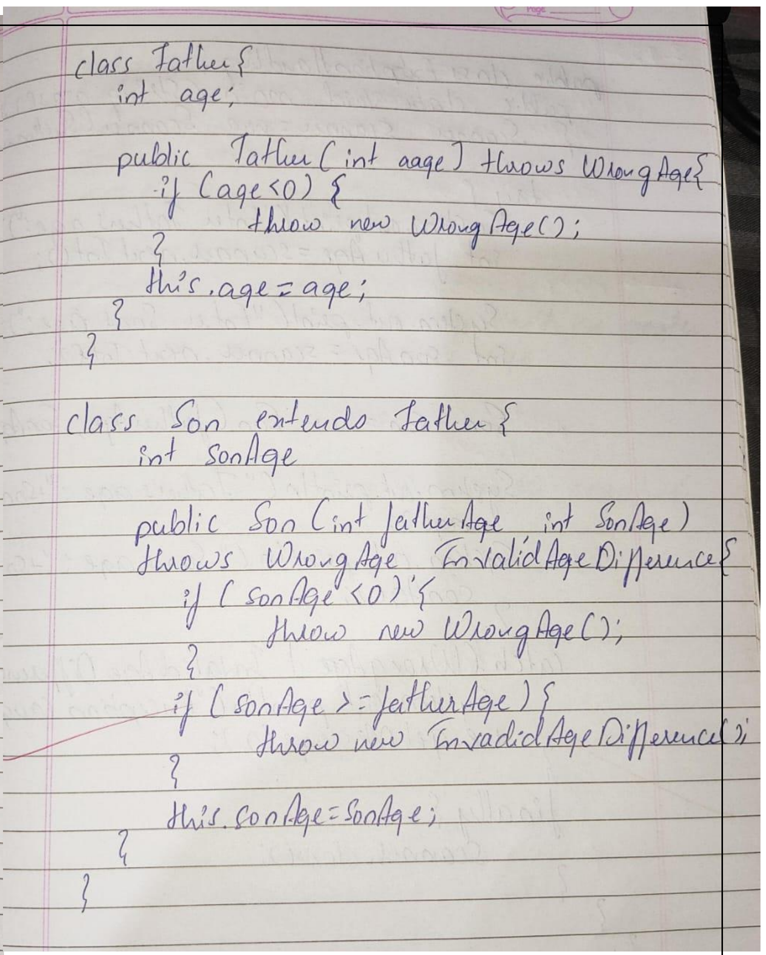


26.11

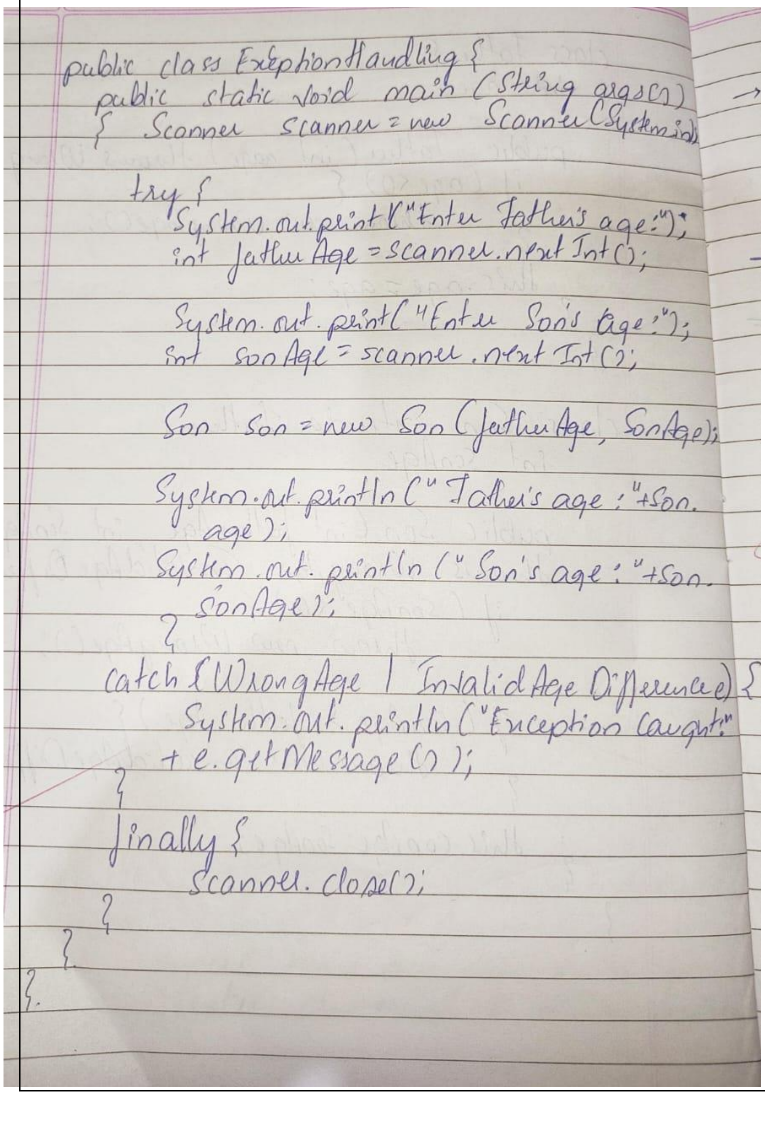
-> Lab-program: 7.

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

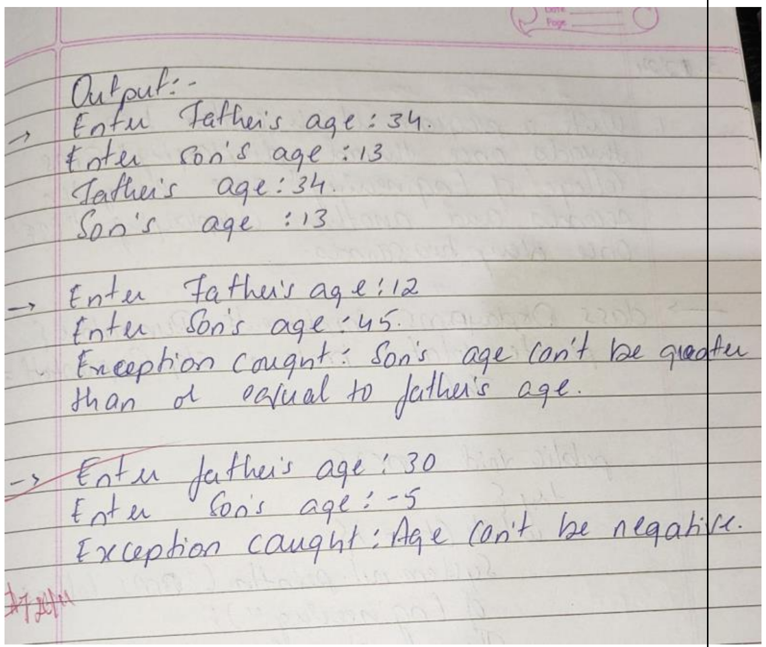
```
-> import java.util.Scanner;

class WrongAge extends Exception {
    public WrongAge() {
        super("Age can not be negative");
    }
}

class InvalidAgeDifference extends Exception {
    public InvalidAgeDifference() {
        super("Son's age can not be
        greater or equal to Father's age");
    }
}
```



```
class Father {
   int age;

   public Father(int aage) throws WrongAge{
     if (age<0) {
          throw new WrongAge();
     }
     this.age=age;
   }
}

class Son extends Father {
   int sonAge

   public Son (int fatherAge int SonAge)
   throws WrongAge InvalidAgeDifference{
      if ( sonAge <0) {
            throw new WrongAge();
      }
      if ( sonAge >= fatherAge) {
           throw new InvadidAgeDifference();
      }
      this.sonAge=SonAge;
   }
}
```



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

    try {
      System.out.print("Enter Father's age:");
      int fatherAge = scanner.nextInt();

      System.out.print("Enter Son's age:");
      int sonAge = scanner.nextInt();

      Son son = new Son (fatherAge, SonAge);

      System.out.println(" Father's age :"+Son.
         age);
      System.out.println (" Son's age: "+Son.
         sonAge);
    }
   catch (WrongAge | InvalidAge Difference e) {
       System.out.println("Exception Caught:"
     + e.getMessage());
   }
   finally {
       Scanner.close();
   }
  }
}.
```



```
Output:-
-> Enter Father's age: 34.
   Enter Son's age : 13
   Father's age: 34.
   Son's age : 13

-> Enter Father's age: 12
   Enter Son's age: 45.
   Exception caught: Son's age can't be greater
   than or equal to father's age.

