B.M.S COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

Submitted in partial fulfillment of the requirements for record of

OBJECT ORIENTED JAVA PROGRAMMING

(23CS3PCOOJ)

Submitted by:

SIRIGIREDDY PRANAV REDDY

1BM22CS281

Faculty incharge:

Dr Seema Patil

Department of Computer Science and Engineering
B.M.S College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019

B.M.S COLLEGE OF ENGINEERING DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Name : SIRIGIREDDY PRANAV REDDY

Section : 3E

USN : 1BM22CS281

INDEX

SNO	Name	Date
1)	Quadratic	12/12/23
2)	SGPA calculation	19/12/23
3)	Book Program	26/12/23
4)	Abstract class Shape	2/1/24
5)	Bank Account	9/1/24
6)	Package	23/1/24
7)	Exception handling	30/1/24
8)	Threads	6/2/24
9)	Applet	20/2/24
10)	Deadlock, ICP	13/2/24

12/12/23 LRA 12-12-23 LAB-1

→ Develope a java program that prints all real solutions to the quadratic equations $ax^2+bx+c=0$. Read in a, b, c & use the quadratic formula. If the discriminate b^2-4ac is negative display a message stating that there are no real solutions.

I/P:

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

```
        Scanner sc = 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 = Root 2 = " + r1);
    }
else if (d > 0)
{
    r1 = ((-b) + (Math.sqrt(d))) / (double)(2*a);
    r2 = ((-b) - (Math.sqrt(d))) / (double)(2*a);
    System.out.println("Roots are real and distinct");
    System.out.println("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.compb();
      System.out.println("S.Pranav Reddy - 1BM22CS281");
    }
  }
```

O/P:

1.) Enter coefficients a, b, c

1
2
1

Roots are real and equal

Root 1 = Root 2 = -1
S.Pranav Reddy - 1BM22CS281

2.) Enter coefficients a, b, c

1
2
3

Root are imaginary
Root 1 = -1 + 1.4142135623731i
Root 2 = -1 - 1.4142135623731i
S.Pranav Reddy - 1BM22CS281

3) Enter coefficients a, b, c

2
3
6

Roots are imaginary

Root1 = -0.75+1.5612494995996i

Root1 = -0.75-1.5612494995996i

S.Pranav Reddy - 1BM22CS281

4) Enter coefficients a, b, c

1
4
1

Roots are Real and distinct

Root 1 = -0.2679491924311228

Root2 = -3.7320508075688877

S.Pranav Reddy - 1BM22CS281

Lab-2

19/12/23

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

$$SGPA = \frac{\sum [(\text{course credits})(\text{Grade points})]}{\sum [\text{course credits}]}$$

T|P

KA19-12-23

```
import java.util.Scanner;

class Subject {

    int marks;
    int credits;
    int points;

}

class Student {

    Subject[] subjects;

    String name;

    String usn;

    double sgpa;

    Scanner sc;

    Public Student() {

        int i;

        subjects = new subject[9];

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

            Subjects[i] = new Subject();

        SC = new scanner(System.in);

    }

    Public void InputDetails() {

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

        name = sc.nextLine();
```

```
System.out.print("Enter student USN: ");
USN = sc.nextLine();

for (int i=0; i<3; i++) {

    Subject[i] = new subject();
    System.out.println("Enter details for subject " + (i+1));
    System.out.println("Enter marks for the subject: ");
    subjects[i].marks = sc.nextInt();
    System.out.print("Enter credits for the subject: ");
    subjects[i].credits = sc.nextInt();
  }
}

Private int calculatepoints (int marks) {
    if (marks > 90) {
        return 10;
    } else if (marks > 80) {
        return 9;
    } else if (marks > 70) {
        return 8;
    } else if (marks > 60) {
        return 7;
    } else if (marks > 50) {
        return 6;
    } else {
        return 0;
    }
}
```

```
Public void calculateSGPA() {
    double totalCredits = 0;
    double weightedSum = 0;
    for (int i=0; i<9; i++) {
        totalCredits += subjects[i].credits;
        subjects[i].points = calculatePoints(subjects[i].
                                             marks);
        weightedSum += subjects[i].credits * subjects[i].points;
    }
    sgpa = weightedSum / totalCredits;
}
Public void displayResult() {
    System.out.println("Student Name: " + name);
    System.out.println("Student USN: " + usn);
    System.out.println("SGPA: " + sgpa);
}
}
Public class Main {
    Public static void main(String[] args) {
        Student student = new Student();
        Student.inputDetails();
        Student.calculateSGPA();
```

```
            Student.displayResult();
        }
    }
```

