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:

PRANEETA M REDDY(1BM22CS205)

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

College of Engineering,

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

INDEX

12/12/23

RA 12-12-23

1. Program Helloworld.

```
class hello
{
    public static void main(String args[])
    {
        System.out.println("hello world");
    }
}
```

Output:
hello world.

classmate
Date
Page

2/12/23

Lab - Program 1
Quadratic Equation

```
import java.util.Scanner;
class Quadratic
{
 Scanner s=new Scanner(System.in);
 int a,b,c;
 double r1,r2,d;
 void getd()
 {
    System.out.println("Praneeta M Reddy, 1BM22CS205");
    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");
        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)
    {
        r1=(-b)/(2*a);
        r2=(Math.sqrt(d))/(2*a);
        System.out.println("roots are imaginary");
        System.out.println("root1="+r1+"i"+r2);
        System.out.println("root2"+r1+"-i"+r2);
    }
  }
}
class QuadraticMain
{
  public static void main(String args[])
  {
      Quadratic q=new Quadratic();
      q.getd();
      q.compute();
  }
}
```

Output:

Praneeta M Reddy, 1BM22CS205

enter the coefficients of a,b,c

1

-5

2

roots are real and distinct

root1=4.561552812828 root2=4.561552812828

Output 2:

Praneeta M Reddy, 1BM22CS205

enter the coefficients of a, b, c

1 2 1

roots are real and equal

root1 = root 2 = -1.0

Output 3:

Praneeta M Reddy, 1BM22CS205

enter the coefficents of a, b, c

0 4 5

not a quadratic equation

enter a non zero value for a:

1

roots are imaginary

root1 = -2.0 + iNaN

root2 = -2.0 - iNaN

19/12/23 Lab-2

SGPA calculator

```
import java.util.Scanner;
class Subject
{
    int subjectmarks;
    int credits;
    int grade;
}
class student
{
    Subject subject[];
    Scanner s=new Scanner(System.in);
    String name;
    String usn;
    double sgpa=0;
    student()
    {
        int i;
        subject = new Subject[9];
        for(i=0;i<9;i++)
        {
            subject[i]= new Subject();
        }
    }
    void getmarks()
    {
        for(int i=0;i<8;i++)
        {
            System.out.println("enter the subjectmarks of the subject"+
                                (i+1)+":");
            subject[i].subjectmarks=s.nextInt();
```

constructor

0...8

```
if(subject[i].subjectmarks>=90)
{
    subject[i].grade=10;
}
else if(subject[i].subjectmarks>=80)
{
    subject[i].grade=9;
}
else if(subject[i].subjectmarks>=70)
{
    subject[i].grade=8;
}
else if(subject[i].subjectmarks>=60)
{
    subject[i].grade=7;
}
else if(subject[i].subjectmarks>=50)
{
    subject[i].grade=6;
}
else if(subject[i].subjectmarks>=40)
{
    subject[i].grade=5;
}
else
{
    subject[i].grade=0;
}
System.out.println("enter the credits of the subject"+(i+1)+":");
subject[i].credits=s.nextInt();
}
}
```

```
void computesgpa()
{
    int totalcredits=0;
    int totalcreditmarks=0;
    for(int i=0; i<8; i++)
    {
        totalcredits+=(subject[i].grade* subject[i].credits);
        totalcreditmarks+=(subject[i].credits);
    }
    sgpa=(double) totalcredits/totalcreditmarks;
    System.out.println("SGPA:"+sgpa);
}
}

class studentMain
{
    public static void main(String args[])
    {
        student s1=new student();
        s1.getstudentdetails();
        s1.getmarks();
        s1.computesgpa();
    }
}
```

Output

Enter the name:

Praneeta

enter the usn:

1BM22CS205

enter the subjectmarks of the subject1:

93

enter the credits of the subject1:
4
enter the subjectmarks of the subject2:
94
enter the credits of the subject2:
4
enter the subjectmarks of the subject3:
91
enter the credits of the subject3:
3
enter the subjectmarks of the subject4:
88
enter the credits of the subject4:
1
enter the subjectmarks of the subject5:
92
enter the credits of the subject5:
1
enter the subjectmarks of the subject6:
94
enter the credits of the subject6:
3
enter the subjectmarks of the subject7:
87
enter the credits of the subject7:
3
enter the subjectmarks of the subject8:
96
enter the credits of the subject8:
1
SGPA: 9.8

26/8/23 Lab 3

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

```
import java.util.Scanner;
class Book
{
    String name;
    String author;
    int price;
    int numpages;
    Book (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="numpages:"+this.numpages+"\n";
        return name+author+price+numpages;
    }
}
```

```
class BookMain
{
  public static void main(String args[])
  {
    System.out.println("Praneeta M Reddy;1BM22CS205");
    Scanner s = new Scanner(System.in);
    System.out.println("enter the no of books:");
    int n = s.nextInt();
    Book b[];
    b = new Book[n];
    for(int i=0; i<n; i++)
    {
      System.out.println("enter the name of book:"+(i+1)+":");
      String name = s.next();
      System.out.println("enter the author of the book");
      String author = s.next();
      System.out.println("enter the price:");
      int price = s.nextInt();
      System.out.println("enter the num of pages:");
      int numpages = s.nextInt();
      b[i] = new Book(name, author, price, numpages);
    }
    System.out.println("details:");
    for(int i=0; i<n; i++)
    {
      system.out.println(b[i].toString());
    }
  }
}
```

Output
Rahecta M Reddy, 1BM22CS205
enter the no of books:
2
enter the name of of book1:
Haarypotter
enter the author of the book
JKRowling
enter the price:
399
enter the num of pages:
511
enter the name of book2:
Divergent
enter the author of the book
Veronica
enter the price:
499
enter the num of pages:
400
details:
book name: Haarypotter
authorname: JKRowling
price: 399
numpages: 511

book name: Divergent
authorname: Veronica
price: 499
numpages: 400

26/12/23

2/1/24 Lab-4

ShapesProgram

```
import java.util.Scanner;
abstract class shape
{
    double dim1, dim2, radius;
    shape(double a, double b)
    {
        dim1 = a;
        dim2 = b;
    }
    shape(double a)
    {
        radius = a;
    }
    abstract void printarea();
}
class rectangle extends shape
{
    rectangle(double a, double b)
    {
        super(a, b);
    }
    void printarea()
    {
        System.out.println("area of rectangle:" + (dim1*dim2));
    }
}
class triangle extends shape
{
    triangle(double a, double b)
    {
        super(a, b);
    }
```

```
    void printarea()
    {
        System.out.println("area of triangle :"+(dim1*dim2)/2);
    }
}
class circle extends shape
{
    circle(double a)
    {
        super(a);
    }
    void printaarea()
    {
        System.out.println("area of circle :"+(3.14*(radius)*(radius)));
    }
}

class ShapeMain
{
    public static void main(String a[])
    {
        Scanner s=new Scanner(System.in);
        System.out.println("Praneeta M Reddy,1BM22CS205");
        System.out.println("enter the length and breadth of rectangle:");
        double l=s.nextInt();
        double b=s.nextInt();
        rectangle r=new rectangle(l,b);
        System.out.println("enter the base and height of triangle:");
        double ba=s.nextInt();
        double h=s.nextInt();
        triangle t=new triangle(ba,h);
        System.out.println("enter the radius:");
```

```
double ra=s.nextInt();
circle c=new circle(ra);
shape sh;
sh=r;
sh.printarea();
sh=t;
sh.printarea();
sh=c;
sh.printarea();
}
}
```

O/P

```
Praneeta M Reddy, 1BM22CS205
enter the length and breadth of rectangle:
2 3
enter the base and height of triangle:
3 4
enter the radius:
5
area of rectangle:6.0
area of triangle:6.0
area of circle:78.5
```

21/11/24

9/1/24 Lab 5

Bank account Program

```
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
                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 acc no:");
        int accno=s.nextInt();
        System.out.println("enter the initial balance:");
        double balance=s.nextDouble();
        int ch;
        double amount1, amount2;
        account ace=new account(name,accno,type,balance);
        savAcct sa=new savAcct(name,accno,balance);
        curAcct ca=new curAcct(name,accno,balance);
        while(true)
        {
            if(ace.type.equals("savings"))
            {
                System.out.println("\nMenu\n1.deposit
                    2.withdraw 3.compute interest 4.display
                    5.exit");
                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 ("1.Menu\n1).deposit 2.withdraw
            3.display 4.exit");
    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);
        }
    }
}
```

O/P

```
enter the name:
praneeta
enter the type (current/savings);
current:
enter the account number;
1234
enter the initial balance:
2000

Menu
1. deposit 2. withdraw 3. display 4. exit
enter the choice
1
enter the amount:
500

Menu
1. deposit 2. withdraw 3. display 4. exit
enter the choice
3
name: praneeta acno: 1234 type: current balance: 2500

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

9/1/24

23/1/24. Lab 6.

Packages Program.

```
package cie;
import java.util.Scanner;
public class Students
{
    protected String usn = new String();
    protected String name = new String();
    protected int sem;
    public void getdetails()
    {
        Scanner s = new Scanner(System.in);
        System.out.println("enter the usn:");
        usn = s.next();
        System.out.println("enter the name:");
        name = s.next();
        System.out.println("enter the sem:");
        sem = s.nextInt();
    }
    public void displaydetails()
    {
        System.out.println("name:"+name+" "+"usn:"
        +usn+" "+"sem:"+sem);
    }
}
```

```
package cie;
import java.util.Scanner;
public class internals extends Students
{
    protected int[] marks = new int[5];
    public void getinternals()
    {
```

```
Scanner s = new Scanner(System.in);
for(int i=0; i<5; i++)
{
    System.out.println("enter the internal marks of course" +(i+1));
    marks[i] = s.nextInt();
}
}
}
```

```
package see;
import cie.internals;
import java.util.Scanner;
public class external extends internals
{
    protected int mark[] = new int[5];
    protected int finalMarks[] = new int[5];
    public void getexternal()
    {
        Scanner s = new Scanner(System.in);
        for(int i=0; i<5; i++)
        {
            System.out.println("enter the external marks of course" +(i+1));
            mark[i] = s.nextInt();
        }
    }
    public void cal()
    {
        for(int i=0; i<5; i++)
            finalMarks[i] = mark[i]/2 + super.marks[i];
    }
```

```
public void displayFinal()
{
    displaydetails();
    for(int i=0; i<5; i++)
        System.out.println("course"+(i+1)+
        ":"+finalMarks[i]);
}
}

import see.external;
class finalMain
{
    public static void main(String args[])
    {
        int no=2;
        external e[]= new external[no];
        for(int i=0; i<no; i++)
        {
            e[i]= new external();
            e[i].getdetails();
            System.out.println("enter cie marks");
            e[i].getinternals();
            System.out.println("enter see marks");
            e[i].getexternal();
        }
        System.out.println("displaying details:");
        for(int i=0; i<no; i++)
        {
            e[i].cal();
            e[i].displayFinal();
        }
    }
}
```

Output enter the usn:
1BM22CS205
enter the name:
Praneeta
enter the sem:
3
enter cie marks
enter internals marks of course1
45
enter internals marks of course2
46
enter internals marks of course3
47
enter internals marks of course4
48
enter internals marks of course 5
49
enter see marks
enter external marks of course1
99
enter external marks of course2
80
enter external marks of course3
98
enter external marks of course4
87
Enter external marks of course 5
98
enter the usn:
1BM22CS206
enter the name:
Ruchila

enter the sem:
3.
enter the cie marks
enter the internals marks of course1
45
enter the internals marks of course 2
43
enter the internals marks of course 3
46
enter the internals marks of course 4
50
enter the internals marks of course 5
44
enter see marks
enter the external marks of course 1
98
enter the external marks of course 2
99
enter the external marks of course 3
47
enter the external marks of course 4
89
enter the external marks of course 5
90
displaying details.
name: Pranata usn: 1BM22CS205 sem: 3
course1: 94
course2: 86
course3: 96
course4: 91
course5: 98
name: Rushila usn: 1BM22CS206 sem: 3

~~display~~
course1: 91
course2: 92
course3: 89
course4: 91
course5 84

23/01/24

30/01/24

Lab program - 7

```
import java.util.Scanner;
class WrongAgeException extends Exception {
    public WrongAgeException (String e){
        super(e);
    }
}
class father{
    private int fage;
    public father (int fage) throws WrongAgeException{
        if (fage < 0){
            throw new WrongAgeException ("age cannot be negative");
        }
        this.fage = fage;
    }
    public void display()
    {
        System.out.println ("father age:" + fage);
    }
}
class son extends father{
    private int sage;
    public son (int sage, int fage) throws WrongAgeException{
        super(fage);
        if (sage > = fage){
            throw new WrongAgeException ("son age
            cannot be greater than or equal to father
            age");
        if (sage < 0)
        {
            throw new WrongAgeException ("age cannot be
            negative");
        }
```

```
        this.sage=sage;
    }
    public void show()
    {
        System.out.println("son age:"+sage);
    }
}

public class ageMain{
    public static void main(String args[]){
        Scanner s=new Scanner(System.in);
        System.out.println("enter the father age:");
        int fatherage=s.nextInt();
        System.out.println("enter son age:");
        int sonage=s.nextInt();
        try{
            Son Son=new Son(sonage,fatherage);
            Son.display();
            Son.show();
        }
        catch(WrongAgeException e){
            System.out.println("error:"+e.getMessage());
        }
    }
}
```

O/P-1 enter the father age.
30
Enter son age: 40
error: son age cannot be greater than father age.

O/P-2 enter the father age: -10
enter the son age: 20
error: age cannot be negative

30/01/24

D/P3

enter father age: 30
enter son age: 20
father age: 30
son age: 20

30/01/24

6/2/24 Lab Program 8

MultiThreading

```
class bms extends Thread{
    public void run(){
        for(int i=1; i<=5; i++){
            System.out.println("bms college of engineering"+i);
            try
            {
                Thread.sleep(10000);
            }
            catch(InterruptedException e){
                e.printStackTrace();
            }
        }
    }
class cse extends Thread{
    public void run(){
        for(int i=1; i<=10; i++){
            System.out.println("cse"+i);
            try{
                Thread.sleep(2000);
            }
            catch(InterruptedException e){
                e.printStackTrace();
            }
        }
    }
class threadMain{
    public static void main(String a[]){
        bms obj1 = new bms();
        cse obj2 = new cse();
```

```
obj1.start();
obj2.start();
}
}
```

Output

bms college of engineering 1
cse 1
cse 2
cse 3
cse 4
cse 5
bms college of engineering 2
cse 6
cse 7
cse 8
cse 9
cse 10
bms college of engineering 3
bms college of engineering 4
bms college of engineering 5.

6/2/24 Lab Program 10

Interprocess communication

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get(){
        while(!valueSet){
            try{
                System.out.println("Consumer
                    waiting");
                wait();
            }catch(InterruptedException e){
            System.out.println("Interrupted Exception
                                caught");

        }
    }
    System.out.println("Got:" +n);
    valueSet = false;
    System.out.println("Intimate Producer\n");
    notify();
    return n;
}
synchronized void put(int n){
    while(valueset){
        try{
            System.out.println("Producer waiting");
            wait();
        }catch(InterruptedException e){
            System.out.println("InterruptedException
                                caught");

    }
}
```

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

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

O/P Put : 0
Intimate Consumer

Producer waiting
Press Control-C to stop.
Got : 0
Intimate Producer

Put : 1
Intimate Consumer

Producer waiting
Consumed : 0
Got : 1
Intimate Producer.

Consumed : 1
Put : 2
Intimate Consumer.

Producer waiting
Got : 2
Intimate Producer.

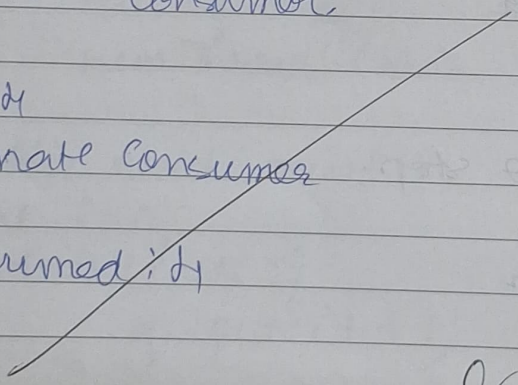
consumed: 2
Put: 3
Intimate consumer

Producer waiting
Got: 3
Intimate consumer

Producer waiting
Put: 4
Intimate consumer

Got: 4
Intimate consumer

consumed: 4



8
6/2/24

13/2/24

Lab Program -10

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.baa(a);
        System.out.println("Back in other thread");
    }
    public static void main(String a[]) {
        new Deadlock();
    }
}
```

O/P

```
MainThread entered A.foo
RacingThread entered B.baa
RacingThread trying to call A.last()
MainThread trying to call B.last()
Inside A.last
Back in Main Thread
Inside A.last
Back in other thread
```

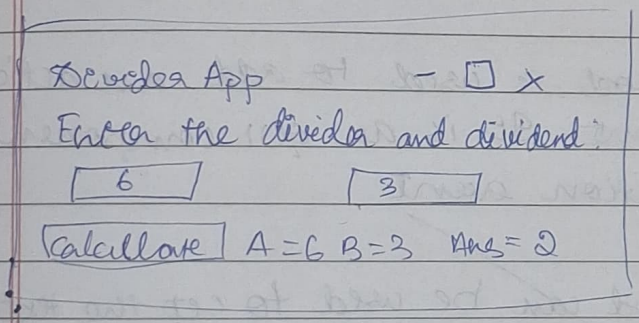
20/2/24 Lab program 9

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

class UserInterface {
    UserInterface(){

        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 answlab = 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(answlab);
```

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

        alab.setText("\n A = " + a);
        blab.setText("\n B = " + b);
        anslab.setText("\n Ans = " + ans);
        err.setText("");
    } catch (NumberFormatException e) {
        displayErrorMessage("Enter Only Integers !");
    } catch (ArithmeticException e) {
        displayErrorMessage("B should be non-zero!!");
    }
}

private void displayErrorMessage(String message) {
    alab.setText("");
    Blab.setText("");
    anslab.setText("");
    err.setText(message);
}
};

button.addActionListener(calculateListener);
jfrm.setVisible(true);
}
```

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

O/P



Functions

JFrame: It is a top-level container in JavaSwing that represents a window with a title bar, border and optional menu bar. It serves as the main container for building GUIs in Swing based application.

setSize: It is used to setsize of the frame. It takes height and width as its parameters

setLayout: It is used to set the layout manager for the JFrame. The Layout manager is responsible for arranging the components that are added to the JFrame

setDefaultCloseOperation: It is used to set the default close operation for the JFrame.

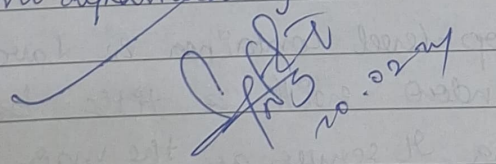
JLabel : It is class, which is used to create a label component

JTextfield : It is used to create a text field component, where the user can enter and edit a single line of text.

add() : This method is used to add components to the JFrame. This method takes a single parameter, which is the component that should be added to the JFrame.

addActionListener : This method is used to add an action listener to a component. It is an GUI component that listens for action event

setText() : This method can be used to set the text of any label component and to change the text of label component dynamically.



Program1

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.

```
//code
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>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();
    }
}
```

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.

```
//code
import java.util.Scanner;
class Subject
{
            int subjectmarks;
            int credits;
            int grade;
}

class student
{
            Subject subject[];
            Scanner s=new Scanner(System.in);
            String name;
            String usn;
            double sgpa=0;
```

```
student()
{
	int i;
	subject=new Subject[9];
	for(i=0;i<9;i++)
	{
		subject[i]=new Subject();
	}
}

void getstudentdetails()
{
	System.out.println("enter the name:");
	name=s.next();
	System.out.println("enter the usn:");
	usn=s.next();
}
void getmarks()
{
	for( int i=0;i<8;i++)
	{
		System.out.println("enter the subjectmarks of the subject"+(i+1)+":");
		subject[i].subjectmarks=s.nextInt();
		if (subject[i].subjectmarks>=90 && subject[i].subjectmarks<=100)
		{
			subject[i].grade=10;
		}
		else if (subject[i].subjectmarks>=80)
		{
			subject[i].grade=9;
		}
		else if (subject[i].subjectmarks>=70)
		{
			subject[i].grade=8;
		}
		else if (subject[i].subjectmarks>=60)
		{
			subject[i].grade=7;
		}
		else if (subject[i].subjectmarks>=50)
		{
			subject[i].grade=6;

		}
```

```
                else if (subject[i].subjectmarks>=40)
                {
                        subject[i].grade=5;

                }
                else
                {
                        subject[i].grade=0;
                }




                System.out.println("enter the credits of the subject"+(i+1)+":");
                subject[i].credits=s.nextInt();

        }
    }
    void computesgpa()
    {
        int totalcredits=0;
        int totalcredithours=0;
        for(int i=0;i<8;i++)
        {

                totalcredits+=(subject[i].grade*subject[i].credits);
                totalcredithours+=(subject[i].credits);

        }
        sgpa=(double) totalcredits/totalcredithours;
        System.out.println("SGPA:"+sgpa);
    }
}
class studentMain
{
    public static void main(String args[])
    {
        student s1=new student();
        s1.getstudentdetails();
        s1.getmarks();
        s1.computesgpa();
    }
}
```

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.

```
//code
import java.util.Scanner;
class Book
{
	String name;
	String author;
	int price;
	int numpages;
	Book(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="numpages:"+this.numpages+"\n";
		return name+author+price+numpages;
	}
}
class BookMain
{
	public static void main(String args[])
	{
		System.out.println("Praneeta M Reddy,1BM22CS205");
		Scanner s=new Scanner(System.in);
		System.out.println("enter the no of books:");
		int n=s.nextInt();
		Book b[];
		b=new Book[n];
		for (int i=0;i<n;i++)
```

```
		{
			System.out.println("enter the name of book"+(i+1)+":");
			String name=s.nextLine();
			System.out.println("enter the author of the book");
			String author=s.next();
			System.out.println("enter the price:");
			int price=s.nextInt();
			System.out.println("enter the num of pages:");
			int numpages=s.nextInt();
			b[i]=new Book(name,author,price,numpages);
		}
		System.out.println("details:");
		for(int i=0;i<n;i++)
		{
			System.out.println(b[i].toString());

		}
	}
}
```

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.

```
//code
import java.util.Scanner;

abstract class shape
{
	double dim1,dim2,radius;
	shape(double a,double b)
	{
		dim1=a;
		dim2=b;
	}
	shape(double a)
	{
		radius=a;
	}
	abstract void printarea();
```

```
}
class rectangle extends shape
{
	rectangle (double a,double b)
	{
		super(a,b);
	}
	void printarea()
	{
		System.out.println("area of rectangle:"+(dim1*dim2));
	}
}
class triangle extends shape
{
	triangle (double a,double b)
	{
		super(a,b);
	}
	void printarea()
	{
		System.out.println("area of triangle:"+(dim1*dim2)/2);
	}
}
class circle extends shape
{
	circle (double a)
	{
		super(a);
	}
	void printarea()
	{
		System.out.println("area of circle:"+(3.14*(radius)*(radius)));
	}
}
class ShapeMain
{
	public static void main(String a[])
	{
		Scanner s=new Scanner(System.in);
		System.out.println("enter the length and breadth of rectangle:");
		double l=s.nextInt();
		double b=s.nextInt();
		rectangle r=new rectangle(l,b);
		System.out.println("enter the base abd height:");
```

```
                double ba=s.nextInt();
                double h=s.nextInt();
                triangle t=new triangle(ba,h);
                System.out.println("enter the radius:");
                double ra=s.nextInt();
                circle c=new circle(ra);
                shape sh;
                sh=r;
                sh.printarea();
                sh=t;
                sh.printarea();
                sh=c;
                sh.printarea();
        }
}
```

Program 5

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

```
//code
import java.util.Scanner;
class account
{
        String 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 5.exit");
		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
4.exit");
		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;
                        }
                }
        }

    }
}
```

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.

```
//code
package cie;
import java.util.Scanner;
public class students
{
        protected String usn=new String();
        protected String name=new String();
        protected int sem;
        public void getdetails()
        {
                Scanner s=new Scanner(System.in);
                System.out.println("enter the usn:");
                usn=s.next();
                System.out.println("enter the name:");
                name=s.next();
                System.out.println("enter the sem:");
                sem=s.nextInt();
        }
```

```
	public void displaydetails()
	{
		System.out.println("name:"+name+" "+"usn:"+usn+" "+"sem:"+sem);
	}
}
package cie;
import java.util.Scanner;
public class internals extends students
{
	protected int marks[]=new int[5];
	public void getinternals()
	{
		Scanner s=new Scanner(System.in);
		for(int i=0;i<5;i++)
		{
			System.out.println("enter the internal marks of course"+(i+1));
			marks[i]=s.nextInt();
		}
	}
}
package see;
import cie.internals;
import java.util.Scanner;
public class external extends internals
{
	protected int mark[]=new int[5];
	protected int finalMarks[]=new int[5];
	public void getexternal()
	{
		Scanner s=new Scanner(System.in);
		for(int i=0;i<5;i++)
		{
			System.out.println("enter the external marks of course"+(i+1));
			mark[i]=s.nextInt();
		}
	}
	public void cal()
	{
		for(int i=0;i<5;i++)
		{

			finalMarks[i]=mark[i]/2+super.marks[i];
		}
	}
```

```
	public void displayFinal()
	{
		displaydetails();
		for(int i=0;i<5;i++)
		{
			System.out.println("course"+(i+1)+":"+finalMarks[i]);

		}
	}
}
import see.external;
class finalMain
{
	public static void main(String args[])
	{
		int no=2;
		external e[]=new external[no];
		for(int i=0;i<no;i++)
		{
			e[i]=new external();
			e[i].getdetails();
			System.out.println("enter cie marks");
			e[i].getinternals();
			System.out.println("enter see marks");
			e[i].getexternal();
		}
		System.out.println("displaying details:");
		for(int i=0;i<no;i++)
		{
			e[i].cal();
			e[i].displayFinal();
		}
	}
}
```

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 greater than or equal to father’s age.

```
//code
import java.util.Scanner;
class WrongAgeException extends Exception{
	public WrongAgeException(String e){
	super(e);
	}
}
class father{
	private int fage;
	 public father(int fage) throws WrongAgeException{
		if(fage<0){
			throw new WrongAgeException ("age cannot be negative");
		}
		this.fage=fage;
	}
	public void display()
	{
		System.out.println("father age:"+fage);
	}
}
class son extends father{
	private int sage;
	 public son( int sage,int fage) throws WrongAgeException{
		super(fage);
		if(sage>=fage){
			throw new WrongAgeException ("son age cannot be greater than
father age");
		}
		if(sage<0)
		{
			throw new WrongAgeException("age cannot be negative");
		}
		this.sage=sage;
	}
	public void show()
	{
		System.out.println(" son age:"+sage);
	}
}
public class ageMain{
	public static void main(String args[]){
		Scanner s=new Scanner(System.in);
		System.out.println("enter the father age:");
		int fatherage=s.nextInt();
```

```
                System.out.println("enter sons age:");
                int sonage=s.nextInt();
                try{
                        son Son=new son(sonage,fatherage);
                        Son.display();
                        Son.show();
                }
                catch(WrongAgeException e){
                        System.out.println("error:"+e.getMessage());
                }
        }
}
```

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.

```
//code
class bms extends Thread{
        public void run(){
                for(int i=1;i<=5;i++){
                        System.out.println("bms college of engineering"+i);
                        try
                        {
                                Thread.sleep(10000);
                        }
                        catch(InterruptedException e){
                                e.printStackTrace();
                        }
                        }
                }
}
class cse extends Thread{
        public void run(){
                for(int i=1;i<=10;i++){
                        System.out.println("cse"+i);
                        try
                        {
                                Thread.sleep(2000);
                        }
```

```
                        catch(InterruptedException e){
                                e.printStackTrace();
                        }
                        }
                }
}
class threadMain{
        public static void main(String a[]){
                bms obj1=new bms();
                cse obj2=new cse();
                obj1.start();
                obj2.start();
        }
}
```

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.

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

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

      // text label
```

```
JLabel jlab = new JLabel("Enter the divider and dividend:");

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

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

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

// add in order :)
jfrm.add(err);  // to display error message
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

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

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

```
        }

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

      button.addActionListener(calculateListener);

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

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

Program 10

Demonstrate Inter process Communication and deadlock

Interprocess Communication

```
class Q {
        int n;
        boolean valueSet = false;
        synchronized int get() {
                while(!valueSet){
                        try {
                                System.out.println("Consumer waiting");
                                wait();
                        } catch(InterruptedException e) {
                        System.out.println("InterruptedException caught");
                        }
```

```
                }
            System.out.println("Got: " + n);
            valueSet = false;
            System.out.println("Intimate Producer\n");
            notify();
            return n;
        }
        synchronized void put(int n) {
            while(valueSet){
                try {
                    System.out.println("Producer waiting");
                    wait();
                    } catch(InterruptedException e) {
                        System.out.println("InterruptedException caught");
                    }

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

}

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

class Consumer implements Runnable {
        Q q;
        Consumer(Q q) {
```

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

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 main thread");
        }
        public void run() {
                b.bar(a);
                System.out.println("Back in other thread");
        }
        public static void main(String args[]) {
                new Deadlock();
        }

}
```