-> Enter father's age: 30
   Enter Son's age: -5
   Exception caught: Age can't be negative.
```

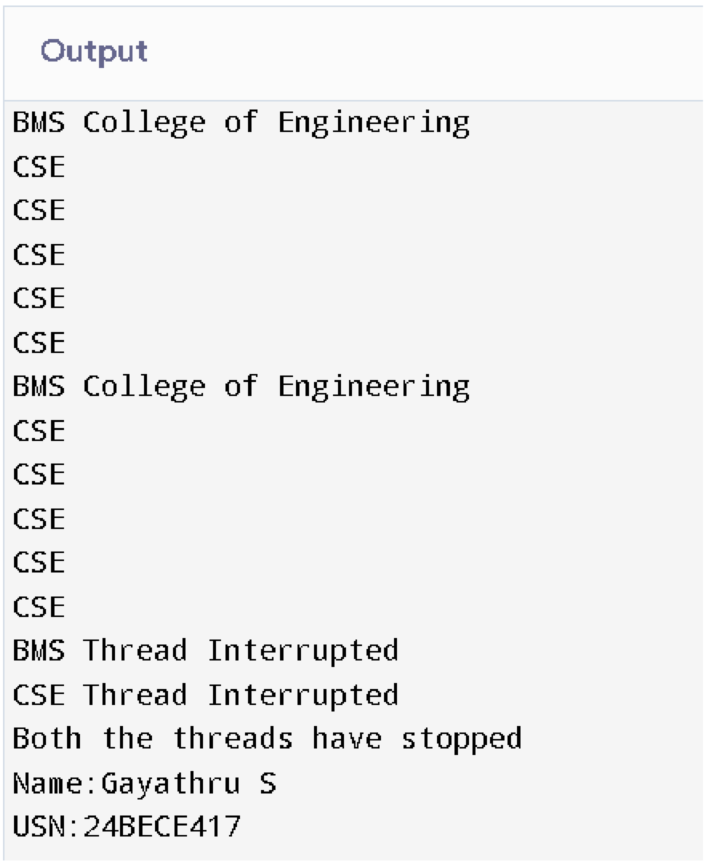
Lab program-8

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.

CODE:

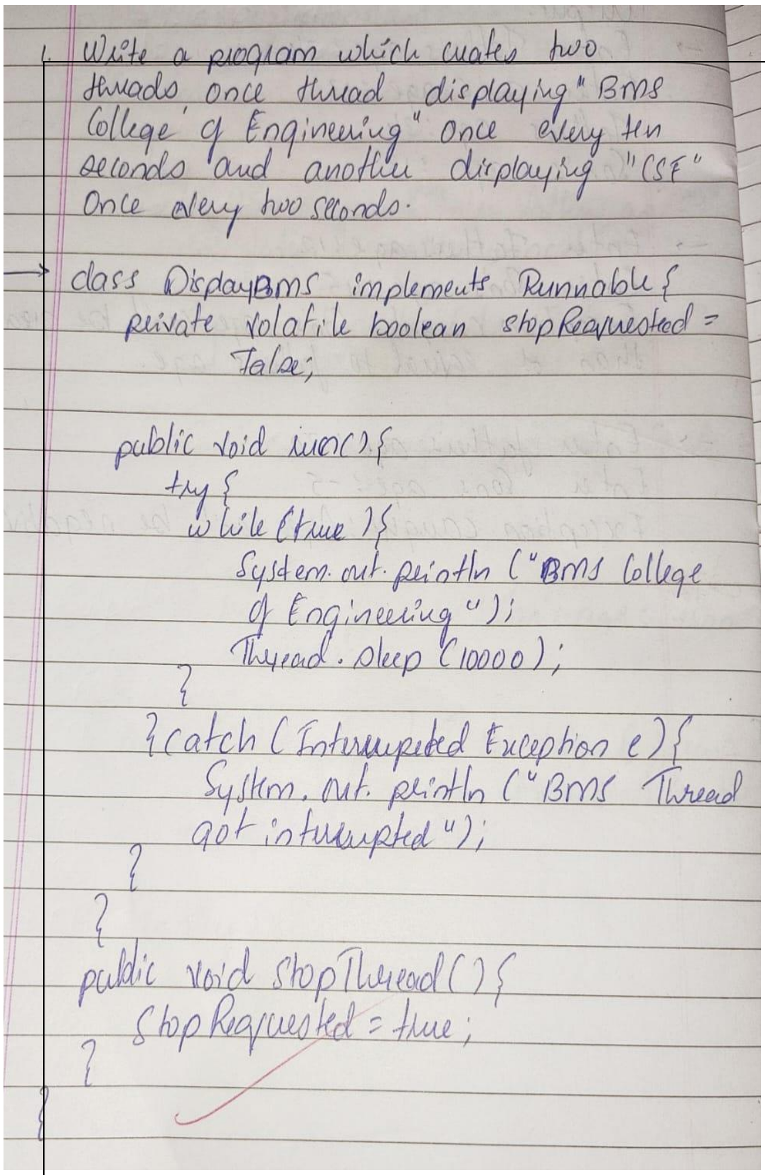
```
class DisplayBMS implements Runnable {
    @Override
    public void run() {
        try {
            while (!Thread.currentThread().isInterrupted()) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            }
        } catch (InterruptedException e) {
            System.out.println("BMS Thread Interrupted");
        }
    }
}

