

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:

Sakshi B R

1BM22CS233

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

College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019

2023-2024.

Lab Program

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

```
import java.util.Scanner;  
class Quadratic  
{
```

```
    int a, b, c;  
    double x1, x2, 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("Root 1 = Root 2 = " + r1);
}

```

```

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 real and distinct");

```

```

    System.out.println("Root 1 = " + r1 + " Root 2 = " + r2);
}

```

```

} else if (d < 0)
{
    class

```

```

        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 2 = " + r1 + " - i" + r2);
}
}
}

```

```

class QuadraticMain
{

```

```

    public static void main(String args[])
    {

```

```

        Quadratic q = new Quadratic();
    }
}

```


q.getd();
q.compute();

q

Output

→ Enter the coefficients of a, b, c

1

2

B1

Roots are ~~imaginary~~ real and equal

Roots 1 = Root 2 = -1.0

→ Enter the coefficients of a, b, c

1

2

Roots are real and equal

Root 1 = Root 2 = -1.0

Sakshi B.R

IBM22CS233

Lab 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.

SGPA

$$SGPA = \frac{\sum [(Course Credits)(Grade Points)]}{\sum [Course Credits]}$$

CGPA

```
import java.util.Scanner;  
class Subject  
{
```

```
    int subjectMarks;  
    int credits;  
    int grade;  
}
```

```
class Student  
{
```

```
    Subject subject[];  
    String name;  
    String usn;  
    Scanner s;  
    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.print("Enter your Name:");  
name = s.next();
```

```
System.out.print("Enter your USN:");
```

```
usn = s.next();
```

```
}
```

```
void getMarks()
```

```
{
```

```
for (int i=0; i<9; i++)
```

```
System.out.print("Enter marks for subject"  
+ (i+1) + ":");
```

```
subject[i].subjectMarks = s.nextInt();
```

```
System.out.print("Enter credits for  
subject " + (i+1) + ":");
```

```
subject[i].credits = s.nextInt();
```

```
subject[i].grade = (subject[i].subjectMarks  
/ 10) + 1;
```

```
if (subject[i].grade == 11)
```

```
subject[i].grade = 10;
```

```
if (subject[i].grade <= 4)
```

```
subject[i].grade = 0;
```

```
}
```

```
}
```

```
void computeSGPA()
```

```
{
```

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

```
{
```

```
St
```

```
int effectiveScore = 0;
```

```
int totalCredits = 0;
```



```

for(int i=0; i<9; i++)
    effectiveScore += (subject[i].grade *
                      subject[i].credits);
totalCredits += subject[i].credits;

SGPA = (double) effectiveScore / (double)
        totalCredits;

```

```

}
class Main
{

```

```

    public static void main(String args[])
    {
        Student s1 = new Student();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();
        System.out.println("Name:" + s1.name);
        System.out.println("USN:" + s1.usn);
        System.out.println("SGPA:" + s1.SGPA);
    }
}

```

Output:

Enter your Name: Sakshi
 Enter your USN: IBM22C.S233
 Enter marks for subject 1: 80
 Enter credits for subject 1: 5
 Enter marks for subject 2: 70
 Enter credits for subject 2: 6
 Enter marks for subject 3: 50
 Enter credits for subject 3: 5

Enter marks for subject 4: 90
Enter credits for subject 4: 9
Enter marks for subject 5: 83
Enter credits for subject 5: 7
Enter marks for subject 6: 65
Enter credits for subject 6: 5
Enter marks for subject 7: 70
Enter credits for subject 7: 6
Enter marks for subject 8: 80
Enter credits for subject 8: 8
Enter marks for subject 9: 50
Enter credits for subject 9: 4
Name: Sakshi

USN: ~~Sakshi~~ IBM22CS233

SGPA: 8.2727

Name: Sakshi B R

USN: IBM22CS233

14/12

Lab 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 books objects.