O/P:

```
Enter student name : Pranav
Enter Student USN : 1BM22CS181
Enter details for student 1
Enter marks for the subject : 89
Enter credits for the subject : 4
Enter details for Studed 2
Enter marks for the subject : 89
Enter credits for the subject : 3
Enter details for student 3
Enter marks for the subject : 99
Enter credits for the subject : 3
Enter details for student 4
Enter marks for the subject : 87
Enter credits for the subject : 3
Enter details for studed 5
Enter marks for the subject : 78
Enter credit for the subject 3
Enter details for student 6
Enter marks for the subject : 80
```

Enter Credits for the Subject: 3

Enter details for the Subject 7

Enter marks for the Subject: 88

Enter Credits for the Subject: 1

Enter details for subject 8

Enter marks for the subject: 90

Enter credits for the subject: 4

Enter details for Subject 9

Enter marks for the subject: 90

Enter credits for the subject: 1

Student Name: Pranav

Student USN: 1BM22CS281

SGPA: 8.863636363

Lab-3 28/12/23

→ 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.

I/P:

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

Books(String name, String author,int price,int numpages){
    this.name=name;
    this.author=author;
    this.price=price;
    this.numpages=numpages;
}
Public string toString(){
    String name,author,price,numpages,
    name="Book name:"+this.name+"\n";
    author="Author name:"+this.author+"\n";
    Price="price:"+this.price+"\n";
}   return name+author+Price+numpages;
```

```
}

class BookMain {

    Public static void main ( String[] args ) {

        Scanner sc = new Scanner (System.in);
        int n;
        String name, author;
        int price, numpages;
        System.out.println ("Enter the number of books:");
        n = sc.nextInt();
        Books b[] = new Books[n];
        System.out.println ("Enter name, author, price and numbe
         of pages:");
        for (int i=0; i<n; i++) {

            name = sc.next();
            author = sc.next();
            Price = sc.nextInt();
            numpages = sc.nextInt();
            b[i] = new Books (name, author, price, numpages);
        }
        System.out.println ("Book details");
        for (int i=0; i<n; i++) {
            System.out.println (b[i].toString());
        }
    }
}
```

o/p:

Name: S Pranav Reddy

USN: 1BM22CS281

Enter the number of book:

1

Enter Name, author, price and number of pages

007

Jhon

750

1275

Book Details:

Book name: 007

author name: Jhon

Price: 750

Number of pages: 1275

LAB-4 URA 2-1-23 2/1/24

→ 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 & 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

I/P

```
import java.util.*;
import java.util.Scanner;
Abstract class Shape {
        int length;
        int width;
        Public abstract void printarea();
    }

class Rectangle extend shape{
        int area;
    Public Rectangle(int length, int width) {
            this.length = length;
            this.width = width;
            this.area = length * width;
    }
    Public void printarea() {
            System.out.println("Area of rectangle is" + area);
    }
}
```

```
class Triangle extends Shape {
    int area;
    Public Triangle (int length, int width){
        this length = length;
        this width = width;
        This.area = (length * width) /2;
    }
    Public void printarea () {
        System.out.println("Area of triangle :" + area
    }
}

class circle extends Shape {
        double are;
        Public circle (int length) {
            this.length = length;
            this.width = 0;
            this.area = 3.14 * length * length;
        }
    Public void printarea () {
        system.out.println("Area of circle" + area);
    }
}

    Public class Main {
        Public static void main (String[] args) {
```

```
int a,b,r;
Scanner Sc = new Scanner(System.in);

System.out.println("Enter length & width and radius");

a = Sc.nextInt();
b= Sc.nextInt();
r= Sc.nextInt();

Rectangle rec = new Rectangle(a,b);
Triangle tri = new Triangle (a,b);
Circle cir = new Circle(r);

rec.printarea();
tri.printarea();
cir.printarea();
}
}
```

LAB-5

9/1/24

→ Develop a java program to create a class Bank that maint... two kinds of account for it customers, one called Savings ... and the other current account. The savings account. The Sav... account provides compound interest and withdrawal facilities b... cheque book facility. The current account provides cheque book ... but no interest. Current account holders should also maintain a m... balance and if the balance falls below this level, a service ... is imposed. Create a class Account that stores customer name, ... number and type of account from this derive the classes 'Cur-a... and Sav-acct to make them more specific to their requir... Include the necessary methods in order to achieve the following tasks:

a) accept deposit from customer and update balance

b) Display the balance

c) compute & deposite interest

d) Permit withdrawal and update the balance

e) Check for the minimum balance, impose penalty if necessary and ... the balance

I/P.

```
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 deposite (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, "Saving", 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 Main
  {
```

```
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 account number:");
  int accno = s.nextInt( );
  System.out.println ("Enter the initial 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("menu 1. deposit 2.withdraw 3.compute
      4. display");
    System.out.println(" enter the choice");
    ch = s.nextInt( );
    switch(ch)
    {
      case 1: System.out.println("enter the amount");
      amount1 = s.nextInt( );
```

```
sa.deposit(amount1);
break;
case 2: System.out.println("Enter the amount");
amount2 = sc.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;
}
}
public static void
{
case 1: System.out.println("Enter the amount");
amount1 = sc.nextInt();
c.deposite(amount);
break;
case 2: System.out.println("Enter the amount");
amount2 = sc.nextInt();
```

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

O/P

```
Enter the name: pranav
Enter the type (current/savings):
current
Enter the account number:
217776
Enter minimum balance.
10000
menu
1 deposite 2. withdraw 3 display
Enter choice
3
name: pranav acno: 217776 type: current
```

balance 9800. menu

1 deposite 2. with draw 3 display

enter the choice:

Enter the name : pranav

Enter the type (current | savings): saving

Enter the account number : 122122

Enter the initial balance : 7800

menu

1. deposite 2 withdraw 3. compute interest 4 display

Enter the choice

2

Enter the amount:

6000

menu

1. deposite 2. withdraw 3. Compute interest 4 display

enter the choice:

3

balance : 1890

menu

1 deposite 2 withdraw 3 compute interest 4. display

enter the choice

name : pranav acno 122122 type: savings balance 1890

Lab-6 16-1-24

1) Demonstrate String length, String literal, String concat

S/P:

```
Public class Main {

    Public Shatic Void main (String [] args) {

        String Str1 = "Hello, World!";
        int length = str1.length()
    System.out.println ("String: " + Str1);
    System.out.println ("length g the string : " + length);

    String Str2 = "Java";
    String Str3 = "Java";

    boolean areEqual = (Str2 == Str3);
    System.out.println ("Are Str2 and Str3 equal?" + areEqual);

    String firstname = "John";
    String lastname = "Doe";
    String fullname = firstname + " " + lastname;
    System.out.println ("Full Name : " + full name);
    }
}
```

O/P:

```
String : Hello, world!
length g the String : 13
Are Str2 and Str3 equal ? true
Full name : John Doe
```

2.) write a Java program to perform sorting of numbers from 1 using Comparator()

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

Public class NumberSorting {

    Public static void main (String[] args) {

Integer[] numbers = new Integer[] { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1};
Arrays.sort(numbers);
System.out.println("Sorted Numbers (Ascending Order): ");
for (Integer number : numbers){

        System.out.print(number + " ");
}

    System.out.println();
    Arrays.sort(numbers, (a,b) -> b.compareTo(a));
    System.out.println("Sorted Numbers (Descending order):");
    for (Integers number : numbers) {

        System.out.println(number + " ");
    }

}
}
```

n/8

sorted Alphabetical (Ascending order.)

1 2 3 4 5 6 7 8

sorted numbers (descending order.)

10 9 8 7 6 5 4 3 2 1

3) write a Java program to create an abstract class Bird with abstract methods fly() and makesound(). Create subclasses Eagle and Hawk that extend the Bird class and implement the respective methods to describe how each bird flies and makes a sound

Ans

```
abstract class Bird {
    abstract void fly();
    abstract void makesound();
}
class Eagle extends Bird {
    void fly() {
        System.out.println("Eagle is soaring high in the
                            sky.");
    }
    void makesound() {
        System.out.println("Eagle makes a screaming sound");
    }
}
```

```
class Hawk extends Bird {

    void fly() {

        System.out.println("Hawk is gliding gracefully thro
        air.");
    }
Void makeSound() {

    System.out.println("Hawk makes a sharp cry.");
    }
}

Public class BirdDemo {

    Public static void main(String[] args) {

     Eagle eagle = new Eagle();
     Hawk hawk = new Hawk();

     eagle.fly();
     eagle.makeSound();
     hawk.fly();
     hawk.makeSound();
    }
}
```

O/P

Eagle is soring high in the sky.

Eagle makes a screeching sound.

Hawk is gliding gracefully through the air.
Hawk makes a sharp cry.

Q.) write a Java program to create a generic class Stack which hold 5 integers and 5 double values

I/P:

```
class GenericStack<T> {
    Private Object[] StackArray;
    Private int top = -1;
    Private static final int Max_Size = 5
    public GenericStack(){
        StackArray = new Object[Max_Size];
    }
    Public void push(T value){
        if(top < max_size-1) StackArray[++top] = value;
        else System.out.println("Stack is full");
    }
    @SuppressWarnings("unchecked")
Public T pop(){
    if(top >= 0)
        return (T) StackArray(top--);
    else {
        System.out.println("Stack is empty")
        return null;
    }
```

```
Public boolean isEmpty(){
    return top == -1;
}
Public boolean isFull(){
    return top == Max_Size-1;
}
}
class Main{
Public static void Main (String[] args){
    GenericStack<Integer> integerStack = new GenericStack<>();
    GenericStack<Double> doubleStack = new GenericStack<>();
    for(int i=1; i<5; i++){
        integerStack.push(i);
    }
    for(double i = 10; i<=5.0; i++){
        doubleStack.push(i);
    }
    System.out.println("Popped integers from the stack:");
    while(!integerStack.isEmpty()){
        System.out.println(integerStack.pop());
    }
    System.out.println("Popped doubles from the stack:");
```

```
while ( !doubleStack.isEmpty()) {
        System.out.println (doubleStack.pop());
    }
  }
}
```

O/P

Popped integers from the stack

5
4
3
2
1

Popped doubles from the stack

5.0
4.0
3.0
2.0
1.0

'b1124'

Lab-6

23-1-24

→ 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 internals 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 a student in all the courses

1] P.

// 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 scanner = new Scanner(System.in);
    System.out.print("Enter USN: ");
    usn = scanner.next();
    System.out.print("Enter Name: ");
```

```
name = scanner.next();
System.out.print("Enter Semester: ");
sem = scanner.nextInt();
}

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

// Internals.java

Package CIE;
import java.util.Scanner;
Public class Internals extends Student {
    Protected int marks[] = new int[5];

    Public Internals() {
        // constructor for Internals
    }

    Public void inputCIEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter Internal Marks for " + name);
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i+1) + " marks: ");
```

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

// Extends.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 setExternalMarks(){
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter the Semester for ...");
    for(int i=0; i<5; i++){
        System.out.println("Enter the ...");
        marks[i] = scanner.nextInt();
    }
  }
}
```

// ~~Test~~.

// Main.java

```
import SEE.External;

public class Main {

    public static void main (String args[]) {

        int numOfStudents = 2;

    External finalMarks[] = new External [numOfStudents];

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

        finalMarks[i] = new External();

        finalMarks[i].inputStudentDetails();

        System.out.println("Enter SEE Marks ");

        finalMarks[i].inputSEEMarks();

        }

    System.out.println("displaying data ");

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

            finalMarks[i].calculateFinalMarks();

            finalMarks[i].displayFinalMarks();

        }

    }

}
```

Q1:

Enter USN 1BM22CS281

Enter Name: dsikfjab

Enter Semester: 5

Enter CIE Marks

Enter Internal marks for ~~dsikfjab~~ dsikfjab

Subject 1 marks: 46

Subject 2 marks: 49

Subject 3 marks: 48

Subject 4 marks: 47

Subject 5 marks: 47

Enter SEE marks:

Enter SEE Marks for dsikfjab

Subject 1 marks: 68

Subject 2 marks: 77

Subject 3 marks: 89

Subject 4 marks: 99

Subject 5 marks: 100

30/11/24

Lab-7

30/1/24

→ write a program that demonstrate 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.

I/P:

```
class wrongAge extends Exception {
    Public WrongAge() {
        super("Age cannot be negative.");
    }
}
class Father {
    Private int age;
    Public father(int age) Throws WrongAge {
        if(age<0) {
            throw new wrongAge();
        }
        THIS.age = age;
    }
    Public int getAge() {
        return age;
    }
}
```

```
class Son extends Father{
    Private int SonAge;
    Public Son(int fatherAge, int sonAge) throws WrongAge{
        super(fatherAge);
        if (SonAge >= fatherAge){
            Throw new WrongAge();
        }
        this.SonAge = SonAge;
    }
    Public int getSonAge(){
        return SonAge;
    }
}
Public class Main{
    Public static void main(String[] args){
    try{
        Father father = new Father(45);
    System.out.println("father's age: " + father.getAge());
    Son son = new Son(45, 20);
    System.out.println("Son's age: " + son.getSonAge());
    Son invalidSon = new Son(45, 50)
System.out.println("Son's age: " + invalidSon.getSonAge());
    }
catch(WrongAge e){
```

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

o/p:

Father's age : 45

Son's age : 20

Error: Age cannot be negative

30/11/24

Lab-8

→ write a program which creates two threads, one thread displaying "BMS college of Engineering" once every ten seconds and another displaying "CSE" once every two seconds

```
Public class ThreadExample {

    Static class DisplayBMS implements Runnable {

        Public void run() {

            while (true){
            System.out.println("BMS College of Engineering");
        try {
            Thread.sleep(10000);
        }
        catch (InterruptException e) {
            e.printStackTrace();

        }
    }
}
static class DisplayCSE implements Runnable {

        public void run() {

            while (true){
                System.out.println("CSE");

        try {
            Thread.sleep(2000);
        } catch (InterruptedException e) {
                e.printStackTrace();
        }
```

```
}
}

public static void main (String[] args) {
    Thread bmsThread = new Thread (new DisplayBMS());
    Thread cseThread = new Thread (new DisplayCSE());

    bmsThread.start();
    cseThread.start();

  }
}
```

O/P:

BMS college of Engineering

CSE

CSE

CSE

CSE

CSE

BMS college of Engineering

CSE

}

{

lab -10

→ Demonstrate Inter process communication and deadlock

(i)

```
class Q {
int
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("In Intimate producer\n");
notify();
return n;
}
synchronized void put(int n){
while(valueSet)
try {
System.out.println("Producer waiting");
wait(); }
catch(InterruptedException caught");
```

```
}
    this.n = n;
    valueSet = true;
    System.out.println("Put: " + n);
    System.out.println("Notifying consumer");
    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<r){
      int x=q.get();
      System.out.println("consumed"+x);
      i++;
    }
  }
}
class Fixed{
  Public static void main(String args[]){
    Q q=new Q();
    new producer(q);
    new producer(q);
    System.out.println("press control-c to stop.");
  }
}
```