class DisplayCSE implements Runnable {
    @Override
    public void run() {
        try {
            while (!Thread.currentThread().isInterrupted()) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println("CSE Thread Interrupted");
        }
    }
}

public class CollegeThreads {
    public static void main(String[] args) {
        Thread thread1 = new Thread(new DisplayBMS());
        Thread thread2 = new Thread(new DisplayCSE());

        thread1.start();
        thread2.start();

        try {
                Thread.sleep(20000);

                thread1.interrupt();
                thread2.interrupt();

                thread1.join();
                thread2.join();
            } catch (InterruptedException e) {
                System.out.println("Main Thread Interrupted");
            }
            System.out.println("Both the threads have stopped")
        }
    }
```

OUTPUT:

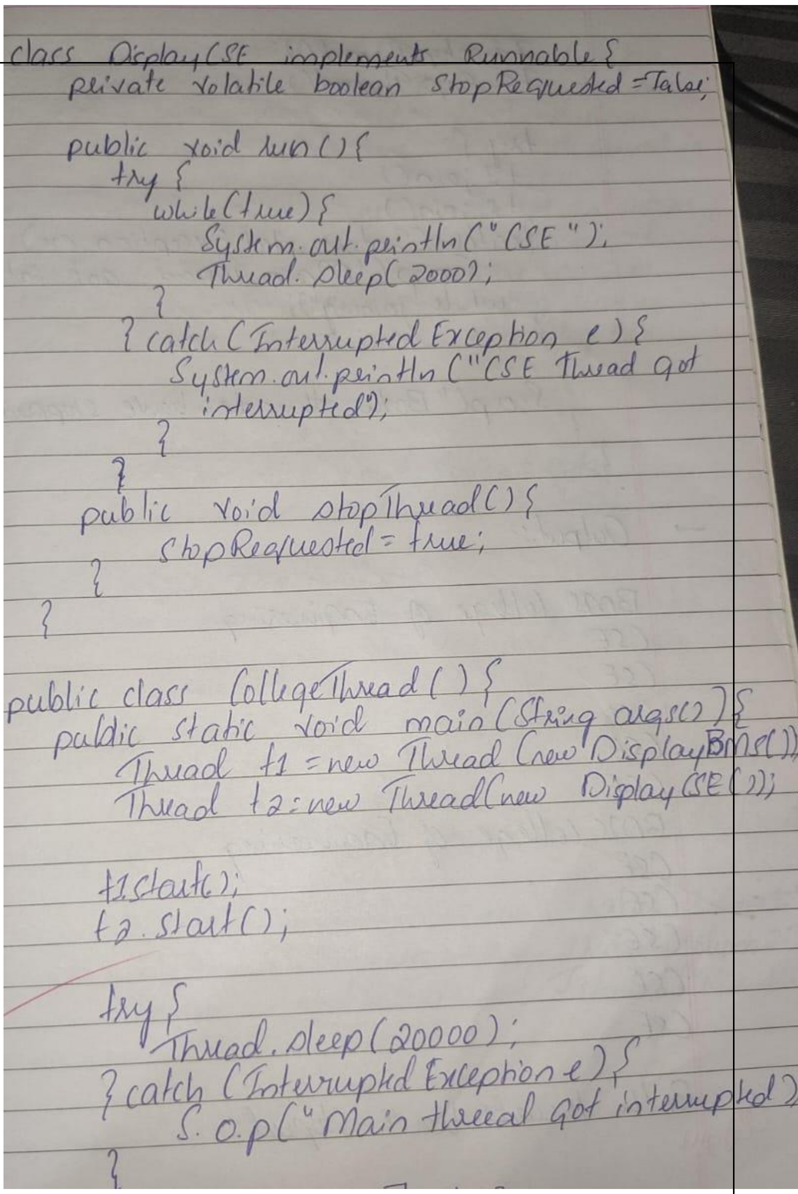


Output

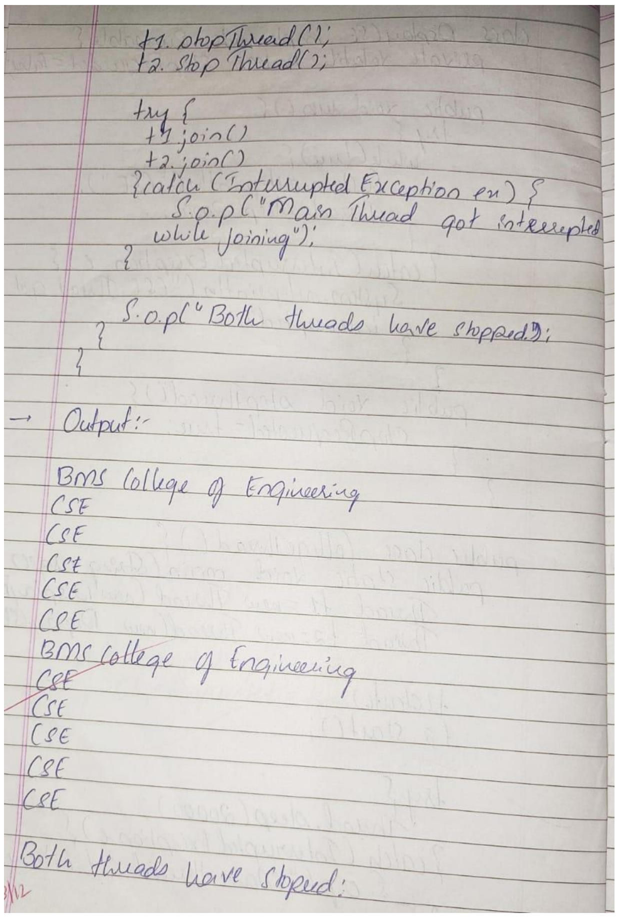
```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS Thread Interrupted
CSE Thread Interrupted
Both the threads have stopped
Name:Gayathru S
USN:24BECE417
```


1. 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 DisplayBMS implements Runnable{
  private volatile boolean stopRequested =
        False;

  public void run(){
    try{
      while(true){
        System.out.println("BMS College
          of Engineering");
        Thread.sleep(10000);
      }
    }catch(InterruptedException e){
        System.out.println("BMS Thread
        got interrupted");
    }
  }
  public void stopThread(){
     stopRequested = true;
  }
}
```



```
class DisplayCSE implements Runnable{
  private volatile boolean stopRequested = False;

  public void run(){
    try{
      while(true){
        System.out.println("CSE");
        Thread.sleep(2000);
      }
    }catch(InterruptedException e){
      System.out.println("CSE Thread got
        interrupted");
    }
  }
  public void stopThread(){
     stopRequested = true;
  }
}

public class CollegeThread(){
  public static void main(String args[]){
    Thread t1 =new Thread(new DisplayBMS());
    Thread t2=new Thread(new DisplayCSE());

    t1.start();
    t2.start();

    try{
      Thread.sleep(20000);
    }catch(InterruptedException e){
      S.o.p("Main thread got interrupted");
    }
```



```
    t1.stopThread();
    t2.stopThread();

    try{
      t1.join()
      t2.join()
    }catch(InterruptedException ex){
      S.o.p("Main Thread got interrupted
      while joining");
    }

    S.o.p("Both threads have stopped");
  }
}
```

→ Output:-

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

Both threads have stoped.
```