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:

Pack/CIE/Internal.Java

```
package CIE;


import java.util.Scanner;


public class Internal extends CIE.Student{
    public int m[] = new int[5];
    CIE.Student student = new
    CIE.Student(); public void accept(){
    student.accept();


        Scanner s1 = new Scanner(System.in);
        System.out.println("Enter Internal
        Marks:"); for(int i=0;i<5;i++){ m[i] =
        s1.nextInt();
        }
    }
```

```
    public void display(){
        student.display();

        for(int i=0;i<5;i++){

            System.out.println("Marks of sub" + (i+1) + " = " + m[i]);

        }

    }

}
```

Pack/CIE/Student.Java

```
package CIE;

import java.util.Scanner;


public class Student{ public

            String usn; public

            String name; public

            int sem; public void

            accept(){

        Scanner s = new Scanner(System.in);

        System.out.println("Enter Name:");

        this.name = s.nextLine();

        System.out.println("Enter usn:");

        this.usn = s.nextLine();
```

```
        System.out.println("Enter sem");
        this.sem = s.nextInt();
           }

           public void display(){
        System.out.println("Name: " + this.name + "\nUSN: " + this.usn + "\nSem: " + this.sem);
           }
}
```

Pack/SEE/External.Java

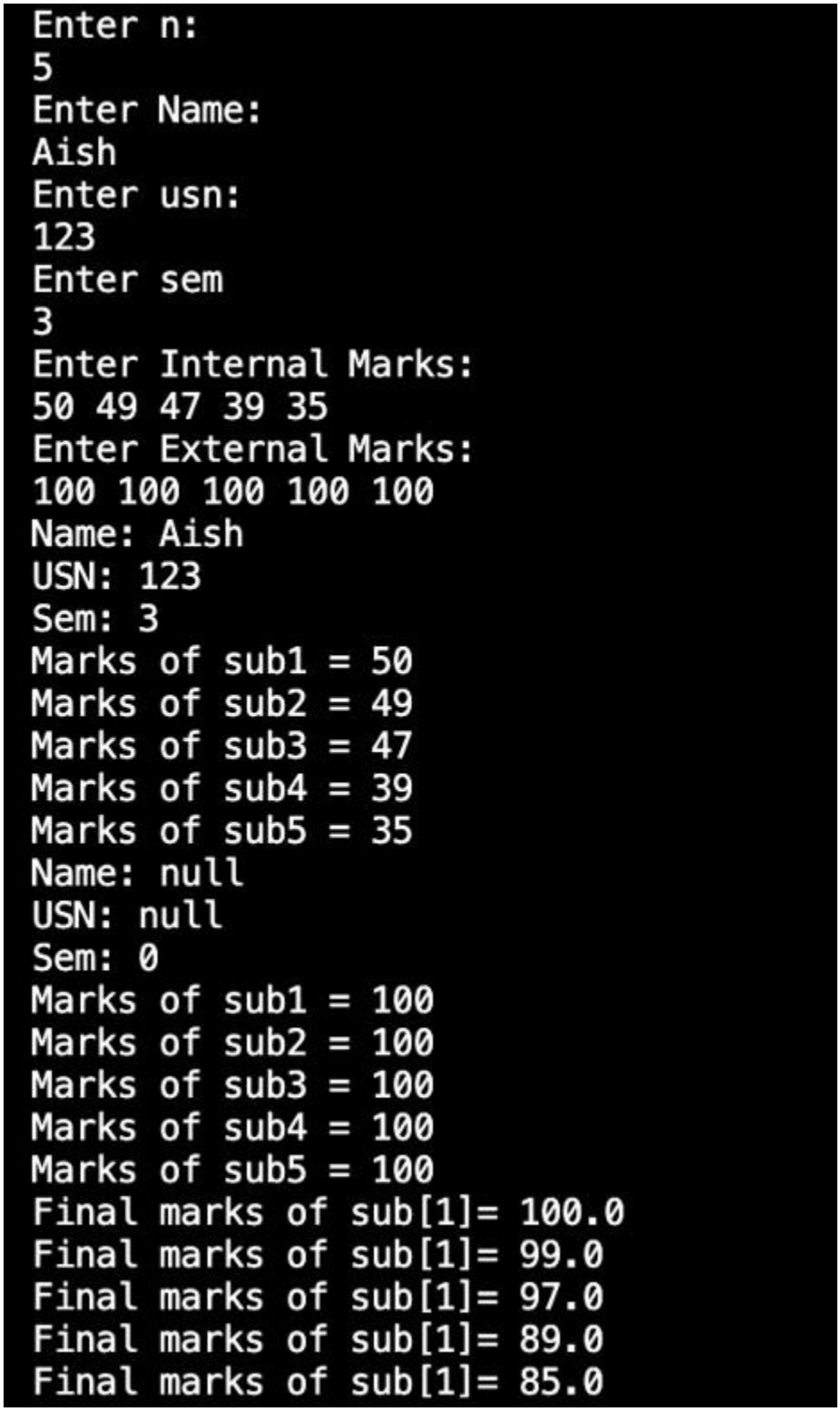
```
package SEE;
import java.util.Scanner;

import CIE.Internal; import CIE.Student;

public class External extends CIE.Student{

public int x[] = new int[5]; public void

accept(){

        Scanner s2 = new Scanner(System.in);
        System.out.println("Enter External Marks:");
        for(int i=0;i<5;i++){
           x[i] = s2.nextInt();
```

```
    }
    }

    public void display(){
  super.display();

  for(int i=0;i<5;i++){
    System.out.println("Marks of sub" + (i+1) + " = " + x[i]);
    }
    }

}
```

Pack/Final.Java import

```
java.util.Scanner;

import CIE.Student;
import CIE.Internal;
import SEE.External;

public class Final{ public static void
    main(String[] args) {
  Scanner n = new Scanner(System.in); System.out.println("Enter n:"); int y = n.nextInt();
```

```
        CIE.Internal[] c1 = new CIE.Internal[y];
        SEE.External[] c2 = new SEE.External[y];

        for(int i=0;i<y;i++){ c1[i] =
            new CIE.Internal(); c2[i] =
            new SEE.External();

            c1[i].accept();
            c2[i].accept();

            // c1[i].accept();c2[i].accept();
            c1[i].display();c2[i].display();

            for(int j=0;j<5;j++){ double calc =
                c1[i].m[j]+((c2[i].x[j])/2);
                System.out.println("Final marks of sub["+(i+1)+"]= "+calc);
            }
            }
            }
}
```
OUTPUT:



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

CODE:

```
import java.util.Scanner;
class quadratic
{
    public static void main (String args[])
    {
      int a; int
      b; int c;
      double
      d;
      Scanner s1=new Scanner(System.in);
      System.out.println("Enter coefficients of quadratic
      equation"); a=s1.nextInt(); b=s1.nextInt(); c=s1.nextInt();
      d=(b*b)-(4*a*c); if(d>0.0)
    {
        double r1=((-b) + Math.pow(d,0.5))/(2.0*a);
        double r2=((-b) - Math.pow(d,0.5))/(2.0*a);
        System.out.println("Roots are real and distinct");
        System.out.println("Root1="+r1+"Root2="+r2);
    }
      else if(d==0.0)
      {
          double r3=(-b)/(2.0*a);
          System.out.println("Roots are real and equal");
          System.out.println("Roots="+r3);
      }
      else
      {
          System.out.println("Roots are imaginary");
      }

    }

}
```

OUTPUT:

```
PS C:\Users\joshi> cd OneDrive
PS C:\Users\joshi\OneDrive> cd Desktop
PS C:\Users\joshi\OneDrive\Desktop> javac quadratic.java
PS C:\Users\joshi\OneDrive\Desktop> java quadratic
Enter coefficients of quadratic equation
1 -2 1
Roots are real and equal
Roots=1.0
PS C:\Users\joshi\OneDrive\Desktop> java quadratic
Enter coefficients of quadratic equation
2 5 7
Roots are imaginary
PS C:\Users\joshi\OneDrive\Desktop> java quadratic
Enter coefficients of quadratic equation
1 3 7
Roots are imaginary
PS C:\Users\joshi\OneDrive\Desktop> java quadratic 1 1 1
Enter coefficients of quadratic equation
1 1 1
Roots are imaginary
PS C:\Users\joshi\OneDrive\Desktop> java quadratic
Enter coefficients of quadratic equation
-2 2 1
Roots are real and distinct
Root1=-0.36602540378443860Root2=1.36602540378443860
```

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 NewThread1 implements Runnable
{
Thread t1;
NewThread1()
{
    t1 = new Thread(this,"Thread1");
    System.out.println("CT:"+t1);
    t1.start();
}
public void run()
{ try
    {
```

```
      for (int n=5; n>0;n--)
      {

       System.out.println("BMS College of Engineering");

Thread.sleep (10000);
}
}
    catch( InterruptedException ie)
{
    System.out.println("Thread1 interrupted");
}

System.out.println ("Thread I quitting");
}
}
class NewThread2 implements Runnable
{
    Thread t2;
    NewThread2()
    {
        t2=new Thread (this,"Thread2");
        System.out.println("CT:"+t2);
        t2.start();
    }
    public void run()
    { try
        {
            for (int n=5; n>0;n--)
            {
                System.out.println("cse");
                Thread.sleep(2000);
            }
        }
        catch(InterruptedException ie)
        {
            System.out.println("Thread 2 Interrupted");
        }
        System.out.println ("Thread 2 quitting");
    }
}
class MainThread {
```

```
    public static void main(String args[]) {
        new NewThread1();
        new NewThread2();

        try {
            Thread.sleep(40000);
            System.out.println("MainThread is awake\n");
        } catch (InterruptedException ie) {
            System.out.println("MainThread Interrupted");
        }
        System.out.println("MainThread exiting");
    }
}
```

OUTPUT:

```
PS C:\Users\joshi\OneDrive\Desktop> javac MainThread.java
PS C:\Users\joshi\OneDrive\Desktop> java MainThread
CT:Thread[#21,Thread1,5,main]
BMS College of Engineering
CT:Thread[#22,Thread2,5,main]
cse
cse
cse
cse
cse
BMS College of Engineering
Thread 2 quitting
BMS College of Engineering
BMS College of Engineering
MainThread is awake

MainThread exiting
BMS College of Engineering
Thread I quitting
```

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:

```
import java.util.Scanner; class
WrongAgeException extends Exception
{
   public WrongAgeException(String message)
   {
     super(message);
   }

}
class father
{
   private int FatherAge; public father(int age)
   throws WrongAgeException
   {
      if(age<0)
      {
         throw new WrongAgeException("Age cannot be Negative");
      }
      this.FatherAge=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;
      System.out.println("Father's Age:"+FatherAge);
      System.out.println("son's Age:"+sonAge);
```

```
    }
}
public class ExceptionDemo
{
    public static void main(String[] args)
    {
        Scanner Scanner= new Scanner(System.in);
        try
        {
            System.out.println("Enter Father's
            age:"); int FatherAge=Scanner.nextInt();
            father father=new father(FatherAge);
            System.out.println("Enter son's Age:");
            int sonAge= Scanner.nextInt(); son son=
            new son(FatherAge,sonAge);
        }
        catch(WrongAgeException e)
        {
            System.out.println("Exception:"+e.getMessage());
        }
    }
}
```

OUTPUT:

```
PS C:\Users\joshi\OneDrive\Desktop> javac ExceptionDemo.java
PS C:\Users\joshi\OneDrive\Desktop> java ExceptionDemo
Enter Father's age:
-1
Exception:Age cannot be Negative
PS C:\Users\joshi\OneDrive\Desktop> java ExceptionDemo
Enter Father's age:
35
Enter son's Age:
45
Exception:son's age should be less than Father's age
```

Program 5:

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current

account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks: a) Accept deposit from customer and update the balance. b) Display the balance. c) Compute and deposit interest d) Permit withdrawal and update the balance Check for the minimum balance, impose penalty if necessary and update the balance.

CODE:

```
import java.util.Scanner;

class Account { String

customerName; int

accountNumber; String

accountType; double

balance;


    public Account(String customerName, int accountNumber, String accountType, double
balance) { this.customerName =

        customerName; this.accountNumber =

        accountNumber; this.accountType =

        accountType; this.balance = balance;

    }

    public void deposit(double amount) {

        balance += amount;

        System.out.println("Deposit of $" + amount + " successful.");
```

```
    }

    public void displayBalance() {
        System.out.println("Balance: $" + balance);
    }
}
class CurAcct extends Account {
    double minBalance; double
    penaltyCharge;

    public CurAcct(String customerName, int accountNumber, double balance)
        { super(customerName, accountNumber, "Current", balance);
        minBalance = 1000; penaltyCharge = 50;
    }

    public void withdraw(double amount) { if
        (balance - amount >= minBalance) {
        balance -= amount;
            System.out.println("Withdrawal of $" + amount + " successful.");
        } else {
            System.out.println("Insufficient balance. Withdrawal failed.");
        }
    }
}
```

```
class SavAcct extends Account { double interestRate; public SavAcct(String
    customerName, int accountNumber, double balance) {
    super(customerName, accountNumber, "Savings", balance); interestRate =
    0.05; // 5% interest rate for savings account
    }
    public void depositInterest() { double
        interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest of $" + interest + " deposited.");
    }
    public void withdraw(double amount)
        { if (balance - amount >= 0) {
        balance -= amount;
            System.out.println("Withdrawal of $" + amount + " successful.");
        } else {
            System.out.println("Insufficient balance. Withdrawal failed.");
        }
    }
}

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

        System.out.println("Enter details for current account:");
```

```
        System.out.print("Customer Name: ");
        String currentCustomerName = scanner.nextLine();
        System.out.print("Account Number: "); int currentAccountNumber =
        Integer.parseInt(scanner.nextLine()); System.out.print("Initial Balance:
        $"); double currentInitialBalance =
        Double.parseDouble(scanner.nextLine());
        CurAcct currentAccount = new CurAcct(currentCustomerName, currentAccountNumber,
currentInitialBalance);
        System.out.println("\nEnter details for savings account:");
        System.out.print("Customer Name: ");
        String savingsCustomerName = scanner.nextLine();
        System.out.print("Account Number: "); int savingsAccountNumber =
        Integer.parseInt(scanner.nextLine()); System.out.print("Initial Balance:
        $"); double savingsInitialBalance =
        Double.parseDouble(scanner.nextLine());
        SavAcct savingsAccount = new SavAcct(savingsCustomerName, savingsAccountNumber,
savingsInitialBalance);
        System.out.print("\nEnter deposit amount for current account: $"); double
        depositAmountCurrent = Double.parseDouble(scanner.nextLine());
        currentAccount.deposit(depositAmountCurrent);

        System.out.print("Enter withdrawal amount for current account: $"); double
        withdrawAmountCurrent = Double.parseDouble(scanner.nextLine());
        currentAccount.withdraw(withdrawAmountCurrent);
        currentAccount.displayBalance();
```

```
        System.out.print("\nEnter deposit amount for savings account: $"); double
        depositAmountSavings = Double.parseDouble(scanner.nextLine());
        savingsAccount.deposit(depositAmountSavings);

        savingsAccount.depositInterest();
        savingsAccount.displayBalance();

        System.out.print("Enter withdrawal amount for savings account: $"); double
        withdrawAmountSavings = Double.parseDouble(scanner.nextLine());
        savingsAccount.withdraw(withdrawAmountSavings);

        savingsAccount.displayBalance();
    }
}
```

OUTPUT:

```
C:\Users\Admin\Desktop>javac BankDemo.java

C:\Users\Admin\Desktop>java BankDemo
Enter details for current account:
Customer Name: manj
Account Number: 123
Initial Balance: $500

Enter details for savings account:
Customer Name: manj
Account Number: 124
Initial Balance: $300

Enter deposit amount for current account: $900
Deposit of $900.0 successful.
Enter withdrawal amount for current account: $1500
Insufficient balance. Withdrawal failed.
Balance: $1400.0

Enter deposit amount for savings account: $300
Deposit of $300.0 successful.
Interest of $30.0 deposited.
Balance: $630.0
Enter withdrawal amount for savings account: $500
Withdrawal of $500.0 successful.
Balance: $130.0
```

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

CODE:

```
import java.util.Scanner;
abstract class shape
{
   int a; int b; abstract void
   printArea();
}
class rect extends shape
{
   void printArea()
{
   System.out.println("Area of rectangle is:"+(a*b));
}
}
class tri extends shape
```

```
{
    void printArea()
    {
        System.out.println("Area of triangle is:"+(0.5*a*b));
    }
}
class cir extends shape
{
    void printArea()
    {
        System.out.println("Area of circle is:"+(314*a*a));
    }
}
class AbstractDemo
{
    public static void main (String args[])
    {
        Scanner s1=new Scanner (System.in);
        System.out.println("Press:\n 1.Rectangle \n 2.Triangle \n
        3.Circle"); int choice; choice=s1.nextInt(); switch(choice)
        {
            case 1: System.out.println("Enter l and b of
            Rectangle"); int l=s1.nextInt(); int br=s1.nextInt(); rect
            r=new rect();
            r.a=l;
            r.b=br;
            r.printArea(); break; case 2: System.out.println("Enter
            l and b of Triangle"); int h=s1.nextInt(); int
            bre=s1.nextInt(); tri t=new tri(); t.a=h;
            t.b=bre;
            t.printArea(); break; case 3:
            System.out.println("Enter r of Circle"); int
            rad=s1.nextInt(); cir c =new cir(); c.a=rad;
            c.printArea(); break;
            default:System.out.println("Enter valid
            choice");
        }
    }
}
```

OUTPUT:

```
PS C:\Users\joshi\OneDrive\Desktop> javac AbstractDemo.java
PS C:\Users\joshi\OneDrive\Desktop> java AbstractDemo
Press:
 1.Rectangle
 2.Triangle
 3.Circle
1
Enter l and b of Rectangle
4 4
Area of rectangle is:16
```

Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to create n book objects.

CODE:

```
import java.util.Scanner;

class book {
    String name;
    String author;
    double price;
    int pages;

    book(String name, String author, double price, int pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.pages = pages;
    }

    void setDetails() {
        Scanner S = new Scanner(System.in);
        System.out.println("Enter name of books:");
        this.name = S.nextLine();
        System.out.println("Author:");
        this.author = S.nextLine();
        System.out.println("Enter price");
        this.price = S.nextDouble();
```

```
        System.out.println("Enter no.of.pages");
        this.pages = S.nextInt();
    }

    void getDetails() {
        System.out.println("Book name:" + this.name);
        System.out.println("Author:" + this.author);
        System.out.println("Price:$" + this.price);
        System.out.println("Number of pages" + this.pages);
    }

    public String toString() {
        return "Book Details:\n" + "Name:" + name + "\n" + "Author:" + author + "\n" + "Prices : $" +
price + "\n"
                + "Number of pages:" + pages;
    }
}

class BookDemo2 {
    public static void main(String args[]) {
        Scanner S1 = new Scanner(System.in);
        System.out.println("Enter number of
        books"); int n = S1.nextInt(); book[] b = new
        book[n]; for (int i = 0; i < n; i++) {
            System.out.println("\n Enter details for book" + (i + 1) + ":");
            b[i] = new book(" ", " ", 0.0, 0);
            b[i].setDetails();
        }
        System.out.println("\n Details of books");
        for (int i = 0; i < n; i++) {
            System.out.println("\n Book" + (i + 1) + ":");
            b[i].getDetails();
        }
        System.out.println("\n Complete details of all books:");
        for (int i = 0; i < n; i++) {
            System.out.println("\n Book" + (i + 1) + ":\n" + b[i]);
        }
    }
}
```

OUTPUT:

```
PS C:\Users\joshi\OneDrive\Desktop> javac BookDemo2.java
PS C:\Users\joshi\OneDrive\Desktop> java BookDemo2
Enter number of books
2

 Enter details for book1:
Enter name of books:
rich dad poor dad
Author:
robert
Enter price
45
Enter no.of.pages
453

 Enter details for book2:
Enter name of books:
xyz
Author:
abc
Enter price
67
Enter no.of.pages
890

 Details of books

 Book1:
Book name:rich dad poor dad
Author:robert
Price:$45.0
Number of pages453

 Book2:
Book name:xyz
Author:abc
Price:$67.0
Number of pages890

 Complete details of all books:

 Book1:
Book Details:
Name:rich dad poor dad
Author:robert
Prices : $45.0
Number of pages:453

 Book2:
Book Details:
Name:xyz
Author:abc
Prices : $67.0
Number of pages:890
PS C:\Users\joshi\OneDrive\Desktop>
```

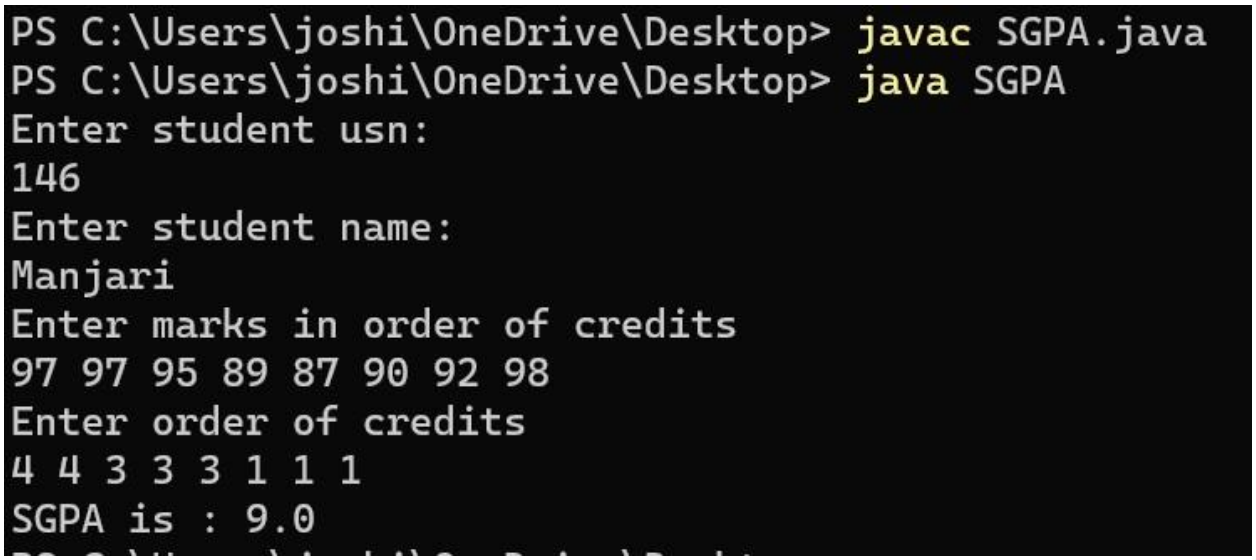
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student

CODE:

```
import java.util.Scanner;

class Details { int usn; String
    name; int[] marks = new
    int[8]; int[] credit = new int[8];
    int[] credit_points = new int[8];
    Scanner s1 = new Scanner(System.in);

    void acceptDetails() {
        System.out.println("Enter student usn:");
        usn = s1.nextInt();
        System.out.println("Enter student name:");
        name = s1.next();
        System.out.println("Enter marks in order of
        credits"); for (int i = 0; i < 8; i++) { marks[i] =
        s1.nextInt();
        }
        System.out.println("Enter order of credits ");
        for (int i = 0; i < 8; i++) {
            credit[i] = s1.nextInt();
        }
    }

    void calculate() {
        for (int i = 0; i < 8; i++) {
            if (marks[i] >= 90) {
                credit_points[i] = 10 * credit[i];
            } else if (marks[i] >= 80) {
                credit_points[i] = 9 * credit[i];
            } else if (marks[i] >= 70) {
                credit_points[i] = 8 * credit[i];
            } else if (marks[i] >= 60) {
                credit_points[i] = 7 * credit[i];
            } else if (marks[i] >= 50) {
                credit_points[i] = 6 * credit[i];
            } else if (marks[i] >= 40) {
                credit_points[i] = 5 * credit[i];
            }
        }
```

```
        int sum = 0; int count = 0;
        double SGPA; for (int j = 0; j <
        8; j++) { sum = sum +
        credit_points[j];
              count=count+credit[j];
        }
        SGPA = sum/count;
        System.out.println("SGPA is : " + SGPA);
    }
}

class SGPA {
    public static void main(String[] args) {
        Details d = new Details();
        d.acceptDetails();
        d.calculate();
    }
}
```

OUTPUT:



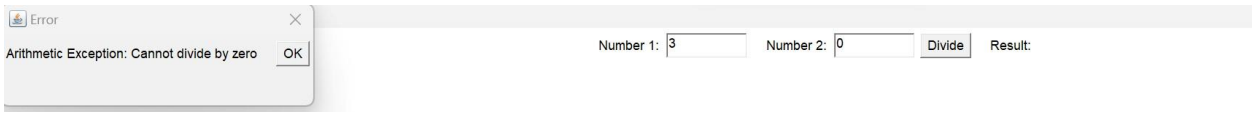
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

CODE:

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

public class DivisionMain1 extends Frame implements ActionListener {
    TextField num1, num2;
    Button divideButton;
    Label resultLabel;

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

        divideButton = new Button("Divide");
        Label number1 = new Label("Number 1:",
        Label.RIGHT); Label number2 = new Label("Number
        2:", Label.RIGHT); num1 = new TextField(5); num2 =
        new TextField(5); resultLabel = new Label("Result:",
        Label.RIGHT);

        add(number1); add(num1);
        add(number2); add(num2);
        add(divideButton); add(resultLabel);
        divideButton.addActionListener(this)
        ;

        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });
    }

    public void actionPerformed(ActionEvent ae) {
        try {
            int n1 = Integer.parseInt(num1.getText());
            int n2 = Integer.parseInt(num2.getText());
            if (n2 == 0) {
                throw new ArithmeticException("Cannot divide by zero");
            }
            int result = n1 / n2;
            resultLabel.setText("Result: " + result);
```

```
        } catch (NumberFormatException e1) { showErrorDialog("Number Format
            Exception: Please enter integers only");
        } catch (ArithmeticException e2) { showErrorDialog("Arithmetic
            Exception: " + e2.getMessage());
        }
    }

    private void showErrorDialog(String message) {
        Dialog dialog = new Dialog(this, "Error", true);
        dialog.setLayout(new FlowLayout()); Label label =
        new Label(message); Button okButton = new
        Button("OK"); okButton.addActionListener(new
        ActionListener() {
            public void actionPerformed(ActionEvent e) {
                dialog.dispose();
            }
        });
        dialog.add(label);
        dialog.add(okButton);
        dialog.setSize(300, 100);
        dialog.setVisible(true);
    }

    public static void main(String[] args) {
        DivisionMain1 divisionMain = new DivisionMain1();
        divisionMain.setSize(new Dimension(400, 200));
        divisionMain.setTitle("Integer Division");
        divisionMain.setVisible(true);
    }
}
```

OUTPUT:



16/12/23

1) d:
2) dir
3) cd cs133
4) dir
5) javac First.java
6) java First

PROGRAM 1 : QUADRATIC EQUATION

```
import java.util.Scanner;
class Quadratic
{
 int a, b, c;
 double r1, r2, d;
 void getd()
 {
  Scanner s = new Scanner(System.in);
  System.out.println("Enter the coefficients of a, b and c");
 a = s.nextInt();
 b = s.nextInt();
 c = s.nextInt();
 }
 void compute()
 {
  while(a==0)
  {
   System.out.println("Not a quadratic equation, enter
                       a non zero value");
   Scanner s=new Scanner(System.in);
   a = s.nextInt();
  }
```

```
d = b*b - 4*a*c;
if (d == 0)
{
r1 = (-b) / (2*a);
System.out.println ("Roots are real and equal");
System.out.println (" Root 1 = Root 2");
}
else if (d > 0)
{
r1 = ((-b) + (Math.sqrt(d))) / (double) (2*a);
r2 = ((-b) - (Math.sqrt(d))) / (double) (2*a);
System.out.println (" Roots are equal and distinct");
System.out.println ("Root 1 = "+r1+" Root 2 = "+r2);
}

else
{
System.out.println ("Roots are imaginary");
r1 = (-b) / (double) (2*a);
r2 = (Math.sqrt (d)) / (double) (2*a);
System.out.println ("Root 1 = "+r1+"+i"+r2);
System.out.println ("Root 1 = "+r1+"-i"+r2);
}
}
}
class QuadraticMain
{
public static void main (String args[])
{
Quadratic q = new Quadratic ();
q.getd ();
q.compute ();
}
}
```

OUTPUT :

Enter the coefficients a, b and c
1 -4 4
Roots are real and equal
Root 1 = Root 2

Enter the coefficients a, b and c
1 5 2
Roots are real and distinct
Root 1 = -0.4384471871911697 Root 2 = -4.561552812808831

Enter the coefficients a, b and c
0 0 0
Not a quadratic equation, enter a non zero value.

1/1/24

LAB SESSION - 2

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

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

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

System.out.println ("Enter number of subjects: ");
int numSub = scanner.nextInt ();
credits = new int[numSub];
marks = new int[numSub];

System.out.println ("Enter details for each subject: ");
for (int i=0; i<numSubjects; i++)
{
 System.out.println ("Enter credits for subject
 " + (i+1) + ": ");
 credits [i] = scanner.nextInt ();
```

```
System.out.print("Enter marks for subject" + (i+1)
                                    + ": ");
    marks[i] = scanner.nextInt();
}
}

void displayDetails(){
System.out.println("\n Student Det
    -ails: ");
  System.out.println("USN: " + usn);
  System.out.println("Name: " + name);

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

double calculateSGPA(){
  int totalCredits = 0;
  double weightedMarks = 0.0;
  for(int i = 0; i < credits.length; i++){
    totalCredits += credits[i];
  weightedMarks += (calculateGradePoint(marks[i])
                    * credits[i]);
  }
  return weightedMarks / totalCredits;
}

private double calculateGradePoints(int marks){
 if (marks >= 90){
   return 10.0;
```

```
} else if (marks >=80) {
    return 9.0;
} else if (marks >= 70) {
    return 8.0;
} else if (marks >= 60) {
    return 7.0;
} else if (marks >= 50) {
    return 6.0;
} else {
    return 0.0;
  }
 }
}

public class studemo {
   public static void main (String [] args) {
     Student student = new Student ();
     Student.acceptDetails ();
     Student.displayDetails ();
     System.out.println ("\n SGPA : " + student.
                  calculateSGPA ());
   }
}
```

OUTPUT :

```
Enter USN :
1bm22cs133
Enter Name :
Kusumanjali
Enter number of subjects :
5
```

Enter details for each subject :
Enter credits for subject 1 : 4
Enter marks for subject 1 : 80
Enter credits for subject 2 : 4
Enter marks for subject 2 : 77
Enter credits for subject 3 : 3
Enter marks for subject 3 : 90
Enter credits for subject 4 : 2
Enter marks for subject 4 : 85
Enter credits for subject 5 : 2
Enter marks for subject 5 : 90

Student Details :
USN : 1bm22cs133
Name : Kusumanjali

Subject - wise Details :
Subject 1 : credits - 4, Marks - 80
Subject 2 : credits - 4, Marks - 77
Subject 3 : credits - 3, Marks - 90
Subject 4 : credits - 2, Marks - 85
Subject 5 : credits - 2, Marks - 90

SGPA : 9.000

11/12/24

2) Write 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.

```
import java.util.Scanner;

class BOOK {
    String name;
    String author;
    double price;
    int numPages;

    public Book (String name, String author, double
        price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
    public void setDetails() {
        Scanner scanner = new Scanner (System.in);

    System.out.println ("Enter book name : ");
    this.name = scanner.nextLine();

    System.out.println ("Enter author name : ");
    this.author = scanner.nextLine();
```

```
System.out.println("Enter price: ");
this.price= scanner.nextDouble();

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

public void getDetails() {
  System.out.println("Book Name: " + name);
  System.out.println("Author: " + author);
  System.out.println("Price: $" + price);
  System.out.println("Number of Pages: " + numPages);
}

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

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

    System.out.print("Enter the number of books: ");
    int n= scanner.nextInt();
    Book[] books= new Book[n];
    for(int i= 0; i<n; i++) {
      System.out.println("\n Enter details for Book "
          + (i+1) + ": ");
```

```
books[i] = new Book("", "", 0.0, 0);
books[i].setDetails();
}
System.out.println("\n Details of all books : ");
for (int i=0; i<n; i++) {
  System.out.println("\n Book " + (i+1) + " : ");
  books[i].getDetails();
}
System.out.println("\n Complete details of all
  books : " );
for (int i=0; i<n; i++) {
  System.out.println("\n Book " + (i+1) + " : \n"
      + books[i].toString());
  }
}
}
```

OUTPUT :

```
Enter the number of books : 2
Enter details for Book 1 :
Enter book name :
twisted love
Enter author name :
Ana Huang
Enter price : 300
Enter number of Pages :
390
Enter the ~~number~~ of book name :
twisted hate
Enter author name :
Ana Huang
```

Price : 300
Number of Pages : 280
Complete details of all books :
Book 1 :
Book Details :
Name : Twisted love
Author : Ana Huang
Price : $300.
Number of Pages : 397
Book 2 :
Book Details :
Name : Twisted Hate
Author : Ana Huang
Price : $300.0
Number of Pages : 282.

8/12/24

LAB SESSION - 3

LAB PROGRAM - 4.

```
import java.util.Scanner;
abstract class Shape
{
    int a;
    int b;
    abstract void printArea();
}
 Class rec extends Shape
 {
    void printArea()
    {
    System.out.println("area of rectangle is :"+(a*b));
    }
}
class tri extends Shape
{
   void printArea()
   {
   System.out.println("area of triangle is :"+(0.5*a*b));
   }
}
class cir extends Shape
{
   void printArea()
   {
    System.out.println("area of triangle is :"+(3.14*a*a
   }
}
```

```
class Main
{
  public static void main(String args[])
  {
  Scanner sl = new Scanner(System.in);
  System.out.println("Choose: \n 1.rectangle \n 2.triangle \n
                                3.circle");
  int choice;
  choice = sl.nextInt();
  switch(choice)
  {
    case 1: System.out.println("Enter l and b of rectang
                                -le :");
          int l=sl.nextInt();
          int br=sl.nextInt();
          rect r=new rect();
          r.a = l;
          r.b = br;
          r.printArea();
          break;
    case 2: System.out.println("Enter h and b of triangle :");
          int h=sl.nextInt();
          int bre=sl.nextInt();
          tri t=new tri();
          t.a = h;
          t.b = bre;
          t.printArea();
          break;
    case 3: System.out.println("Enter r of circle:");
          int rad=sl.nextInt();
          cir c = new cir();
          c.a = rad;
```

```
c.printArea();
break;
default: System.out.println("enter valid choice");
}
}
}
```

OUTPUT:

```
Choose:
1. rectangle
2. triangle
3. circle
1
Enter l and b of rectangle:
3 2
area of rectangle is: 6

Enter h and b of triangle:
4 4
area of triangle is: 4.

Enter r of circle:
3
area of circle is: 28.2599.
```

Q Uses of super keyword.

```
class A
{
 int a;
 int b;
 private int c;
 A()
 {
  System.out.println("A's constructor");
  a = 10;
  b = 20;
  c = 30;
 }
 A(int a1)
 {
  c = a1;
 }
 A(int p, int q, int r)
 {
 a = p; b = q; c = r;
 }
 void display()
 {
  System.out.println(a+b+c);
 }
}
class B extends A
{
  int b;
  b();
  {
```

```
System.out.println("B's Constructor");
b = 20;
}
B (int x, int y, int z, int p)
{
    super(z);
    a = x;
    super.b = y;
    b = p;
}
void display()
{
    super.display();
    System.out.println(b);
}
}
class Main
{
    public static void main(String args[])
    {
        A a1 = new A();
        B b1 = new B();
        A a2 = new A(100, 200, 300);
        B b2 = new B(1, 2, 3, 4);
        a1.display();
        a2.display();
        b1.display();
        b2.display();
    }
}
```

OUTPUT:

A's constructor
B's constructor
B's constructor
B's constructor
60
600
60
26
6
4.

22/1/23

LAB SESSION - 4

// File Final Java //

```
import java.util.Scanner;
import CIE.Student;
import CIE.Internal;
import SEE.External;
public class Final {
 public static void main (String [] args) {
  Scanner n = new Scanner (System.in);
  System.out.println ("Enter n : ");
  int y = n.nextInt ();
  CIE.Internal [] c1 = new CIE.Internal [y];
  SEE.External [] c2 = new SEE.External [y];
  for (int i=0; i<y; i++) {
    c1 [i] = new CIE.Internal ();
    c2 [i] = new SEE.External ();
    c1 [i].accept ();
    c2 [i].accept ();
    c1 [i].accept (); c2 [i].accept ();
    c1 [i].display (); c2 [i].display ();
    for (int j=0; j<5; j++) {
    double calc = c1 [i].m [j] + ((c2 [i].x [j])/2);
    System.out.println ("Final marks of sub [" + (j+1) + "] =
                                    " + calc ();
  }
 }
}
}
```

// File Student //

```
package CIE;
import java.util.Scanner;
public class Student {
public String usn;
public String name;
public int sem;
public void accept() {
Scanner s = new Scanner(System.in);
System.out.println("Enter Name: ");
this.name = s.nextLine();
System.out.println("Enter usn: ");
this.usn = s.nextLine();
System.out.println("Enter sem: ");
this.sem = s.nextInt();
}
public void display() {
System.out.println("Name: " + this.name + "\n USN: "
+ this.usn + "\nSem: " + this.sem);
}
}
```

// FP u Internal.java //

```
package CIE;
import java.util.Scanner;
public class Internal extends CIE.Student {
public int m[] = new int[5];
CIE.Student Student = new CIE.Student();
public void accept();
Student.accept();
```

```
Scanner s1 = new Scanner (System.in);
System.out.println ("Enter Internal Marks: ");
for (int i=0; i<5; i++) {
 m[i] = s1.nextInt();
}
}
public void display() {
  Student.display();
 for (int i=0; i<5; i++) {
  System.out.println ("Marks of sub" + (i+1) + "=" 
                                    + m[i]);
}
}
}

// File SEE Java //.

package SEE;
import java.util.Scanner;
import CIE.Internal;
import CIE.Student;
public class External extends CIE.Student {
public int x[]= new int [5];
public void accept() {
Scanner s2 = new Scanner (System.in);
System.out.println ("Enter External Marks: ");
for (int i=0; i<5; i++) {
  x[i] = s2.nextInt();
}
}
public void display() {
 for (int i=0; i<5; i++){
```

```
System.out.println("Marks of sub" + (i+1) + "=" + x[i]);
}
}.
```

OUTPUT :

```
Enter n :
2
ENTER Name :
A
Enter USN :
10
Enter Sem :
2
Enter Internal Marks :
40
41
42
43
44
Enter External Marks :
50
49
47
46
44
Name : A
USN : 10
Sem : 2
Marks of Sub1 = 40
```

Marks of sub2 = 41
Marks of sub3 = 42
Marks of sub4 = 43
Marks of sub 5 = 44
Name: B
usn : 11
sem : 2

Marks of sub1 = 50	marks of sub1 = 343
Marks of sub 2 = 49	marks of sub 2 = 23
Marks of sub 3 = 47	Marks of sub 3 = 2
Marks of sub 4 = 46	Marks of sub 4 = 2
Marks of sub 5 = 44	Marks of sub 5 = 2

Final marks of sub[1] = 172
Final marks of sub[2] = 13
Final marks of sub[3] = 44
Final marks of sub[4] = 5
Final marks of sub[5] = 6.

→ EXCEPTION HANDLING.

```
import java.util.Scanner;
class WrongAgeException extends Exception {
public WrongAgeException (String message) {
 super(message);
}
}
class Father {
private int fatherAge ;
public Father (int age) throws WrongAgeException {
if (age < 0) {
throw new WrongAgeException ("Age cannot be negative
}
```

```
        this.fatherAge = age;
    }
}
class Son extends Father {
    private int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAge
                                                    Exception {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age should be less tha
                    Father's age");
        }
        this.sonAge = sonAge;
        System.out.println("Father's Age: " + fatherAge);
        System.out.println("Son's Age: " + sonAge);
    }
}

public class ExceptionInheritanceDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.println("Enter Father's Age: ");
            int fatherAge = scanner.nextInt();
            Father father = new Father(fatherAge);
            System.out.print("Enter Son's Age: ");
            int sonAge = scanner.nextInt();
            Son son = new Son(fatherAge, sonAge);
        }
        catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

OUTPUT:

Enter Father's Age : 40

Enter Son's Age : 20

Father's Age : 40

Son's Age : 20

22/1/24

29/1/24

classmate
Date
Page

LAB SESSION - 5

```
class Gener<T, G>
{
 T ob;
 G ob1;
 Gener(T o, G o1)
 {
  ob = o;
  ob1 = o1;
 }
 void showType()
 {
  System.out.println(ob.getClass().getName());
  System.out.println(ob1.getClass().getName());
 }
 T retObj()
 {
  return ob;
 }
 G retObj1()
 {
  return ob1;
 }
}
class GenMain
{
  public static void main(String args[])
  {
   Gener<Integer, Double> g1;
   g1 = new Gener<Integer, Double>(100, 1.588);
   g1.showType();
```

```
int x = g1.retObj();
System.out.println("x = " + x);
double f = g1.retObj1();
System.out.println("f = " + f);
Gener1 <Double, Double> g2 = new Gener1 <Double,
                                Double> (253.963, 5963.535);
g2.showType();
double d1 = g2.retObj();
System.out.println(d1 + "  " + d1);
double d2 = g2.retObj1();
System.out.println(d2 + "  " + d2);
}
}
```

OUTPUT:

```
java.lang.Integer
java.lang.Double
x = 100
f = 1.588
java.lang.Double
java.lang.Double
253.963   253.963
5963.535  5963.535
```

EXCEPTION HANDLING:

```
Import java.util.Scanner;
class SalaryExc extends Exception {
SalaryExc(String err)
{
  System.out.println(err);
}
class Employee
{
  int empid;
   void accept1(){
  Scanner sc = new Scanner(System.in);
  System.out.println("Enter the Employee Id :");
  empid = sc.nextInt();
}
}
class Manager extends Employee
{
  int salary;
  void accept2(){
  super.accept1();
  Scanner sc = new Scanner(System.in);
  System.out.println("Enter the Salary of Manager:");
  salary = sc.nextInt();
  System.out.println("Details of Manager: \n Empid:
  "+empid+"\n Manager Salary:" + salary);
}
}

class Worker extends Employee{
  int salary;
```

```
void accept3 (int min) throws SalaryExc {
super.accept1();
Scanner sc = new Scanner (System.in);
System.out.println ("Enter the salary of worker: ");
Salary = sc.nextInt();
if (min < this.salary)
{
  throw new SalaryExc ("worker's salary cannot be greater than Manager's salary!");
}
System.out.println("Details of worker:\n empid +"
\n Worker salary :" + salary);
}
}

class Excp {
public static void main (String args[])
{
  int min = 1000000;
  Scanner sc = new Scanner (System.in);
  System.out.println ("Enter the number of employees:");
  int n = sc.nextInt();
  Manager a[] = new Manager [n];
  Worker b[] = new Worker [n];
  for (int i=0; i<n; i++)
  {
    a[i] = new Manager();
    a[i].accept2();
  if (a[i].salary < min)
  min = a[i].salary;
  b[i] = new Worker();
  try {
```

```
b[i] = accept3(mis);
}
catch (SalaryExc e)
{
System.out.println("Salary exception has been
    handled");
}
catch (Exception ee)
{
System.out.println("unrupted error: "+ee);
}
}
}
}
```

OUTPUT!

```
Enter the number of employees:
2
Enter the Employee Id:
01
Enter the Salary of Manager:
50000
Details of Manager:
Empid: 1
Manager Salary: 50000
Enter the Employee Id:
04
Enter the Salary of Worker:
30000
Details of Worker:
Empid: 4
Worker Salary: 30000
```

Enter the Employee Id :
02
Enter the Salary of Manager :
40000
Details of Manager :
Empid : 2
Manager Salary : 40000
Enter the Employee Id :
02
Enter the Salary of Worker :
30000
Details of Worker :
Empid : 3
Worker Salary : 30000.

5/2/24

LAB - SESSION 6

THREADS

```
class NewThread1 Implements Runnable
{
  Thread t1;
  NewThread1()
  {
    t1 = new Thread(this, "Thread1");
    System.out.println("CT: " + t1);
    t1.start();
  }
  public void run()
  {
    try
    {
    for(int n=5; n>0; n--)
    {
    System.out.println("BMS College Of Engineering");
    Thread.sleep(10000);
    }
    }
    catch (InterruptedException Ie)
    {
    System.out.println("Thread1 Interrupted");
    }
    System.out.println("Thread1 quitting");
  }
}
```

```
class NewThread2 implements Runnable
{
  Thread2;
  NewThread2()
  {
  t2 = new Thread(this, "Thread2");
  System.out.println("CT: " + t2);
  t2.start();
  }
  public void run()
  {
   try
  {
  for(int n=5; n>0; n--)
  {
  System.out.println("CSE");
  Thread.sleep(1000);
  }
  }
  catch (InterruptedException ie)
  {
  System.out.println("Thread 2 Interrupted");
  }
  System.out.println("Thread 2 quitting");
  }
}
class MainThread
{
  public static void main(String ss[])
  {
  new NewThread1();
```

```
        new NewThread2 ();
    }
}
```

OUTPUT:

```
CT: Thread [#29, Thread1, 5, main]
BMS College of Engineering
CT: Thread [#30, Thread2, 5, main]
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
Thread 2 quitting
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
Thread 1 quitting.
```

5/2/2024

19/2/24

LAB SESSION - 7

LAB PROGRAM - 5.

