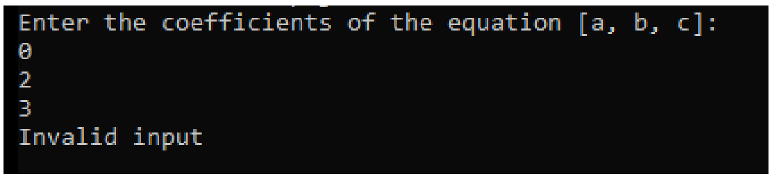
Q1.Develop a Java program that prints all real solutions to the quadratic equation ax 2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b 2 - 4ac is negative, display a message stating that there are no real solutions.

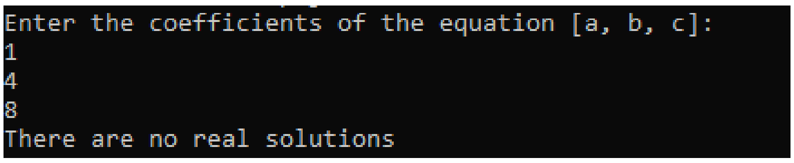
Solution:

```
import java.util.Scanner;

public class Quad {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the coefficients of the equation [a, b, c]:");
        double a = sc.nextDouble();
        double b = sc.nextDouble();
        double c = sc.nextDouble();

        sc.close();
        if (a == 0) {
            System.out.println("Invalid input");
            return;
        }
        double d = b * b - 4 * a * c;
        double r1, r2;
        if (d > 0) {
            System.out.println("The roots are real and distinct");
            r1 = (-b + Math.sqrt(d)) / (2 * a);
            r2 = (-b - Math.sqrt(d)) / (2 * a);
            System.out.println("r1 = " + r1);
            System.out.println("r2 = " + r2);
        } else if (d == 0) {
            System.out.println("The roots are real and equal.");
            r1 = r2 = -b / (2 * a);
            System.out.println("r1 = r2 = " + r1);
        } else {
            System.out.println("There are no real solutions");
        }
    }
}
```

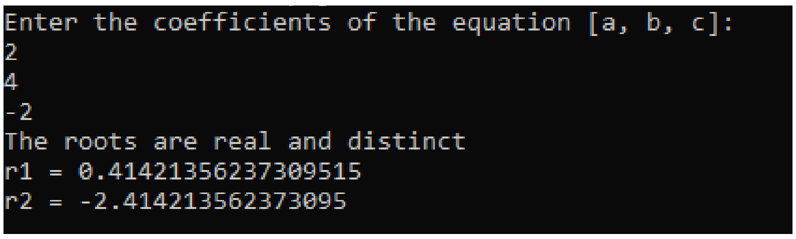
Output:



```
Enter the coefficients of the equation [a, b, c]:
0
2
3
Invalid input
```



```
Enter the coefficients of the equation [a, b, c]:
1
4
8
There are no real solutions
```



```
Enter the coefficients of the equation [a, b, c]:
2
4
-2
The roots are real and distinct
r1 = 0.41421356237309515
r2 = -2.414213562373095
```

Q2. 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 &amp; CGPA of a student.

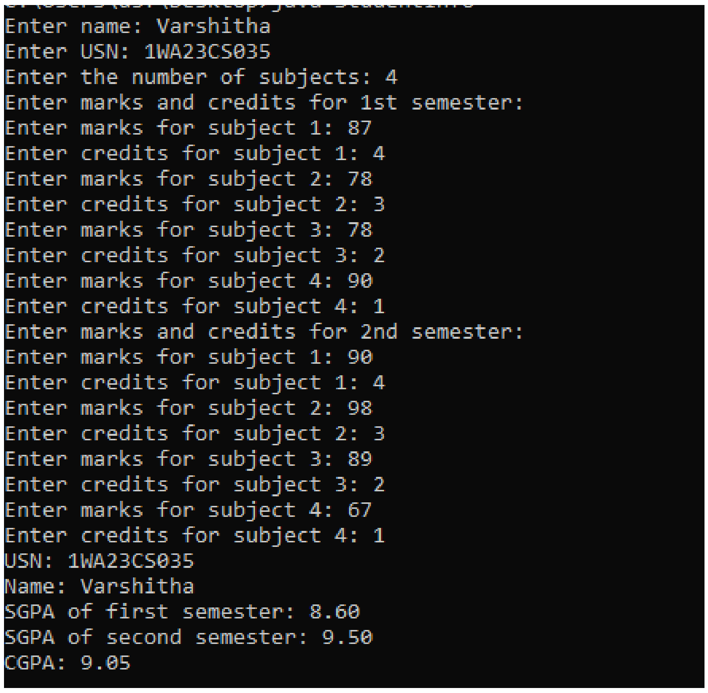
NOTE: Add Semester 1 &amp; Semester 2 marks of all subjects and compute SGPA and CGPA

