VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHREYA SATHYANARAYANA (1BM23CS318)

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-Jan 2025

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

CERTIFICATE

This is to certify that the Lab work entitled "**OBJECT ORIENTED JAVA PROGRAMMING**" carried out by **SHREYA SATHYANARAYANA (1BM23CS318),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

Dr. Nandhini Vineeth

Associate Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

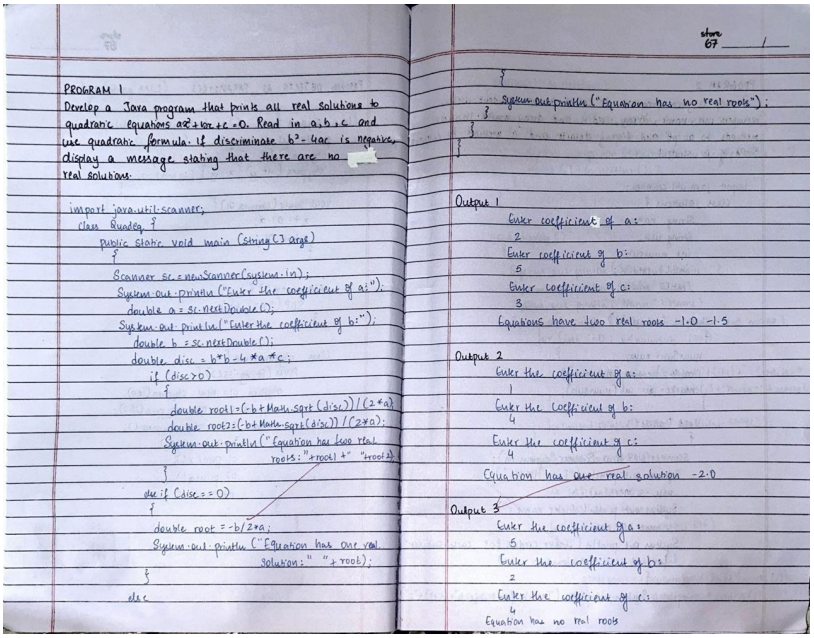
JAVA LAB REPORT

index

Program 1

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

Observation:



PROGRAM 1

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

```
import java.util.scanner;
class Quadeq {
  public static void main (String[] args)
  {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the coefficient of a:");
    double a = sc.nextDouble();
    System.out.println("Enter the coefficient of b:");
    double b = sc.nextDouble();
    double disc = b*b-4*a*c;
    if (disc>0)
    {
      double root1=(-b+Math.sqrt(disc))/(2*a);
      double root2=(-b+Math.sqrt(disc))/(2*a);
      System.out.println("Equation has two real
                          roots:"+root1+" "+root2);
    }
  else if (disc==0)
  {
    double root = -b/2*a;
    System.out.println("Equation has one real
                        solution:" "+root);
  }
  else
  {
    System.out.println("Equation has no real roots");
  }
}
```

Output 1

```
Enter coefficient of a:
2
Enter coefficient of b:
5
Enter coefficient of c:
3
Equations have two real roots -1.0 -1.5
```

Output 2

```
Enter the coefficient of a:
1
Enter the coefficient of b:
4
Enter the coefficient of c:
4
Equation has one real solution -2.0
```

Output 3

```
Enter the coefficient of a:
5
Enter the coefficient of b:
2
Enter the coefficient of c:
4
Equation has no real roots
```

Code:

```
import java.util.Scanner;

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

```
        }
        else{
            System.out.println("It has two distinct roots:");
            double r1=((-b+Math.sqrt(d))/(2*a));
            System.out.println("x1="+r1);
            double r2=((-b-Math.sqrt(d))/(2*a));
            System.out.println("x2="+r2);
        }
        System.out.println("\nDo you want to calculate again?(yes=0 and no=1): ");
        y=sc.nextInt();
      }while(y==0);
   }
}
```

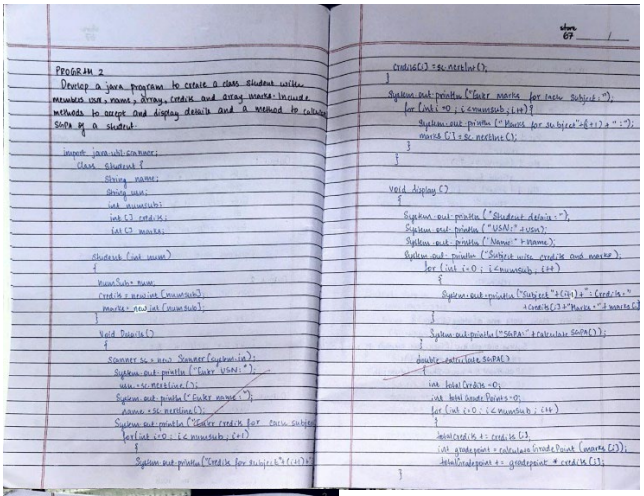
Output:

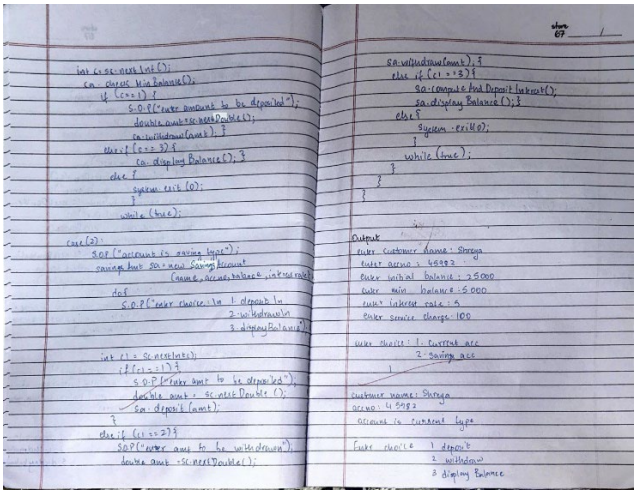
```
C:\295>java QuadraticEquation
Enter the coefficients of a, b, c:
Enter coefficient a: 8 50 20
Enter coefficient b: Enter coefficient c: Roots are real and unique.
Root 1: -0.42951766839402206
Root 2: -5.820482331605978

C:\295>javac QuadraticEquation.java

C:\295>java QuadraticEquation
Enter the coefficients of a, b, c:
Enter coefficient a: 3 4 5
Enter coefficient b: Enter coefficient c: Roots are imaginary.
Root 1: -0.6666666666666666 + 1.1055415967851332i
Root 2: -0.6666666666666666 - 1.1055415967851332i

C:\295>javac QuadraticEquation.java

C:\295>java QuadraticEquation
Enter the coefficients of a, b, c:
Enter coefficient a: 1 2 5
Enter coefficient b: Enter coefficient c: Roots are imaginary.
Root 1: -1.0 + 2.0i
Root 2: -1.0 - 2.0i
```

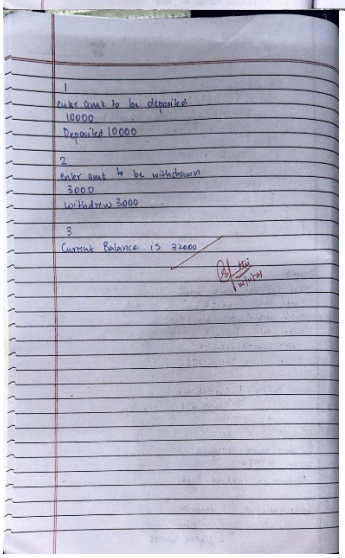
Program 2

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:







Code:

```
import java.util.Scanner;
class Subject {
    int subM;
    int cred;
    int grade;

    void setSubDet(int marks, int cred) {
        this.subM = marks;
        this.cred = cred;
        if (subM >= 90) {
            grade = 10;
```

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

class Student {
    Scanner s = new Scanner(System.in);
    Subject[] subjects = new Subject[8];
    Student() {
        for (int i = 0; i < subjects.length; i++) {
            subjects[i] = new Subject();
        }
    }

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

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

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

        if (totalCred > 0) {
```

```
            SGPA = Score / totalCred;
        } else {
            SGPA = 0;
        }
        return SGPA;
    }
}
public class StudentDetails {
    public static void main(String[] arg) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of semesters: ");
        int numSems = sc.nextInt();
        Student[] students = new Student[numSems];
        double cumulativeSGPA = 0.0;
        System.out.print("Enter USN: ");
        String usn = sc.next();
        System.out.print("Enter Name: ");
        String name = sc.next();

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

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

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

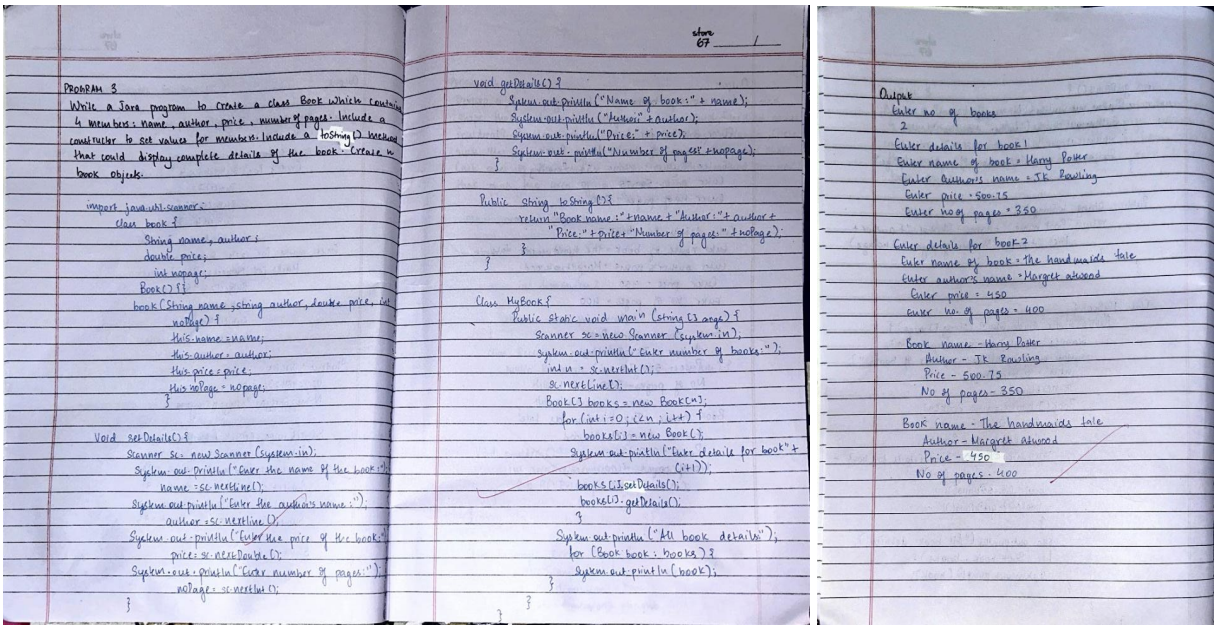
Output:

```
C:\jdk\programs>java StudentDetails
Enter number of semesters: 1
Enter USN: 1BM23CS318
Enter Name: Shreya Sathyanarayana
Enter details for semester 1
Enter marks for subject 1: 100
Enter credit for subject 1: 4
Enter marks for subject 2: 99
Enter credit for subject 2: 4
Enter marks for subject 3: 100
Enter credit for subject 3: 3
Enter marks for subject 4: 98
Enter credit for subject 4: 3
Enter marks for subject 5: 97
Enter credit for subject 5: 2
Enter marks for subject 6: 100
Enter credit for subject 6: 1
Enter marks for subject 7: 100
Enter credit for subject 7: 2
Enter marks for subject 8: 100
Enter credit for subject 8: 1
USN: 1BM23CS318
Name: Shreya
SGPA for sem 1: 10.0
CGPA: 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:



Code:

```
import java.util.Scanner;
class Book {
    String name, author;
    double price;
    int noPage;
    Book() {}
    Book(String name, String author, double price, int noPage) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.noPage = noPage;
    }
    void setDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter name of book: ");
        name = sc.nextLine();
        System.out.println("Enter author name: ");
        author = sc.nextLine();
        System.out.println("Enter price of book: ");
        price = sc.nextDouble();
        System.out.println("Enter number of pages: ");
        noPage = sc.nextInt();
    }
    void getDetails() {
```

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

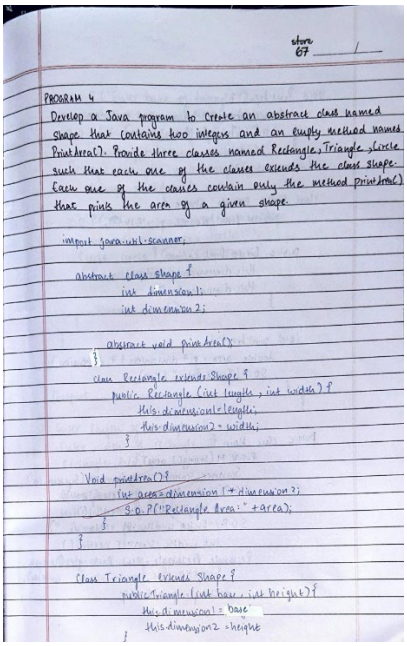
Output:

```
C:\Users\Admin\Documents\23cs310>java MyBook
Enter number of books:
2
Enter details for book 1
Enter name of book:
HarryPotter
Enter author name:
JK Rowling
Enter price of book:
500.75
Enter number of pages:
350
Name of book: HarryPotter
Author: JK Rowling
Price: 500.75
Number of pages: 350
Enter details for book 2
Enter name of book:
Magic of Lost Lamp
Enter author name:
Sudha Murthy
Enter price of book:
650
Enter number of pages:
400
Name of book: Magic of Lost Lamp
Author: Sudha Murthy
Price: 650.0
Number of pages: 400
All book details:
Book name: HarryPotter
Author: JK Rowling
Price: 500.75
Number of pages: 350

Book name: Magic of Lost Lamp
Author: Sudha Murthy
Price: 650.0
Number of pages: 400
```

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.

Observation:

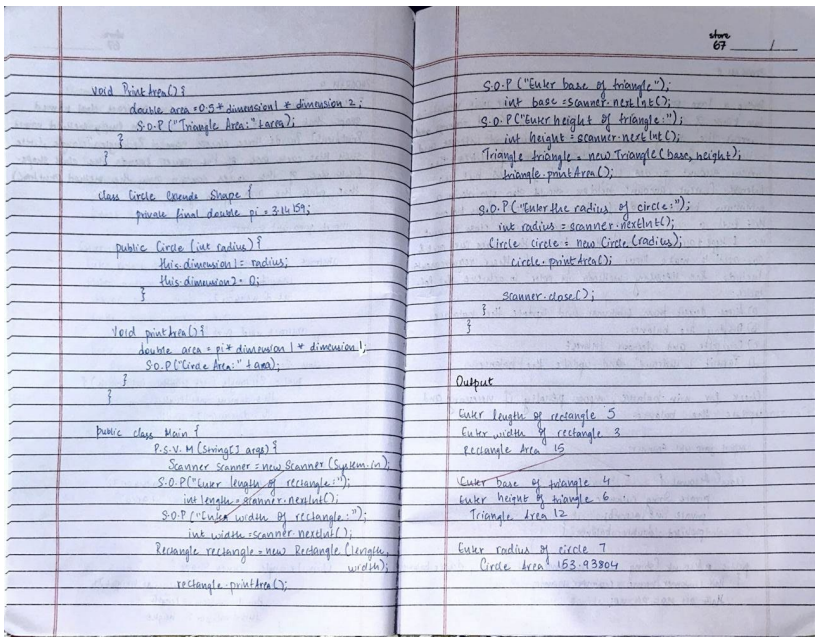


PROGRAM 4
Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named PrintArea(). Provide three classes named Rectangle, Triangle, Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method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

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;
S.O.P("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 * dimension 2;
S.O.P ("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;
S.O.P("Circle Area:" + area);
}
}
public class Main {
P.S.V.M (String[] args) {
Scanner scanner = new Scanner (System.in);
S.O.P("Enter length of rectangle:");
int length = scanner.nextInt();
S.O.P ("Enter width of rectangle:");
int width = scanner.nextInt();
Rectangle rectangle = new Rectangle (length, width);
rectangle.printArea();

S.O.P ("Enter base of triangle:");
int base = scanner.nextInt();
S.O.P("Enter height of triangle:");
int height = scanner.nextInt();
Triangle triangle = new Triangle(base, height);
triangle.printArea();

S.O.P("Enter the radius of circle:");
int radius = scanner.nextInt();
Circle circle = new Circle(radius);
circle.printArea();

scanner.close();
}
}

Output

Enter length of rectangle 5
Enter width of rectangle 3
Rectangle Area 15

Enter base of triangle 4
Enter height of triangle 6
Triangle Area 12

Enter radius of circle 7
Circle Area 153.93804

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.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:

```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Documents\23cs310>javac Main.java

C:\Users\Admin\Documents\23cs310>java Main
Enter length of rectangle: 5
Enter width of rectangle: 3
Rectangle Area: 15
Enter base of triangle: 4
Enter height of triangle: 6
Triangle Area: 12.0
Enter radius of circle: 7
Circle Area: 153.93791
```

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.

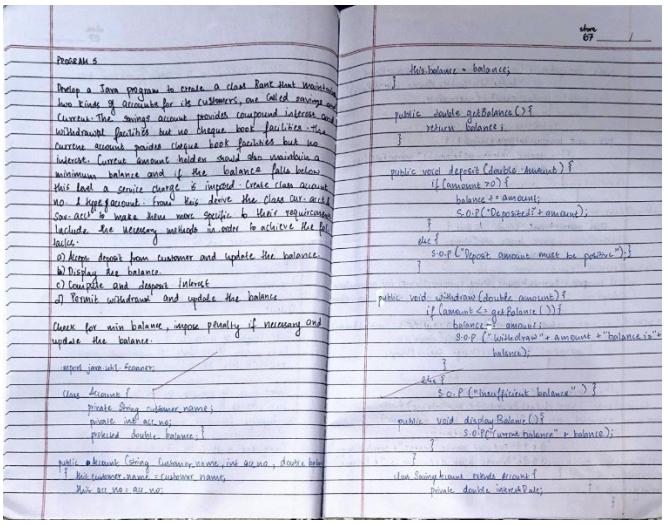
b) Display the balance.

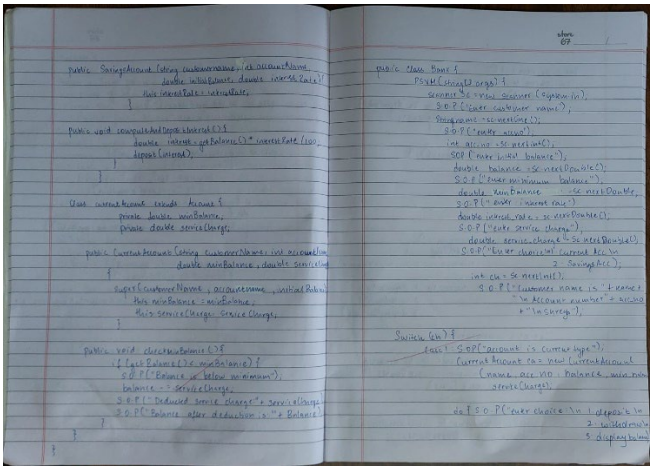
c) Compute and deposit interest

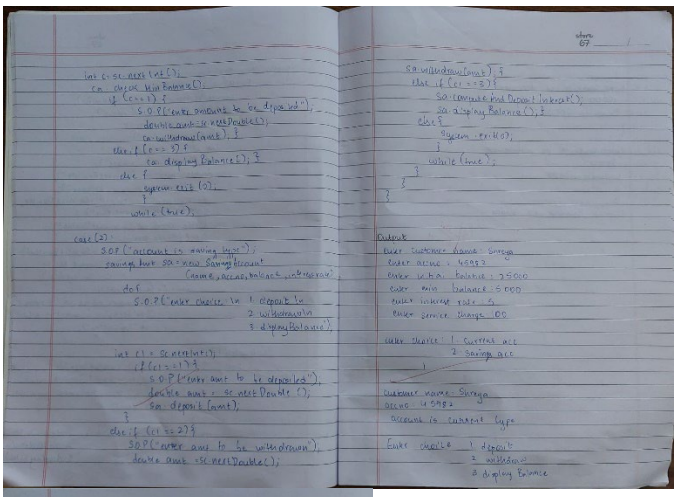
d) Permit withdrawal and update the balance

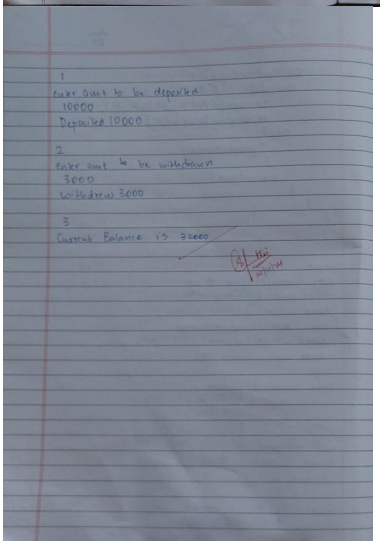
Check for the minimum balance, impose penalty if necessary and update the balance.

Observation:









Code:

```
import java.util.Scanner;

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

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

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
```

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

class SavingsAccount extends Account {
    private double interestRate;

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

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

    public CurrentAccount(String customerName, int accountNumber, double initialBalance,
double minimumBalance, double serviceCharge) {
        super(customerName, accountNumber, initialBalance);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }
    public void checkMinimumBalance() {
        if (getBalance() < minimumBalance) {
            System.out.println("Balance is below minimum");
```

```
            balance-=serviceCharge;
            System.out.println("Deducted service charge:" +serviceCharge);
            System.out.println("Balance after deduction is:"+balance);
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("enter customer name:");
        String name=sc.nextLine();
        System.out.println("enter accno:");
        int acc_no=sc.nextInt();
        System.out.println("enter initial balance:");
        double balance=sc.nextDouble();
        System.out.println("enter minimum balance:");
        double minimum_balance=sc.nextDouble();
        System.out.println("enter interest rate:");
        double interest_rate=sc.nextDouble();
        System.out.println("enter service charge:");
        double service_charge=sc.nextDouble();
        System.out.println("Enter choice:\n 1.Current acc\n 2.Savings acc");
        int ch=sc.nextInt();
        System.out.println("Customer name is:"+ name+"\nAccount
number:"+acc_no+"\nBhoomika BG-1BM23CS067");