Put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
Got: 4

Put: 5
Got: 5

```
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, "Racing Thread");
t.start();
a.foo(b);
thread.
System.out.println("Back in main");
}
```

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

o/p

```
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
inside A.last
Back in main Thread
RacingThread trying to call A.last()
inside A.last
Back in other Thread
```

13/02/24

Lab-9

Q.) Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text field, Num1 and Num2. The division of Num1 by 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("Divide app");
        jfrm.setTitle("Divided app");
        jfrm.setSize(275,150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JLabel jlab = new JLabel("Enter the divide and dividend");
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);
```

```
JButton button = new JButton("calculate")

JLabel ajlb = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();
jfrm.add(ajlb);
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

ActionListner 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=a1b;
ala b.setText(" \nA=" +a)
 bla b.setText ("\nB="+b);
 ansla b.setTextf ("\nAns="+ans);
}
catch (NumberFormatException e){
        ala b.setText(" ");
        bla b.setText(" ");
        ansla b.setText(" ");
        err.setText(" Enter only Integers! ");
      }
    catch (ArithmeticException e){
          ala b.setText(" ");
          bla b.setText(" ");
          ansla b.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 CalculatorGUI();
  }
});
}
}
```

o/p:

Enter The divider and dividend

[868] [2]

[Calculate] A=868 B=2 Ans: 434

JFrame: represent main windows of application

JLabel: used to display text labels on GUI

add() this is used to add a swing component (button, label, text filed) to a container (JFrame)

setText(text): It is used to set the text of text based component (such as JLabel) to specified string

getText(): it retrieves text from text based component & returns the represented String representing current text.

addActionListener(ActionListener listener)

This method adds action listener to the component that generates action events (JButton).

setSize(int height, int width): It is used to set size of components such as JFrame to specified height & width in pixels

setLayout(LayoutManager layout): It sets layout manager for container responsible for arranging elements inside it

30/2/24

lab 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 * (double) a);
         System.out.println("Roots are real and equal");
         System.out.println("Root1 = Root2 = " + r1);
      } else if (d > 0) {
```

```
            r1 = ((-b) + (Math.sqrt(d))) / (2 * (double) a);
            r2 = ((-b) - (Math.sqrt(d))) / (2 * (double) 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 * (double) a);
            r2 = Math.sqrt(-d) / (2 * (double) a);
            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();
        System.out.println("Name: Sirigireddy Pranav Reddy\nUSN: 1BM22CS281");
    }
}
```

Lab 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 marks;
    int credits;
    int points; // Added field for points
}

class Student {
    Subject[] subjects;
    String name;
    String usn;
    double sgpa;
    Scanner sc;

    public Student() {
        int i;
        subjects = new Subject[9];
        for (i = 0; i < 9; i++)
            subjects[i] = new Subject();
        sc = new Scanner(System.in);
    }

    public void inputDetails() {
        System.out.print("Enter student name: ");
        name = sc.nextLine();
        System.out.print("Enter student USN: ");
        usn = sc.nextLine();

        for (int i = 0; i < 9; i++) {
            subjects[i] = new Subject();
            System.out.println("Enter details for subject " + (i + 1));
            System.out.print("Enter marks for the subject: ");
            subjects[i].marks = sc.nextInt();
```

```
        System.out.print("Enter credits for the subject: ");
        subjects[i].credits = sc.nextInt();
    }
}

private int calculatePoints(int marks) {
    if (marks > 90) {
        return 10;
    } else if (marks > 80) {
        return 9;
    } else if (marks > 70) {
        return 8;
    } else if (marks > 60) {
        return 7;
    } else if (marks > 50) {
        return 6;
    } else {
        return 0; // You can modify this based on your specific grading system
    }
}

public void calculateSGPA() {
    double totalCredits = 0;
    double weightedSum = 0;

    for (int i = 0; i < 9; i++) {
        totalCredits += subjects[i].credits;
        subjects[i].points = calculatePoints(subjects[i].marks);
        weightedSum += subjects[i].credits * subjects[i].points;
    }
```

```
        sgpa = weightedSum / totalCredits;
    }

    public void displayResult() {
        System.out.println("Student Name: " + name);
        System.out.println("Student USN: " + usn);
        System.out.println("SGPA: " + sgpa);
    }
}

public class Main {
    public static void main(String[] args) {
        Student student = new Student();
        student.inputDetails();
        student.calculateSGPA();
        student.displayResult();
    }
}
```

Lab 3

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

```
import java.util.Scanner;

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

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

```
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

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

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

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

Lab 4

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

```
import java.util.Scanner;

class InputScanner {
    Scanner s;

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

abstract class shape extends InputScanner {
    double a;
    double b;
    double r;

    abstract void getInput();

    abstract void DisplayArea();
}
```

```
class Rectangle extends shape {
    void getInput() {
        System.out.println("Enter the sides of the rectangle");
        a = s.nextDouble();
        b = s.nextDouble();
    }

    void DisplayArea() {
        System.out.println("The area of the rectangle is " + a * b + "");
    }
}

class Triangle extends shape {
    void getInput() {
        System.out.println("Enter the sides of the triangle");
        a = s.nextDouble();
        b = s.nextDouble();
    }

    void DisplayArea() {
        System.out.println("The area of the triangle is " + (0.5 * a * b) + "");
    }
}

class Circle extends shape {
    void getInput() {
        System.out.println("Enter the radius of the circle");
        r = s.nextDouble();
    }
```

```
    void DisplayArea() {
        System.out.println("The area of the circle  is " + 3.14 * r * r + "");
    }
}

class Shapemain {
    public static void main(String args[]) {
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
        r.getInput();
        r.DisplayArea();
        t.getInput();
        t.DisplayArea();
        c.getInput();
        c.DisplayArea();
        System.out.println("Sohan A R-1BM22CS285");

    }
}
/*
 * java -cp /tmp/qYrVKCZE0d Shapemain
 * Enter the sides of the rectangle
 * 6
 * 3
 * The area of the rectangle is 18.0
 * Enter the sides of the triangle
 * 5
 * 3
 * The area of the triangle is 7.5
 * Enter the radius of the circle
```

```
* 1
* The area of the circle is 3.14
* Sohan A R-1BM22CS285
*
*/
```

Lab 5

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

a)Accept deposit from customer and update the balance.

b)Display the balance.

c)Compute and deposit interest

d)Permit withdrawal and update the balance Check for the minimum balance, impose penalty if necessary and update the balance

```
import java.util.Scanner;


class account {
   String name;
   int accno;
   String type;
   double balance;


   account(String name, int accno, String type, double balance) {
      this.name = name;
      this.accno = accno;
      this.type = type;
      this.balance = balance;
   }
```

```
    void deposit(double amount) {
        balance += amount;
    }

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

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

class savAcct extends account {

    private static double rate = 5;

    savAcct(String name, int accno, double balance) {
        super(name, accno, "savings", balance);

    }

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

```
}

class curAcct extends account {

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

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

   }

   void checkmin() {

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

   }

}

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

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

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

        case 3:
            ca.display();
            break;
        case 4:
            System.exit(0);
        default:
```

```
                System.out.println("invalid input");
                break;
            }
        }
    }

  }
}
/*
 * java -cp /tmp/qYrVKCZE0d accountMain
 *
 * enter the name :
 * pranav
 * enter the type(current/savings):
 * current
 * enter the account number:
 * 654654
 * enter the intial balance:
 * 100000
 * Menu
 * 1.deposit 2.withdraw 3.display
 * enter the choice:
 * 1
 * enter the amount:
 * 25000
 * Menu
 * 1.deposit 2.withdraw 3.display
 * enter the choice:
 * 2
 * enter the amount:
 * 15000
```

```
* Menu
* 1.deposit 2.withdraw 3.display
* enter the choice:3
* name:pranavaccno:654654type:currentbalance:110000.0
*
* Menu
* 1.deposit 2.withdraw 3.displayenter the choice:
*
*/
```

Lab 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 a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

```
// Student.java
package CIE;

import java.util.Scanner;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    public void inputStudentDetails() {
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.print("Enter USN: ");
        usn = scanner.next();
        System.out.print("Enter Name: ");
        name = scanner.next();
        System.out.print("Enter Semester: ");
        sem = scanner.nextInt();
    }

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

// Internals.java
package CIE;

import java.util.Scanner;

public class Internals extends Student {
    protected int marks[] = new int[5];

    public Internals() {
        // Constructor for Internals
    }

    public void inputCIEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter Internal Marks for " + name);
        for (int i = 0; i < 5; i++) {
```

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

// Externals.java
package SEE;

import CIE.Internals;

import java.util.Scanner;

public class Externals extends Internals {
    protected int marks[] = new int[5];
    protected int finalMarks[] = new int[5];

    public Externals() {
    }

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

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

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

            System.out.println("Displaying data:\n");

            for (int i = 0; i < numOfStudents; i++) {
                finalMarks[i].calculateFinalMarks();
```

```
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

Lab 7

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age=father’s age

```
import java.util.Scanner;


// Custom exception class
class WrongAge extends Exception {
    // Parameterized constructor with user-defined message
    public WrongAge(String message) {
        super(message);
    }
}


// Base class
class Father {
    private int fatherAge;


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

```
    // Method to display father's age
    public void display() {
        System.out.println("Father's age: " + fatherAge);
    }
}

// Derived class
class Son extends Father {
    private int sonAge;

    // Constructor for Son class
    public Son() throws WrongAge {
        super(); // Call to superclass constructor
        Scanner s = new Scanner(System.in);
        System.out.print("Enter son's age: ");
        sonAge = s.nextInt();
        if (sonAge >= super.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");
    }

    // Method to display son's age
    public void display() {
        super.display(); // Call to superclass method
        System.out.println("Son's age: " + sonAge);
    }
}

// Main class
```

```
public class ExceptionHandlingInheritance {
    public static void main(String[] args) {
        try {
            // Creating Son object
            Son son = new Son();
            // Displaying son's age
            son.display();
        } catch (WrongAge e) {
            // Handling the custom exception
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Lab 8

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

```
class CollegeRunnable implements Runnable {
    public void run() {
        while (true) {
            try {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000); // Sleep for 10 seconds
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

class DepartmentRunnable implements Runnable {
```

```
    public void run() {
        while (true) {
            try {
                System.out.println("CSE");
                Thread.sleep(2000); // Sleep for 2 seconds
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

public class java_lab_8 {
    public static void main(String[] args) {
        // Creating instances of runnables
        Runnable collegeRunnable = new CollegeRunnable();
        Runnable departmentRunnable = new DepartmentRunnable();

        // Creating threads using runnables
        Thread collegeThread = new Thread(collegeRunnable);
        Thread departmentThread = new Thread(departmentRunnable);

        // Starting threads
        collegeThread.start();
        departmentThread.start();
    }
}
```

Lab 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() {

      // create jframe container

      JFrame jfrm = new JFrame("Divider App");

      jfrm.setSize(300, 200);

      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 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!");
            }
        }
    });

    // 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 SwingDemo();
      }
    });
  }
}
```

Lab 10

10.A)Demonstrate Inter process Communication and deadlock

```
class Q {
  int n;

  boolean valueSet = false;
  synchronized int get() {
```

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

```
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run() {
        int i = 0;
        while (i < 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 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.");
```

```
    }
}
```

10B)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("M
    ainThread");

    Thread t = new Thread(this,
    "RacingThread");

    t.start();

    a.foo(b); // get lock on a in this
    thread.

    System.out.println("Back in main
```

```
    thread");

    }

    public void run() {

    b.bar(a); // get lock on b in other
    thread.

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

    }

    public static void main(String args[]) {

      new Deadlock();

    }

}
```