Solution:

```
import java.util.Scanner;
```

```
class Student {
    String usn;
    String name;
    int[] marks1, marks2;
    int[] credits1, credits2;
    int sub;

    void acceptDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter name: ");
        name = sc.next();
        System.out.print("Enter USN: ");
        usn = sc.next();
        System.out.print("Enter the number of subjects: ");
        sub = sc.nextInt();

        marks1 = new int[sub];
        credits1 = new int[sub];
        marks2 = new int[sub];
        credits2 = new int[sub];

        System.out.println("Enter marks and credits for 1st semester:");
        acceptMarksAndCredits(sc, marks1, credits1);

        System.out.println("Enter marks and credits for 2nd semester:");
        acceptMarksAndCredits(sc, marks2, credits2);
    }

    private void acceptMarksAndCredits(Scanner sc, int[] marks, int[] credits) {
        for (int i = 0; i < sub; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = sc.nextInt();
            while (marks[i] < 0 || marks[i] > 100) {
                System.out.print("Invalid marks (between 0-100). Enter again: ");
                marks[i] = sc.nextInt();
            }
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = sc.nextInt();
            while (credits[i] <= 0) {
                System.out.print("Invalid credits (must be greater than 0). Enter again: ");
                credits[i] = sc.nextInt();
            }
        }
    }

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

    double calcSgpa(int sem) {
        int[] marks = (sem == 1) ? marks1 : marks2;
        int[] credits = (sem == 1) ? credits1 : credits2;

        double gradePoints = 0;
        double totalCredits = 0;

        for (int i = 0; i < sub; i++) {
            gradePoints += convertMarksToGradePoints(marks[i]) * credits[i];
            totalCredits += credits[i];
        }

        return totalCredits == 0 ? 0 : gradePoints / totalCredits;
    }

    double calcCgpa() {
        double sgpa1 = calcSgpa(1);
        double sgpa2 = calcSgpa(2);
        return (sgpa1 + sgpa2) / 2;
    }

    private double convertMarksToGradePoints(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 class StudentInfo {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.acceptDetails();
        s1.displayDetails();
        double sgpa1 = s1.calcSgpa(1);
        double sgpa2 = s1.calcSgpa(2);
        double cgpa = s1.calcCgpa();
        System.out.printf("SGPA of first semester: %.2f\n", sgpa1);
        System.out.printf("SGPA of second semester: %.2f\n", sgpa2);
        System.out.printf("CGPA: %.2f\n", cgpa);
    }
}
```

Output:



```
Enter name: Varshitha
Enter USN: 1WA23CS035
Enter the number of subjects: 4
Enter marks and credits for 1st semester:
Enter marks for subject 1: 87
Enter credits for subject 1: 4
Enter marks for subject 2: 78
Enter credits for subject 2: 3
Enter marks for subject 3: 78
Enter credits for subject 3: 2
Enter marks for subject 4: 90
Enter credits for subject 4: 1
Enter marks and credits for 2nd semester:
Enter marks for subject 1: 90
Enter credits for subject 1: 4
Enter marks for subject 2: 98
Enter credits for subject 2: 3
Enter marks for subject 3: 89
Enter credits for subject 3: 2
Enter marks for subject 4: 67
Enter credits for subject 4: 1
USN: 1WA23CS035
Name: Varshitha
SGPA of first semester: 8.60
SGPA of second semester: 9.50
CGPA: 9.05
```

Q3.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. Display the complete details of the book. Develop a Java program to create n book objects.