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

```
class Books {
```

```
    String name, author;
```

```
    int price, 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 = "Books name is: " + this.name +  
              "\n";
```

```
        price = "Price :
```

```
        author = "Author name : " + this.author +  
                "\n";
```



```

price = "Price : " + this.price + "\n";
numPages = "Number of pages : " + this.numPages + "\n";
return name + author + price + numPages;
}
}

```

```

class BookMain {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n;
        String name, author;
        int price, numPages;
        System.out.println("Enter the number of books");
        n = sc.nextInt();
        Books b[] = new Books[n];
        System.out.println("Enter name, author, price and number of pages");
        for (int i = 0; i < n; i++) {
            name = sc.next();
            author = sc.next();
            price = sc.nextInt();
            numPages = sc.nextInt();
            b[i] = new Books(name, author, price, numPages);
        }
        System.out.println("Book details:");
        for (int i = 0; i < n; i++) {
            System.out.println(b[i].toString());
        }
    }
}

```


Output

Enter the number of books:

2

Enter Name, author, price and number of pages:

ABC murder

Agatha

300

400

Panchatantra

Vishnu

200

300

Book details:

Book name is: ABC murder

Author name: Agatha

Price: 300

Number of pages: 400

Book name: Panchatantra

Author name: Vishnu

Price: 200

Number of pages: 300

Sakshi B.R

IBM0000033

Lab Program 4

```
import java.util.*;

class InputScanner {
    Scanner sc;
    InputScanner() {
        sc = new Scanner(System.in);
    }
}

abstract class Shape extends InputScanner {
    double a;
    double b;
    abstract void getInput();
    abstract void displayArea();
}

class Rectangle extends Shape {
    void getInput() {
        System.out.println("Enter the length  
and breadth:");
        a = sc.nextDouble();
        b = sc.nextDouble();
    }
    void displayArea() {
        System.out.println("Area of  
rectangle is: " + (a * b));
    }
}

class Triangle extends Shape {
    void getInput() {
        System.out.println("Enter the length  
and height");
    }
}
```



```
a = sc.nextDouble();  
b = sc.nextDouble();
```

```
}
```

```
void displayArea() {  
    System.out.println("Area of  
    rectangle is: " + (a * b * 0.5));  
}
```

```
}
```

```
}
```

```
class Circle extends Shape {
```

```
    void getInput() {
```

```
        System.out.println("Enter the  
        radius:");
```

```
        a = sc.nextDouble();
```

```
}
```

```
void displayArea() {
```

```
    System.out.println("Area of  
    rectangle is: " + (a * a * 3.14));  
}
```

```
}
```

```
}
```

```
class ShapeMain {
```

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

```
        Rectangle r = new Rectangle();
```

```
        Triangle t = new Triangle();
```

```
        Circle c = new Circle();
```

```
        r.getInput();
```

```
        r.displayArea();
```

```
        t.getInput();
```

```
        t.displayArea();
```

```
        c.getInput();
```

```
        c.displayArea();  
    }
```

```
}
```

```
}
```


Output

Enter the length and breadth:

80

40

Area of rectangle is : 3200.0

Enter the length and height:

40

20

Area of rectangle is : 400.0

Enter the radius:

15

Area of rectangle is : 706.5

Sakshi B.R

IBM22CS233

~~Was~~
7-1-2024

Lab 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 Acc 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:

- Accept deposit from customer and update the balance
- Display the balance
- Compute and deposit interest
- Permit withdrawal and update 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(i)
{
    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  
initial balance:");
        double balance = s.nextDouble();
        int cli;
        double amount 1, amount 2;
        account acc = new account (name,  
accno, type, balance);
    }
}

```



