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:

PRADEEP P T

(1BM22CS197)

Department of Computer Science and Engineering,

B.M.S College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019

2023-2024.

INDEX

Lab Prog ①: Quadratic Equation

```
import java.util.Scanner;
class Quadratic {
int a, b, c;
double r1, r2, d;
void getd()
{
Scanner s = new Scanner (System.in);
System.out.println ("Enter the coefficients
of a, b, 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);
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("Root 2 = "+r1+" - i"+r2);
}
}
}
class QuadraticMain
{
public static void main(String args[])
{
Quadratic q = new Quadratic();
q.getd();
q.compute();
}
}
```

Output:

i) Enter the coefficients of a, b, c :
1 -3 2
Roots are real and equal. distinct
Root 1 = 2 Root 2 = +1

ii) Enter the coefficients of a, b, c
0 2 3
Not a quadratic equation
Enter a non zero value of a

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

LAB PROG-(2) : Develop a Java Program to display the name, USN and input the marks, credits from the user calculate the SGPA and display with the name and usn

```
import java.util.Scanner;
class Subject
{
int marks;
int credits;
int grade;
}
class student
{
String name;
string usn;
double SGPA;
Subject[] subject;
Scanner s;
student()
{
int i;
subject = new Subject[8];
```

```
for(i=0;i<8;i++)
subject[i]= new subject();
s= new Scanner(System.in);
}
void getStudentDetails()
{
System.out.println("Enter the name of student");
name = s.nextline();
System.out.println("Enter the USN");
usn = s.nextline();
}
void getMarks()
{
System.out.println("Enter the marks and credits of each subject");
for(int i=0;i<8;i++)
{
subject[i].marks = s.nextInt();
subject[i].credits = s.nextInt();
if(subject[i].marks >=90)
  subject[i].grade = 10;
else if(subject[i].marks >=80 && subject[i].marks <90)
 subject[i].grade =9;
else if(subject[i].marks >=70 && subject[i].marks <80)
 subject[i].grade =8;
else if(subject[i].marks >=60 && subject[i].marks <70)
 subject[i].grade =7;
else if(subject[i].marks >=50 && subject[i].marks <60)
```

```
subject[i].grade = 6;
else if (subject[i].marks >= 40 && subject[i].marks < 50)
 subject[i].grade = 5;
else
{
System.out.println("Invalid marks");
break;
}
}
}
}
void computeSGPA()
{
double creditsgained = 0;
double totalcredits = 0;
for (int i=0; i<8; i++)
{
totalcredits += subject[i].credits;
creditsgained += subject[i].grade *
subject[i].credits;
}
SGPA = creditsgained / totalcredits;
}
void display()
{
System.out.println("Name of the
student = " + name);
System.out.println("USN = " + usn);
System.out.println("SGPA = " + SGPA);
}
}
```

```
public class Main {
public static void main (String args[])
{
Student S1 = new student ();
S1. getStudentDetails ();
S1. getMarks ();
S1. computeSGPA;
S1. display ();
}
}
```

Output:

```
Enter the name of student
Pradeep
Enter the USN
1BM22CS197
Enter the marks and credits of each
subject
95
4
90
4
89
3
96
3
94
3
89
1
98
1
93
1
```

Name of the student = Pradeep
USN = 1BM22CS197
SGPA = 9.8.

19/12/23

LAB PROG-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 object. Include a toString() method that could display the complete details of the book. Develop a Java program to create n book objects.

```
import java.util.Scanner
class Book
{
String name, author;
int price;
int num_pages;

Book(string name, String author, int price, int num_pages) {
  this.name = name;
  this.author = author;
  this.price = price;
  this.num_pages = num_pages;
}
public String toString() {
  String n, a, p, N;
n = "\n" + "Name of Book: " + name + "\n";
a = "Author of Book: " + author + "\n";
p = "Price of Book: " + price + "\n";
N = "Number of pages: " + num_pages +
"\n";
Return n + a + p + N;
}
}
```

```
class Books
 {
 public static void main (String args[])
 {
  Scanner sc = new Scanner (System.in);
 System.out.println ("\n Enter number of
 books: ");
 int n = sc.nextInt ();
 Book b[] = new Book[n];
 String name, author;
 int price, num;
 sc.nextLine ();
 for (int i=0; i<n; i++) {
 System.out.println ("Enter name of book:");
 name = sc.nextLine ();
 System.out.println ("Enter author's name:");
 author = sc.nextLine ();
 System.out.println ("Enter price: ");
 price = sc.nextInt ();
 system.out.println( "Enter number of
 pages: ");
 num = sc.nextInt ();
 b[i] = new Book (name, author, price, num)
 }
 System.out.println ("Books details: ");
 for (int i=0; i<n; i++) {
 System.out.println (b[i].toString ());
  }
 }
 }
```

Output:

```
Enter numbers of books:
2
Enter name of book:
Invincible
Enter author's name:
Jack
Enter price:
1500
Enter numbers of pages:
120
Enter name of book:
Raise
Enter author's name:
Tim
Enter price:
1800
Enter number of pages:
250

Book Details:
Name of Book: Invincible
Author of Book: Jack
Price of Book: 1500
Number of pages: 120

Name of Book: Raise
Author of Book: Tim
Price of Book: 1800
Enter number of pages: 250
```

25

28/12/23

02/01/24 LAB 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
abstract class shape{
double dim1, dim2;
shape (double a, double b) {
dim1 = a;
dim2 = b;
}
abstract void printArea();
}
class Rectangle extends shape{
Rectangle (double a, double b) {
super (a, b);
}
void printArea() {
System.out.println ("area of rectangle is" + (dim1 * dim2));
}
}
class Triangle extends shape {
Triangle (double a, double b) {
super (a, b);
}
```

```
void printArea() {
System.out.println("Area of Triangle
is" + (dim1 * dim2) /2);
}
}
class Circle extends Shape {
Circle (double a, double b) {
super(a, b);
}
final double pi = 3.14159;
void printArea() {
System.out.println("Area of Circle is"
+ pi * Math.pow(dim1, 2));
}
}
class Main {
 public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
double a, b;
System.out.println("Enter sides of
rectangle:");
a = sc.nextDouble();
b = sc.nextDouble();
Rectangle r = new Rectangle(a, b);
System.out.println("Enter height and
base of triangle:");
a = sc.nextDouble();
b = sc.nextDouble();
Triangle t = new Triangle(a, b);
```

```
System.out.println("Enter radius of
Circle:");
a = sc.nextDouble();
Circle c = new Circle(a,1);
Shape s;
S = r;
s.printArea();
S = t;
S.printArea();
s = c;
S.printArea();
}
}
```

Output:

```
Enter sides of rectangle:
10
20
Enter height and base of triangle:
30
40
Enter radius of circle:
6
Area of Rectangle is 200.0
Area of Triangle is 600.0
Area of Circle is 113.09724
```

02/01/24

09/01/24 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.

```
import java.util.Scanner;

class Account {
    String customerName;
    int accnumber;
    string acctype;
    double balance;

Account (string name; int accno,
    string actype) {
customerName = name;
accnumber = accno;
acctype = actype;
balance = 0;
}
void deposit(double amount) {
    balance += amount;
```

```
System.out.println("Deposit of Rs"+
amount+" successful. New balance: Rs"+
balance);
}
void display(){
System.out.println("Account balance for
"+customerName+": Rs"+balance);
}
}
class CurrAcct extends Account{
  double minbal;
  double serchar;

CurrAcct(String name, int accno, double
  minbal){
  super(name, accno, "current");
  minimumBalance = minbal;
  serchar = 10.0;
}
void checkminbal(){
 if(balance < minbal){
    balance -= serchar;
System.out.println("Service charge of Rs"+
serchar+" applied. New balance: Rs"+
balance);
}
}
void withdraw(double amount){
 if(amount <= balance){
 balance -= amount;
System.out.println("Withdrawal of
  Rs"+amount+" successful. New
```

```
balance : Rs" + balance);
}
else {
System.out.println("Insufficient funds. Withdrawal failed");
}
}
}
class SavAcct extends Account {
double interestRate;
SavAcct(String name, int accno, double interestRate) {
super(name, accno, "Savings");
this.interestRate = interestRate;
}
void computeInterest() {
double interest = balance * (interestRate/100);
balance += interest;
System.out.println("Interest of Rs" + interest + "applied. New balance : Rs" + balance);
}

void withdraw(double amount) {
if (amount <= balance) {
balance -= amount;
System.out.println("Withdrawal of Rs" + amount + "successful. New balance : Rs" + balance);
}
else
{
```

```
System.out.println ( " Insufficient funds.
  Withdrawal failed");
 }
}
}
public class Bank {
public static void main (String [] args)
{
Scanner scanner = new Scanner (System.in);
System.out.println ("Enter your name :");
String customerName = scanner.nextLine();
System.out.print ("Enter your account
number: ");
int accountNumber = scanner.nextInt();
CurrAcct currentAccount = new CurrAcct
( customer.name, accno, 1000.0 );
SavACct savingsAccount = new SavACct
savingsAccount = new savAcc (customer
Name, accno, 5.0);

int choice;
do {
System.out.println ("\nMenu:\n1. Deposit
\n2. Withdraw \n3. Compute interest for
savings account \n4. Display Account
details \n5. Exit");
System.out.println ("Enter your choice");
choice = scanner.nextInt ();

switch (choice) {
case 1: System.out.print ("Enter deppsit
  : Rs");
```

```
double depositAmount = scanner.nextDouble();
currentAccount.deposit(depositAmount);
break;

case 2: System.out.print("Enter withdrawal amount: Rs");
double withdrawalAmount = scanner.nextDouble();
currentAccount.withdraw(withdrawAmount);
break;
case 3: savingsAccount.computeInterest();
break;
case 4: currentAccount.displayBalance();
break;
case 5: System.out.println("Exiting program");
break;
default: System.out.println("Invalid choice");
}
} while (choice != 5);
}
}
```

Output:

```
Enter customer name: Rahul
Enter account number: 78
Enter customer name: Rohan
Enter account number: 18

--MENU--
1. Deposit
```

2. Withdraw
3. Compute interest for SavingsAccount
4. Display account details
5. Exit

Enter your choice : 1
Enter the type of account : saving
Enter the deposit amount : 1000

--Menu--

1. Deposit
2. Withdraw
3. Compute interest for savingsAccount
4. Display accountdetails
5. Exit

Enter your choice = 2
Enter the withdrawal amount : 200

--menu--

1. Deposit
2. Withdraw
3. Compute interest for SavingsAccount
4. Display accountdetails
5. Exit

Enter your choice = 4
Customer name : Rahul
Account number = 78
Type of account = saving
Balance = 800.0

10/01/24

23.01.24

Lab Program ⑥:

Create a package CIE which has two classes - student and Internals. The class student has members like usn, name, sem. The class Internals 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 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 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.println("Enter the usn");
  usn = s.nextLine();
```

```
System.out.println("Enter the student name");
name = s.nextLine();
System.out.println("Enter the semester");
sem = s.nextInt();
}
public void displayStudentDetails()
{
System.out.println("USN = "+usn);
System.out.println("Student name = "+
name);
System.out.println("Semester = "+sem);
}
}
// Internals.java

package CIE;
import CIE.Student;
import java.util.Scanner;
public class Internals extends Student
{
protected int marks[] = new int[5];
public void inputCIEmarks()
{
Scanner s = new Scanner(System.in);
System.out.println("Enter the marks of
each subject");
for(int i=0; i<5; i++)
marks[i]=s.nextInt();
}
}
```

Externals.java

```
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);
  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

import SEE.Externals;
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:

Enter the usn
1BM22CS197
Enter the student name
Ajay
Enter the semester
3.
Enter CIE Marks.
Enter the marks of each subject
48
46
45
49
50
Enter SEE Marks.
Subject 1 marks : 46
Subject 2 marks : 43
Subject 3 marks : 48
Subject 4 marks : 49
Subject 5 marks : 44

Enter the usn
1BM22CS198
Enter the student name
Ram
Enter the semester
3
Enter CIE Marks
Enter the marks of each subject
44
49

48
47
43
Enter SEE marks
Subject 1 Marks: 49
Subject 2 Marks: 47
Subject 3 Marks: 45
Subject 4 Marks: 46
Subject 5 Marks: 42
~~Subject 6 Mar~~

Displaying data:
USN = 1BM22CS197
Student name = Ajay
Semester = 3
Subject 1: 71
Subject 2: 67
Subject 3: 69
Subject 4: 73
Subject 5: 72

USN = 1BM22CS198
Student name = Ram
Semester = 3
Subject 1: 93
Subject 2: 96
Subject 3: 88
Subject 4: 83
Subject 5: 85

23/01/24

30/1/24 Lab 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() {
        super("Age Error");
    }
    public WrongAge(String message) {
        super(message);
    }
}
class InputScanner {
    Scanner scanner;
    public InputScanner() {
        scanner = new Scanner(System.in);
    }
    public int nextInt() {
        return scanner.nextInt();
    }
}
```

```
class Father extends InputScanner {
  int fatherAge;
  public Father() throws WrongAge {
  System.out.print("Enter father's age:");
   fatherAge = nextInt();
   if (fatherAge < 0) {
   throw new WrongAge("Age cannot be
   negative");
   }
}

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

class Son intends Father {
    int sonAge;
  public Son() throws WrongAge {
     super();
  System.out.print("Enter son's age:");
  sonage = 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 Exception {
    public static void main (String[] args) {
        try {
            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Exception" + e.getMessage());
        }
    }
}
```

Output:

Enter father's age : -48
Exception : Age cannot be negative

Enter father's age : 49
Enter son's age : 60
Exception : Son's age cannot be greater than father's age.

Enter father's age : 56
Enter son's age : -24
Exception : Age cannot be negative

Enter father's age : 45
Enter son's age : 18
Father's age : 45
Son's age : 18

30.01.24

06/02/24 Lab 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 Dr1 extends Thread {
  public void run() {
    while (true) {
      System.out.println("BMS College of
      Engineering");
      try {
        Thread.sleep(10000);
      }
      catch (InterruptedException e) {
      System.out.println("Interrupted:"+e);
      }
    }
  }

class Dr2 extends Thread {
  public void run() {
    while (true) {
      System.out.println("CSE");
      try {
        Thread.sleep(2000);
      }
      catch (InterruptedException e) {
      System.out.println("Interrupted:"+e);
      }
    }
  }
```

```
public class Main {
  public static void main (String args[])
  {
    Ob1 bms = new ob1();
    ob2 bsc = new ob2();
    bms.start();
    csc.start();
  }
}
```

Output:

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

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 a NumberFormatException. If Num2 were Zero, the program would throw an ArithmeticException Display the exception in a message dialog box.

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

class SwingDemo {
    SwingDemo() {
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275,150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JLabel jlab = new JLabel("Enter the divider and dividend:");
    JTextField ajtf = new JTextField(8);
    JTextField bjtf = new JTextField(8);
    JButton button = new JButton("Calculate");
```

```
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();

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

ActionListener l = new ActionListener()
{
public void actionPerformed(ActionEvent
                                ae)
{
System.out.println("Action event from
a text field");
}
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);

button.addActionListener(new ActionListener
(){
public void actionPerformed(ActionEvent
ae){
try{
int a = Integer.parseInt(ajtf.getText());
int b = Integer.parseInt(bjtf.getText());
int ans = a/b;
```

```
alab.setText("\nA = "+a);
blab.setText("\nB = "+b);
anslab.setText("\nAns = "+ans);
}
catch(NumberFormatException e){
 alab.setText("");
 blab.setText("");
 anslab.setText("");
 err.setText("Enter Only Integers!");
}
catch(ArithmeticException e){
 alab.setText("");
 blab.setText("");
 anslab.setText("");
 err.setText("B should be NON zero!");
}
}
});
 jfrm.setVisible(true);
}
public static void main(String args[]){
 SwingUtilities.invokeLater(new Runnable(){
 public void run(){
  new SwingDemo();
});}
}
```

Output:

Enter the divider and divident

20 | 4

Calculate A = 20 B = 4 Ans = 5

Enter the divider and dividend:

10		0

Calculate

B should be NON Zero!

20.02.24

AWT functions:

1. Jframe: It is a class in Java that is part of the swing library, which is used for creating graphical users interfaces in Java application.

2. setSize(): setSize() is a method which is used with components such as Jframe Jpanel etc to set their size.

3. setLayout(): It is a method which is used to set the layout manager for a container, such as Jfram or any other container component.

4. Jlabel: It is a class which is used to display non-editable text or image on GUI

5. setdefaultCloseOperation: It is a method in Java swing used to specify the default close operation for a 'Jframe'.

6. JTextfield : Class in Java Swing provides a text input field in a GUI. It allows users to enter and edit single text line.

7. addActionListener : It is a method in Java Swing that is used to register an action for a component, eg. button that generates action events.

8. setText : It is a method used in Java Swing to set the text content of text based component.

20.02.24

Lab Program (10): Demonstrate Inter process communication and deadlock

i) Implementation of a producer and consumer

```
class Q {
int n;  boolean valueset = false;
synchronized int get() {
```

```
while(!valueSet)
try {
System.out.println("\n Consumer waiting");
wait();
}
catch(InterruptedException e) {
System.out.println("InterruptedException
caught");
}
System.out.println("Got:"+n);
valueSet = false;
System.out.println("\n Intimate Producer\n");
notify();
return n;
}
synchronized void put(int n)
{
while(valueSet)
try {
System.out.println("\n Producer waiting\n");
wait();
}
catch(InterruptedException e) {
System.out.println("InterruptedException
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<15) {
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<15) {
int r = q.get();
System.out.println("consumed :"+r);
i++;
}
}
}
class PC Fixed {
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
Consumed : 1
Put : 2
```

ii) 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("Main Thread");
Thread t = new thread(this, "RacingThread");
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();
}}
```

Output:

```
RacingThread entered B.bar
Main Thread entered A.foo
RacingThread trying to call A.last()
Inside A.last
Back in otherthread
MainThread trying to call B.last()
Inside A.last
Back in mainthread
```

13.02.24

LAB 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("Roo1 = Root2 = " + r1);
}
else if(d&gt;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("Roo1 = " + r1 + " Root2 = " + r2);
}
else if(d&lt;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

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, credits, grade;}
class Student {
    String name;
    String usn;
    double SGPA;
    Scanner s;
    subject subjects[];
Student()
{
int i;
subjects = new subject[9];
for(i=0;i<8;i++)
subjects[i] = new subject();
s = new Scanner(System.in);
}
public void getStudentDetails(){
System.out.println("Enter student name:");
name=s.nextLine();
System.out.println("Enter Student USN:");
usn=s.nextLine();}
public void getMarks(){
int i;
for(i=0;i<8;i++){
```

```
System.out.println("Enter marks of subject"+(i+1)+":");
subjects[i].subjectMarks= s.nextInt();
if(subjects[i].subjectMarks>=40&&subjects[i].subjectMarks<=100){
subjects[i].grade=calculateGrade(subjects[i].subjectMarks);}
else{
System.out.println("Invalid Marks. Marks should be between 40 and 100");}
System.out.println("enter credits:");
subjects[i].credits=s.nextInt();
}
}
public int calculateGrade(int marks){
if (marks>=90)
return 10;
else if(marks>=70&&marks<=80)
return 9;
else if(marks>=60&&marks<=70)
return 8;
else if(marks>=50&&marks<=60)
return 7;
else
return 6;
}
public void computeSGPA() {
        int totalscore = 0;
        int totalcred = 0;
        for (int i = 0; i < 8; i++) {
            totalscore += subjects[i].grade * subjects[i].credits;
            totalcred += subjects[i].credits;
        }
        SGPA = (double) totalscore / (double) totalcred;
    }
```

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

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 book objects.

```
import java.util.Scanner;
class Book {
    private String name;
    private String author;
    private double price;
   private int numPages;
    public Book(String name, String author, double price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public void setAuthor(String author) {
        this.author = author;
    }
    public String getAuthor() {
        return author;
    }
    public void setPrice(double price) {
```

```
        this.price = price;
    }
    public double getPrice() {
        return price;
    }
    public void setNumPages(int numPages) {
        this.numPages = numPages;
    }
    public int getNumPages() {
        return numPages;

    }
    public String toString() {
        return "Book Details: \nName: " + name + "\nAuthor: " + author + "\nPrice: INR" + price +
"\nNumber of Pages: " + numPages;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number of books: ");
        int n = scanner.nextInt();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Book " + (i + 1) + ":");
            scanner.nextLine();
            System.out.println("Enter name: ");
            String name = scanner.nextLine();
            System.out.println("Enter author: ");
            String author = scanner.nextLine();
            System.out.println("Enter price: ");
            double price = scanner.nextDouble();
```

```
            System.out.println("Enter number of pages: ");
            int numPages = scanner.nextInt();
            books[i] = new Book(name, author, price, numPages);
        }
        System.out.println("\nDetails of all books:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nBook " + (i + 1) + ":\n" + books[i]);
        }
        scanner.close();
    }
}
```

LAB 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 classShape. Each one of the classes contain only the method printArea( ) that prints the area of the given shape.

```
import java.util.Scanner;
class InputScanner {
    Scanner s = new Scanner(System.in);
    int getInput(String prompt) {
        System.out.println(prompt);
        return s.nextInt();
    }
}
class shape extends InputScanner {
    double dim1;
    double dim2;
    shape(double a, double b) {
        dim1 = a;
        dim2 = b;
    }
}
class Rectangle extends shape {
    Rectangle() {
        super(0, 0);
        dim1 = getInput("Enter length");
        dim2 = getInput("Enter breadth");
    }
    double area() {
        System.out.println("Inside Area for Rectangle.");
        return dim1 * dim2;
```

```
        }
}
class Triangle extends shape {
    Triangle() {
        super(0, 0);
        dim1 = getInput("Enter length");
        dim2 = getInput("Enter base");
    }
    double area() {
        System.out.println("Inside Area for Triangle.");
        return dim1 * dim2 / 2;
    }
}
class Circle extends shape {
    Circle() {
        super(0, 0);
        dim1 = getInput("Enter the radius");
        dim2 = dim1;
    }
    double area() {
        System.out.println("Inside Area for Circle.");
        return Math.PI * dim1 * dim2;
    }
}
public class Areas {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle();
        System.out.println("Area of Rectangle: " + rectangle.area());

        Triangle triangle = new Triangle();
        System.out.println("Area of Triangle: " + triangle.area());
```

```
        Circle circle = new Circle();
        System.out.println("Area of Circle: " + circle.area());
    }
}
```

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 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 customerName;
    int accountNumber;
    String accountType;
    double balance;
    Account(String name, int number, String type, double initialBalance) {
        customerName = name;
        accountNumber = number;
        accountType = type;
        balance = initialBalance;
    }
    void deposit(double amount) {
        balance += amount;
        System.out.println("Deposit of INR " + amount + " successful");
```

```
    }
    void displayBalance() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Customer Name: " + customerName);
        System.out.println("Account Type: " + accountType);
        System.out.println("Balance: INR " + balance);
    }
    void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawal of INR " + amount + " successful");
        } else {
            System.out.println("Insufficient funds");
        }
    }

    void computeInterest() {
    }
    void checkMinimumBalance(double minBalance, double serviceCharge) {
    }
}
class SavAcct extends Account {
    double interestRate = 0.05;
    SavAcct(String name, int number, String type, double initialBalance) {
        super(name, number, type, initialBalance);
    }
    void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest of INR " + interest + " added to the account");
    }
```

```
}
class CurAcct extends Account {
    double minBalance = 1000;
    double serviceCharge = 50;
    CurAcct(String name, int number, String type, double initialBalance) {
        super(name, number, type, initialBalance);
    }
    void checkMinimumBalance(double minBalance, double serviceCharge) {
        if (balance < minBalance) {
            System.out.println("Service charge of INR " + serviceCharge + " imposed");
            balance -= serviceCharge;
        }
    }
}
public class Bank {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of users: ");
        int numUsers = scanner.nextInt();
        Account[] accounts = new Account[numUsers];
        for (int i = 0; i < numUsers; i++) {
            System.out.println("\nUser " + (i + 1));
            System.out.print("Enter customer name: ");
            scanner.nextLine();
            String name = scanner.nextLine();
            System.out.print("Enter account number: ");
            int accNumber = scanner.nextInt();
            System.out.print("Enter initial deposit amount: INR ");
            double initialDeposit = scanner.nextDouble();
            System.out.print("Enter account type (Savings/Current): ");
            scanner.nextLine();
```

```
    String accType = scanner.nextLine();
    if (accType.equalsIgnoreCase("Savings")) {
        accounts[i] = new SavAcct(name, accNumber, accType, initialDeposit);
    } else if (accType.equalsIgnoreCase("Current")) {
        accounts[i] = new CurAcct(name, accNumber, accType, initialDeposit);
    } else {
        System.out.println("Invalid account type entered. Defaulting to Account.");
        accounts[i] = new Account(name, accNumber, "Account", initialDeposit);
    }
}
boolean exit = false;
while (!exit) {
    System.out.println("\nChoose an option:");
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Display Balance");
    System.out.println("4. Compute Interest (Savings only)");
    System.out.println("5. Exit");
    System.out.print("Enter your choice: ");
    int choice = scanner.nextInt();
    switch (choice) {
        case 1:
            System.out.print("Enter account number: ");
            int accNum = scanner.nextInt();
            System.out.print("Enter deposit amount: INR ");
            double depositAmount = scanner.nextDouble();
            for (Account acc : accounts) {
                if (acc.accountNumber == accNum) {
                    acc.deposit(depositAmount);
                }
            }
```

```
    break;
case 2:
    System.out.print("Enter account number: ");
    accNum = scanner.nextInt();
    System.out.print("Enter withdrawal amount: INR ");
    double withdrawAmount = scanner.nextDouble();
    for (Account acc : accounts) {
        if (acc.accountNumber == accNum) {
            acc.withdraw(withdrawAmount);
        }
    }
    break;
case 3:
    System.out.print("Enter account number: ");
    accNum = scanner.nextInt();
    for (Account acc : accounts) {
        if (acc.accountNumber == accNum) {
            acc.displayBalance();
        }
    }
    break;
case 4:
    System.out.print("Enter account number (for Savings account): ");
    accNum = scanner.nextInt();
    for (Account acc : accounts) {
        if (acc.accountNumber == accNum && acc instanceof SavAcct) {
            ((SavAcct) acc).computeInterest();
        }
    }
    break;
case 5:
```

```
                exit = true;
                break;
            default:
                System.out.println("Invalid choice. Please enter a valid option.");
        }
    }
  }
}
```

LAB 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;
public class Student {
    public String usn;
    public String name;
    public int sem;
    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
}
package CIE;
public class Internals extends Student {
    public int[] internalMarks;
    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
}
```

```
package SEE;
```

```
import CIE.Student;
public class External extends Student {
    public int[] seeMarks;
    public External(String usn, String name, int sem, int[] seeMarks) {
        super(usn, name, sem);
        this.seeMarks = seeMarks;
    }
}

import CIE.Internals;
import SEE.External;
import java.util.Scanner;
public class FinalMarks {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = scanner.nextInt();
        Internals[] cieStudents = new Internals[n];
        External[] seeStudents = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for CIE of student " + (i + 1));
            System.out.print("USN: ");
            String usn = scanner.next();
            System.out.print("Name: ");
            String name = scanner.next();
            System.out.print("Semester: ");
            int sem = scanner.nextInt();
            int[] cieMarks = new int[5];
            System.out.print("Enter CIE marks for 5 courses: ");
            for (int j = 0; j < 5; j++) {
```

```
            cieMarks[j] = scanner.nextInt();
        }
        cieStudents[i] = new Internals(usn, name, sem, cieMarks);
    }
    for (int i = 0; i < n; i++) {
        System.out.println("Enter details for SEE of student " + (i + 1));
        System.out.print("USN: ");
        String usn = scanner.next();
        System.out.print("Name: ");
        String name = scanner.next();
        System.out.print("Semester: ");
        int sem = scanner.nextInt();
        int[] seeMarks = new int[5];
        System.out.print("Enter SEE marks for 5 courses: ");
        for (int j = 0; j < 5; j++) {
            seeMarks[j] = scanner.nextInt();
         }
        seeStudents[i] = new External(usn, name, sem, seeMarks);
    }
    System.out.println("\nFinal Marks of Students:");
    for (int i = 0; i < n; i++) {
        System.out.println("\nDetails of Student " + (i + 1));
        System.out.println("USN: " + cieStudents[i].usn);
        System.out.println("Name: " + cieStudents[i].name);
        System.out.println("Semester: " + cieStudents[i].sem);
        System.out.println("CIE Marks: ");
        for (int j = 0; j < 5; j++) {
            System.out.print(cieStudents[i].internalMarks[j] + " ");
        }
        System.out.println("\nSEE Marks: ");
```

```
            for (int j = 0; j < 5; j++) {
                System.out.print(seeStudents[i].seeMarks[j] + " ");
            }
        }
    }
}
```

LAB 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 Father {
    protected int fatherAge;
    public Father(int age) throws WrongAge {
        fatherAge = age;
        if (fatherAge < 0) {
            throw new WrongAge("Father's age cannot be negative");
        }
    }
}
class Son extends Father {
    private int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAge {
        super(fatherAge);
        this.sonAge = sonAge;
        if (sonAge <= 0) {
            throw new WrongAge("Son's age cannot be negative or zero");
        }
        if (sonAge >= fatherAge) {
```

```
            throw new WrongAge("Son's age cannot be greater than or equal to father's age");
        }
    }
}
 public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.print("Enter father's age: ");
            int fatherAge = scanner.nextInt();
            System.out.print("Enter son's age: ");
            int sonAge = scanner.nextInt();
            Son son = new Son(fatherAge, sonAge);
            System.out.println("Father's age: " + fatherAge);
            System.out.println("Son's age: " + sonAge);
        } catch (WrongAge e) {
            System.out.println("Exception caught: " + e);
            System.out.println("Exception caught: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("Error: " + e);
            System.out.println("Error: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
 }
```

LAB 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 DisplayThread extends Thread {
    private String message;
    private int interval;
    private boolean running = true;
    public DisplayThread(String message, int interval) {
        this.message = message;
        this.interval = interval;
    }
    public void run() {
        while (running) {
            System.out.println(message);
            try {
                Thread.sleep(interval);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
    public void stopThread() {
        running = false;
    }
}
public class ThreadEx {
    public static void main(String[] args) {
        DisplayThread bmsThread = new DisplayThread("BMS College of Engineering", 10000);
        DisplayThread cseThread = new DisplayThread("CSE", 2000);
```

```
        bmsThread.start();
        cseThread.start();
        System.out.println("Press Enter to stop the threads...");
        try {
            System.in.read();
        } catch (Exception e) {
            e.printStackTrace();
        }
        bmsThread.stopThread();
        cseThread.stopThread();
    }
}
```

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 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 SwingDemo{
SwingDemo(){
JFrame jfrm = new JFrame("Divider App");
jfrm.setSize(275, 150);
jfrm.setLayout(new FlowLayout());
jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JLabel jlab = new JLabel("Enter the divider and divident:");
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
JButton button = new JButton("Calculate");
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();
jfrm.add(err); // to display error bois
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
```

```
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);
ActionListener l = new ActionListener() {
public void actionPerformed(ActionEvent evt) {
System.out.println("Action event from a text field");
}
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent evt) {
try{
int a = Integer.parseInt(ajtf.getText());
int b = Integer.parseInt(bjtf.getText());
int ans = a/b;

alab.setText("\nA = " + a);
blab.setText("\nB = " + b);
anslab.setText("\nAns = "+ ans);
}
catch(NumberFormatException e){
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("Enter Only Integers!");
}
catch(ArithmeticException e){
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("B should be NON zero!");
```

```
}
}
});
jfrm.setVisible(true);
}
public static void main(String args[]){
SwingUtilities.invokeLater(new Runnable(){
public void run(){
new SwingDemo();
}
});
}
}
```

LAB PROGRAM 10

Demonstrate Inter process Communication and deadlock.

IPC

```
class Q {
int n;
boolean valueSet = false;
synchronized int get() {
while(!valueSet)
try {
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
System.out.println("Got: " + n);
valueSet = false;
notify();
return n;
}
synchronized void put(int n) {
while(valueSet)
try {
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
this.n = n;
valueSet = true;
System.out.println("Put: " + 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<15) {
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<15) {
int r=q.get();
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);
System.out.println("Back in mainthread");
}
public void run() {
b.bar(a);
System.out.println("Back in other thread");
}
public static void main(String args[]) {
new Deadlock();
}
}
```