NOTE: 1: Use normal display method

2: Use Override toString method

Eg: public String toString() {Write code to display data }

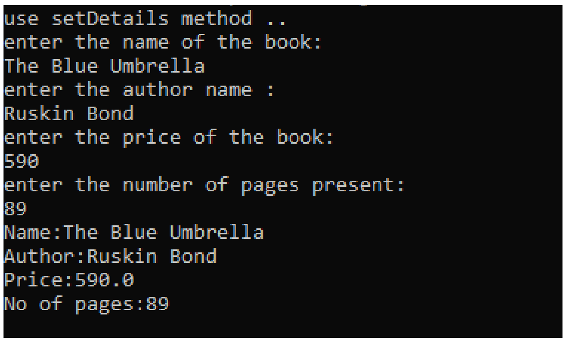
Solution:

```
import java.util.Scanner;
```

```
class Book{
        String name;
        String author;
        double price;
        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 Book()
        {
        System.out.println("use setDetails method ..");
        }
        public void setDetails()
        {
                Scanner sc  = new Scanner(System.in);
                System.out.println("enter the name of the book:");
                this.name  = sc.nextLine();
                System.out.println("enter the author name :");
                this.author = sc.nextLine();
                System.out.println("enter the price of the book:");
                this.price = sc.nextDouble();
                System.out.println("enter the number of pages present:");
                this.num_pages = sc.nextInt();
        }
        public void getDetails()
        {
```

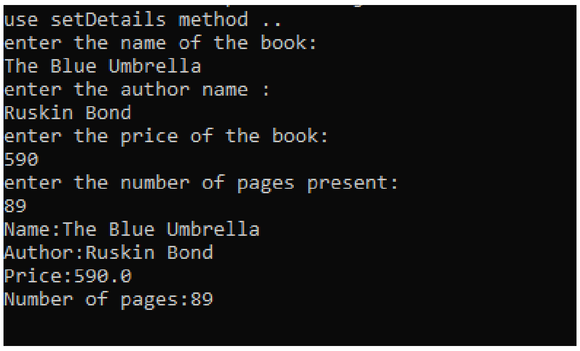
```
		System.out.println("Name:"+this.name);
		System.out.println("Author:"+this.author);
		System.out.println("Price:"+this.price);
		System.out.println("No of pages:"+this.num_pages);
	}
	public String toString(){
			return
"Name:"+this.name+"\nAuthor:"+this.author+"\nPrice:"+this.price+"\nNumber of
pages:"+this.num_pages+"\n";
	}
}

public class BookDetails{
	public static void main(String args[]){
		Book b[] = new Book[1];
		for(int i=0;i<1;i++){
				b[i] = new Book();
				b[i] .setDetails();
		}
		for(int i=0;i<1;i++){
			//b[i].getDetails();
			System.out.println(b[i]);
		}
	}
}
```
Output:

1.using normal display method



```
use setDetails method ..
enter the name of the book:
The Blue Umbrella
enter the author name :
Ruskin Bond
enter the price of the book:
590
enter the number of pages present:
89
Name:The Blue Umbrella
Author:Ruskin Bond
Price:590.0
No of pages:89
```

2. using toString method



```
use setDetails method ..
enter the name of the book:
The Blue Umbrella
enter the author name :
Ruskin Bond
enter the price of the book:
590
enter the number of pages present:
89
Name:The Blue Umbrella
Author:Ruskin Bond
Price:590.0
Number of pages:89
```

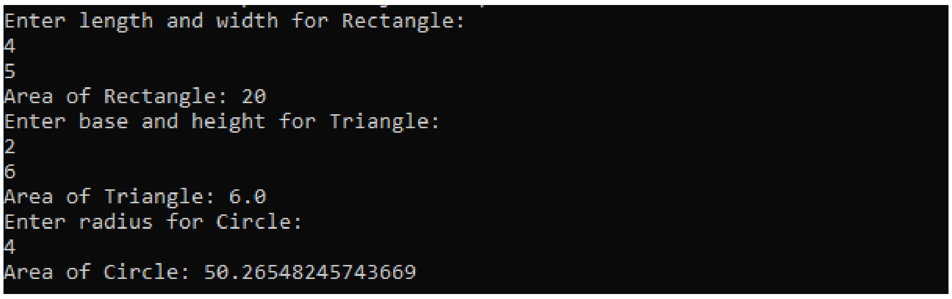
Q4. Develop a Java program to create an abstract class named Shape that contains two Integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the Classes contain only the method printArea( ) that prints the area of the given shape.

Solution:

```
import java.util.Scanner;
abstract class Shape {
int dim1;
int dim2;
Shape()
{
this.dim1 = 0;
this.dim2 = 0;
}
Shape(int dim1, int dim2) {
this.dim1 = dim1;
this.dim2 = dim2;
}
abstract void printArea();
}
class Rectangle extends Shape
{
Rectangle(int length, int width)
{
super(length, width);
}
void printArea()
{
int area = dim1 * dim2;
System.out.println("Area of Rectangle: " + area);
}


}
class Triangle extends Shape {
Triangle(int base, int height) {
super(base, height);
}
void printArea() {
double area = 0.5 * dim1 * dim2;
System.out.println("Area of Triangle: " + area);
}
}
class Circle extends Shape {
Circle(int radius) {
super(radius, 0);
}
```

```
void printArea() {
double area = Math.PI * dim1 * dim1;
System.out.println("Area of Circle: " + area);
}
}
public class Shapes {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
System.out.println("Enter length and width for Rectangle:");
int length = sc.nextInt();
int width = sc.nextInt();
Rectangle a1 = new Rectangle(length, width);
a1.printArea();
System.out.println("Enter base and height for Triangle:");
int base = sc.nextInt();
int height = sc.nextInt();
Triangle a2= new Triangle(base, height);
a2.printArea();
System.out.println("Enter radius for Circle:");
int radius = sc.nextInt();
Circle a3 = new Circle(radius);

a3.printArea();
}
}
```

Output:


```
Enter length and width for Rectangle:
4
5
Area of Rectangle: 20
Enter base and height for Triangle:
2
6
Area of Triangle: 6.0
Enter radius for Circle:
4
Area of Circle: 50.26548245743669
```

Q5. 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.

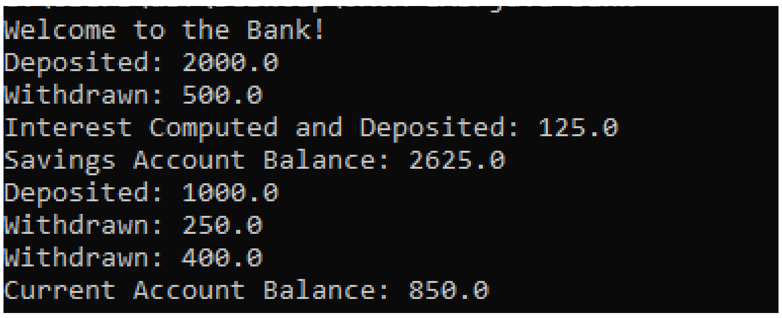
Solution:

```
import java.util.Scanner;
abstract class Account
{
String customerName;
String accountNumber;
double balance;
public Account(String customerName, String accountNumber, double
initialBalance) {
this.customerName = customerName;
this.accountNumber = accountNumber;
this.balance = initialBalance;
}
abstract void deposit(double amount);
abstract void displayBalance();
abstract void withdraw(double amount);
}
class SavingsAccount extends Account
{
double interestRate;

public SavingsAccount(String customerName, String accountNumber, double
initialBalance, double interestRate)
{
super(customerName, accountNumber, initialBalance);
this.interestRate = interestRate;
}
@Override
public void deposit(double amount)
{
balance += amount;
System.out.println("Deposited: " + amount);
}
@Override
public void displayBalance() {
System.out.println("Savings Account Balance: " + balance);
}
public void computeAndDepositInterest() {
```

```
double interest = balance * interestRate / 100;
balance += interest;
System.out.println("Interest Computed and Deposited: " + interest);
}
@Override
public void withdraw(double amount) {
if (amount <= balance) {
balance -= amount;
System.out.println("Withdrawn: " + amount);
}
else {
System.out.println("Insufficient balance!");
}
}
}

class CurrentAccount extends Account {
private double minimumBalance;
private double serviceCharge;
public CurrentAccount(String customerName, String accountNumber,
double initialBalance, double minimumBalance, double serviceCharge) {
super(customerName, accountNumber, initialBalance);
this.minimumBalance = minimumBalance;
this.serviceCharge = serviceCharge;
}
@Override
public void deposit(double amount) {
balance += amount;
System.out.println("Deposited: " + amount);
}
```

```
@Override
public void displayBalance() {
System.out.println("Current Account Balance: " + balance);
}
@Override
public void withdraw(double amount) {
if (amount <= balance) {
balance -= amount;
System.out.println("Withdrawn: " + amount);
checkMinimumBalance();
} else {
System.out.println("Insufficient balance!");
}
}
private void checkMinimumBalance() {
if (balance < minimumBalance) {

balance -= serviceCharge;
System.out.println("Service charge imposed: " + serviceCharge);
}
}
}
class Bank {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Welcome to the Bank!");
SavingsAccount savingsAccount = new SavingsAccount("Alice",
"S123", 1000.0, 5.0);
CurrentAccount currentAccount = new CurrentAccount("Bob", "C456",
500.0, 300.0, 20.0);
savingsAccount.deposit(2000);
savingsAccount.withdraw(500);
savingsAccount.computeAndDepositInterest();
savingsAccount.displayBalance();

currentAccount.deposit(1000);
currentAccount.withdraw(250);
currentAccount.withdraw(400);
currentAccount.displayBalance();
scanner.close();
}
}
```

Output:



```
Welcome to the Bank!
Deposited: 2000.0
Withdrawn: 500.0
Interest Computed and Deposited: 125.0
Savings Account Balance: 2625.0
Deposited: 1000.0
Withdrawn: 250.0
Withdrawn: 400.0
Current Account Balance: 850.0
```

Q6.Create a package CIE which has two classes- Student and Internals. The class Student 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

Solution:

```
package CIE;

import java.util.Scanner;
```

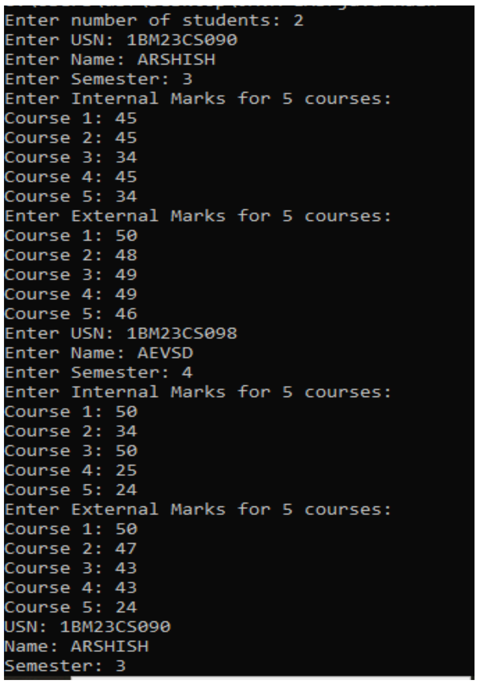
```
public class Student {
protected String usn;
protected String name;
protected int sem;
public void inputStudentDetails(){
Scanner s = new Scanner(System.in);
System.out.print("Enter USN: ");
this.usn = s.nextLine();
System.out.print("Enter Name: ");
this.name = s.nextLine();
System.out.print("Enter Semester: ");
this.sem = s.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 s = new Scanner(System.in);
System.out.println("Enter Internal Marks for 5 courses: ");
for (int i = 0; i < 5; i++) {
System.out.print("Course " + (i + 1) + ": ");
marks[i] = s.nextInt();
```

```
}
}
public void displayCIEmarks() {
System.out.println("Internal Marks: ");
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() {
externalMarks = new int[5];
finalMarks = new int[5];
}
public void inputSEEmarks() {
Scanner s = new Scanner(System.in);
System.out.println("Enter External Marks for 5 courses: ");
for (int i = 0; i < 5; i++) {
System.out.print("Course " + (i + 1) + ": ");
externalMarks[i] = s.nextInt();
}
}
public void calculateFinalMarks() {
for (int i = 0; i < 5; i++) {
```