```

savingsAcc sa = new savingsAcc(name,
                                accno, balance);
(currAcc ca = new currAcc(name,
                             accno, balance);
while(true)
{
    if (accType.equals("savings"))
    {
        System.out.println("1. Menu\n1. deposit 2. withdraw 3. compute\ninterest 4. display");
        System.out.println("enter the\nchoice:");
        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: System.exit(0);
            case 5: sa.display();
                    break;
            case 6: System.exit(0);
        }
    }
}

```


default: System.out.println
("Invalid input");
break;

}

else
{

System.out.println("Menu | 1. 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

Enter the name, type (current/savings),
account number, initial balance,

Sakshi

savings

321

60000

Menu

1. Deposit 2. Withdraw 3. Compute interest
4. Display

Enter the choice:

1

Enter the amount:

5000

Menu

1. Deposit 2. Withdraw 3. Compute interest
4. Display

2

Enter the amount:

500

Menu

1. Deposit 2. Withdraw 3. Compute
interest 4. Display

3.

Balance: \$7006.00

Menu

1. Deposit 2. Withdraw 3. Compute interest
4. Display

Enter the choice

4.

Name: Sakshi

Acc no: 3218

Type: Savings

Balance: 59006.0

Enter the name, type (current/saving),
acc no, initial balance

Sakshi

Current

101

50000

Menu

1. Deposit 2. Withdraw 3. display

Enter the choice

1

Enter the amount

5000

Menu

1. Deposit 2. Withdraw 3. display

Enter the choice

2

Enter the amount

500

Menu

1. Deposit 2. Withdraw 3. display

Enter the choice

3

Name : Sakshi

Acc no: 101

Type : current

Balance : 504500.0

Sakshi-B.P

1BM22CS233

8
16/11/2024

Lab - 6

Create a package CTE which has two classes - Student and Internal. The class Student has members like usn, name, sem. The class Internal derived from Student 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.

```
//Student.java
package CTE;
import java.util.*;
public class student
{
    protected string usn = new string();
    protected string name = new string();
    protected int

    public void inputStudentDetails()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter student usn");
        usn = sc.next();
        System.out.println("Enter student name");
    }
}
```



```
name = sc.next();  
system.out.println("Enter semester");  
sem = sc.nextInt();
```

```
}  
public void displayStudentDetails()  
{  
    System.out.println("Student user: " + user);  
    System.out.println("Student name: " + name);  
    System.out.println("Student semester: " + sem);  
}
```

```
// Internals.java
```

```
package CTE;
```

```
import java.util.*;
```

```
public class Internals extends Student
```

```
{  
    protected int marks[] = new int[5];
```

```
    public void inputCTE marks()  
    {
```

```
        Scanner sc = new Scanner(System.in);
```

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

```
        {  
            System.out.println("Enter 5 subject  
            marks");
```

```
            marks[i] = sc.nextInt();  
        }  
    }
```



```
11 External java  
package SEE;  
import CIE Internals;  
import java.util.Scanner;  
public class External extends Internals  
{
```

```
    protected int marks[];  
    protected int finalmarks[];  
    public External()  
{
```

```
        marks = new int[5];  
        finalmarks = new int[5];  
    }
```

```
    public void inputSEEmark()  
{
```

```
        Scanner sc = new Scanner(System.in);  
        for(int i=0; i<5; i++)  
{
```

```
            System.out.println("Subject" + (i+1) + "  
            marks:");
```

```
            marks[i] = sc.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 para
import SEE.Externals;
class main {
    public static void main(String args[])
    {
        int num of students = 2;
        Externals finalMarks[] = new Externals[num of students];
        for(int i=0; i<num of students; i++)
        {
            finalMarks[i] = new Externals();
            finalMarks[i].inputStudentDetails();
            System.out.println("Enter I/E marks");
            finalMarks[i] = inputI/E marks();
            System.out.println("Enter SEE marks");
            finalMarks[i] = inputSEE marks();
        }
        System.out.println("Displaying data: /n");
        for(int i=0; i<num of students; i++)
        {
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
}

```


Output

Enter student usn

IBM22CS233

Enter student name

Sakshi

Enter student semester

3

Enter CIE marks

Enter 5 marks : 39

Enter 5 marks : 38

Enter 5 marks : 26

Enter 5 marks : 40

Enter 5 marks : 36

Enter SEE marks

Subject 1 marks : 78

Subject 2 marks : 89

Subject 3 marks : 90

Subject 4 marks : 87

Subject 5 marks : 92

Enter student usn

IBM22CS237

Enter student name

Rani

Enter student semester

3

Enter CIE marks

Enter 5 marks : 39

Enter 5 marks : 38

Enter 5 marks : 30

Enter 5 marks : 30

Enter 5 marks : 25

Enter SEE marks

Subject 1 marks : 56

Subject 2 marks : 78

Subject 3 marks : 89

Subject 4 marks : 90

Subject 5 marks : 92

Displaying data

Student name :

Student ID :

Student semester :

Subject 1 :

Subject 2 :

Subject 3 :

Subject 4 :

Subject 5 :

Student name :

Student ID :

Student semester :

Subject 1 :

Subject 2 :

Subject 3 :

Subject 4 :

Subject 5 :

Sakshi B.R

1BM22CS233

Lab - 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 takes both father and son's age and throws an exception if son's age is \geq 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 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  
    age");
```

```
}
```

```
else if (sonAge < 0)
```

```
{
```

```
    throw new WrongAge("son's  
    Age: " + sonAge);
```

```
}
```

```
}
```

```
public class Main
```

```
{
```

```
    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:

25

Enter son's age:

50

Error: Son's age cannot be greater than father's age.

Sakshi B.R

IBM02CS233

Lab - 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 DisplayMessageThread extends Thread {
    private String message;
    private int interval;
    public DisplayMessageThread(String message,
        int interval) {
        this.message = message;
        this.interval = interval;
    }

    public void run() {
        while (true) {
            System.out.println(message);
            try {
                Thread.sleep(interval * 1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```
public class ThreadExample {
    public static void main(String[] args) {
        DisplayMessageThread thread1 =
```