Input :

```
import java.util.Scanner;

class Account{
String customerName;
int accountNumber;
String accountType;
double balance;

public Account(String customerName, int accountNumber,
String accountType, double balance)
{
  this.customerName = customerName;
  this.accountNumber = accountNumber;
  this.accountType = accountType;
  this.balance = balance;
}

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

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

```
    }
}
class CurAcct extends Account
{
 double minBalance;
 double penaltyCharge;

 public CurAcct (String customerName, int accountNumber,
                 double balance)
 {
   super (customerName, accountNumber, "current", balan
                                                  -ce);
   minBalance = 1000;
   penaltyCharge = 50;
 }
 public void withdraw (double amount)
 {
   if (balance - amount >= minBalance)
   {
     balance -= amount;
     System.out.println ("withdrawal of $" + amount +
                         "successful");
   }
   else
   {
    System.out.println ("Insufficient balance. Withdrawal
                                              failed");
   }
  }
}
```

```
class SavAcct extends Account
{
    double InterestRate;
    public SavAcct(String customerName, int accountNumber,
                   double balance)
    {
      super(customerName, accountNumber, "Savings", balance);
      InterestRate = 0.05;
    }
    public void depositInterest()
    {
      double Interest = balance * InterestRate;
      balance += Interest;
      System.out.println("Interest of $" + Interest +
                         " deposited");
    }
    public void withdraw(double amount)
    {
      if (balance - amount >= 0)
      {
        balance -= amount;
        System.out.println("Withdrawal of $" + amount
                           + " successful");
      }
    }
}

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

```
System.out.println("creating a current account");
System.out.println("Enter customer name:");
String curName = s.next();
System.out.println("Enter account number");
int curNumber = scanner.nextInt();
System.out.println("Enter Initial Balance:");
double curBalance = s.nextDouble();
CurAcct currentAccount = new CurAcct(curName, curName,
                                     curBalance);
System.out.println("creating a Savings account");
Scanner s.next();
System.out.println("Enter Customer Name:");
String savName = s.next();
System.out.println("Enter account number:");
int savNumber = s.nextInt();
System.out.println("Enter Initial Balance:");
double savBalance = s.nextDouble();
SavAcct savingsAccount = new SavAcct(savName, savNumber,
                                     savBalance);

int choice;
do {
    System.out.println("choose: 1. Deposit to Current
Account \n 2. withdraw from current Account \n 3.
Deposit to Savings Account \n 4. withdraw from Savings
Account \n 5. Display Current Account Balance \n 6. Display
Savings Account Balance \n 7. Exit");
choice = s.nextInt();
switch(choice)
{
case 1:
    System.out.println("Enter amount to deposit:");
    double curDeposit = s.nextDouble();
```

```
        currentAccount.deposit(currDeposit);
        break;
    case 2:
        System.out.println("Enter amount to withdraw from current
                            account: ");
        double currWithdraw = s.nextDouble();
        currentAccount.withdraw(currWithdraw);
        break;
    case 3:
        System.out.println("Enter amount to deposit");
        double savDeposit = s.nextDouble();
        savingsAccount.deposit(savDeposit);
        break;
    case 4:
        System.out.println("Enter amount to withdraw: ");
        double savWithdraw = s.nextDouble();
        savingsAccount.withdraw(savWithdraw);
        break;
    case 5:
        System.out.println("Current Account Balance: ");
        currentAccount.displayBalance();
        break;
    case 6:
        System.out.println("Savings Account Balance: ");
        savingsAccount.displayBalance();
        break;
    case 7:
        System.out.println("Exiting...");
        break;
    default:
        System.out.println("Invalid choice");
    }
```

```
while (choice != 7);
  {
  }
}.
```

OUTPUT :

```
Creating Current Account :
Enter customer name : abc
Enter account number : 1234
Enter Initial Balance : 10000
Creating a Savings account.
Enter customer name : efg
Enter account number : 5678
Enter Initial Balance : 15000
Choose :
1. Deposit to current account
2. Withdraw from current account
3. Deposit to Savings Account
4. Withdraw from Savings Account
5. Display Current Account Balance
6. Display Savings Account Balance
7. Exit
5
Current Account Balance :
Balance : $10000.0
Choose :
1. Deposit to current account
2. Withdraw from Current account
3. Deposit to Savings account
4. Withdraw from Savings account
5. Display Current Account Balance
6. Display Savings Account Balance
7 Exit
```

7

Exiting....

PROGRAM 9 - AWT

Input :

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

public class DivisionMain1 extends Frame implements
  ActionListener{
      TextField num1, num2;
      Button divideButton;
      Label resultLabel;

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

  divideButton = new Button("Divide");
  Label number1 = new Label("Number 1: ", Label.RIGHT);
Label number2 = new Label("Number 2: ", Label.RIGHT);
num1 = new TextField(5);
num2 = new TextField(5);
resultLabel = new Label("Result: ", Label.RIGHT);

add(number1);
add(num1);
add(number2);
add(num2);
add(divideButton);
add(resultLabel);
```

```
divideButton.addActionListener(this);
addWindowListener(new WindowAdapter() {
public windowClosing(WindowEvent we) {
    System.exit(0);
  }
});
}
public void actionPerformed(ActionEvent ae) {
  try {
    int n1 = Integer.parseInt(num1.getText());
    int n2 = Integer.parseInt(num2.getText());
    if(n2==0) {
      throw new ArithmeticException("Cannot divide By zero");
    }
    int result = n1/n2;
resultLabel.setText("Result: " + result);
} catch (NumberFormatException e1) {
    showErrorDialog("Number Format Exception: Please enter
                    Integers only");
} catch (ArithmeticException e2) {
    showErrorDialog("Arithmetic Exception: "+e2.getMessage());
  }
}
private void showErrorDialog(String message) {
  Dialog dialog = new Dialog(this, "Error", true);
  dialog.setLayout(new FlowLayout());
  Label label = new Label(message);
  Button okButton = new Button("Ok");
  okButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      dialog.dispose();
  }
```

```
        });
        dialog.add(label);
        dialog.add(okButton);
        dialog.setSize(300,100);
        dialog.setVisible(true);
    }
    public static void main(String[] args) {
        DivisionMain1 divisionMain = new DivisionMain1();
        divisionMain.setSize(new Dimension(400,200));
        divisionMain.setTitle("Integer Division");
        divisionMain.setVisible(true);
    }
}
```

OUTPUT :