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

```
            System.exit(0);
           }while(true);

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

Output:

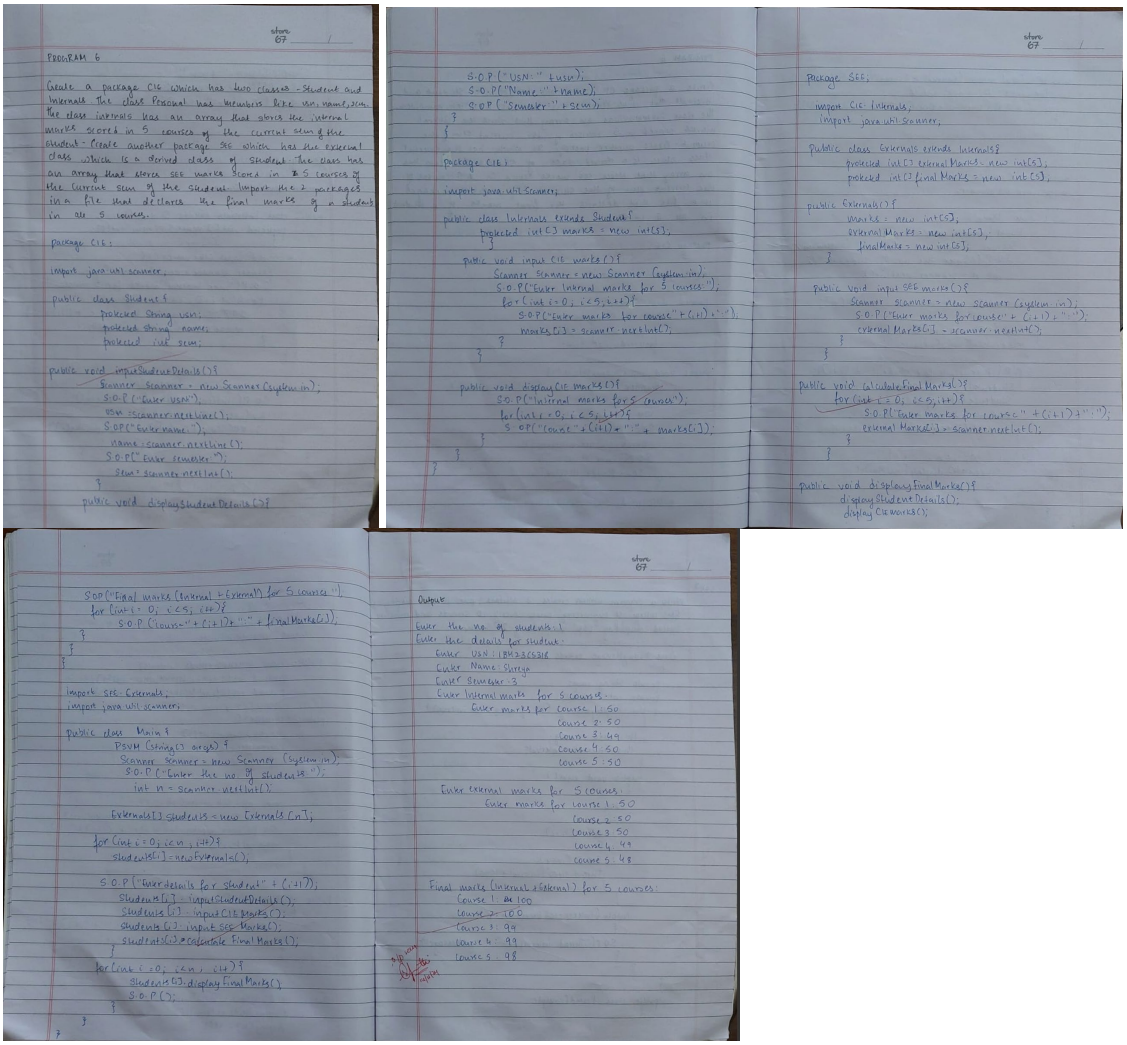
```
C:\jdk\programs>java Bank
enter customer name:
Shreya
enter accno:
1
enter initial balance:
100000
enter minimum balance:
1000
enter interest rate:
2
enter service charge:
2
Enter choice:
 1.Current acc
 2.Savings acc
1
Customer name is:Shreya
Account number:1
Bhoomika BG-1BM23CS067
account is current type
enter choice:
 1.deposit
 2.withdraw
 3.display balance
1
enter amount to be deposited:
200000
Deposited: 200000.0
```

```
enter choice:
 1.deposit
 2.withdraw
 3.display balance
2
enter amount to withdraw:
2000
withdrew:2000.0 balance is:298000.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
3
Current Balance: 298000.0
enter choice:
 1.deposit
 2.withdraw
 3.display balance
```

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.

Observation:



Code:

```
package CIE;
import java.util.Scanner;

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

    public void inputStudentDetails() {
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.print("Enter USN: ");
        usn = scanner.nextLine();
        System.out.print("Enter Name: ");
        name = scanner.nextLine();
        System.out.print("Enter Semester: ");
        sem = scanner.nextInt();
    }
    public void displayStudentDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}
package CIE;
import java.util.Scanner;

public class Internals extends Student {
    protected int[] marks = new int[5];
    public void inputCIEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter Internal marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for Course " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }
    public void displayCIEmarks() {
        System.out.println("Internal Marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + marks[i]);
        }
    }
}
package SEE;
import CIE.Internals;
import java.util.Scanner;

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

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

```
    public void inputSEEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter External marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for Course " + (i + 1) + ": ");
            externalMarks[i] = scanner.nextInt();
        }
    }
    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = marks[i] + externalMarks[i];
        }
    }
    public void displayFinalMarks() {
        displayStudentDetails();
        displayCIEmarks();
        System.out.println("Final Marks (Internal + External) for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}
import SEE.Externals;
import java.util.Scanner;

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

        Externals[] students = new Externals[n];

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

}

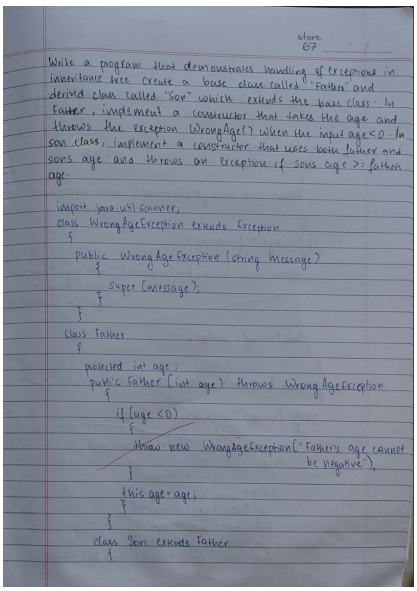
Output:

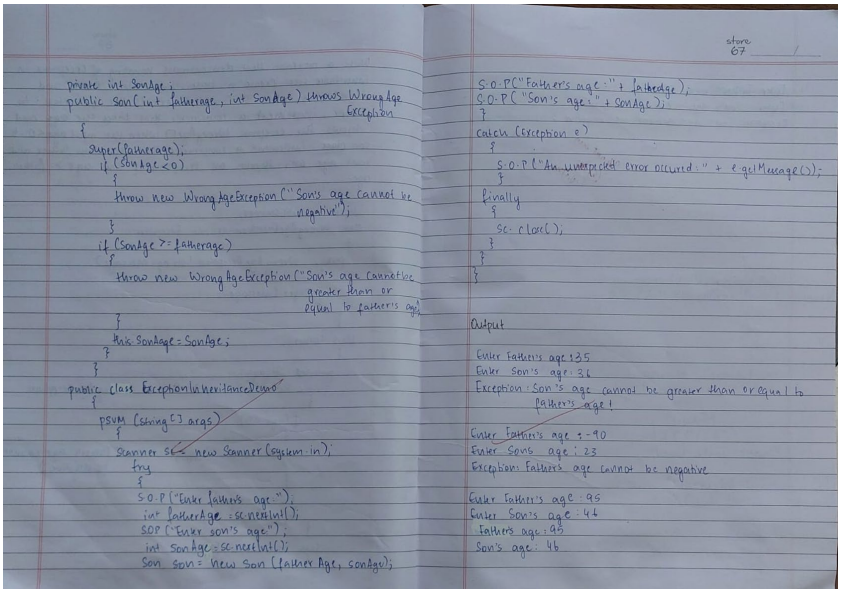
```
C:\Users\Admin\Documents\23cs310\lab6 packages>java Main.java
Enter the number of students: 1
Enter details for student 1
Enter USN: 1bm23cs310
Enter Name: joey
Enter Semester: 3
Enter Internal marks for 5 courses:
Enter marks for Course 1: 45
Enter marks for Course 2: 35
Enter marks for Course 3: 42
Enter marks for Course 4: 26
Enter marks for Course 5: 39
Enter External marks for 5 courses:
Enter marks for Course 1: 50
Enter marks for Course 2: 43
Enter marks for Course 3: 20
Enter marks for Course 4: 38
Enter marks for Course 5: 41
USN: 1bm23cs310
Name: joey
Semester: 3
Internal Marks for 5 courses:
Course 1: 45
Course 2: 35
Course 3: 42
Course 4: 26
Course 5: 39
Final Marks (Internal + External) for 5 courses:
Course 1: 95
Course 2: 78
Course 3: 62
Course 4: 64
Course 5: 80
```

Program 7

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

Observation:





Code:

```
class WrongAge extends Exception {
    String message;
    WrongAge(String message) {
        this.message = message;
    }

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

class Son extends Father {
    int sAge;
```

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

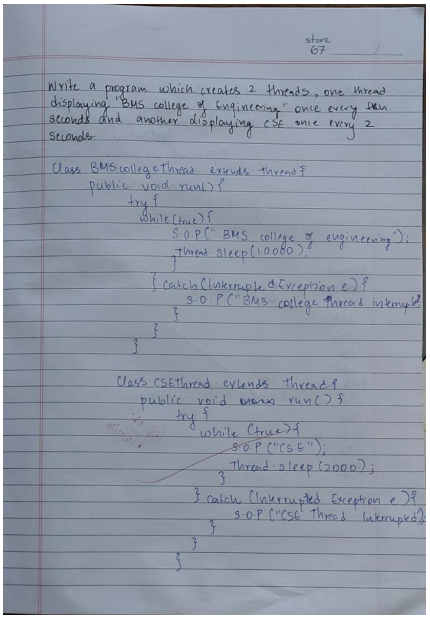
Output:

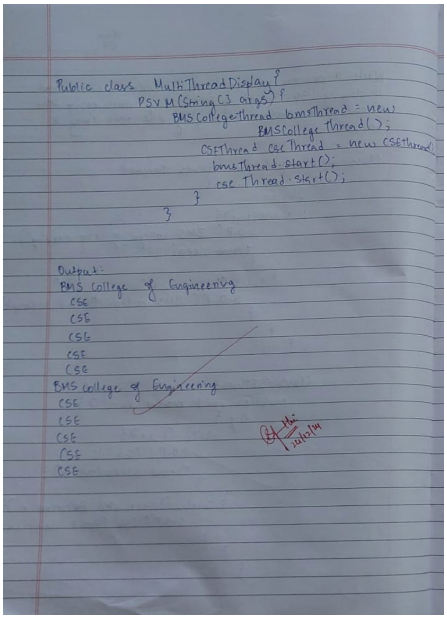
```
C:\jdk\programs>java fatherson
Father's age: 40, Son's age: 20
WrongAge Exception: Father's age cannot be negative!
WrongAge Exception: Son's age cannot be greater than or equal to Father's age!
WrongAge Exception: Son's age cannot be negative!
```

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:



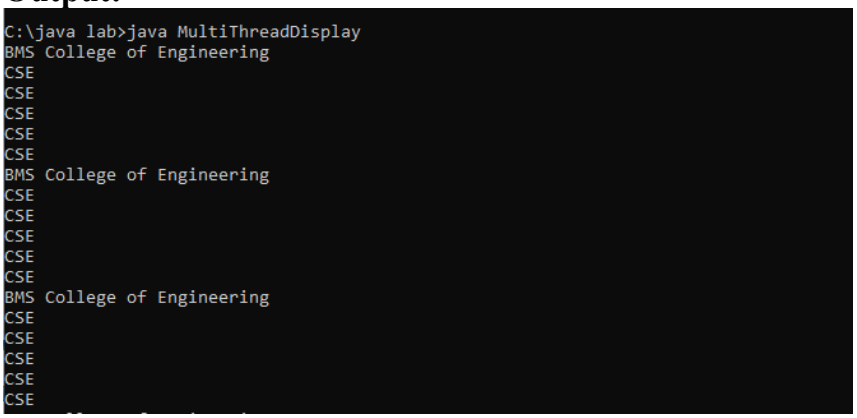


Code:

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

class CSEThread extends Thread {
    public void run() {
        try {
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000); // 2 seconds
            }
        } catch (InterruptedException e) {
            System.out.println("CSEThread interrupted.");
        }
    }
```

```
}

public class MultiThreadDisplay {
   public static void main(String[] args) {
      BMSCollegeThread bmsThread = new BMSCollegeThread();
      CSEThread cseThread = new CSEThread();
      bmsThread.start();
      cseThread.start();
   }
}
```

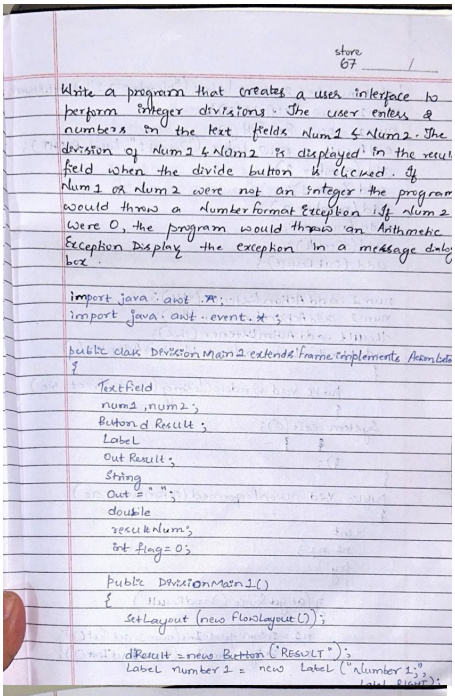
Output:



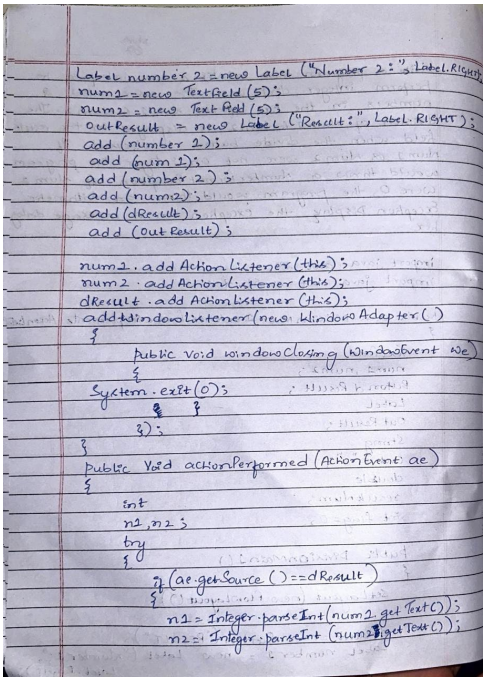
Program 9

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.

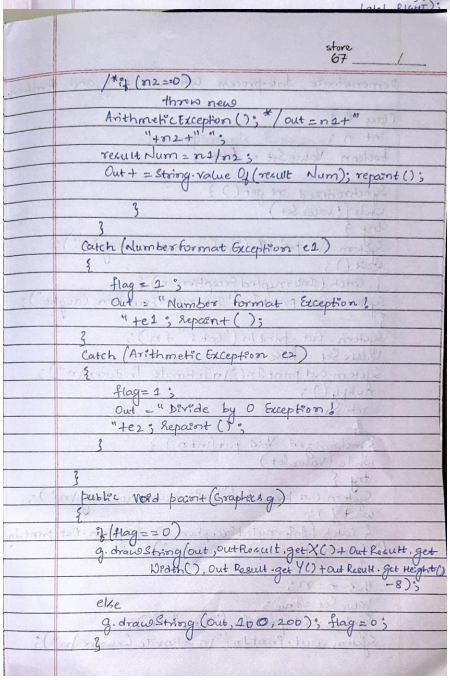
Observation:



```
store
67
Write a program that creates a user interface to
perform integer divisions. The user enters 2
numbers in the text fields Num1 & Num2. The
division of Num1 & 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 Numberformat Exception if Num2
were 0, the program would throw an Arithmetic
Exception Display the exception in a message dialog
box

import java.awt.*;
import java.awt.event.*;

public class DivisionMain2 extends Frame implements ActionLis
{
  TextField
  num1,num2;
  Button dResult;
  Label
  OutResult;
  String
  Out = " ";
  double
  resultNum;
  int flag=0;

  Public DivisionMain1()
  {
    setLayout (new FlowLayout());

    dResult = new Button ("RESULT");
    Label number1 = new Label ("Number 1:",
```



```
Label number 2 = new Label ("Number 2:", Label.RIGHT);
num1 = new TextField (5);
num2 = new TextField (5);
outResult = new Label ("Result:", Label.RIGHT);
add (number 1);
add (num 1);
add (number 2);
add (num2);
add (dResult);
add (outResult);

num1.addActionListener(this);
num2.addActionListener(this);
dResult.addActionListener(this);
addWindowListener(new WindowAdapter()
{
    public void windowClosing (WindowEvent we)
    {
System.exit(0);
    }
});
}
public void actionPerformed (ActionEvent ae)
{
    int
    n1,n2;
    try
    {
      if(ae.getSource()==dResult)
      {
        n1 = Integer.parseInt(num1.getText());
        n2 = Integer.parseInt(num2.getText());
```



```
store
67
/*if (n2==0)
    throw new
ArithmeticException(); */out = n1+"
  "+n2+" ";
resultNum = n1/n2;
Out+ = String.valueOf(resultNum); repaint();

    }
  }
  Catch (NumberFormatException e1)
  {
    flag = 1;
    Out = "Number format Exception!
    "+e1; Repaint();
  }
  Catch (Arithmetic Exception e2)
  {
    flag=1;
    Out = "Divide by 0 Exception!
    "+e2; Repaint();
  }
}
Public void paint(Graphics g)
{
  if(flag==0)
  g.drawString(out, outResult.getX()+outResult.get
    Width(), outResult.getY()+outResult.getHeight()
    -8);
  else
  g.drawString(Out, 100, 200); flag=0;
}
```

Code :

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

public class DivisionMain1 extends Frame implements ActionListener
{
        TextField
        num1,num2; Button
        dResult;
        Label
        outResult;
        String out="";
        double
        resultNum; int
        flag=0;

        public DivisionMain1()
        {
                setLayout(new FlowLayout());

                dResult = new Button("RESULT");
                Label number1 = new Label("Number
                1:",Label.RIGHT); Label number2 = new
                Label("Number 2:",Label.RIGHT);
                num1=new TextField(5);
                num2=new TextField(5);
                outResult = new Label("Result:",Label.RIGHT);

                add(number1);
                add(num1);
                add(number2);
                add(num2);
                add(dResult);
                add(outResult)
                ;
```

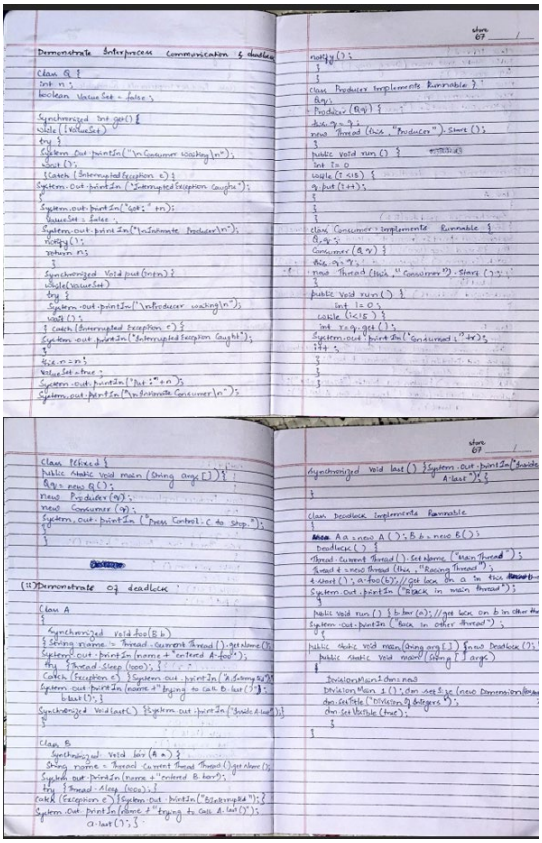
```
        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new
        WindowAdapter()
        {
                public void windowClosing(WindowEvent we)
                {
                        System.exit(0);
                }
        });
}
public void actionPerformed(ActionEvent ae)
{
        int
        n1,n2;
        try
        {
                if (ae.getSource() == dResult)
                {
                        n1=Integer.parseInt(num1.getText());
                        n2=Integer.parseInt(num2.getText());

                        /*if(n2==0)
                                throw new
                        ArithmeticException();*/ out=n1+"
                        "+n2+" ";
                        resultNum=n1/n2;
                        out+=String.valueOf(resultN
                        um); repaint();

                }
        }
        catch(NumberFormatException e1)
        {
                flag=1;
                out="Number Format Exception!
                "+e1; repaint();
        }
        catch(ArithmeticException e2)
```

```
        {
                flag=1;
                out="Divide by 0 Exception!
                "+e2; repaint();
        }

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

Program 10

Demonstrate Interprocess communication and deadlock

Observation:



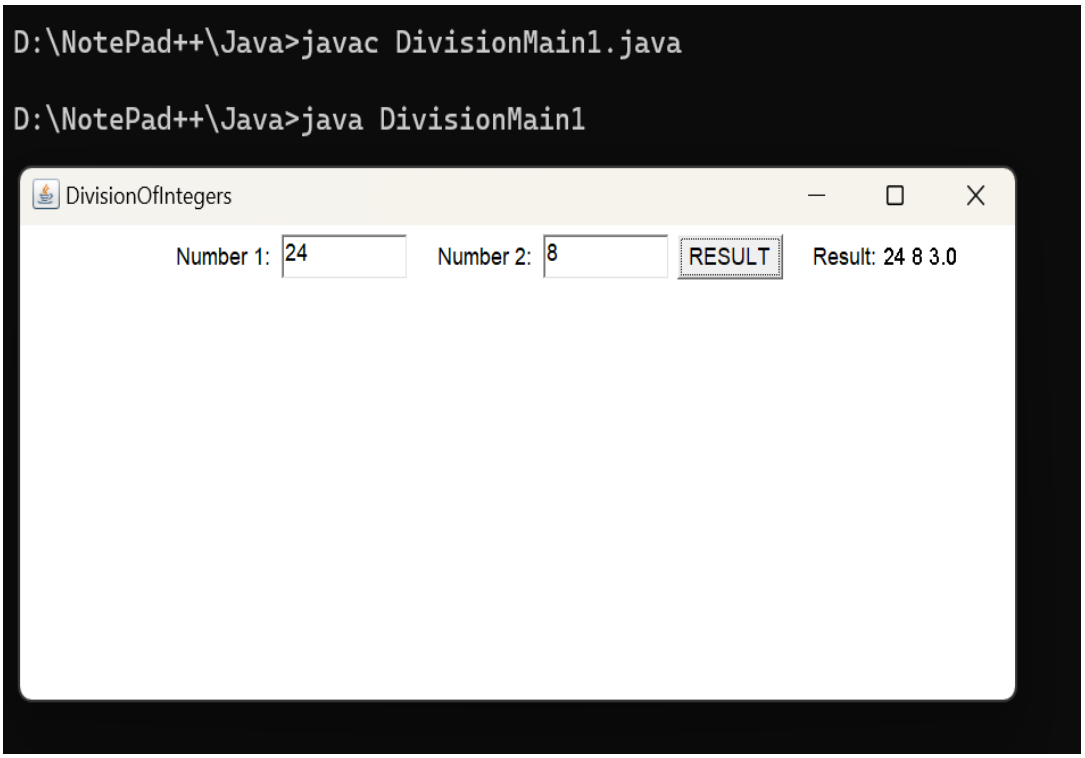
Code:

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

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

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

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

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

```
}
}
}

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

OUTPUT



ii. Demonstration of deadlock

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

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

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

}


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


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