new DisplayThread ("BMS College of
Engineers")

DisplayMessage Thread thread 2 = new
DisplayMessage Thread ("CSE", 2);
thread 1.start();
thread 2.start();

3
4

Output

BMS College of Engineering

CSE

CSE

CSE

CSE

CSE

BMS College of Engineering

CSE

CSE

CSE

CSE

CSE

BMS College of Engineering

Thread 1 Interrupted

Thread 2 Interrupted

Main thread existing

Sakshi B.R

1 BMS CS 233

6-2-24

20/2/24

Lab 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 an Arithmetic Exception. Display the exception in a message dialog box.

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

```
public class DivisionMain extends Frame
implements ActionListener
{
```

```
    TextField num1, num2;
    Button dResult;
    Label outResult;
    String out = " ";
    double resultNum;
    int flag = 0;
```

```
    public DivisionMain()
    {
```

```
        setLayout(new FlowLayout());
```

```
        dResult = new Button("RESULT");
```

```
        Label number1 = new Label("Number 1:", Label.RIGHT);
```



```
Label number2 = new Label("Number  
2:", Label.RIGHT);  
num1 = new TextField(5);  
num2 = new TextField(5);  
outResult = new Label("Result:",  
Label.RIGHT);
```

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

```
num1.addActionListener(this);  
num2.addActionListener(this);  
dResult.addActionListener(this);  
addWindowListener(new Window-  
Adapter()
```

```
{
```

```
public void windowClosing(Window-  
Event we)
```

```
{
```

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

```
}
```

```
public void void actionPerformed(ActionEvent ae)
```

```
{
```

```
int n1, n2;
```

```
try
```

```
{
```

```
if(ae.getSource() == dResult)
```


5

```
n1 = Integer.parseInt(num1.getText());
n2 = Integer.parseInt(num2.getText());
```

```
/* if (n2 == 0)
    throw new ArithmeticException(); */
```

```
out = n1 + " / " + n2;
resultNum = n1 / n2;
out += String.valueOf(resultNum);
repaint();
```

```
}
catch (NumberFormatException e1)
```

```
flag = 1;
out = "Number Format Exception! " + e1;
repaint();
```

```
}
catch (ArithmeticException e2)
```

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

```
}
}
public void paint(Graphics g)
```

```
{
    if (flag == 0)
        g.drawString(out, outResult.getX() +
            outResult.getWidth(), outResult.getY());
}
```



```
+ outResult.getHeight() - 2);  
else  
g.drawString(out, 100, 200);  
flag = 0;
```

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

Output

Number 1: Number 2:
Result 6 41.0 ~~Result 64~~

Sakshi B.R

IBM22CS233.

20/11/2024

13/2/24

Lab Program 10

Implementation of producer and consumer

```
class Q {
```

```
    int n;
```

```
    boolean valueSet = false;
```

```
    Synchronised int get() {
```

```
        while (!valueSet)
```

```
        try {
```

```
            System.out.println("\n Consumer waiting\n");
```

```
            wait();
```

```
        } catch (InterruptedException e) {
```

```
            System.out.println("InterruptedException caught");
```

```
        }
```

```
        System.out.println("Got: " + n);
```

```
        valueSet = false;
```

```
        System.out.println("\n Intimate\n Producers\n");
```

```
        notify();
```

```
        return n;
```

```
    }
```

```
    synchronised void put(int n) {
```

```
        while (valueSet)
```

```
        try {
```

```
            System.out.println("\n Producer\n waiting\n");
```

```
            wait();
```

```
        } catch (InterruptedException e) {
```

```
            System.out.println("InterruptedException caught");
```

```
        }
```



```

this.n = n;
valueSet = true;
System.out.println("Put: " + n);
System.out.println("Intimate consumer");
notify();
}
}

```

class Producer implements Runnable {

@ q;

Producer(@ q) {

this.q = q;

new Thread(this, "Producer").start();

{

public void run() {

int i = 0;

while (i < 15) {

q.put(i++);

}

}

}

class Consumer implements Runnable {

@ q;

Consumer(@ q) {

this.q = q;

new Thread(this, "Consumer").start();

{

public void run() {

int i = 0;

while (i < 15) {

int r = q.get();

System.out.println("consumed: " + r);

}

9. "Practical" (1992)

2. (g) $\frac{1}{2}$ minutes

152

Deadlocks

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.bar()");

b.bar();

}

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.bar()");


```
a.last();
```

```
}
```

```
void last() {
```

```
    System.out.println("Inside A.last()");
```

```
}
```

```
}
```

```
class Deadlock implements Runnable {
```

```
    Aa = new A();
```

```
    Bb = new B();
```

```
    Deadlock() {
```

```
        Thread.currentThread().setName("Main Thread");
```

```
        Thread t = new Thread(this, "Racing Thread");
```

```
        t.start();
```

```
        a.foo(b);
```

```
        System.out.println("Back in main thread");
```

```
}
```

```
public void run() {
```

```
    b.bar(a);
```

```
    System.out.println("Back in other thread");
```

```
}
```

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

```
    new Deadlock();
```

```
}
```

```
}
```


o/p MainThread entered A.foo
 Racing Thread entered B.bar
 MainThread trying to call B.bar()
~~Inside A.bar~~
 Back in main thread
 Racing Thread trying to call A.bar()
 Inside A.bar
 Back in other thread

Sakshi . B . R

1 BM22CS 233 .

Ans
 15-2-24

LAB PROGRAM 1:

```
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("&quot;Enter the coefficients of a,b,c&quot;");
        a = s.nextInt();
        b = s.nextInt();
        c = s.nextInt();
    }
    void compute()
    {
        while(a==0)
        {
            System.out.println("&quot;Not a quadratic equation&quot;");
            System.out.println("&quot;Enter a non zero value for a:&quot;");
            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("&quot;Roots are real and equal&quot;");
            System.out.println("&quot;Roo1 = Root2 = &quot; + r1);
        }
    }
}
```



```

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 real and distinct"");
System.out.println(""Root1 = " + r1 + " Root2 = " + r2);
}
else if(d<0)
{
System.out.println(""Roots are imaginary"");
r1 = (-b)/(2*a);
r2 = Math.sqrt(-d)/(2*a);
System.out.println(""Root1 = " + r1 + " + i"+r2);
System.out.println(""Root1 = " + r1 + " - i"+r2);
}

}

}

class QuadraticMain
{
public static void main(String args[])
{
Quadratic q = new Quadratic();
q.getd();

q.compute();
}
}

```


LAB PROGRAM 2:

```
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();

    }

}

```

LAB PROGRAM 3:

```
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 Mainbook
{
    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]);
}
```



```
        }  
    }  
}
```

LAB PROGRAM 4:

```
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);
        System.out.println();
    }
}

```

```

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

```



```
{  
  
    super();  
  
}  
  
public void printArea()  
{  
  
    a=getInput("Enter the side 1: ");  
    b=getInput("Enter the side 2: ");  
    double area=0.5*a*b;  
    System.out.println("Area of the Triangle: " +area);  
    System.out.println();  
}  
  
}
```

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);  
        System.out.println();  
    }  
}
```



```
}
```

```
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();
```

```
    }
```

```
}
```

LAB PROGRAM 5:

```
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);

```



```

Scanner sc=new scanner(system.in);

System.out.println("Enter student USN");

usn=sc.next();

System.out.println("Enter student name");

name = sc.next();

System.out.println("Enter semester");

Scanner=sc.nextInt();

}

public void display StudentDetails()

{

    System.out.println("Student usn:"+usn);

    System.out.println("Student name:"+name);

    System.out.println("Student semester:"+sum);

}

}

//Internals.java

package CIE;

import java.util.*

public class Internals extends student

{

    protected int marks[]=new int[5];

    public void input CIE marks()

    {

        scanner sc=new scanner(system.in);

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

        {

            System.out.println("Enter 5 subject marks");

            marks[i]=sc.nextInt();

        }

    }

}

}

```

```

//External.java

package SEE;

import CIE.Internals;

import java.util.Scanner;

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

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

    public void input SEE marks()
    {
        Scanner sc=new scanner(System.in);
        for(int i=0;i<5;i++)
        {
            System.out.println("Subject"+(i+1)+"marks:");
            marks[i]=sc.nextInt();
        }
    }

    public void Calculate Final Marks[]
    {
        for(int i=0;i<5;i++)
        {
            final marks[i]=Marks[i]/2+super.marks[i];
        }
    }

    public void display FinalMarks()
    {

```



```

        display StudentDetails();
    }
    for(int i=0;i<5;i++)
    {
        System.out.println("Subject" +(i+1) + "final Marks[i]");
    }
}
}
//Main.java
import SEE.Externals
class Main
{
    public static void main(string args[])
    {
        int num of students=2;
        Externals final Marks[]=new Externals[num of students];
        for(int i=0;i<num of students;i++)
        {
            final marks[i]=newExternals();
            final marks[i].input student Details();
            System.out.println("Enter CIE marks");
            final marks[i].input CIE marks();
            System.out.println("Enter SEE marks");
            final marks[i].input SEE marks();
        }
        System.out.println("Displaying data:/n");
        for(int i=0;i<num of students;i++)
        {
            final Marks[i].Calculate FinalMarks();
            final Marks[i].display FinalMarks();
        }
    }
}

```

```
}
```

LAB PROGRAM 7:

```
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());
        }
    }
}

```

LAB PROGRAM 8:

```

class BMSThread extends Thread {

```



```

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

```

```

public class threadEx {
    public static void main(String[] args) {
        BMSThread bms = new BMSThread();
        bms.start();
    }
}

```

```

        CSEThread cse = new CSEThread();
        cse.start();
    }
}

```

LAB PROGRAM 9:

```

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

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

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

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

        add(number1);
        add(num1);

```



```

        add(number2);

        add(num2);

        add(dResult);

        add(outResult);


        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent we)
            {
                System.exit(0);
            }
        });
    }


    public void actionPerformed(ActionEvent ae)
    {
        int n1,n2;
        try
        {
            if (ae.getSource() == dResult)
            {
                n1=Integer.parseInt(num1.getText());
                n2=Integer.parseInt(num2.getText());

                /*if(n2==0)
                    throw new ArithmeticException();*/

                out=n1+" "+n2;
                resultNum=n1/n2;
            }
        }
        catch (Exception e)
        {
            out="Error!";
        }
    }
}

```

```

        out+=String.valueOf(resultNum);

        repaint();

    }

}

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

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

}

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

        g.drawString(out,outResult.getX()+outResult.getWidth(),outResult.getY()+outResult.getHeig
ht()-8);

    else
        g.drawString(out,100,200);
    flag=0;
}

public static void main(String[] args)

```



```

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

}

```

LAB PROGRAM 10:

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<2) {
            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.");
    }
}

```

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();
}
}
```