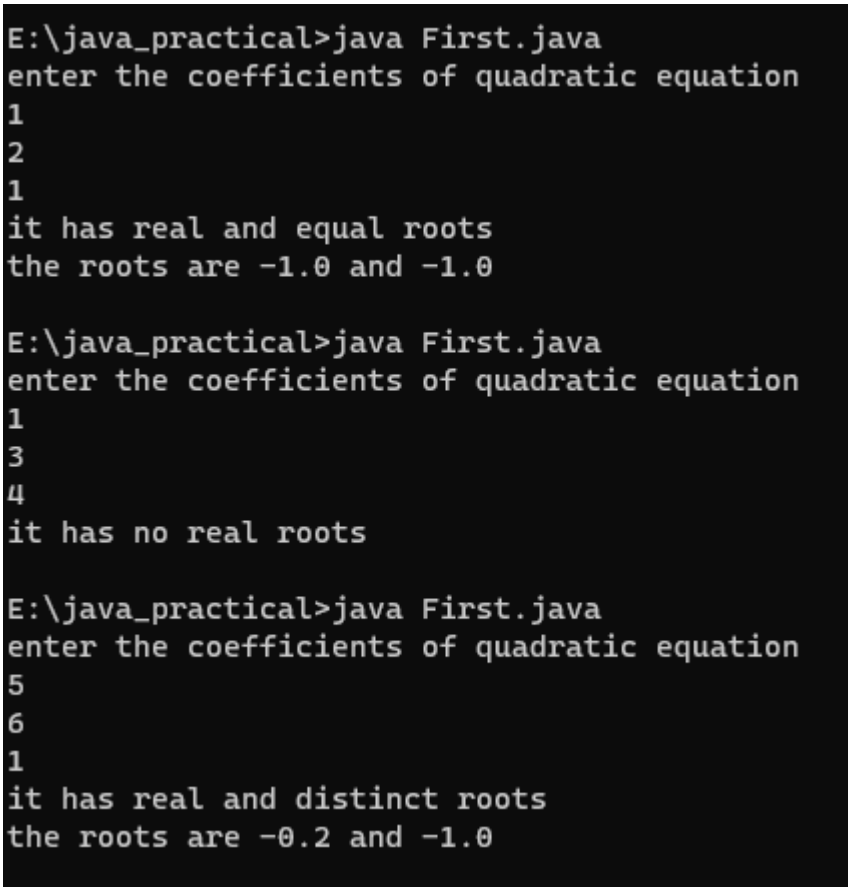
1) **Develop a Java program that prints all real solutions to the quadratic equation ax+bx+c= 0. Read in a, b, c and use the quadratic formula. If the discriminate b-4ac is negative, display a message stating that there are no real solutions.**

CODE:=

```
import java.util.Scanner;
class First{
public static void main(String args[]){
int a,b,c;
double r1,r2,d;
System.out.println("enter the coefficients of quadratic equation");
Scanner s1= new Scanner(System.in);
a=s1.nextInt();
b=s1.nextInt();
c=s1.nextInt();
if(a==0){
System.out.println("coefficients are invalid");
}
else{
d=b*b-(4*a*c);
if(d>0){
System.out.println("it has real and distinct roots");
r1=((-b+Math.sqrt(d)))/(2*a);
r2=((-b-Math.sqrt(d)))/(2*a);
System.out.println("the roots are "+r1+" and "+r2);
}
else if(d==0){
System.out.println("it has real and equal roots");
r1=(-b)/(2*a);
System.out.println("the roots are "+r1+" and "+r1);
}
else{
```

```
System.out.println("it has no real roots");
}
}
}
}
```

Output:



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;
class Student {
   String usn;
   String name;
   int[] credits = new int[5];
   int[] marks = new int[5];

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

      System.out.println("Enter marks for 5 subjects:");
      for (int i = 0; i < 5; i++) {
         System.out.print("Subject " + (i + 1) + " credits: ");
         credits[i] = scanner.nextInt();
         System.out.print("Subject " + (i + 1) + " marks: ");
         marks[i] = scanner.nextInt();
      }
      scanner.close();
   }

   void displayDetails() {
      System.out.println("USN: " + usn);
      System.out.println("Name: " + name);
      for (int i = 0; i < 5; i++) {
         System.out.println("Subject " + (i + 1) + ": Credits=" + credits[i]
+ ", Marks=" + marks[i]);
      }
   }

   double calculateSGPA() {
      double totalCredits = 0, totalGradePoints = 0;
      for (int i = 0; i < 5; i++) {
         totalCredits += credits[i];
         totalGradePoints += calculateGradePoint(marks[i]) * credits[i];
      }
```

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

OUTPUT:=

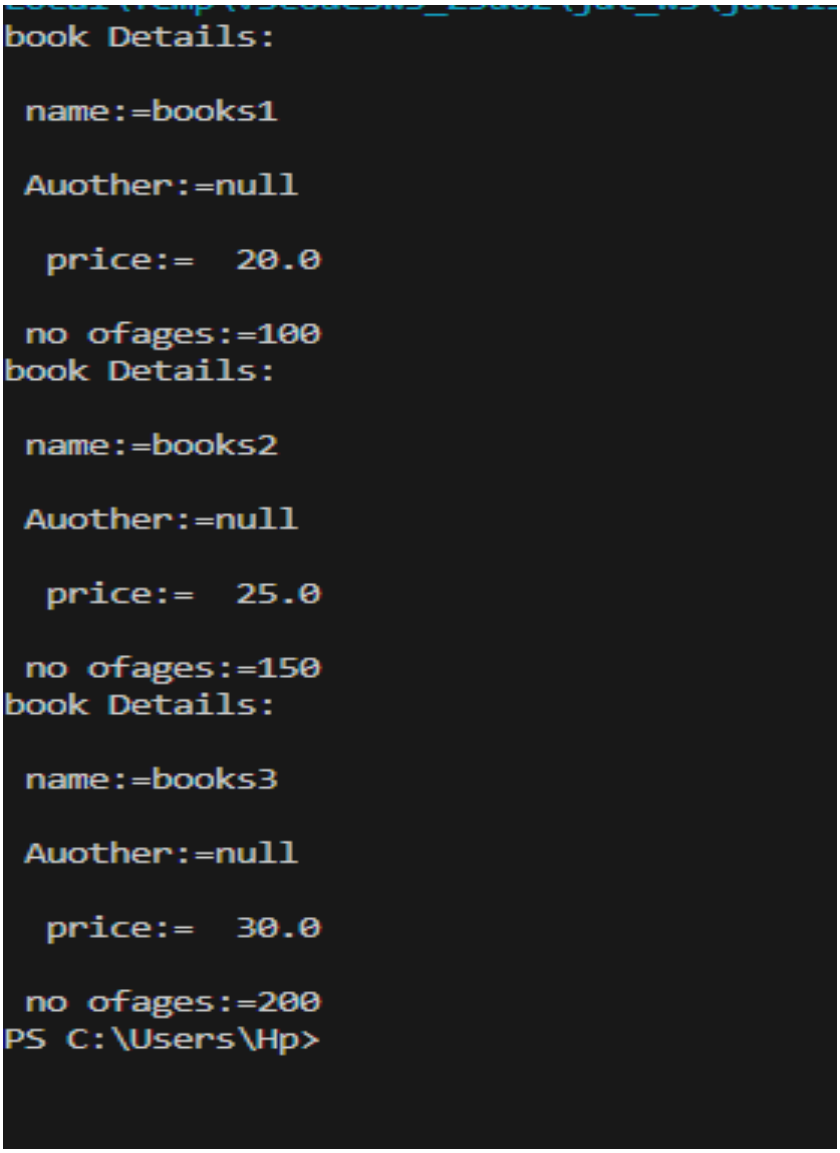
```
book Details:

 name:=books1

 Auother:=null

  price:=  20.0

 no ofages:=100
book Details:

 name:=books2

 Auother:=null

  price:=  25.0

 no ofages:=150
book Details:

 name:=books3

 Auother:=null

  price:=  30.0

 no ofages:=200
```

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:=

```
public class Book{
String name;
String author;
double price;
int numpages;

public Book(String name,String auother,double price,int
numpages){
this.name=name;
this.author=author;
this.price=price;
this.numpages=numpages;}

public void setDetais(String name,String author,double price, int
numpages){
this.name=name;
this.author=author;
this.price=price;
this.numpages=numpages;}

public String toString(){
return "book Details: \n \n name:="+ name +" \n \n
Auother:="+author+"\n \n  price:=  "+price+"\n \n no
ofages:="+numpages;
}
public static void main (String args[]){
int n=3;
Book[] books=new Book[n];
for(int i=0;i<n;i++){
books[i]=new
Book("books"+(i+1),"Auother"+(i+1),20.0+i*5,100+i*50);
}
for(int i=0;i<n;i++){
System.out.println(books[i].toString());
}
}
}
```

OUTPUT:=


book Details:
name:=books1
Auother:=null
price:= 20.0
no ofages:=100
book Details:
name:=books2
Auother:=null
price:= 25.0
no ofages:=150
book Details:
name:=books3
Auother:=null
price:= 30.0
no ofages:=200
PS C:\Users\Hp>

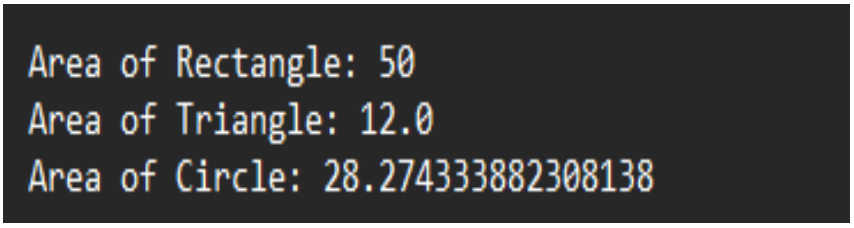
4) Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea()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

Code:=

```
abstract class Shape {
    protected int side1;
    protected int side2;

    public Shape(int side1, int side2) {
        this.side1 = side1;
        this.side2 = side2;
    }

    public abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }

    @Override
    public void printArea() {
        System.out.println("Area of Rectangle: " + (side1 *
side2));
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }

    @Override
    public void printArea() {
        System.out.println("Area of Triangle: " + (0.5 * side1
* side2));
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        super(radius, radius);
    }

    @Override
    public void printArea() {
        System.out.println("Area of Circle: " + (Math.PI *
side1 * side2));
    }
}
```

```
public class
shape
{
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle(5, 10);
        rectangle.printArea();

        Triangle triangle = new Triangle(4, 6);
        triangle.printArea();

        Circle circle = new Circle(3);
        circle.printArea();
    }
}
```

OUTPUT-



5)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:

1)Accept deposit from customer and update the balance.
2)Display the balance.
3)Compute and deposit interest
4)Permit withdrawal and update the balance

code:=

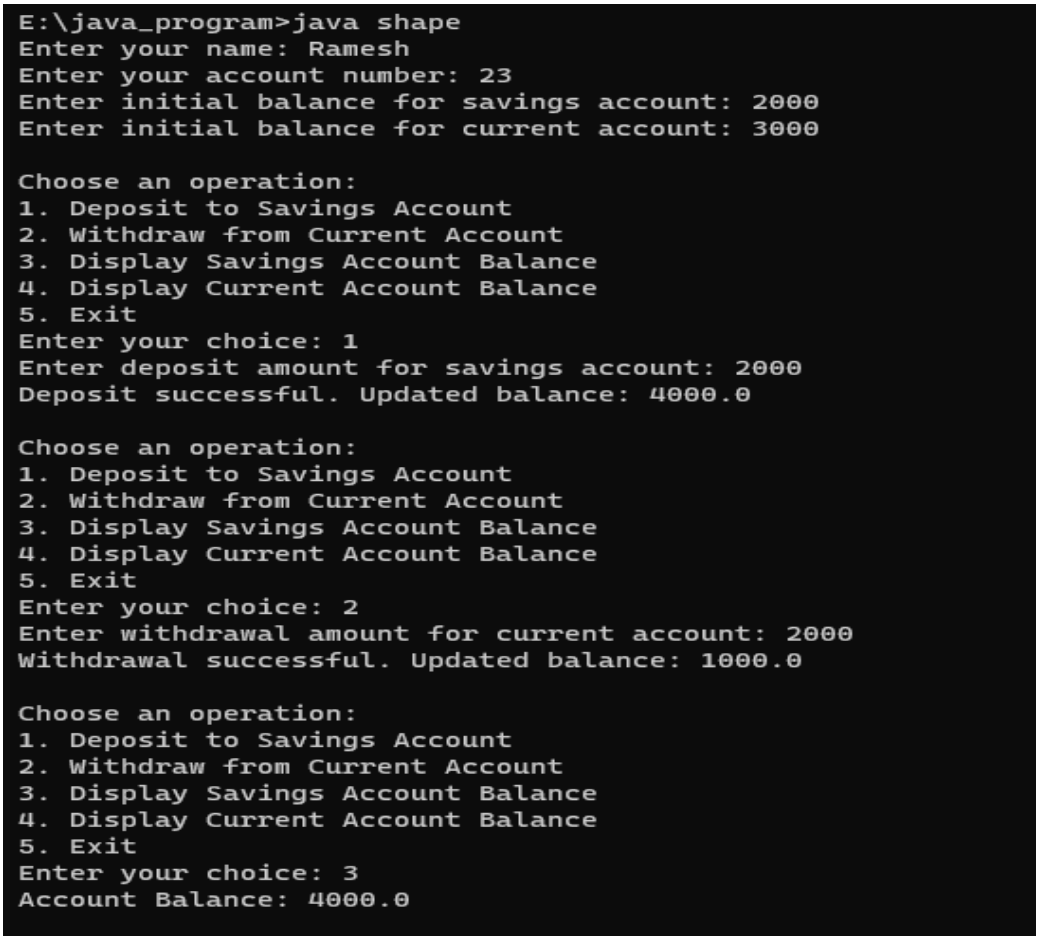
```
import java.util.Scanner;
class Account {
   private String customerName;
   private int accountNumber;
   private String accountType;
   protected double balance;

   public Account(String name, int accNo, String type, double
initialBalance) {
      customerName = name;
      accountNumber = accNo;
      accountType = type;
      balance = initialBalance;
   }

   public void deposit(double amount) {
      balance += amount;
      System.out.println("Deposit successful. Updated balance: " +
balance);
   }

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

   public void computeInterest() {
      // Implemented in child classes
   }

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

```
class CurrentAccount extends Account {
    private double minimumBalance;
    private double serviceCharge;

    public CurrentAccount(String name, int accNo, double
initialBalance) {
        super(name, accNo, "Current", initialBalance);
        minimumBalance = 1000; // example minimum balance
        serviceCharge = 50; // example service charge
    }

    @Override
    public void withdraw(double amount) {
        if (balance - amount >= minimumBalance) {
            balance -= amount;
            System.out.println("Withdrawal successful. Updated balance: "
+ balance);
        } else {
            System.out.println("Insufficient funds. Service charge of " +
serviceCharge + " applied.");
            balance -= serviceCharge;
        }
    }

    @Override
    public void computeInterest() {
        // Current account does not earn interest
        System.out.println("Current account does not earn interest.");
    }
}

class SavingsAccount extends Account {
    private double interestRate;

    public SavingsAccount(String name, int accNo, double
initialBalance) {
        super(name, accNo, "Savings", initialBalance);
        interestRate = 0.05; // example interest rate (5%)
    }

    @Override
    public void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest computed and deposited. Updated
balance: " + balance);
    }
}
```

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

      // Prompt user to enter personal information
      System.out.print("Enter your name: ");
      String name = scanner.nextLine();
      System.out.print("Enter your account number: ");
      int accountNumber = scanner.nextInt();

      // Creating savings account
      System.out.print("Enter initial balance for savings account: ");
      double savingsInitialBalance = scanner.nextDouble();
      SavingsAccount savingsAccount = new SavingsAccount(name,
accountNumber, savingsInitialBalance);

      // Creating current account
      System.out.print("Enter initial balance for current account: ");
      double currentInitialBalance = scanner.nextDouble();
      CurrentAccount currentAccount = new CurrentAccount(name,
accountNumber, currentInitialBalance);

      // Perform operations
      boolean exit = false;
      while (!exit) {
         System.out.println("\nChoose an operation:");
         System.out.println("1. Deposit to Savings Account");
         System.out.println("2. Withdraw from Current Account");
         System.out.println("3. Display Savings Account Balance");
         System.out.println("4. Display Current Account Balance");
         System.out.println("5. Exit");

         System.out.print("Enter your choice: ");
         int choice = scanner.nextInt();

         switch (choice) {
            case 1:
               System.out.print("Enter deposit amount for savings
account: ");
               double savingsDepositAmount = scanner.nextDouble();
               savingsAccount.deposit(savingsDepositAmount);
               break;
            case 2:
               System.out.print("Enter withdrawal amount for current
account: ");
               double currentWithdrawalAmount =
scanner.nextDouble();
               currentAccount.withdraw(currentWithdrawalAmount);
               break;
            case 3:
```

```
                savingsAccount.displayBalance();
                break;
            case 4:
                currentAccount.displayBalance();
                break;
            case 5:
                exit = true;
                break;
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }

    scanner.close();
  }
}
```

Output:=



```
Choose an operation:
1. Deposit to Savings Account
2. Withdraw from Current Account
3. Display Savings Account Balance
4. Display Current Account Balance
5. Exit
Enter your choice: 4
Account Balance: 1000.0

Choose an operation:
1. Deposit to Savings Account
2. Withdraw from Current Account
3. Display Savings Account Balance
4. Display Current Account Balance
5. Exit
Enter your choice: 5
```

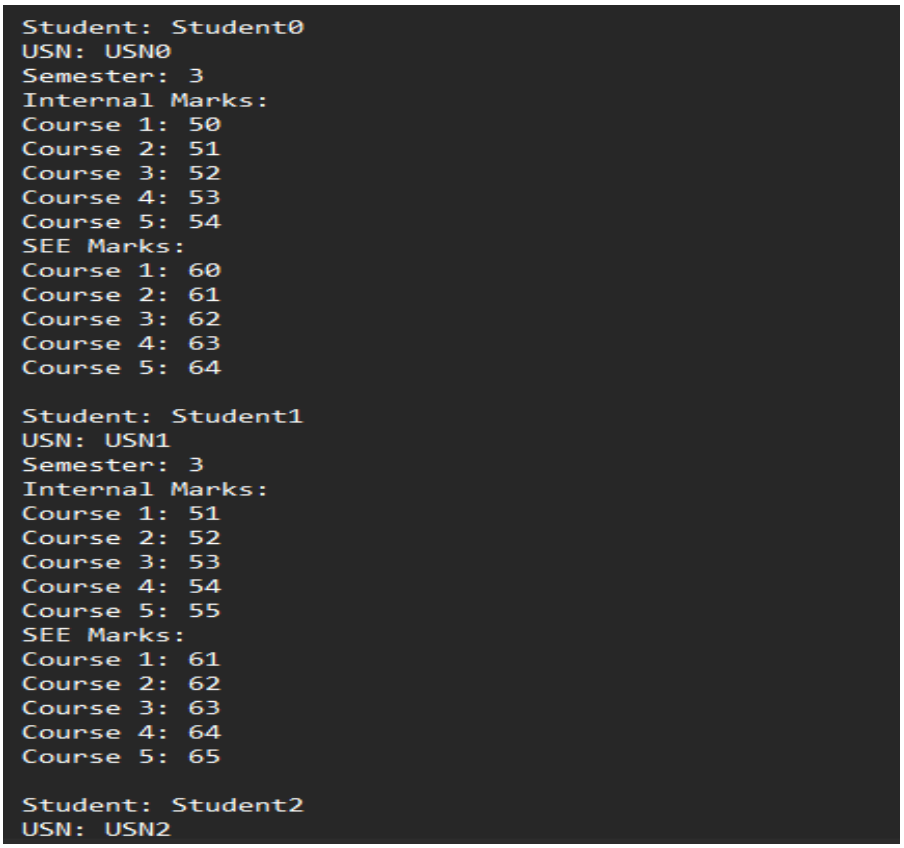
6) Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External 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 {
    protected String usn;
    protected String name;
    protected int sem;
}

package CIE;

public class Internals {
    protected int[] internalMarks = new int[5];
}
package SEE;
import CIE.Student;

public class External extends Student {
    protected int[] seeMarks = new int[5];
}
import CIE.Student;
import CIE.Internals;
import SEE.External;

public class MainFile {
    public static void main(String[] args) {
        int n = 5;

        External[] students = new External[n];

        for (int i = 0; i < n; i++) {
            students[i] = new External();
            students[i].usn = "USN" + i;
            students[i].name = "Student" + i;
            students[i].sem = 3; // Assuming all students are in the same
semester

            for (int j = 0; j < 5; j++) {
                students[i].internalMarks[j] = 50 + i;
            }
            for (int j = 0; j < 5; j++) {
                students[i].seeMarks[j] = 60 + i;
            }
```

```
        }
        for (int i = 0; i < n; i++) {
            System.out.println("Student: " + students[i].name);
            System.out.println("USN: " + students[i].usn);
            System.out.println("Semester: " + students[i].sem);

            System.out.println("Internal Marks:");
            for (int j = 0; j < 5; j++) {
                System.out.println("Course " + (j + 1) + ": " +
students[i].internalMarks[j]);
            }
            System.out.println("SEE Marks:");
            for (int j = 0; j < 5; j++) {
                System.out.println("Course " + (j + 1) + ": " +
students[i].seeMarks[j]);
            }
            System.out.println();
        }
    }
}
```

OUTPUT:=



```
Student: Student0
USN: USN0
Semester: 3
Internal Marks:
Course 1: 50
Course 2: 51
Course 3: 52
Course 4: 53
Course 5: 54
SEE Marks:
Course 1: 60
Course 2: 61
Course 3: 62
Course 4: 63
Course 5: 64

Student: Student1
USN: USN1
Semester: 3
Internal Marks:
Course 1: 51
Course 2: 52
Course 3: 53
Course 4: 54
Course 5: 55
SEE Marks:
Course 1: 61
Course 2: 62
Course 3: 63
Course 4: 64
Course 5: 65

Student: Student2
USN: USN2
```

```
Semester: 3
Internal Marks:
Course 1: 52
Course 2: 53
Course 3: 54
Course 4: 55
Course 5: 56
SEE Marks:
Course 1: 62
Course 2: 63
Course 3: 64
Course 4: 65
Course 5: 66

Student: Student3
USN: USN3
Semester: 3
Internal Marks:
Course 1: 53
Course 2: 54
Course 3: 55
Course 4: 56
Course 5: 57
SEE Marks:
Course 1: 63
Course 2: 64
Course 3: 65
Course 4: 66
Course 5: 67

Student: Student4
USN: USN4
Semester: 3
Internal Marks:
Course 1: 54
Course 2: 55
Course 3: 56
Course 4: 57
Course 5: 58
SEE Marks:
Course 1: 64
Course 2: 65
Course 3: 66
Course 4: 67
Course 5: 68
```

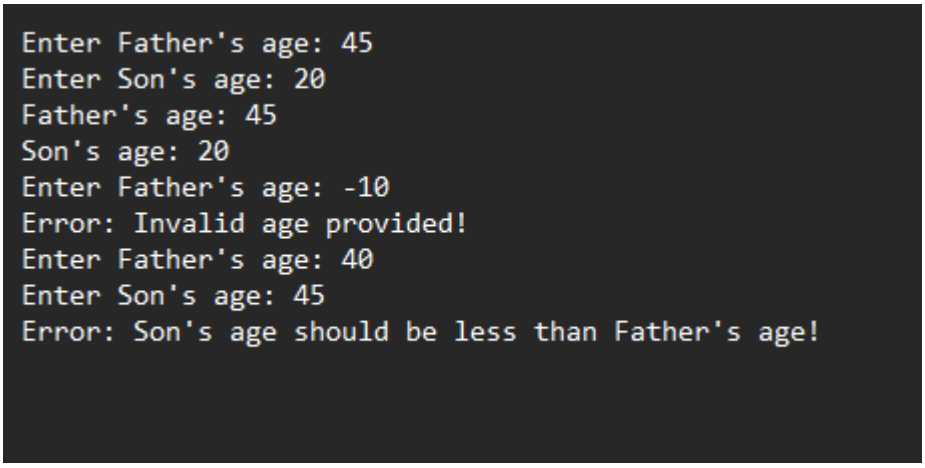
7)Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age<0. In Son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

Code:=

```
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}

class Father {
    private int age;

    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Age cannot be negative");
        }
        this.age = age;
    }

    public int getAge() {
        return age;
    }
}

class Son extends Father {
    private int sonAge;

    public Son(int fatherAge, int sonAge) throws WrongAgeException {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age should be less than
Father's age");
        }
        this.sonAge = sonAge;
    }

    public int getSonAge() {
        return sonAge;
    }
}

public class InheritanceExceptionExample {
    public static void main(String[] args) {
        try {
```

```
            int fatherAgeInput = Integer.parseInt(args[0]);
            int sonAgeInput = Integer.parseInt(args[1]);

            Father father = new Father(fatherAgeInput);
            Son son = new Son(fatherAgeInput, sonAgeInput);

            System.out.println("Father's age: " + father.getAge());
            System.out.println("Son's age: " + son.getSonAge());
        } catch (NumberFormatException e) {
            System.out.println("Invalid input. Please enter valid ages as command-line arguments.");
        } catch (WrongAgeException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

OUTPUT:=



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 {
    public void run() {
        try {
            while (true) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted");
        }
    }
}
class DisplayCSE implements Runnable {
    public void run() {
        try {
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted");
        }
    }
}
public class Main {
    public static void main(String[] args) {
        Thread bmsThread = new Thread(new DisplayBMS());
        Thread cseThread = new Thread(new DisplayCSE());

        bmsThread.start();
        cseThread.start();
    }
}
```

Output:=

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

9.write a program thar 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.

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(resultNum);
				repaint();

			}
		}
		catch(NumberFormatException e1)
		{
			flag=1;
			out="Number Format Exception!  "+e1;
			repaint();
		}
		catch(ArithmeticException e2)
		{
			flag=1;
			out="Divide by 0 Exception!  "+e2;
			repaint();
		}

	}

	public void paint(Graphics g)
	{
		if(flag==0)

	g.drawString(out,outResult.getX()+outResult.getWidth(),outResult.getY()+ou
tResult.getHeight()-8);
		else
		g.drawString(out,100,200);
		flag=0;
	}

	public static void main(String[] args)
	{
```

```
                DivisionMain1 dm=new DivisionMain1();
                dm.setSize(new Dimension(800,400));
                dm.setTitle("DivionOfIntegers");
                dm.setVisible(true);
            }

        }
```

Output: