B.M.S COLLEGE OF ENGINEERING BENGALURU

Autonomous Institute, Affiliated to VTU

LAB REPORT

23CS3PCOOJ

Submitted in partial fulfilment of the requirements for Lab

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

S GAJANANA NAYAK(1BM22CS227)

Department of Computer Science and Engineering, B.M.S

College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019 2023-2024.

INDEX

Sl.No.	Title	Date
1	Complete scanned Observation Book	12/12/2023 - 20/02/2024
2	Lab 1	12/12/2023
3	Lab 2	19/12/2023
4	Lab 3	26/12/2023
5	Lab 4	02/01/2024
6	Lab 5	09/01/2024
7	Lab 6	23/01/2024
8	Lab 7	30/01/2024
9	Lab 8	06/02/2024
10	Lab 9	20/02/2024
11	Lab 10	13/02/2024

⇒ Lab Program 1 (Quadratic Eqn) — B. Gajanana Nayak 1BM22CS227 12/12/23

output1:

```
enter the coefficients of a, b, c
1
-5
9
roots are real and distinct
root1 = 4.561553128128
root2 = 4.561552128128
```

output 2:

```
enter the coefficients of a, b, c
1 2 1
roots are real and equal
root1 = root2 = -1.0
```

output 3:

```
enter the coefficients of a, b, c
0 4 5
not a quadratic equation
enter a non-zero value for a:
1
roots are imaginary
root1 = -2.0 + i NaN
root2 = -2.0 - i NaN
```

7) Find sum of 5 digits number

```
import java.util.*;
class digits {
    public static void main (String args[]){
    long number, sum;
    Scanner sc = new Scanner (System.in);
    System.out.println("Enter a 5-digit number:");
    number = sc.nextLong();
    for (sum = 0; number != 0; number = number/10) {
        sum = sum + number %10;
    }
    System.out.println("sum of digits: "+sum);
    }
}
```

Output (S Gajanana Nayak 1BM22CS227)

```
Enter a 5-digit number:
12345
Sum of digits: 15
```

//LAB-PROGRAM-(2) 19/12/23

```
import java.util.Scanner;
class Subject
{
  int subjectMarks;
  int credits;
  String grade;
}
class Student
{
  String name;
  String usn;
  double SGPA;
  Scanner s;
  Subject subject[];
  Student()
  {
    int i;
    Subject = new Subject[9];
    for(i=0; i<9; i++)
        subject[i] = new Subject();
    s = new Scanner(System.in);
  }
void getStudentDetails()
  {
    System.out.println("Enter your name:");
    name = s.nextLine();
    System.out.println("Enter your usn:");
    usn = s.nextLine();
  }
```

```
void getMarks()
{
 int i;
 for(i=0; i<8; i++)
 {
   System.out.println("Enter the marks & credits for
                       course " + i + ":");
   System.out.println("marks:");
   int marks = s.nextInt();
   System.out.println("credits:");
   int credit = s.nextInt();
   subject[i].subjectMarks = marks;
   subject[i].credits = credit;

   if(marks>=90)
   {
     subject[i].grade="O";
   }
   else if(marks>=80){
       subject[i].grade="A+";
   }
   else if(marks>=70){
       subject[i].grade="A";
   }
   else if(marks>=60){
       subject[i].grade="B+";
   }
   else if(marks>=50){
       subject[i].grade="B";
   }
   else if(marks<=40){
       subject[i].grade="C";
   }
   else{
       subject[i].grade="F";
   }
```

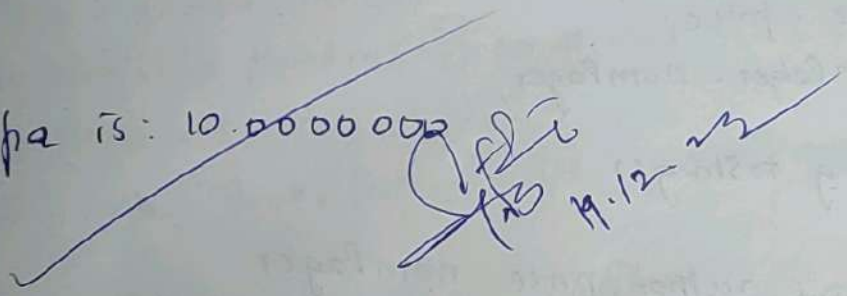
```
    }
}
void computeSGPA()
{
  int i;
  double sgpa;
  double totalcredits=0
  double totalgp=0;
  for(i=0;i<8;i++)
  {
    totalcredits += subject[i].credits;
    switch(subject[i].grade)
    {
      case "O": totalgp += 10* subject[i].credits;
                break;
      case "A+": totalgp += 9* subject[i].credits;
                 break;
      case "A": totalgp += 8* subject[i].credits;
                break;
      case "B+": totalgp += 7* subject[i].credits;
                 break;
      case "B": totalgp += 6* subject[i].credits;
                break;
      case "C": totalgp += 5* subject[i].credits;
                break;
      case "F": totalgp += 0* subject[i].credit;
                break;
    }
  }
  sgpa = totalgp/totalcredits;
  System.out.println("the sgpa is :"+sgpa);
 }
}
class sgpa
{
  public static void main(String args[])
  {
```

```
        Student s1 = new Student();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();
    }
}
```

Output: (S Gajarana Nayak 1BM22CS227)

```
Enter your name:
Gajarana
Enter your usn:
1BM22CS227
Enter the marks and credits for course 0:
Marks:
90
credits:
4
Enter the marks and credits for course 1:
Marks:
91
credits:
4
Enter the marks and credit for course 2:
Marks:
99
credits:
3
Enter the marks & credits for course 3:
Marks:
91
Credit:
3
Enter the marks & credits for course 4:
Marks:
93
Credits:
2
```

Enter the marks & credits for course 5:
Marks: 90
Credits:
01
Enter the marks & credits for course 6:
Marks:
92
Credits:
01
Enter the marks & credits for course 7:
Marks:
96
credits:
02
the sgpa is: 10.000000



// LAB - PROGRAM - (3) 26/12/23

```
import java.util.Scanner;
class Books
{
  String name;
  String author;
  int price;
  int numPages;

  Books (String name, String author, int price, int numPages)
  {
    this.name = name;
    this.author = author;
    this.price = price;
    this.numPages = numPages;
  }
  public String toString()
  {
    String name, author, price, numPages;
    name = "Book name: " + this.name + "\n";
    author = "Author name: " + this.author + "\n";
    price = "Price: " + this.price + "\n";
    numPages = "Number of pages: " + this.numPages + "\n";
    return name + author + price + numPages;
  }
}
public class Mainbooks
{
  public static void main(String args[])
  {
    Scanner s = new Scanner(System.in);
    int n;
    int i;
    String name;
    String author;
    int price;
    int numPages;
```

```
System.out.println ("Enter the number of books: ");
n = s.nextInt();
Books b[];
b = new Books[n];

for(i=0; i<n; i++)
{
  System.out.println("Enter the details of book" + (i+1)
                                    + ": ");
  System.out.println("Enter the name of the book: ");
  name = s.next();
  System.out.println ("Enter the author name: ");
  author = s.next();
  System.out.println ("Enter the price: ");
  price = s.nextInt();
  System.out.println("Enter the number of pages: ");
  numPages = s.nextInt();
  b[i] = new Books(name, author, price, numPages);
}
System.out.println("Book Details: ");
for(i=0; i<n; i++)
{
  System.out.println(b[i]);
}
}
}
```

Output:

```
Enter the number of books:
2
Enter the details of book1:
Enter the name of the book:
Secret
Enter the author name:
william
```

Enter the price:
350
Enter the number of pages:
123
Enter the details of book2:
Enter the name of the book:
revolution
Enter the author name:
chetan
Enter the price:
200
Enter the number of pages:
301
Book Details:
Book name: secret
Author name: william
Price: 350
Number of Pages: 123

Book name: revolution
Author name: chetan
Price: 200
Number of pages: 301

26.12.22

LAB - PROGRAM (4) - Shape 2/1/24

```
import java.util.Scanner;

class inputsconner
{
  protected sconner s;
  public inputSconnar()
  {
     s = new sconner (system.in);
  }
  public int getInput(string message)
  {
     system.out.println(message);
     return sconner.nextInt();
  }
}

abstract class Shape extends inputsconner
{
  protected int a, b;
  public Shape ()
  {
     super();
  }
  abstract public void printArea();
}

class Rectangle extends Shape
{
  protected int a, b;
  public Rectangle()
  {
     super();
  }
  public void printArea()
  {
    a = getInput ("Enter the length:");
    b = getInput ("Enter the breadth:");
    int area = a * b;
    System.out.println("Area of the Rectangle :"+area);
  }
}
```

```
class Triangle extends Shape
{
    protected int a, b;
    public Triangle()
    {
        super();
    }
    public void printArea()
    {
        a = getInput("Enter the side1:");
        b = getInput("Enter the side 2:");
        double area = 0.5 * a * b;
        System.out.println("Area of the Triangle:" + area);
    }
}
class Circle extends Shape
{
    protected int a;
    public Circle()
    {
        super();
    }
    public void printArea()
    {
        a = getInput("Enter the radius:");
        double area = 3.14 * a * a;
        System.out.println("Area of the circle:" + area);
    }
}
public class MainShape
{
    public static void main(String[] args)
    {
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
```

```
        r.printArea();
        t.printArea();
        c.printArea();
    }
}
```

Output: (S Gajanana Nayak 1BM22CS227)

```
Enter the length:
2
Enter the bredth:
4
Area of the Rectangle: 8
Enter the side 1:
2
Enter the side 2:
2
Area of the Triangle: 2.0
Enter the radius:
5
Area of the Circle: 78.5
```

2/1/24

9/1/24

LAB - PROGRAM (5)

```
import java.util.Scannor;
class account
{
   String name;
   int accno;
   String type;
   double balance;

   account(String name, int accno, String type, double balance)
   {
      this.name=name;
      this.accno=accno;
      this.type=type;
      this.balance=balance;
   }
   void deposit(double amount)
   {
      balance+=amount;
   }
   void withdraw(double amount)
   {
      if(balance-amount)>=0)
      {
         balance-=amount;
      }
      else
      {
         System.out.println("Insufficient balance, cant withdraw");
      }
   }

   void display()
   {
      System.out.println("name:"+name+" accno:"+accno+
      "type:"+type+" balance:"+balance);
   }
}
```

```
class savAcct extends account
{
 private static double rate = 5;
 savAcct ( String name, int accno, double balance)
 { super( name, accno, "savings", balance);
 }
 void interest()
 {
   balance += balance * (rate)/100;
   System.out.println("balance:" + balance);
 }
}

class currAcct extends account
 {
 private double minBal = 500;
 private double serviceCharges = 50;
 currAcct (String name, int accno, double balance)
 { super (name, accno, "current", balance);
 }
 void checkmain()
 {
   if (balance < minBal)
   {
     System.out.println("balance is less than min
     balance, service charges imposed," + servicecharges);
     balance -= servicecharges;
     System.out.println("balance is" + balance);
   }
 }
}
```

```
class accountmain
{
  public static void main (String a[])
  {
    Scanner s = new Scanner (System.in)
    System.out.println ("Enter the name :");
    String name = s.next();
    System.out.println ("Enter the type :");
    String type = s.next();
    System.out.println ("Enter acc no:");
    int accno = s.nextInt();
    System.out.println ("Enter initial balance :");
    double balance = s.nextDouble();

    int ch;
    double amt1, amt2;
    account acc = new account (name, accno, type, balance);
    SavAcct sa = new SavAcct (name, accno, balance);
    curAcct ca = new curAcct (name, accno, balance);

    while (true)
    {
      if (acc.type.equals ("savings"))
      {
        System.out.println ("\n Menu \n 1. deposit 2. withdraw
        3. compute interest 4. display");
        System.out.println ("Enter your choice :");
        ch = s.nextInt();
        switch (ch)
        {
          case 1: System.out.println ("Enter amount :");
                  amt1 = s.nextInt();
                  sa.deposit (amt1);
                  break;
```

```
case 2 : System.out.println("Enter amount : ");
         amt2 = s.nextInt();
         sa.withdraw(amt2);
         break;
case 3 : sa.intrest();
         break;
case 4 : sa.display();
         break;
case 5 : System.exit(0);
default : System.out.println("Invalid input");
         break;
}
}
else
{
System.out.println("\n menu\n 1. deposit 2. withdraw
 3. display");
System.out.println("Enter your choice:");
ch = s.nextInt();
switch(ch)
{
 case 1 : System.out.println("Enter amount:");
         amt1 = s.nextInt();
         ca.deposit(amt1);
         break;
 case 2 : System.out.println("Enter amount:");
         amt2 = s.nextInt();
         ca.withdraw(amt2);
         ca.checkmin();
         break;
case 3 : ca.display();
         break;
```

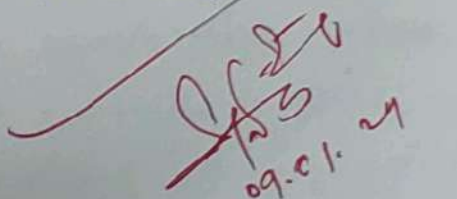
```
        case 4: System.exit(0);
        default : System.out.println("Invalid Input");
                  break;
        }
    }
  }
}
```

Output: (S Gajanana Nayak 1BM22CS227)

```
Enter the name :
john
enter the type:
current
enter the acc no:
1
enter the initial balance:
1000
menu
1. deposit 2. withdraw 3. display
enter choice:1
enter the amount:
500

menu
1. deposit 2. withdraw 3. display
enter choice:
2
enter amount:
600

menu
1. deposit 2. withdraw 3. display
enter the choice:
3
```

name: john accno: 1 type: current balance: 900

menu
1. deposit 2. withdraw 3. display
enter the choice:
H



WEEK-6 : (S Gajanana Nayak 1BM22CS227) 23/01/24

```
// Student java file

package CIE;
import java.util.Scanner;

public class Student
{
    protected String usn = new String();
    protected String name = new String();
    protected int sem;

    public void inputStudentDetails()
    {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = s.next();
        System.out.print("Enter Name: ");
        name = s.next();
        System.out.print("Enter Semester: ");
        sem = s.nextInt();
    }

    public void displayStudentDetails()
    {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}
```

// Internals java file

```
package CIE;
import java.util.Scanner;
public class Internals extends Student
{
   protected int marks[] = new int[5];
   public Internals(){}
   public void inputCIEMarks()
   {
      Scanner s = new Scanner(System.in);
      System.out.println("Enter Internal Marks for" + name);
      for(int i=0; i<5; i++)
      {
         System.out.print("Subject " + (i+1) + "marks:");
         marks[i] = s.nextInt();
      }
   }
}
```

// External java file.

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals
{
   protected int marks[];
   protected int finalMarks[];
   public Externals()
   {
      marks = new int[5];
      finalMarks = new int[5];
   }
```

```
public Externals()
{
}

public void inputSEEMarks()
{
    Scanner s = new Scanner(System.in);
    System.out.println("Enter SEE Marks for:" + name);
    for(int i=0; i<5; i++)
    {
        System.out.print("subject "+(i+1)+" marks:");
        marks[i] = s.nextInt();
    }
}
public void calculateFinalMarks()
{
    for(int i=0; i<5; i++)
        finalMarks[i] = marks[i]/2 + super.marks[i];
}
public void displayFinalMarks()
{
    displayStudentDetails();
    for(int i=0; i<5; i++)
        System.out.println("subject"+(i+1)+":"+
                           finalMarks[i]);
}
}

// main java file
import SEE.Externals;
public class Main
{
    public static void main(String args[])
    {
```

```
int numofstudents = 2;
External finalMarks[] = new External[numofStudents];
for(i=0 ; i<numofstudents ; i++)
{
    finalMarks[i] = new External();
    finalMarks[i].inputStudentDetails();
    System.out.println("Enter CIE marks :");
    finalMarks[i].inputCIEMarks();
    System.out.println("Enter SEE marks :");
    finalMarks[i].inputSEEmarks();
}

System.out.println("Displaying data :\n");
for(int i=0; i<numofstudents; i++)
{
    finalMarks[i].calculateFinalMarks();
    finalMarks[i].displayFinalMarks();
}
}
}
```

Output:

```
Enter USN : 111
Enter Name : Ram
Enter Semester : 3
Enter CIE Marks:
Enter Internal marks for Ram
Subject 1 marks : 33
Subject 2 marks : 36
Subject 3 marks : 28
Subject 4 marks : 31
Subject 5 marks : 40
```

Enter SEE marks for Ram
Subject 1 marks: 89
Subject 2 marks: 91
Subject 3 marks: 78
Subject 4 marks: 84
Subject 5 marks: 90

Enter USN: 112
Enter Name: Raju
Enter Semester: 3
Enter CIE marks:
Enter Internal marks for Raju
Subject 1 marks: 33
Subject 2 marks: 28
Subject 3 marks: 23
Subject 4 marks: 27
Subject 5 marks: 36

Enter SEE marks:
Enter SEE marks for Raju
Subject 1 marks: 84
Subject 2 marks: 85
Subject 3 marks: 99
Subject 4 marks: 92
Subject 5 marks: 77

Displaying data:
USN: 111
Name: Ram
Semester: 3
Subject 1: 77
Subject 2: 81
Subject 3: 67
Subject 4: 73

Subject 5 : 85

USN : 112

Name : Raju

Semester : 3

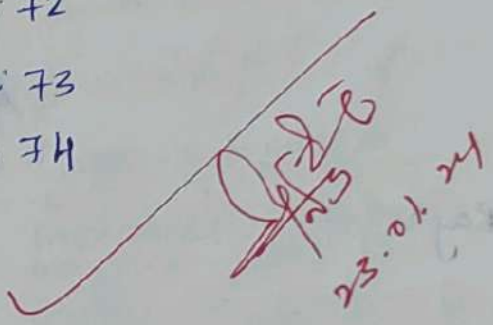
Subject 1 : 75

Subject 2 : 70

Subject 3 : 72

Subject 4 : 73

Subject 5 : 74



WEEK-7 : (S Gopavarna Nayak IBM22CS227) 30/1/24

```
import java.util.Scanner;
class WrongAge extends Exception
{
  public WrongAge(String message)
  {
    super(message);
  }
}
class InputScanner
{
  protected Scanner s;
  public InputScanner()
  {
    s = new Scanner(System.in);
  }
}
class Father extends InputScanner
{
  protected int fatherAge;
  public Father() throws WrongAge
  {
    System.out.println("Enter Father's Age:");
    fatherAge = s.nextInt();
    if (fatherAge<0)
    {
      throw new WrongAge("Age cannot be negative:");
    }
  }
  public void display()
  {
    System.out.println("Father's Age:" + fatherAge);
  }
}
```

```
class Son extends Father
{
  private int sonAge;
  public Son() throws WrongAge
  {
    super();
    System.out.println("Enter son's age:");
    sonAge = s.nextInt();
    if (sonAge > fatherAge)
    {
      throw new WrongAge("Son's age cannot be greater than
                          father's age");
    }
    else if (sonAge<0)
    {
      throw new WrongAge("Age cannot be negative");
    }
  }
  public void display()
  {
    super.display();
    System.out.println("Son's Age: " + sonAge);
  }
}
public class FatherSonAge
{
  public static void main(String args[])
  {
    try
    {
      Son son = new Son();
      son.display();
    }
```

```
        catch (wrongAge e)
        {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:

```
Enter Father's Age:
45
Enter Son's age:
20
Father's Age: 45
Son's Age: 20
```

```
Enter Father's Age:
56
Enter Son's age:
78
Error: Son's age cannot be
       greater than father's age.
```

30.01.24

(06/02/24)

WEEK-8: (Thread Program)

```
class BMSThread extends Thread
{
  public void run()
  {
    while (true)
    {
      System.out.println("BMS College of Engineering");
      try
      {
        Thread.sleep(10000);
      }
      catch (InterruptedException e)
      {
        e.printStackTrace();
      }
    }
  }
}
class CSEThread extends Thread
{
  public void run()
  {
    while (true)
    {
      System.out.println("CSE");
      try
      {
        Thread.sleep(2000);
      }
      catch (InterruptedException e)
      {
        e.printStackTrace();
      }
    }
  }
}
```

```
public class ThreadExample
{
    public static void main(String[] args)
    {
        BMSThread bmsThread = new BMSThread();
        bmsThread.start();

        CSEThread cseThread = new CSEThread();
        cseThread.start();
    }
}
```

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
```

(20/2/24)

LAB - (9):

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

class UserInterface {
    UserInterface() {
    // create JFrame container
    JFrame jfrm = new JFrame("Divider App");
    jfrm.setSize(275,150);
    jfrm.setLayout(new FlowLayout());
    // to terminate on close
    jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    // text label
    JLabel jlab = new JLabel("Enter the divider & dividend:");

    // add text field for both numbers
    JTextField ajtf = new JTextField(8);
    JTextField bjtf = new JTextField(8);

    // calc button
    JButton button = new JButton("calculate");

    // labels
    JLabel err = new JLabel();
    JLabel alab = new JLabel();
    JLabel blab = new JLabel();
    JLabel anslab = new JLabel();

// add in order
    jfrm.add(err); // to display error message
    jfrm.add(jlab);
    jfrm.add(ajtf);
```

```
jfrm.add(bjtb);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

ActionListner calculateListner = new ActionListner() {
  public void actionPerformed(ActionEvent evt) {
    try {
      int a = Integer.parseInt(ajtb.getText());
      int b = Integer.parseInt(bjtb.getText());
      if(b==0) {
        throw new ArithmeticException();
      }
      int ans = a/b;
      alab.setText("\n A=" +a);
      blab.setText("\n B=" +b);
      anslab.setText("\n Ans=" +ans);
      err.setText(" "); // Clear any prev errors
    } catch (NumberFormatException e) {
      displayErrorMessage("Enter only Integers!");
    } catch (ArithmeticException e) {
      displayErrorMessage("B should be nonzero");
    }
  }
private void displayErrorMessage(String message) {
    alab.setText(" ");
    blab.setText(" ");
    anslab.setText(" ");
    err.setText(message);
}}
```

```
button.addActionListener(calculateListner);

//display frame
jfrm.setVisible(true);
}
public static void main(String args[]) {
    // create frame on event dispatching thread
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            new UserInterface();
        }
    });
}
}
```

Output:

Enter the divider and dividend:

6 3

Calculate A = 6 B = 3 Ans = 2

Functions Used:

JFrame: It is a top level container in Java Swing that represents a window with a title bar, border and optional menubar.

setSize: It is used to set size of the frame

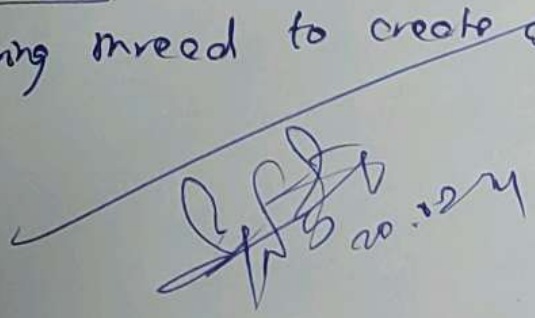
setLayout: This line sets the layout manager for the frame to FlowLayout, which arranges components from left to right in a flow like manner.

add: This line adds the error label to the frame

setText: This line sets the text of the 'A' label to display the value of 'A'.

setVisible: This line makes the frame visible

invokeLater: This line schedules a job for the event-dispatching thread to create and show the GUI



(06/02/24)

LAB - PROGRAM 10:

IPC:

```
class Q
{ int n;
boolean valueset = false;
synchronized int get()
{
   while (!valueset)
   try
   {
     System.out.println("\n Consumer waiting\n");
     wait();
   }
   catch (InterruptedException e)
   {
     System.out.println("Interrupted Exception caught");
   }
   System.out.println("Got: "+n);
   valueset = false;
   System.out.println("\nIntimate Producer\n");
   notify();
  return n;
}
synchronized void put(int n)
{
   while (valueSet)
   try
   {
     System.out.println("\nProducer waiting\n");
     wait();
   }
```

```
        catch (InterruptedException e)
        {
          System.out.println("Interrupted Exception caught");
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\n Intimate Consumer\n");
        notify();
    }
}

class Producer implements Runnable
{
  Q q;
  Producer(Q q)
  {
    this.q = q;
    new Thread(this, "Producer").start();
  }
  public void run()
  {
    int i=0;
    while (i<5)
    {
      q.put(i++);
    }
  }
}
```

```
class Consumer implements Runnable
{
  Q q;
  Consumer(Q q)
  {
    this.q = q;
    new Thread(this, "Consumer").start();
  }
  public void run()
  {
    int i = 0;
    while(i<5)
    {
      int r = q.get();
      System.out.println("Consumed: " + r);
      i++;
    }
  }
}
class PCFixed {
public static void main(String args[])
{
    Q q = new Q();
    new Producer(q);
    new Consumer(q);
    System.out.println("Press Control-C to stop.");
  }
}
```

Output:

Press Control - C to stop

Put: 0

Intimate Consumer

Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer

Producer waiting

Consumed : 0

Got : 1

Intimate Producer

Consomed : 1

Put: 2

Intimate Consumer

Producer waiting

Got: 2

Intimate Producer

Cosumed : 2

Put: 3

Intimate Consumer

Producer waiting

Got: 3

Intimate Producer

Consumed : 3

Put : 4

Intimate Consumer

Got : 4

Intimate Producer

Consumed : 4

06.12.24

LAB - PROGRAM - (10) (Deadlock) (13|2|24)

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

```
catch (Exception e)
{
    System.out.println ("B Interrupted");
}
System.out.println (name + "trying to call A.last()");
a.last();
}
void last()
{ System.out.println ("Inside A.last");
}
}
class Deadlock implements Runnable
{
    A a = new A();
    B b = new B();
    Deadlock()
    {
        Thread.currentThread().setName ("MainThread");
        Thread t = new Thread (this, "RacingThread");
        t.start();
        a.foo(b); //get lock on a in this thread
        System.out.println ("Back in main thread");
    }
    public void run()
    {
        b.bar(a); //get lock on b in other thread
        System.out.println ("Back in other thread");
    }
```

```
public static void main (String args[])
{
    new Deadlock();
}
}
```

Output:

```
Main Thread entered A.foo
Racing Thread entered B.bar
Racing Thread trying to call B.last()
Inside A.last
Back in main thread
Back in other thread.
```

LAB PROGRAMS (OOJ)

S Gajanana Nayak(1BM22CS227)

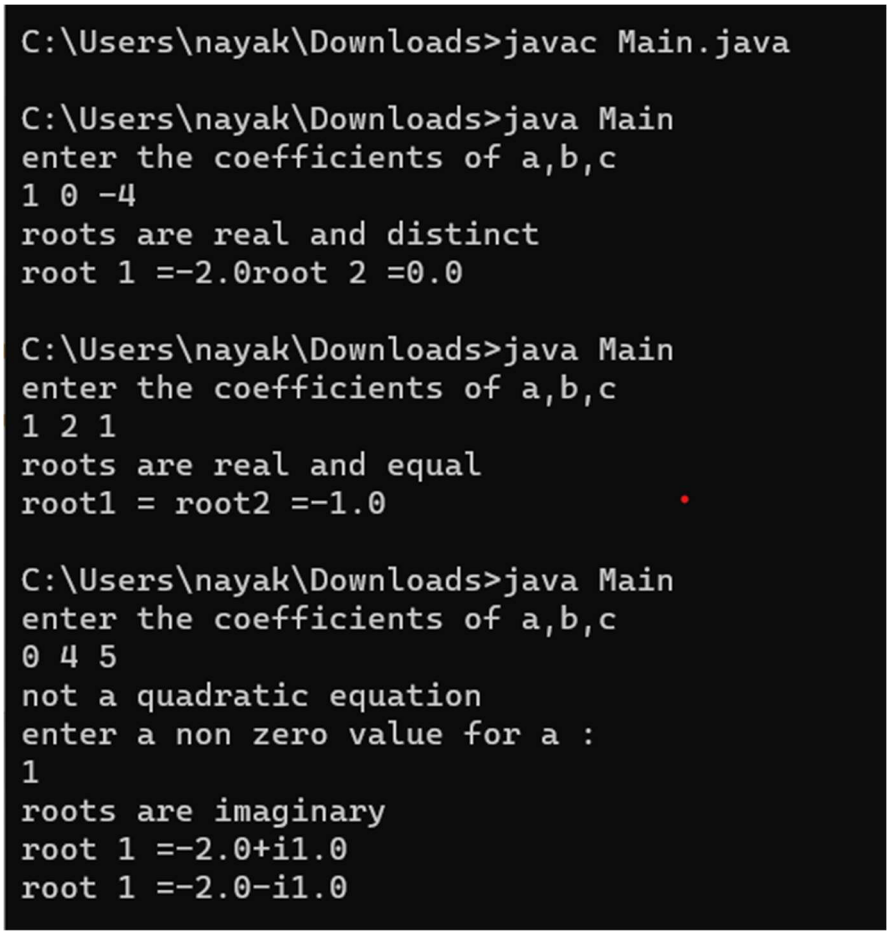
//PROGRAM 1

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.

```
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,c");
        a=s.nextInt();
        b=s.nextInt();
        c=s.nextInt();
}
void compute()
{
        while(a==0)
        {
                System.out.println("not a quadratic equation");
                System.out.println("enter a non zero value for a : ");
                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("root1 = root2 ="+r1);
        }
        else if(d>0)
        {
                r1=((-b)+(Math.sqrt(d)))/(double)(2*a);
                r1=((-b)-(Math.sqrt(d)))/(double)(2*a);
                System.out.println("roots are real and distinct");
                System.out.println("root 1 ="+r1+"root 2 ="+r2);
        }
        else if(d<0)
        {
                System.out.println("roots are imaginary");
                r1=(-b)/(2*a);
                r2=Math.sqrt(-d)/(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:



//PROGRAM 2

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

```
import java.util.Scanner;


class Subject
{
int subjectMarks;
int credits;
String grade;
}
```

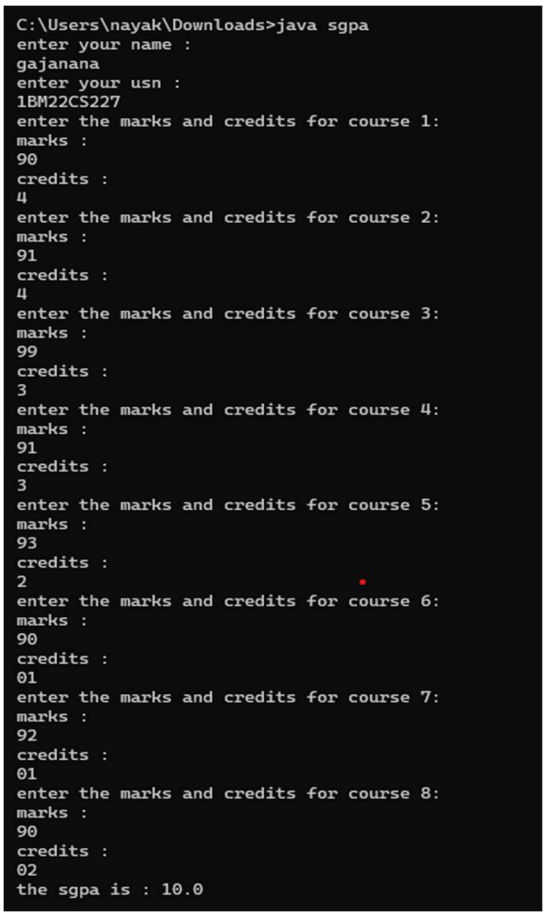
```
class Student
{
String name;
String usn;
double SGPA;
Scanner s;
Subject subject[];
Student()
{
        int i;
        subject = new Subject[9];
        for(i=0;i<9;i++)
                subject[i] = new Subject();
        s = new Scanner(System.in);
}

void getStudentDetails()
{
        System.out.println("enter your name : ");
        name = s.nextLine();
        System.out.println("enter your usn : ");
        usn = s.nextLine();
}

void getMarks()
{
        int i;
        for(i=0;i<8;i++)
        {
                System.out.println("enter the marks and credits for course " + (i+1) + ":");
                System.out.println("marks : ");
```

```
int marks = s.nextInt();
System.out.println("credits : ");
int credit = s.nextInt();
subject[i].subjectMarks = marks;
subject[i].credits = credit;

if(marks >= 90 && marks<=100)
{
        subject[i].grade = "O";
}
else if(marks>=80 && marks<90)
{
        subject[i].grade = "A+";
}
else if(marks>=70 && marks<80)
{
        subject[i].grade = "A";
}
else if(marks>=60 && marks<70)
{
        subject[i].grade = "B+";
}
else if(marks>=50 && marks<60)
{
        subject[i].grade = "B";
}
else if(marks>=40 && marks<50)
{
        subject[i].grade = "C";
}
else if(marks>=0 && marks<40)
{
```

```
                    subject[i].grade = "F";
                }
        }
}
void computeSGPA()
{
        int i;
        double sgpa;
        double totalcredits = 0;
        double totalgradepoints = 0;

        for(i=0;i<8;i++)
        {
                totalcredits += subject[i].credits;
                switch(subject[i].grade)
                {
                        case "O" : totalgradepoints += 10*subject[i].credits;
                        break;
                        case "A+" : totalgradepoints += 9*subject[i].credits;
                        break;
                        case "A" : totalgradepoints += 8*subject[i].credits;
                        break;
                        case "B+" : totalgradepoints += 7*subject[i].credits;
                        break;
                        case "B" : totalgradepoints += 6*subject[i].credits;
                        break;
                        case "C" : totalgradepoints += 5*subject[i].credits;
                        break;
                        case "F" : totalgradepoints += 0*subject[i].credits;
                        break;
                }
        }
```

```
        sgpa = totalgradepoints/totalcredits;
        System.out.println("the sgpa is : "+sgpa);
}
}
class sgpa
{
public static void main(String args[])
{
        Student s1 = new Student();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();
}
}
```

OUTPUT:



//PROGRAM 3

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

```
import java.util.Scanner;

class Books
{
   String name;
   String author;
   int price;
   int numPages;

   Books(String name, String author, int price, int numPages)
   {
      this.name = name;
      this.author = author;
      this.price = price;
      this.numPages = numPages;
   }

   public String toString()
   {
      String name, author, price, numPages;
      name = "Book name s: " + this.name + "\n";
      author = "Author name : " + this.author + "\n";
      price = "Price : " + this.price + "\n";
      numPages = "Number of pages : " + this.numPages + "\n";
      return name + author + price + numPages;
   }
```

```
}

public class Mainbooks
 {
   public static void main(String[] args)
   {
      Scanner s = new Scanner(System.in);
      int n;
 int i;
      String name;
String author;
      int price;
int numPages;
      System.out.println("Enter the number of books:");
      n = s.nextInt();
      Books b[];
b=new Books[n];
      for (i = 0; i < n; i++)
      {
         System.out.println("Enter the details of book" + (i+1) + ":");

         System.out.println("Enter the name of the book:");
         name = s.next();
         System.out.println("Enter the author name:");
         author = s.next();
         System.out.println("Enter the price:");
         price = s.nextInt();
         System.out.println("Enter the number of pages:");
         numPages = s.nextInt();

         b[i] = new Books(name, author, price, numPages);
      }
```

```
        System.out.println("Book details:");
        for (i = 0; i < n; i++)
        {
            System.out.println(b[i]);
        }
    }
}
```

OUTPUT:

```
C:\Users\nayak\Downloads>javac Mainbooks.java

C:\Users\nayak\Downloads>java Mainbooks
Enter the number of books:
2
Enter the details of book1:
Enter the name of the book:
secret
Enter the author name:
william
Enter the price:
350
Enter the number of pages:
123
Enter the details of book2:
Enter the name of the book:
revelution
Enter the author name:
chetan
Enter the price:
200
Enter the number of pages:
301
Book details:
Book name s: secret
Author name : william
Price : 350
Number of pages : 123

Book name s: revelution
Author name : chetan
Price : 200
Number of pages : 301
```

//PROGRAM 4

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

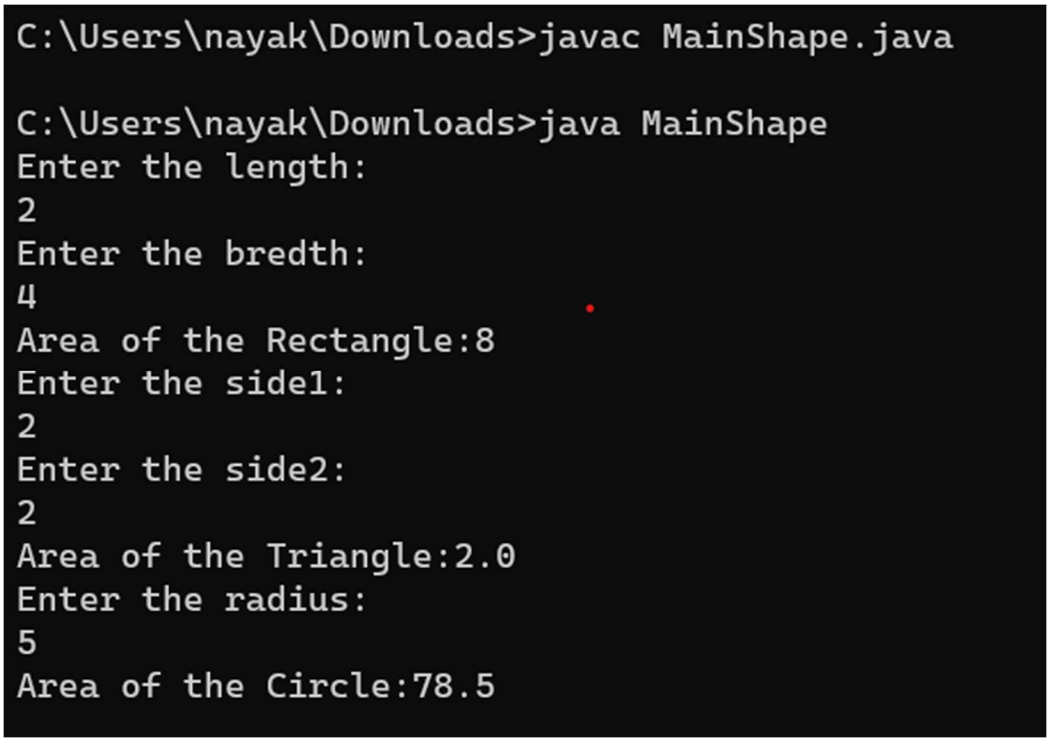
```
import java.util.Scanner;

class inputScanner
{

protected Scanner s;

public inputScanner()
{
        s = new Scanner(System.in);
}

public int getInput(String message)
{
        System.out.println(message);
        return s.nextInt();
}



}

abstract class Shape extends inputScanner
{
protected int a,b;
```

```
public Shape()
{
        super();
}

abstract public void printArea();
}

class Rectangle extends Shape
{
protected int a,b;
public Rectangle()
{
        super();
}

public void printArea()
{

        a=getInput("Enter the length:");
        b=getInput("Enter the bredth:");
        int area= a*b;
        System.out.println("Area of the Rectangle:" +area);
}
}

class Triangle extends Shape
{
protected int a,b;
public Triangle()
{
        super();
```

```
}

public void printArea()
{
        a=getInput("Enter the side1:");
        b=getInput("Enter the side2:");
        double area=0.5*a*b;
        System.out.println("Area of the Triangle:" +area);
}

}

class Circle extends Shape
{
protected int a;
public Circle()
{
        super();
}

public void printArea()
{
        a=getInput("Enter the radius:");
        double area=3.14*a*a;
        System.out.println("Area of the Circle:" +area);

}

}

public class MainShape
{
```

```
public static void main(String[] args)
{
        Rectangle r=new Rectangle();
        Triangle t=new Triangle();
        Circle c=new Circle();

        r.printArea();
        t.printArea();
        c.printArea();
}
}
```

OUTPUT:



//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.

```
import java.util.Scanner;
class account
{
	String name;
	int accno;
	String type;
	double balance;

	account(String name,int accno,String type,double balance)
	{
		this.name=name;
		this.accno=accno;
		this.type=type;
		this.balance=balance;
	}
	void deposit(double amount)
	{
		balance+=amount;
	}
	void withdraw(double amount)
	{
```

```
                if((balance-amount)>=0)
                {
                        balance-=amount;
                }
                else
                {
                        System.out.println("insufficient balance,cant withdraw");
                }
        }

        void display()
        {
                System.out.println("name:"+name+"accno:"+accno+"type:"+type+"balance:"+balance);
        }
}
class savAcct extends account
{

        private static double rate=5;
        savAcct(String name,int accno,double balance)
        {
                super(name,accno,"savings",balance);


        }

        void interest()
        {
                balance+=balance*(rate)/100;
                System.out.println("balance:"+balance);
        }



}
```

```
class curAcct extends account
{

	private double minBal=500;
	private double serviceCharges=50;

	curAcct(String name,int accno,double balance)
	{
		super(name,accno,"current",balance);

	}


	void checkmin()
	{

		if(balance<minBal)
		{
			System.out.println("balance is less than min balance,service charges imposed:"+serviceCharges);
			balance-=serviceCharges;
			System.out.println("balance is:"+balance);
		}

	}

}
class accountMain
{
	public static void main(String a[])
	{
		Scanner s=new Scanner(System.in);
		System.out.println("enter the name :");
```

```
String name=s.next();
System.out.println("enter the type(current/savings):");
String type=s.next();
System.out.println("enter the account number:");
int accno=s.nextInt();
System.out.println("enter the intial balance:");
double balance=s.nextDouble();
int ch;
double amount1,amount2;
account acc=new account(name,accno,type,balance);
savAcct sa=new savAcct(name,accno,balance);
curAcct ca=new curAcct(name,accno,balance);
while(true)
{
        if(acc.type.equals("savings"))
        {
                System.out.println("\nMenu\n1.deposit 2.withdraw 3.compute interest
                4.display");
                System.out.println("enter the choice:");
                ch=s.nextInt();
                switch(ch)
                {
                        case 1:System.out.println("enter the amount:");
                                amount1=s.nextInt();
                                sa.deposit(amount1);
                                break;
                        case 2:System.out.println("enter the amount:");
                                amount2=s.nextInt();
                                sa.withdraw(amount2);
                                break;
                        case 3:sa.interest();
                                break;
                        case 4:sa.display();
```

```
                                        break;
                                case 5:System.exit(0);
                                default:System.out.println("invalid input");
                                        break;
                        }
                }
                else
                {
                        System.out.println("\nMenu\n1.deposit 2.withdraw  3.display");
                        System.out.println("enter the choice:");
                        ch=s.nextInt();
                        switch(ch)
                        {
                                case 1:System.out.println("enter the amount:");
                                        amount1=s.nextInt();
                                        ca.deposit(amount1);
                                        break;
                                case 2:System.out.println("enter the amount:");
                                        amount2=s.nextInt();
                                        ca.withdraw(amount2);
                                        ca.checkmin();
                                        break;

                                case 3:ca.display();
                                        break;
                                case 4:System.exit(0);
                                default:System.out.println("invalid input");
                                        break;
                        }
                }
        }
    }
}
```

OUTPUT:

```
C:\Users\nayak\Downloads>javac accountMain.java

C:\Users\nayak\Downloads>java accountMain
enter the name :
john
enter the type(current/savings):
current
enter the account number:
1
enter the intial balance:
1000

Menu
1.deposit 2.withdraw  3.display
enter the choice:
1
enter the amount:
500

Menu
1.deposit 2.withdraw  3.display
enter the choice:
2
enter the amount:
600

Menu
1.deposit 2.withdraw  3.display
enter the choice:
3
name:johnaccno:1type:currentbalance:900.0

Menu
1.deposit 2.withdraw  3.display
enter the choice:
4
```

//PROGRAM 6

Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current

semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

```
package CIE;

import java.util.Scanner;

public class Student
{
   protected String usn = new String();
   protected String name = new String();         protected
int sem;

   public void inputStudentDetails()
     {
          Scanner s = new Scanner(System.in);
         System.out.print("Enter USN: ");
         usn = s.next();
          System.out.print("Enter Name: ");
         name = s.next();
         System.out.print("Enter Semester: ");
         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 Internals() {}
   public void inputCIEmarks()
            {
      Scanner s = new Scanner(System.in);
      System.out.println("Enter Internal Marks for " + name);
      for (int i = 0; i < 5; i++)
      {
         System.out.print("Subject " + (i + 1) + " marks: ");
         marks[i] = s.nextInt();
      }
   }
}

package SEE;

import CIE.Internals;

import java.util.Scanner;

public class Externals extends Internals
{
   protected int marks[];
   protected int finalMarks[];
```

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

    public void inputSEEmarks()
            {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter SEE Marks for " + name);
        for (int i = 0; i < 5; i++)
      {
            System.out.print("Subject " + (i + 1) + " marks: ");
            marks[i] = s.nextInt();
       }
    }

    public void calculateFinalMarks()
            {
        for (int i = 0; i < 5; i++)
        finalMarks[i] = marks[i] / 2 + super.marks[i];
    }

    public void displayFinalMarks()
            {
        displayStudentDetails();
       for (int i = 0; i < 5; i++)
        System.out.println("Subject " + (i + 1) + ": " + finalMarks[i]);
    }
}

import SEE.Externals;
```

```
public class Main
{
	public static void main(String args[])
	{
		 int numOfStudents=2;
		Externals finalMarks[]=new Externals[numOfStudents];

		for(int i=0;i<numOfStudents;i++)
		{
			finalMarks[i]=new Externals();
			finalMarks[i].inputStudentDetails();
			System.out.println("Enter CIE marks:");
			finalMarks[i].inputCIEmarks();
			 System.out.println("Enter SEE marks:");
		  finalMarks[i].inputSEEmarks();
		}

		   System.out.println("DIsplaying data:\n");
		  for(int i=0;i<numOfStudents;i++)
		{
			finalMarks[i].calculateFinalMarks();
			finalMarks[i].displayFinalMarks();
		}
	}
}
```

OUTPUT:

```
C:\Users\nayak\Downloads>javac Main1.java

C:\Users\nayak\Downloads>java Main1
Enter USN: 111
Enter Name: ram
Enter Semester: 3
Enter CIE marks:
Enter Internal Marks for ram
Subject 1 marks: 33
Subject 2 marks: 36
Subject 3 marks: 28
Subject 4 marks: 31
Subject 5 marks: 40
Enter SEE marks:
Enter SEE Marks for ram
Subject 1 marks: 89
Subject 2 marks: 91
Subject 3 marks: 78
Subject 4 marks: 84
Subject 5 marks: 90
Enter USN: 112
Enter Name: raju
Enter Semester: 3
Enter CIE marks:
Enter Internal Marks for raju
Subject 1 marks: 33
Subject 2 marks: 28
Subject 3 marks: 23
Subject 4 marks: 27
Subject 5 marks: 36
Enter SEE marks:
Enter SEE Marks for raju
Subject 1 marks: 84
Subject 2 marks: 85
Subject 3 marks: 99
Subject 4 marks: 92
Subject 5 marks: 77
DIsplaying data:

USN: 111
Name: ram
Semester: 3
Subject 1: 77
Subject 2: 81
Subject 3: 67
Subject 4: 73
Subject 5: 85
USN: 112
Name: raju
Semester: 3
Subject 1: 75
Subject 2: 70
Subject 3: 72
Subject 4: 73
Subject 5: 74
```

//PROGRAM 7

Write a program that demonstrates handling of exceptions in inheritance tree.

Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws

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

```
import java.util.Scanner;

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

class InputScanner
{
 protected Scanner s;
 public InputScanner()
 {
  s = new Scanner(System.in);
 }
}


class Father extends InputScanner
{
 protected int fatherAge;
 public Father() throws WrongAge
 {
  System.out.println("Enter Father's Age:");
  fatherAge=s.nextInt();
```

```
 if(fatherAge<0)
 {
  throw new WrongAge("Age cannot be negetive:");
 }
}

public void display()
{
 System.out.println("Father's Age:" + fatherAge);
}

}

class Son extends Father
{
 private int sonAge;

 public Son() throws WrongAge
 {
  super();
  System.out.println("Enter Son's age:");
  sonAge=s.nextInt();


  if(sonAge>fatherAge)
  {
   throw new WrongAge("Son's age cannot be greater than father's age");
  }
  else if (sonAge<0)
  {
   throw new WrongAge("Age cannot be negative");
```

```
  }
 }

 public void display()
 {
  super.display();
  System.out.println("Son's Age: " + sonAge);
 }

}



public class FatherSonAge
{
 public static void main(String args[])
 {
  try
  {
   Son son=new Son();
   son.display();
  }

  catch (WrongAge e)
  {
   System.out.println("Error: " + e.getMessage());
  }
 }

}
```

OUTPUT:

```
C:\Users\nayak\Downloads>javac FatherSonAge.java

C:\Users\nayak\Downloads>java FatherSonAge
Enter Father's Age:
45
Enter Son's age:
20
Father's Age:45
Son's Age: 20

C:\Users\nayak\Downloads>java FatherSonAge
Enter Father's Age:
56
Enter Son's age:
78
Error: Son's age cannot be greater than father's age
```

//PROGRAM 8

Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displaying "CSE" once every two seconds.

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

```
                    e.printStackTrace();
                }
        }
}
}

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


public class ThreadExample
{
public static void main(String[] args)
{
        BMSThread bmsThread=new BMSThread();
        bmsThread.start();
```

```
        CSEThread cseThread=new CSEThread();
        cseThread.start();
}
}
```

OUTPUT:

```
C:\Users\nayak\Downloads>javac ThreadExample.java

C:\Users\nayak\Downloads>java ThreadExample
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
```

//PROGRAM 9

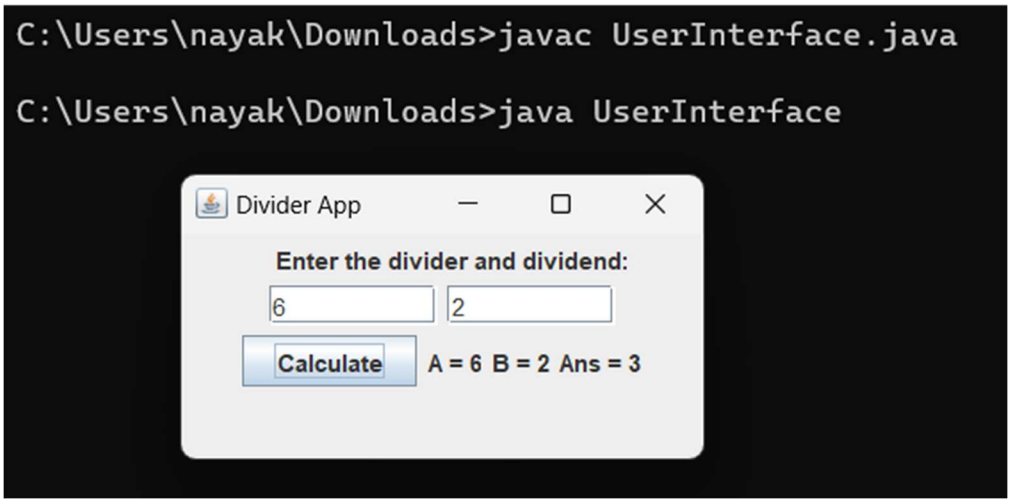
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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

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

class UserInterface {
   UserInterface() {
      // create JFrame container
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 150);
      jfrm.setLayout(new FlowLayout());
      // to terminate on close
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // text label
      JLabel jlab = new JLabel("Enter the divider and dividend:");

      // add text field for both numbers
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);

      // calc button
      JButton button = new JButton("Calculate");

      // labels
      JLabel err = new JLabel();
      JLabel alab = new JLabel();
      JLabel blab = new JLabel();
      JLabel anslab = new JLabel();

      // add in order
      jfrm.add(err);  // to display error message
      jfrm.add(jlab);
```

```
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

ActionListener calculateListener = new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        try {
            int a = Integer.parseInt(ajtf.getText());
            int b = Integer.parseInt(bjtf.getText());
            if (b == 0) {
                throw new ArithmeticException();
            }
            int ans = a / b;

            alab.setText("\nA = " + a);
            blab.setText("\nB = " + b);
            anslab.setText("\nAns = " + ans);
            err.setText(""); // Clear any previous error message
        } catch (NumberFormatException e) {
            displayErrorMessage("Enter Only Integers!");
        } catch (ArithmeticException e) {
            displayErrorMessage("B should be non-zero!");
        }
    }

    private void displayErrorMessage(String message) {
        alab.setText("");
        blab.setText("");
        anslab.setText("");
```

```
            err.setText(message);
        }
    };

    button.addActionListener(calculateListener);

    // display frame
    jfrm.setVisible(true);
  }

  public static void main(String args[]) {
    // create frame on event dispatching thread
    SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        new UserInterface();
      }
    });
  }
}
```

OUTPUT:



//PROGRAM 10 (Part 1)

Demonstrate Inter process Communication

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

```
        }
        catch(InterruptedException e)
        {
                System.out.println("InterruptedException caught");
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\nIntimate Consumer\n");
        notify();
}
}


class Producer implements Runnable
{
Q q;
Producer(Q q)
{
        this.q = q;
        new Thread(this, "Producer").start();
}
public void run()
{
        int i = 0;
        while(i<5)
        {
                q.put(i++);
        }
}
}
```

```
class Consumer implements Runnable
{
Q q;
Consumer(Q q)
{
	this.q = q;
	new Thread(this, "Consumer").start();
}
public void run()
{
	int i=0;
	while(i<5)
	{
		int r=q.get();
		System.out.println("consumed:"+r);
		i++;
	}
}
}


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

OUTPUT:

```
C:\Users\nayak\Downloads>javac PCFixed.java

C:\Users\nayak\Downloads>java PCFixed
Press Control-C to stop.
Put: 0

Intimate Consumer


Producer waiting

Got: 0

Intimate Producer

consumed:0
Put: 1

Intimate Consumer


Producer waiting

Got: 1

Intimate Producer

consumed:1
Put: 2

Intimate Consumer


Producer waiting

Got: 2

Intimate Producer

consumed:2
Put: 3

Intimate Consumer


Producer waiting

Got: 3

Intimate Producer

consumed:3
Put: 4

Intimate Consumer

Got: 4

Intimate Producer

consumed:4
```

//PROGRAM 10 (Part 2)

Demonstrate deadlock

```
class A
{
synchronized void foo(B b)
{
        String name =Thread.currentThread().getName();
        System.out.println(name + " entered A.foo");
        try
        {
                Thread.sleep(1000);
        }
        catch(Exception e)
        {
                System.out.println("A Interrupted");
        }

        System.out.println(name + " trying to call B.last()");
        b.last();

}

void last()
{
        System.out.println("Inside A.last");
}
}

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

```
        try
        {
                Thread.sleep(1000);
        }
        catch(Exception e)
        {
                System.out.println("B Interrupted");
        }

        System.out.println(name + " trying to call A.last()");
        a.last();
}

void last()
{
        System.out.println("Inside A.last");
}
}

class Deadlock implements Runnable
{
A a = new A();
B b = new B();
Deadlock()
{
        Thread.currentThread().setName("MainThread");
        Thread t = new Thread(this,"RacingThread");
        t.start();
        a.foo(b); // get lock on a in this thread.
        System.out.println("Back in main thread");
}
```

```
public void run()
{
        b.bar(a); // get lock on b in other thread.
        System.out.println("Back in other thread");
}

public static void main(String args[])
{
        new Deadlock();
}

}
```

OUTPUT: