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:

Rachana N

(2023BMS02608)

Department of Computer Science and Engineering,

B.M.S College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560019

2023-2024.

INDEX

Sl.No.	Title Date	Complete scanned Observation Book
1	Complete scanned Observation Book	12/12/2023 - 20/02/2024
2	Lab 1	12/12/2023
3	Lab 2	19/12/2023
4	Lab 3	26/12/2023
5	Lab 4	02/01/2024
6	Lab 5	09/01/2024
7	Lab 6	16/01/2024
8	Lab 7	23/01/2024
9	Lab 8	30/01/2024
10	Lab 9	06/02/2024
11	Lab 10	20/02/2024

1) Quadratic :- Week 1

```
import java.util.Scanner;
class Quadratic
{
 int a, b, c;
 double r1, r2, d;
 void getId()
 {
  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 & equal");
   System.out.println("Root 1 = Root2 = " + r1);
  }
  else if(d>0){
   r1 = ((-b) + (Math.sqrt(d)))/(double)(2*a);
   r2 = ((-b) - (Math.sqrt(a)))/(double)(2*a);
   System.out.println("Root 1 = " + r1 + "Root 2 = " + r2);
  }
  else if(d<0)
  {
   System.out.println("Roots are imaginary");
   r1 = (-b)/(2*a);
   r2 = Math.sqrt(-d)/(2*a);
   System.out.println("Root1 = " + r1 + " + i " + r2);
   System.out.println("Root1 = " + r1 + " - i " + r2);
  }
 }
}
class quadraticMain{
 public static void main(String args[]){
  Quadratic q = new Quadratic();
  q.getId();
  q.compute();
  System.out.println("2023BMS02608");
 }
}
```

```
import java.util.Scanner;

class Subject {
    int subjectMarks;
    int credits;
    int grades;
}
class Student {
    String name;
    String usn;
    double SGPA;
    Scanner s;
    Subject[] subjects;

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

    void getStudentDetails() {
        System.out.println("Enter student name:");
        name = s.nextLine();
        System.out.println("Enter student usn:");
        usn = s.nextLine();
    }
    void getMarks() {
        for (int i = 0; i<9; i++) {
        System.out.println("Enter marks subject " + (i+1) + ":");
        subjects[i].subjectMarks = s.nextInt();

        subjects[i].credits = 3;

        if(subjects[i].subjectMarks >= 90) {
            subjects[i].grade = 10;
        } else if (subjects[i].subjectMarks >= 75) {
            subjects[i].grade = 9;
        } else if (subjects[i].subjectMarks >= 60) {
            subjects[i].grade = 8;
        } else if (subjects[i].subjectMarks >= 50) {
            subjects[i].grade = 7;
```

```
} else if (subjects[i].subjectMarks >= 40) {
        subjects[i].grade = 6;
    } else {
        subjects[i].grade = 0;
    }
  }
}

void computeSGPA() {
    double totalCredits = 0;
    double totalGradePoints = 0;

    for (int i=0; i<9; i++) {
        totalGradePoints += subjects[i].grade * subjects[i].credits;
        totalCredits += subjects[i].credits;
    }
    SGPA = totalGradePoints / totalCredits;
  }
}

public class Main {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.getMarks();
        s1.computeSGPA();

        System.out.println("\nStudent Details:");
        System.out.println("Name : " + s1.name);
        System.out.println("USN : " + s1.usn);
        System.out.println("SGPA : " + s1.SGPA);
    }
}
```

Output :-

```
Enter student name:
Prachana
Enter student USN:
609
Enter marks for subject 1:
95
Enter marks for subject 2:
65
Enter marks for subject 3:
92
Enter marks for subject 4:
81
Enter marks for subject 5:
90
Enter marks for subject 6:
89
Enter marks for subject 7:
88
Enter marks for subject 8:
82
```

Enter marks for subject 9:
95

Student Details:
Name: rachana
USN: 608
SGPA: 9.333333334

Week 3: 19/12/23

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 books objects.

26/12/23

→

```
import java.util.Scanner;
class Book{
String name;
String author;
int price;
int numPages;
public 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 bookDetails="Book name:"+this.name+"\n"
 +"Author name:"+this.author+"\n"
 +"price:"+this.price+"\n"
 +"Number of pages:"+this.numPages+"\n";
 return bookDetails;
 }
}
class Main{
public static void main(String[] args){
Scanner scanner=new Scanner(System.in);
System.out.print("enter the number of books:");
```

```
int n = scanner.nextInt();
Book[] books = new Book[n];
for(int i=0; i<n; i++){
System.out.print("Enter name of the book:");
String name = scanner.next();
System.out.print("enter author of the book:");
String author = scanner.next();

System.out.println("enter price of the book:");
int price = scanner.nextInt();

System.out.print("enter the number of pages:");
int numPages = scanner.nextInt();

books[i] = new Book(name, author, price, numPages);
}
System.out.println("\n Book Details:");
for(int i=0; i<n; i++){
System.out.println("Book" + (i+1)+":\n" + books[i]);
}
}
}
```

output :-

```
enter the number of books: 2
enter name of the book: ooj
enter author of the book: jhon
enter price of the book: 2300
enter the number of pages: 100
enter the name of the book: dst
enter author of the book: von
enter price of the book: 457
enter the number of pages: 120
```

```
Book Details:
Book1:
Book name: ooj
Author name: jhon
price: 2300
Number of pages: 100
Book2:
Book name: dst
Author name: von
price: 457
Number of pages: 120
```

Week 4

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, and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given shape.

=>

```
import java.util.Scanner;
abstract class Shape {
    int a, b;
    Shape(int a, int b){
        this.a = a;
        this.b = b;
    }
    abstract void printArea();
}
class Rectangle extends Shape {
    Rectangle(int a, int b) {
        super(a, b);
    }
    void printArea() {
        System.out.println("Area of Triangle:" + (0.5 * a * b));
    }
}
class Triangle extends Shape {
    Triangle(int a, int b) {
        super(a, b);
    }
    void printArea() {
        System.out.println("Area of Circle:" + (3.14159265358979 * a * a));
    }
}
class Circle extends Shape{
    Circle(int a, int b) {
        super(a, b);
    }
    void printArea() {
        System.out.println("Area of Circle:" + (3.14159265358979 * a * a));
    }
}
```

```
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter length & breadth of Rectangle:");
        int length = scanner.nextInt();
        int breadth = scanner.nextInt();
        Rectangle rectangle = new Rectangle(length, breadth);
        rectangle.printArea();

        System.out.println("Enter base and height of Triangle:");
        int base = scanner.nextInt();
        int height = scanner.nextInt();
        Triangle triangle = new Triangle(base, height);
        triangle.printArea();

        System.out.println("Enter radius of Circle:");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius, radius);
        circle.printArea();
    }
}
```

output:

```
Enter length & breadth of Rectangle:
5 2
Area of Rectangle: 10
Enter base and height of Triangle:
6 2
Area of Triangle: 6.0
Enter radius of Circle:
10
Area of Circle: 314.1592653589793
```

2/1/24

Week 5

9/1/24

1) Develop a java program to create a class Bank that maintains two kinds of account for its customers, one called saving account and the other current account. The saving 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 & update the balance.
b) display the balance
c) Compute and deposit interest
d) permit withdrawal & update the balance
e) check the minimum balance, impose penalty if necessary & update the balance.

→

```
import java.util.Scanner;

class BankAccount {
    protected String customerName;
    protected int accountNumber;
    protected double balance;
public BankAccount(String customerName, int accountNumber, double
                                                    balance) {

        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.balance = balance;
    }
public void displayAccountDetails() {
        System.out.println("customer name:" + customerName);
        System.out.println("Account number:" + accountNumber);
        System.out.println("Type of Account: Savings Account");
        System.out.println("balance - " + balance);
    }
}
class SavingsAccount extends BankAccount {
    private double interestRate;
public SavingsAccount(String customerName, int accountNumber,
    double balance, double interestRate) {
            super(customerName, accountNumber, balance);
            this.interestRate = interestRate;
    }
public void deposit(double amount) {
            balance += amount;
public void withdraw(double amount) {
            balance -= amount;
public void computeInterest() {
            balance += balance * interestRate / 100;
public void displayAccountDetails() {
        super.displayAccountDetails();
        System.out.println("Interest Rate " + interestRate + "%");
    }
}
class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        SavingsAccount[] accounts = new SavingsAccount[2];
        int choice;
        String customerName;
```

```
int accountNumber;
double balance;
double withdrawalAmount;
double interestRate;

for (int i=0;  i<2;  ;++){
    System.out.print("Enter customer name: ");
    customerName = scanner.next();
    System.out.print(" Enter account Number ");
    accountNumber = scanner.nextInt();
    System.out.print("Enter the deposit amount: ");
    balance = scanner.nextDouble();
    accounts[i] = new SavingsAccount(customerName,
    accountNumber, balance, 2);
}
do {
    System.out.println("\n ---MENU ---- ");
    System.out.println(" 1. Deposit");
    System.out.println("2. withdraw")
    System.out.println("3. compute interest for Savings
                        Account");
    System.out.println("4. Display account details");
    System.out.println("5. Exit");
    System.out.print("Enter your choice");
    choice = scanner.nextInt();

switch (choice){

    case 1:
        System.out.print("Enter the type of account:");
        String accountType = scanner.next();
        System.out.print("Enter the deposit amount:")
        depositAmount = scanner.nextDouble();
        accounts[0].deposit(depositAmount);
        break;
    case 2:
        System.out.println("Enter the type of account:");
        accountType = scanner.next();
        System.out.print("Enter the withdrawal amount:");
        withdrawalAmount = scanner.nextDouble();
        accounts[0].withdraw(withdrawalAmount);
        break;
case 3:
        accounts[0].computeInterest();
        break;
```

```
case 4:
    accounts[0].displayAccountDetails();
    break;
case 5:
    System.out.println("Exiting...");
    break;
default:
    System.out.println("Invalid choice. Please try again.");
}
} while (choice != 5);
scanner.close();
}
}
```

o/p:

```
Enter customer name: Rachana
Enter account Number: 112
Enter the deposit amount: 20000
Enter customer name: Prakruthi
Enter account Number: 213
Enter the deposit amount: 30000
--- MENU ---
1. Deposit
2. withdraw
3. compute interest for saving Account
4. Display account details.
5. Exit
Enter your choice 1
Enter the type of account: savings
Enter the deposit amnt: 30000
--- MENU ---
1. Deposit
2. withdraw
3. Compute interest for savings Account
4. Display account details
5. Exit
Enter your choice 3
--MENU ---
1. Deposit
2. withdraw
3. Compute interest for savings Account
4. Display account details
5. Exit
Enter your choice = 4
Customer name: Rachana
Account number: 112
Type of Account: savings account
balance = 53040.0
Interest Rate: 2.0%
```

09.01.24

Week 6

lab6

23/1/24

Student.java

```
package CIE;
import java.util.Scanner;
class Student{
protected String usn = new String();
protected String name = new String();
protected int sem;
public void inputStudentDetails()
{
Scanner s = new Scanner(System.in);
System.out.println("Enter the usn");
usn = s.next(); System.out.println("Enter the name"); name = s.next();
System.out.println("Enter the Semester");
sem = s.nextInt();
}
public void displayStudentDetails()
{
System.out.println("USN:" + usn);
System.out.println("Name IS :" + name);
System.out.println("Semester" + sem);
}
}
```

complete

23-01-24

Internals.java

```
package CIE;
import java.util.Scanner;
public class Internals extends Student
{
protected int marks[] = new int[5];
public void inputCIEmarks()
{
Scanner s = new Scanner(System.in);
System.out.println("Enter the marks of 5 subjects:");
for(int i=0; i<5; i++)
{
System.out.println("subject" + (i+1) + ":");
marks[i] = s.nextInt();
}
}}
```

Externals.java

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals
{
    protected int marks[];
    protected int finalMarks[];

    public Externals()
    {
        marks = new int[5];
        finalMarks = new int[5];
    }
    public void inputSEEmarks()
    {
        Scanner s = new Scanner(System.in);
        for(int i=0; i<5; i++)
        {
            System.out.print("Subject" + (i+1) + " marks: ");
            marks[i] = s.nextInt();
        }
    }
public void calculateFinalMarks()
    {
        for(int i=0; i<5; i++)
    finalMarks[i] = marks[i]/2 + super.marks[i];
    }
public void displayfinalMarks()
    {
        displayStudentDetails();
        for(int i=0; i<5; i++)
        System.out.println("Subject" + (i+1) + ": " + finalMarks[i]);
    }
}
```

Main.java

```
import SEE.Externals;
class Main {
  public static void main(String args[])
    {
     int numofStudents = 2;
    Externals finalMarks[] = new Externals[numofStudents];
    for(int i=0; i<numofStudents; i++)
     {
      finalMarks[i] = new Externals();
      finalMarks[i].inputStudentDetails();
      System.out.println("Enter CIE marks");
      finalMarks[i].inputCIEmarks();
      System.out.println("Enter SEE marks");
      finalMarks[i].inputSEEmarks();
     }
   System.out.println("Displaying data :\n");
   for(int i=0; i<numofStudents; i++)
     {
      finalMarks[i].calculateFinalMarks();
      finalMarks[i].displayFinalMarks();
     }
   }
}
```

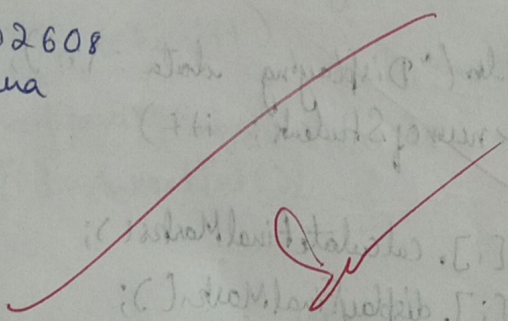
O/P:

```
enter the usn
BMSC2608
enter the name
Rachana
enter the semester
3
Enter CIE marks
Enter internal marks of CIE: 1
67
Enter internal marks of CIE: 2
30
Enter internal marks of CIE: 3
81
Enter internal marks of CIE: 4
98
Enter internals marks of CIE: 5
78
```

```
Enter SEE marks
Subject 1 marks: 78
subject 2 marks: 68
Subject 3 marks: 68
Subject 4 marks: 35
Subject 5 marks: 57
```

```
usn : BMSC2608
Name is : Rachana
semester 3
Subject 1 marks :
Subj
```

```
enter the usn
2023BMS02608
enter the name
Rachana
Enter the semester
3
Enter CIE marks
enter the marks for 5 subjects:
subject1: 24
subject2: 27
subject3: 16
subject4: 28
subject5: 29
Enter SEE marks
Subject 1 marks: 89
        2 marks: 98
        3 marks: 88
        4 marks: 87
        5 marks: 86

USN: 2023BMS02608
NAME IS: Rachana
Semester 3
Subject 1: 68
Subject 2: 76
Subject 3: 60
Subject 4: 71
Subject 5: 72
```



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

Lab program 7

30/1/24

```
import java.util.Scanner;
class WrongAge extends Exception
    {
     public WrongAge(String message)
            {
              super(message);
            }
    }
class Father
    {
      int fatherAge;

      Father() throws WrongAge
            {
              Scanner s = new Scanner(System.in);
              System.out.println("Enter father's age: ");
              fatherAge = s.nextInt();
              if(fatherAge < 0)
                  {
                    throw new WrongAge("Age cannot be negative");
                  }
            }
          void display()
              {
                System.out.println("Father's age is:" + fatherAge);
              }
    }
class Son extends Father
    {
      int sonAge;
      son() throws WrongAge
            {
              super();
              Scanner s = new Scanner(System.in);
              System.out.println("Enter son's age: ");
              sonAge = s.nextInt();
```

```
        if (sonAge >= fatherAge)
          {
            throw new WrongAge("Son's age cannot be greater
                                than father's age");
          }
        else if (sonAge < 0)
          {
            throw new WrongAge("Age cannot be negative");
          }
      }
    void display()
      {
        super.display();
        System.out.println("son's age is :" + sonAge);
      }
}

class Main
  {
    public static void main(String[] args)
      {
        try
        {
          Son s = new Son();
          s.display();
        }
        catch (WrongAge e)
          {
            System.out.println(e.getMessage());
          }
      }
}
```

o/p :

```
Enter Father's age :
40
Enter Son's age :
20
Father's age is : 40
Son's age is : 20

Enter Father's age :
20
Enter Son's age :
40
Son's age cannot be greater than father's age

Enter Father's age :
-23
Age cannot be negative

Enter father's age :
40
Enter Son's age :
-34
Age cannot be negative
```

30.01.24

6/2/24

Lab program 8:

write a program which creates two threads, One thread displaying "BMS college of Engineering" once every ten seconds & another displaying "CSE" once every two seconds.

```
class College extends Thread
  {
    public void run()
      {
        for(int i=0;  i<5;  i++)
          {
            System.out.println("BMS college of Engineering");
            try {
                Thread.sleep(10000);
                }
            catch(InterruptedException e)
              {
                e.printStackTrace();
              }
          }
      }
  }
class Dept extends Thread
  {
    public void run()
      {
        for(int i=1;  i<=25;  i++)
          {
            System.out.println("CSE");
              try{
                  Thread.sleep(2000);
                 }
            catch(InterruptedException e)
              {
                e.printStackTrace();
              }
          }
      }
```

```
class Main {
    public static void main(String args[])
    {
        College c1 = new College();
        c1.start();
        Dept d1 = new Dept();
        d1.start();
    }
}
```

o/p:

```
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
```

30.02.24

Week 9 20/2/24

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

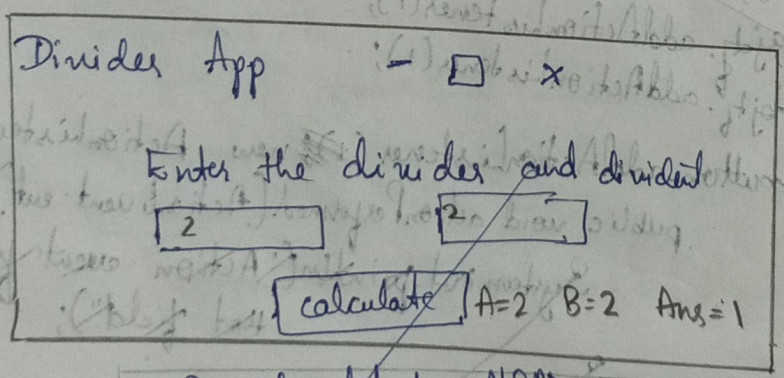
→

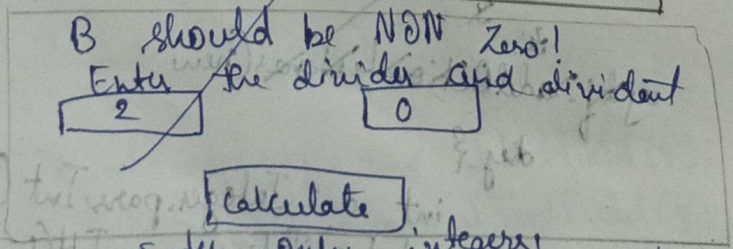
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo{
    SwingDemo(){
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275,150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

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

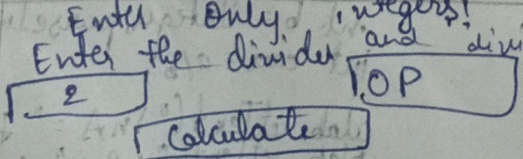
```
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();

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

ActionListener l = new ActionListener()
{
public void actionPerformed(ActionEvent ent)
{
System.out.println("Action event from a text field");
}
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener(){
public void actionPerformed(ActionEvent ent){
System.out.println("Action event from a
text field");
}
});
ajtf.addActionListener(new
try{
int a = Integer.parseInt(ajtf.getText());
int b = Integer.parseInt(bjtf.getInt());
int ans = a/b;
alab.setText("InA = " + a);
blab.setText("InB = " + b);
anslab.setText("In Ans = "+ans);
```

```
catch (ArithmeticException e){
        alab.setText(" ");
        blab.setText(" ");
        anslab.setText(" ");
        err.setText("B should be NON Zero!");
      }
    }
});

    jfrm.setVisible(true);
}

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

output:







1) JFrame: 'JFrame' is a class defined in the 'javax.swing' package. It represents a window with decorations such as a title, borders, and buttons for closing, minimizing, and maximizing. It serves as a fundamental component for building graphical user interface (GUIs) in Java Swing.

2) setSize(int width, int height):
This method is used to set the size of a component, such as a 'JFrame', in pixels. It takes two parameters: the width & height of the component in pixels.

3) setLayout(LayoutManager layout):
This method sets the layout manager for a container, such as a 'JFrame'. A layout manager is responsible for arranging the components within the container. The 'setLayout' method allows you to specify which layout manager to use. Common layout managers include 'FlowLayout', 'BorderLayout', 'GridLayout', and 'GridBagLayout'.

4) setDefaultCloseOperation(int operation):
This method sets the default operation that will be performed when the user closes the window using the close button. The 'operation' parameter specifies the action to take. Common operations include 'EXIT_ON_CLOSE', 'DISPOSE_ON_CLOSE', 'HIDE_ON_CLOSE', and 'DO_NOTHING_ON_CLOSE'.

5) JLabel: 'JLabel' is a class in the 'javax.swing' package that represents a display area for a short text string or an image, or both. It is commonly used to display text or icons on a GUI.

6) JTextField: 'JTextField' is a class in the 'javax.swing' package that allows editing of a single line of text. It provides a text input box where the users can enter or edit text.

7) addActionListener(ActionListener listener):

This method adds an action listener to a component. An action listener is notified whenever the component generates an action event, such as when a button is pressed. By adding an action listener, you can define the actions to be performed in response to such events.

8) setText(String text):

This method sets the text of a 'JLabel', 'JTextField' or similar text-based component to the specified string 'text'. It updates the content displayed by the component to the new text provided.

Week 10

2)

```
class Q
{
  int n;
  boolean valueSet = false;
  synchronized int get()
  {
    while(!valueset)
    try {
      System.out.println("Consumer waiting\n");
      wait();
    }
    Catch(InterruptedException e)
    {
      Syste.out.println("InterruptedException caught");
    }
    System.out.println("Got:" + n);
    valueSet = false;
    Syte.out.println("Intimate Producer");
    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("Intime Consumes");
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.");
    }
}
```

o/p:

Press Control-C to stop.

Put: 0
Intimate Consumer
Producer waiting
Got: 0
Intimate Producer
.....
put: 14
Intimate Consumer
Got: 14
Intimate Producer
Consumed: 14

3) 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();
}
}
```

o/p:

RacingThread entered B.bar
MainThread entered A.foo
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

WEEK 1

1. **Develop a Java program that prints all real solutions to the quadratic equation ax2+bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions.**

Program:

```
import java.util.Scanner;

class Quadratic {
   int a, b, c;
   double r1, r2, d;

   void getCoefficients() {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter the coefficients of a, b, and c:");
      a = scanner.nextInt();
      b = scanner.nextInt();
      c = scanner.nextInt();
   }

   void computeRoots() {
      while (a == 0) {
         System.out.println("Not a quadratic equation");
         System.out.println("Enter a non-zero value for a:");
         Scanner scanner = new Scanner(System.in);
         a = scanner.nextInt();
      }

      d = b * b - 4 * a * c;

      if (d == 0) {
         r1 = -b / (2.0 * a);
         System.out.println("Roots are real and equal");
         System.out.println("Root1 = Root2 = " + r1);
      } else if (d > 0) {
         r1 = (-b + Math.sqrt(d)) / (2.0 * a);
```

```
            r2 = (-b - Math.sqrt(d)) / (2.0 * a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root1 = " + r1 + " Root2 = " + r2);
        } else {
            System.out.println("Roots are imaginary");
            r1 = -b / (2.0 * a);
            r2 = Math.sqrt(-d) / (2.0 * a);
            System.out.println("Root1 = " + r1 + " + i" + r2);
            System.out.println("Root2 = " + r1 + " - i" + r2);
        }
    }
}

class QuadraticMain {
    public static void main(String[] args) {
        System.out.println("My Name is Rachana.N");
        System.out.println("My USN is 2023BMS02608");
        Quadratic quadratic = new Quadratic();
        quadratic.getCoefficients();
        quadratic.computeRoots();
    }
}
```

WEEK2

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.

Program:

```
import java.util.Scanner;
class Subject
{
int subjectmarks;
int credits;
int grade;
}
class Student
{
String name;
String usn;
double SGPA;
Scanner s;
Subject[] subjects;

Student()
{
int i;
subjects=new Subject[9];
for (i=0; i<9; i++)
{
subjects[i]=new Subject();
}
s=new Scanner(System.in);
}
void getStudentdetails()
{
System.out.println("Enter student name:");
name=s.nextLine();
System.out.println("Enter Student usn:");
usn=s.nextLine();
}
void getMarks(){
for(int i=0; i<9; i++)
{
```

```
System.out.println("Enter marks for subject"+(i+1)+":");
subjects[i].subjectmarks=s.nextInt();
subjects[i].credits=4;
if(subjects[i].subjectmarks>=90){
subjects[i].grade=10;}
else if(subjects[i].subjectmarks>=75){
subjects[i].grade=9;}
else if(subjects[i].subjectmarks>=60){
subjects[i].grade=8;}
else if(subjects[i].subjectmarks>=50){
subjects[i].grade=7;}
else if(subjects[i].subjectmarks>=40){
subjects[i].grade=6;}
else{
subjects[i].grade=0;
}
}
}
void computeSGPA()
{
double totalcredits=0;
double totalgradepoints=0;
for(int i=0; i<9; i++){
totalgradepoints += subjects[i].grade*subjects[i].credits;
totalcredits += subjects[i].credits;
}
SGPA= totalgradepoints/totalcredits;
}
}
class Main
{
public static void main(String args[])
{
Student s2=new Student();
s2.getStudentdetails();
s2.getMarks();
s2.computeSGPA();

System.out.println("\n Student details");
System.out.println("Name:"+s2.name);
System.out.println("usn:"+s2.usn);
System.out.println("SGPA:"+s2.SGPA);
}
}
```

WEEK3

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.

Program:

```
import java.util.Scanner;

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

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

public class Main{

   public  static  void  main(String[]  args)  {
      Scanner scanner = new Scanner(System.in);

      System.out.print("Enter the number of books: ");
```

```
        int n = scanner.nextInt();

        Book[] books = new Book[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter name of the book: ");
            scanner.nextLine(); // consume the newline character
            String name = scanner.nextLine();

            System.out.print("Enter author of the book: ");
            String author = scanner.nextLine();

            System.out.print("Enter the price of the book: ");
            int price = scanner.nextInt();

            System.out.print("Enter the number of pages of the book: ");
            int numPages = scanner.nextInt();

            books[i] = new Book(name, author, price, numPages);
        }

        System.out.println("\nBook Details:");
        for (int i = 0; i < n; i++) {
            System.out.println("Book " + (i + 1) + ":\n" + books[i]);
        }
    }
}
```

WEEK4

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.

Program:

```
import java.util.Scanner;

abstract class Shape {
int a, b;

Shape(int a, int b)
{
        this.a = a;
        this.b = b;
}

abstract void printArea();
}

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

        void printArea()
        {
        System.out.println("Area of Rectangle: " + (a * b));
        }
}

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

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

class Circle extends Shape
{
        Circle(int a, int b)
        {
        super(a, b);
        }

void printArea()
         {
        System.out.println("Area of Circle: " + (3.141592653589793 * a * a));
        }
}

class Main
{
public static void main(String[] args)
      {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter length and breadth of Rectangle: ");
        int length = scanner.nextInt();
        int breadth = scanner.nextInt();

        Rectangle rectangle = new Rectangle(length, breadth);
        rectangle.printArea();

        System.out.println("Enter base and height of Triangle: ");
        int base = scanner.nextInt();
        int height = scanner.nextInt();
        Triangle triangle = new Triangle(base, height);
        triangle.printArea();
        System.out.println("Enter radius of Circle: ");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius, radius);
        circle.printArea();
        }
}
```

WEEK5

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.

Program:

```
import java.util.Scanner;

abstract class Account {
   String customerName;
   long accountNumber;
   String accountType;
   double balance;

   public Account(String customerName, long accountNumber, String
accountType, double balance) {
      this.customerName = customerName;
      this.accountNumber = accountNumber;
      this.accountType = accountType;
      this.balance = balance;
   }

   public void displayBalance() {
      System.out.println("Account Balance: $" + balance);
   }
```

```
    // Abstract method for withdrawal
    public abstract void withdraw(double amount);
}

class CurrAcct extends Account {
    double minimumBalance;
    double serviceCharge;

    public CurrAcct(String customerName, long accountNumber, double balance) {
        super(customerName, accountNumber, "Current Account", balance);
        this.minimumBalance = 1000; // Set minimum balance
        this.serviceCharge = 50;   // Set service charge
    }

    @Override
    public void withdraw(double amount) {
        if (balance - amount >= minimumBalance) {
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining balance: $" +
balance);
        } else {
            System.out.println("Insufficient funds. Service charge of $" +
serviceCharge + " applied.");
            balance -= serviceCharge;
            System.out.println("Remaining balance after service charge: $" + balance);
        }
    }
}

class SavAcct extends Account {
    double interestRate;

    public SavAcct(String customerName, long accountNumber, double balance) {
        super(customerName, accountNumber, "Savings Account", balance);
        this.interestRate = 0.05; // Set interest rate (5%)
    }

    @Override
    public void withdraw(double amount) {
        if (balance - amount >= 0) {
```

```
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining balance: $" +
balance);
        } else {
            System.out.println("Insufficient funds. Cannot complete withdrawal.");
        }
    }

    public void depositInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest deposited. Updated balance: $" + balance);
    }
}

public class Bank1 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter account number: ");
        long accountNumber = scanner.nextLong();

        System.out.print("Enter initial balance: ");
        double initialBalance = scanner.nextDouble();

        System.out.print("Enter account type (Current/Savings): ");
        String accountType = scanner.next();

        Account account;
        if (accountType.equalsIgnoreCase("Current")) {
            account = new CurrAcct(customerName, accountNumber, initialBalance);
        } else if (accountType.equalsIgnoreCase("Savings")) {
            account = new SavAcct(customerName, accountNumber, initialBalance);
        } else {
            System.out.println("Invalid account type. Exiting program.");
            return;
        }
```

```
        int choice;
        do {
            System.out.println("\n1. Deposit");
            System.out.println("2. Display Balance");
            System.out.println("3. Deposit Interest for Savings Account");
            System.out.println("4. Withdraw");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter deposit amount: ");
                    double depositAmount = scanner.nextDouble();
                    account.balance += depositAmount;
                    System.out.println("Deposit successful. Updated balance: $" +
account.balance);
                    break;

                case 2:
                    account.displayBalance();
                    break;

                case 3:
                    if (account instanceof SavAcct) {
                        ((SavAcct) account).depositInterest();
                    } else {
                        System.out.println("This option is applicable for Savings Account
only.");
                    }
                    break;

                case 4:
                    System.out.print("Enter withdrawal amount: ");
                    double withdrawalAmount = scanner.nextDouble();
                    account.withdraw(withdrawalAmount);
                    break;

                case 5:
```

```
                System.out.println("Exiting  program.  Goodbye!");
                break;

            default:
                System.out.println("Invalid choice. Please enter a valid option.");
        }

    } while (choice != 5);

    scanner.close();
  }
}
```

WEEK6

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.

Program:

File: Student.java

```
package CIE;
import java.util.Scanner;
class Student{
protected String usn=new String();
protected String name=new String();
protected int sem;
public void inputStudentDetails()
{
        Scanner s=new Scanner(System.in);
        System.out.println("enter the usn");
        usn=s.next();
        System.out.println("enter the name");
        name=s.next();
        System.out.println("enter the semester");
        sem=s.nextInt();

}
public void displayStudentDetails()
{
System.out.println("USN:" +usn);
System.out.println("NAME IS:"+ name);
System.out.println("semester" + sem);

}
}
```

File: Internals.java

```
package CIE;
import java.util.Scanner;
public class Internals extends Student
{
protected int marks[]=new int[5];
public void inputCIEmarks()
{
        Scanner s=new Scanner(System.in);

        for(int i=0; i<5; i++)
        {
        System.out.println("Enter internal marks of CIE:"+(i+1));
        marks[i]=s.nextInt();
        }

}
}
```

File: Externals.java

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals
{
        protected int marks[];
        protected int finalMarks[];

        public  Externals()
        {
        marks = new int[5];
        finalMarks = new int[5];
        }
public void inputSEEmarks()
        {
        Scanner s = new Scanner(System.in);
        for(int i=0; i<5; i++)
        {
        System.out.print("Subject "+(i+1)+" marks: ");
```

```
        marks[i] = s.nextInt();
        }
}
public void calculateFinalMarks() {
for(int i=0; i<5; i++)
finalMarks[i] = marks[i]/2 + super.marks[i];
}
public void displayFinalMarks()
{
displayStudentDetails();
for(int i=0; i<5; i++)
System.out.println("Subject " + (i+1) + ": " + finalMarks[i]);
}
}
```

File: Main.java

```
import SEE.Externals;

class Main {

public static void main(String args[])

{

        int numOfStudents = 2;
        Externals finalMarks[] = new
        Externals[numOfStudents];
        for(int i=0;i<numOfStudents;i++)
        {
        finalMarks[i] = new Externals();
        finalMarks[i].inputStudentDetails();
        System.out.println("Enter CIE marks");
        finalMarks[i].inputCIEmarks();
        System.out.println("Enter SEE marks");
        finalMarks[i].inputSEEmarks();
}
System.out.println("Displaying data:\n");
for(int i=0;i<numOfStudents;i++)
```

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

}

}
```

WEEK7

7.Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that cases both father and son's age and throws an exception if son's age is >=father's age.

Program:

```
import java.util.Scanner;

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

class Father
{
int fatherAge;
Father() throws WrongAge
 {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Father's age: ");
        fatherAge = s.nextInt();
        if (fatherAge < 0)
         {
        throw new WrongAge("Age cannot be negative");
        }
}

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

class Son extends Father
{
```

```
int sonAge;
Son() throws WrongAge
        {
        super();
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Son's age: ");
        sonAge = s.nextInt();
        if (sonAge >= fatherAge) {
        throw new WrongAge("Son's age cannot be greater than father's age");
        }
        else if (sonAge < 0)
        {
        throw new WrongAge("Age cannot be negative");
        }
}

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

class Main {
public static void main(String[] args)
 {
        try
        {
        Son s = new Son();
        s.display();
        }
        catch (WrongAge e)
        {
        System.out.println(e.getMessage());
        }
}
}
```

WEEK8

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.

Program:

```
class College extends Thread
{
public void run()
 {
        for (int i = 0; i <5; i++) {
        System.out.println("BMS COLLEGE OF ENGINEERING");
        try
        {
        Thread.sleep(10000);
        }
        catch (InterruptedException e)
        {
        e.printStackTrace();
        }
}
}
}

class Dept extends Thread
 {
public void run()
 {
        for (int i = 1; i <= 25; i++) {
        System.out.println("CSE");
        try
        {
        Thread.sleep(2000);
        }
         catch (InterruptedException e)
        {
        e.printStackTrace();
        }
}

}
```

```
}

class Main {
        public static void main(String args[])
        {
        College c1 = new College();
        c1.start();
        Dept d1 = new Dept();
        d1.start();
        }
}
```

WEEK9

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.

Program:

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

class SwingDemo{
   SwingDemo(){
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 200);
      jfrm.setLayout(new FlowLayout());
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

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

      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);

      JButton button = new JButton("Calculate");

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

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

ActionListener l = new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
   }
};
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());
         if (b == 0) {
            throw new ArithmeticException();
         }
         int ans = a/b;

         alab.setText("A      =      "      +      a);
         blab.setText("B      =      "      +      b);
         anslab.setText("Ans = "+ ans);
         err.setText("");
      }
      catch(NumberFormatException          e){
         clearLabels();
         err.setText("Enter Only Integers!");
      }
      catch(ArithmeticException                e){
         clearLabels();
```

```
                err.setText("B should be NON zero!");
            }
        }

        private void clearLabels() {
            alab.setText("");
            blab.setText("");
            anslab.setText("");
        }
    });

    jfrm.setVisible(true);
  }

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

WEEK10

10.Demonstrate Inter process Communication and deadlock

Program:

A.Deadlock

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

```
}
void last()
        {
        System.out.println("Inside A.last");
        }
}
class Deadlock implements Runnable
{
A a = new A();
B b = new B();
Deadlock() {
Thread.currentThread().setName("MainThread");
Thread t = new Thread(this,"RacingThread");
t.start();
a.foo(b); // get lock on a in this thread.
System.out.println("Back in main thread");
}
public void run()
 {
b.bar(a); // get lock on b in other thread.
System.out.println("Back in otherthread");
  }
public static void main(String args[])
        {
        new Deadlock();
        }
}
```

B. InterprocessCommunication:

```
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.");
	}
}
```