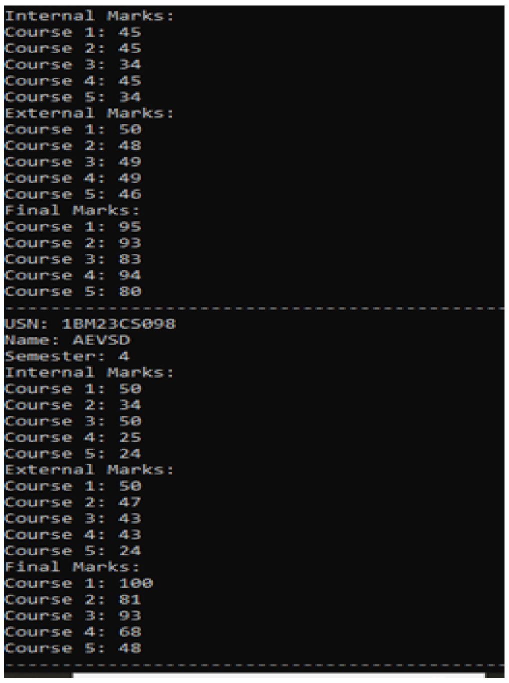
```
finalMarks[i] = marks[i] + externalMarks[i]; // Internal + External
}
}
public void displayFinalMarks() {
displayStudentDetails();
displayCIEmarks();
System.out.println("External Marks: ");
for (int i = 0; i < 5; i++) {
System.out.println("Course " + (i + 1) + ": " + externalMarks[i]);
}
System.out.println("Final Marks: ");
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 sc = new Scanner(System.in);
System.out.print("Enter number of students: ");
int n = sc.nextInt();

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

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:



```
Enter number of students: 2
Enter USN: 1BM23CS090
Enter Name: ARSHISH
Enter Semester: 3
Enter Internal Marks for 5 courses:
Course 1: 45
Course 2: 45
Course 3: 34
Course 4: 45
Course 5: 34
Enter External Marks for 5 courses:
Course 1: 50
Course 2: 48
Course 3: 49
Course 4: 49
Course 5: 46
Enter USN: 1BM23CS098
Enter Name: AEVSD
Enter Semester: 4
Enter Internal Marks for 5 courses:
Course 1: 50
Course 2: 34
Course 3: 50
Course 4: 25
Course 5: 24
Enter External Marks for 5 courses:
Course 1: 50
Course 2: 47
Course 3: 43
Course 4: 43
Course 5: 24
USN: 1BM23CS090
Name: ARSHISH
Semester: 3
```



```
Internal Marks:
Course 1: 45
Course 2: 45
Course 3: 34
Course 4: 45
Course 5: 34
External Marks:
Course 1: 50
Course 2: 48
Course 3: 49
Course 4: 49
Course 5: 46
Final Marks:
Course 1: 95
Course 2: 93
Course 3: 83
Course 4: 94
Course 5: 80
------------------------------------------
USN: 1BM23CS098
Name: AEVSD
Semester: 4
Internal Marks:
Course 1: 50
Course 2: 34
Course 3: 50
Course 4: 25
Course 5: 24
External Marks:
Course 1: 50
Course 2: 47
Course 3: 43
Course 4: 43
Course 5: 24
Final Marks:
Course 1: 100
Course 2: 81
Course 3: 93
Course 4: 68
Course 5: 48
------------------------------------------
```

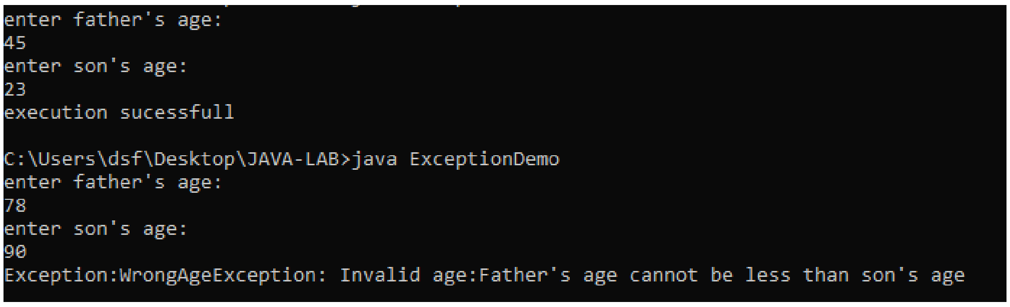
Q7. Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age&lt;0. In Son class, implement a constructor that uses both father and son's age and throws an exception if son's age is &gt;=father's age.

//WrongeAgeException.java

```
public class WrongAgeException extends Exception{
            public WrongAgeException(String message)
                        {
                        super(message);
                        }
            }
//Q1.Father.java
import java.util.Scanner;
class Father{
            int FatherAge;

            public Father(int FatherAge) throws WrongAgeException{
            if(FatherAge<=0){
            throw new WrongAgeException("Invalid age:age cannot be zero");
                                    }
                        this.FatherAge = FatherAge;
                        }
            }
class Son extends Father{
             int SonAge;

            public Son(int FatherAge,int SonAge) throws WrongAgeException
            {
                        super(FatherAge);
                        if(SonAge<=0){
            throw new WrongAgeException("Invalid age:age cannot be zero");
                                    }
                        this.SonAge = SonAge;
            }
            public void compute() throws WrongAgeException{
                        if(this.FatherAge<=this.SonAge){
                                    throw new WrongAgeException("Invalid age:Father's age cannot
be less than son's age");
                                    }
                        }
            }
public class Q1{
            public static void main(String args[]){
                        Scanner sc = new Scanner(System.in);
                        try{
                                    System.out.println("enter father's age:");
                                    int a = sc.nextInt();
                                    Father f1 = new Father(a);
                                    System.out.println("enter son's age:");
                                    int b = sc.nextInt();
                                    Son s1  = new Son(a,b);
                                    s1.compute();
                                    System.out.println("execution sucessfull");
                                    }catch(WrongAgeException e){
                                    System.out.println("Exception:"+e);
```

```
                    }
            }
    }
```

Output:



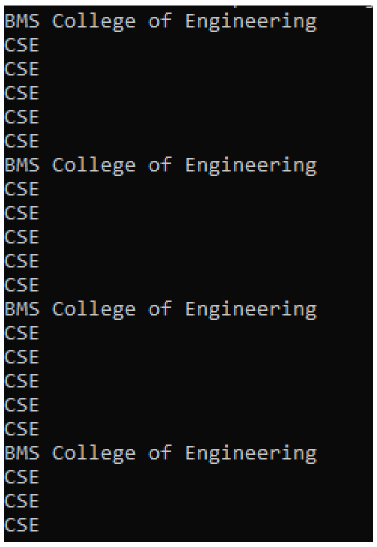
```
enter father's age:
45
enter son's age:
23
execution sucessfull

C:\Users\dsf\Desktop\JAVA-LAB>java ExceptionDemo
enter father's age:
78
enter son's age:
90
Exception:WrongAgeException: Invalid age:Father's age cannot be less than son's age
```

Q8. 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.\

Solution:

```
class MyThread implements Runnable{
        Thread t;
        String message;
        int interval;
        MyThread(String message,int interval)
        {
                this.message = message;
                this.interval = interval;
                t = new Thread(this,message);
                t.start();
        }

        public void run()
        {

                try{
                while(true){
                System.out.println(message);
                Thread.sleep(interval);
                }
                }catch(InterruptedException e){
                        System.out.println("Thread interrupted..");
                        }
                }
        }
class Q6{
        public static void main(String args[]){
                new MyThread("BMS College of Engineering",10000);
```

```
                        new MyThread("CSE",2000);

                                    }
                        }
```
Output:

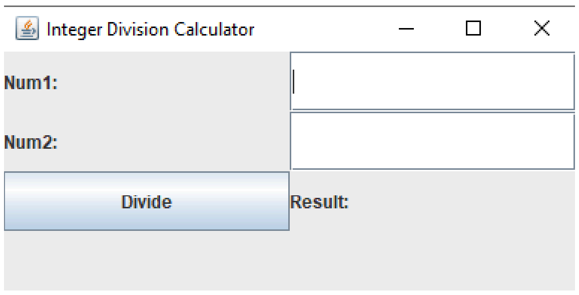


```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
```

Q9. 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.(Open ended program).

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

public class DivisionApp {
    public static void main(String[] args) {

        JFrame frame = new JFrame("Integer Division Calculator");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 200);


        JLabel num1Label = new JLabel("Num1:");
        JTextField num1Field = new JTextField(10);

        JLabel num2Label = new JLabel("Num2:");
        JTextField num2Field = new JTextField(10);
        JButton divideButton = new JButton("Divide");
        JLabel resultLabel = new JLabel("Result: ");

        divideButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    int num1 = Integer.parseInt(num1Field.getText());
                    int num2 = Integer.parseInt(num2Field.getText());

                    int result = num1 / num2;

                    resultLabel.setText("Result: " + result);
                } catch (NumberFormatException nfe) {
                    JOptionPane.showMessageDialog(frame, "Please enter valid integers.", "Input
Error", JOptionPane.ERROR_MESSAGE);
                } catch (ArithmeticException ae) {
                    JOptionPane.showMessageDialog(frame, "Division by zero is not allowed.",
"Arithmetic Error", JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(4, 2));

        panel.add(num1Label);
        panel.add(num1Field);
        panel.add(num2Label);
        panel.add(num2Field);
        panel.add(divideButton);
        panel.add(resultLabel);

        frame.add(panel);


        frame.setVisible(true);
    }
}
```

Output:

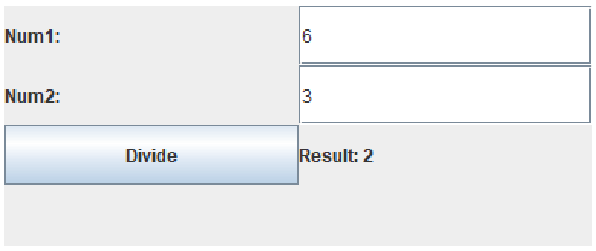


Integer Division Calculator

Num1:

Num2:

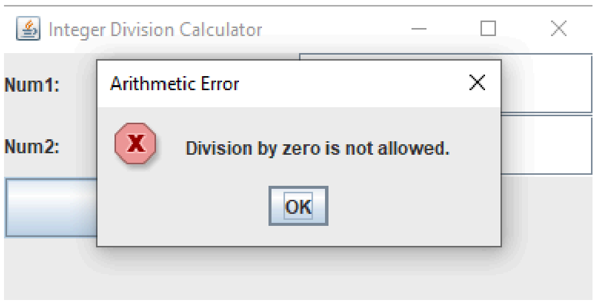
Result:



Num1: 6

Num2: 3

Result: 2



Integer Division Calculator

Num1:

Num2:

Arithmetic Error

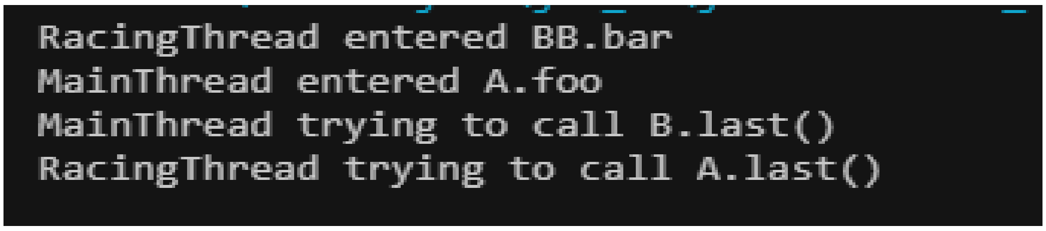
Division by zero is not allowed.

Q10.Demonstrtate Interprocess Communication and Deadloack.

```
class AA {
  synchronized void foo(BB 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 BB {
  synchronized void bar(AA a) {
    String name = Thread.currentThread().getName();
        System.out.println(name + " entered BB.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 {
        AA a = new AA();
        BB b = new BB();
        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();
	}
}
```

Output:



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