**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“JnanaSangama”, Belgaum -590014, Karnataka.**

****

**LAB REPORT**

**on**

**Object Oriented Java Programming (23CS3PCOOJ)**

***Submitted by***

**Aswath Ram Pavith Reddy (1BM19CS028)**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**

****

**B.M.S. COLLEGE OF ENGINEERING**

**(Autonomous Institution under VTU)**

**BENGALURU-560019**

**Sep-2024 to Jan-2025**

**B.M.S. College of Engineering,**

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

**Department of Computer Science and Engineering**

****

**CERTIFICATE**

This is to certify that the Lab work entitled “Object Oriented Java Programming (23CS3PCOOJ)” carried out by **Aswath Ram Pavith Reddy (1BM19CS028),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

|  |  |
| --- | --- |
| Lab faculty Incharge Name: Surabhi  Assistant Professor: Dr Seema Patil  Department of CSE, BMSCE | Dr. Jyothi S Nayak  Professor & HOD  Department of CSE, BMSCE |

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 | 30/09/2024 | Implement Quadratric Equation | 4-7 |
| 2 | 7/10/2024 | SGPA calculation | 8-13 |
| 3 | 14/10/2024 | Book Store | 14-19 |
| 4 | 21/10/2024 | Area of shapes | 20-25 |
| 5 | 28/10/2024 | Bank- Savings and current | 26-35 |
| 6 | 11/11/2024 | Packages | 36-43 |
| 7 | 28/11/2024 | Exceptions | 44-49 |
| 8 | 28/11/2024 | Threads | 50-53 |
| 9 | 28/11/2024 | Interface – Open End Swing Demo | 54-58 |
| 10 | 28/11/2024 | Demonstrate IPC and Deadlock | 59-68 |

Github Link:

https://github.com/Aswathramreddy/javalab.git

**Program 1**

Implement Quadratic Equation

Algorith:

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

Code:

import java.lang.Math;

class quadratic{

public static void main(String args[]){

int a,b,c;

double r1,r2;

Scanner scan = new Scanner(System.in);

System.out.println("Enter the value of a:");

a=scan.nextInt();

System.out.println("Enter the value of b:");

b = scan.nextInt();

System.out.println("Enter the value of c:");

c=scan.nextInt();

scan.close();

float d=b\*b-(4\*a\*c);

System.out.println("The value of d is:" +d);

if(a==0){

System.out.println("Not a quadratic equation");

}

if(d==0){

r1=(-b)/(2\*a);

System.out.println("Roots are real and equal");

System.out.println("Root 1 is" +r1+ "\nRoot 2 is" +r1);

}

else if(d>0)

{

r1=((-b) + (Math.sqrt(d)))/(double)(2\*a);

r2=((-b) - (Math.sqrt(d)))/(double)(2\*a);

System.out.println("Root 1 is" +r1+ "\nRoot 2 is" +r1);

}

else if(d<0)

{

System.out.println("Roots are imaginary");

r1=(-b)/(2\*a);

r2 = Math.sqrt(-d)/(2\*a);

System.out.println("Root 1 is " +r1+ "+" +r2+"i");

}

}

}

Output:

A screenshot of a computer program

Description automatically generated

Program 2

SGPA calculation

Algorithm:

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

Code:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Set the number of subjects to 8

int numberOfSubjects = 8;

Student student = new Student(numberOfSubjects);

student.acceptDetails();

student.displayDetails();

scanner.close();

}

}

import java.util.Scanner;

class Student {

public String usn;

public String name;

public int[] credits;

public int[] marks;

public Student(int n) {

credits = new int[n];

marks = new int[n];

}

public void acceptDetails() {

Scanner scanner = new Scanner(System.in);

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

usn = scanner.nextLine();

System.out.println("Enter Name: ");

name = scanner.nextLine();

for (int i = 0; i < credits.length; i++) {

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

credits[i] = scanner.nextInt();

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

marks[i] = scanner.nextInt();

}

}

public double calculateSGPA() {

double totalPoints = 0;

int totalCredits = 0;

for (int i = 0; i < credits.length; i++) {

int gradePoint = convertMarksToGradePoints(marks[i]);

totalPoints += gradePoint \* credits[i];

totalCredits += credits[i];

}

return totalCredits == 0 ? 0 : totalPoints / totalCredits;

}

private int convertMarksToGradePoints(int marks) {

if (marks >= 90) return 10;

if (marks >= 80) return 9;

if (marks >= 70) return 8;

if (marks >= 60) return 7;

if (marks >= 50) return 6;

if (marks >= 40) return 5;

return 0;

}

public void displayDetails() {

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

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

System.out.println("Subjects Details:");

for (int i = 0; i < credits.length; i++) {

System.out.println("Subject " + (i + 1) + ": Credits = " + credits[i] + ", Marks = " + marks[i]);

}

double sgpa = calculateSGPA();

System.out.printf("SGPA: %.2f\n", sgpa);

}

Output:

A screenshot of a computer screen

Description automatically generated

Program 3

Book store

Algorithm:

A notebook with writing on it

Description automatically generated]

A piece of lined paper with writing

Description automatically generated

A lined notebook with writing

Description automatically generated

Code:

import java.util.Scanner;

public class Main

{

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);

int n;

System.out.println("enter how many books");

n = scanner.nextInt();

scanner.nextLine();

Book[] books = new Book[n];

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

{

books[i] = new Book(" "," ",0,0);

System.out.println("enter the details for the book " + (i+1));

books[i].getdetails(scanner);

}

System.out.println("here is your book details: ");

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

{

books[i].showdetails();

}

scanner.close();

}

}

import java.util.Scanner;

public class Book

{

public String name;

public String author;

public int price;

public int num\_pages;

public Book(String name,String author,int price,int num\_pages)

{

this.name = name;

this.author = author;

this.price = price;

this.num\_pages = num\_pages;

}

public void getdetails(Scanner scanner)

{

System.out.println("enter the book name: ");

name = scanner.nextLine();

System.out.println("enter the author: ");

author = scanner.nextLine();

System.out.println("enter the price: ");

price = scanner.nextInt();

scanner.nextLine();

System.out.println("enter the number of pages : ");

num\_pages = scanner.nextInt();

scanner.nextLine();

}

public void showdetails()

{

System.out.println(this);

}

public String toString()

{

return

"Book name: " + this.name + "\n" +

"Author name: " + this.author + "\n" +

"Price: " + this.price + "\n" +

"Number of pages: " +this.num\_pages + "\n";

}

}

Output:

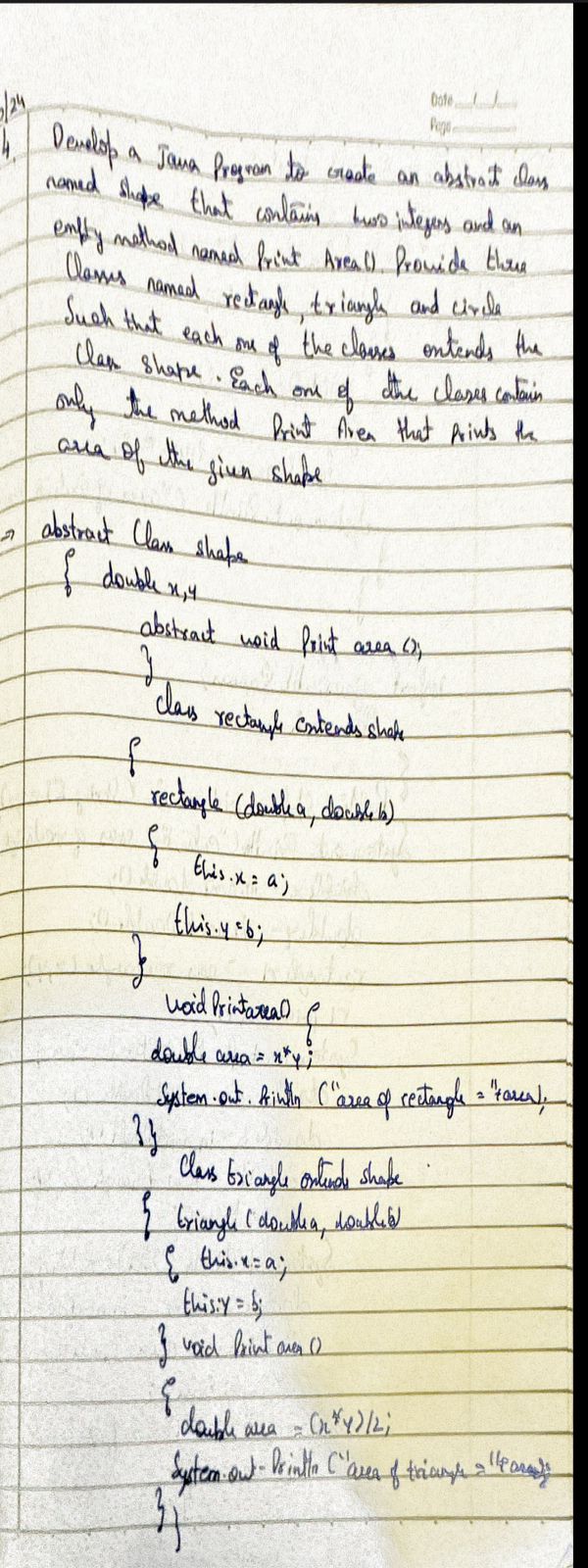
A screenshot of a computer

Description automatically generated

Program 4

Area of shapes

Algorithm:



A close-up of a notebook

Description automatically generated

A piece of lined paper with writing on it

Description automatically generated

Code:

import java.util.Scanner;

class Main

{

public static void main(String[] args)

{

Scanner in = new Scanner(System.in);

System.out.println("enter the sides of rectangle: ");

double x = in.nextDouble();

double y = in.nextDouble();

Rectangle r1 = new Rectangle(x,y);

r1.printarea();

System.out.println("enter the sides of triangle: ");

double a = in.nextDouble();

double b = in.nextDouble();

Triangle t1 = new Triangle(a,b);

t1.printarea();

System.out.println("enter the radius of circle : ");

double radius = in.nextDouble();

Circle c1 = new Circle(radius);

c1.printarea();

System.out.println("END OF PROGRAM");

in.close();

}

}

abstract class Shape

{

double x,y;

abstract void printarea();

}

class Rectangle extends Shape

{

Rectangle(double a,double b)

{

this.x = a;

this.y = b;

}

void printarea()

{

double area = x\*y;

System.out.println("area of rectangle = " + area);

}

}

class Triangle extends Shape

{

Triangle(double a,double b)

{

this.x = a;

this.y = b;

}

void printarea()

{

double area = (x\*y)/2;

System.out.println("area of triangle = " + area);

}

}

class Circle extends Shape

{

Circle(double radius)

{

this.x = radius;

}

void printarea()

{

double area = 3.1416\*x\*x;

System.out.println("area of circle = " + area);

}

}

Output:  
A computer screen shot of a program

Description automatically generated

Program 5

Bank account

Algorithm:

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

Code:

import java.util.Scanner;

interface Bank

{

void deposit(double amount);

void displayBalance();

void withdraw(double amount);

}

class Account

{

String name;

int accountNumber;

String accountType;

double balance;

Account(String name, int accountNumber, String accountType, double initialBalance)

{

this.name = name;

this.accountNumber = accountNumber;

this.accountType = accountType;

this.balance = initialBalance;

}

// Method to get the balance

public double getBalance()

{

return balance;

}

}

class SavAcct extends Account implements Bank

{

double interestRate;

SavAcct(String name, int accountNumber, double initialBalance, double interestRate)

{

super(name, accountNumber, "Savings", initialBalance);

this.interestRate = interestRate;

}

public void deposit(double amount)

{

balance += amount;

System.out.println("Deposited: " + amount);

}

public void displayBalance()

{

System.out.println("Current balance: " + getBalance());

}

public void computeAndDepositInterest()

{

double interest = getBalance() \* interestRate;

balance += interest;

System.out.println("Interest of " + interest + " added. New balance: " + getBalance());

}

public void withdraw(double amount)

{

if (amount > getBalance())

{

System.out.println("Insufficient funds.");

}

else

{

balance -= amount;

System.out.println("Withdrew: " + amount + ". New balance: " + getBalance());

}

}

}

class CurAcct extends Account implements Bank

{

double minBalance = 500.0;

double serviceCharge = 50.0;

CurAcct(String name, int accountNumber, double initialBalance)

{

super(name, accountNumber, "Current", initialBalance);

}

public void deposit(double amount)

{

balance += amount;

System.out.println("Deposited: " + amount);

}

public void displayBalance()

{

System.out.println("Current balance: " + getBalance());

}

public void withdraw(double amount)

{

if (amount > getBalance())

{

System.out.println("Insufficient funds.");

}

else

{

balance -= amount;

System.out.println("Withdrew: " + amount + ". New balance: " + getBalance());

}

checkMinimumBalance();

}

void checkMinimumBalance()

{

if (getBalance() < minBalance)

{

balance -= serviceCharge;

System.out.println("Service charge imposed. New balance: " + getBalance());

}

}

}

public class BankAccount

{

public static void main(String[] args)

{

Scanner scanner = new Scanner(System.in);

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

String name = scanner.nextLine();

System.out.println("Enter account number: ");

int accountNumber = scanner.nextInt();

System.out.println("Choose account type (1 for Savings, 2 for Current): ");

int accountType = scanner.nextInt();

Bank account;

if (accountType == 1)

{

System.out.println("Enter initial balance: ");

double initialBalance = scanner.nextDouble();

System.out.println("Enter interest rate (as a decimal): ");

double interestRate = scanner.nextDouble();

account = new SavAcct(name, accountNumber, initialBalance, interestRate);

}

else

{

System.out.println("Enter initial balance: ");

double initialBalance = scanner.nextDouble();

account = new CurAcct(name, accountNumber, initialBalance);

}

account.displayBalance();

// Immediate interest calculation for savings account

if (account instanceof SavAcct)

{

((SavAcct) account).computeAndDepositInterest();

}

System.out.println("Enter deposit amount: ");

double depositAmount = scanner.nextDouble();

account.deposit(depositAmount);

account.displayBalance();

System.out.println("Enter withdrawal amount: ");

double withdrawalAmount = scanner.nextDouble();

account.withdraw(withdrawalAmount);

account.displayBalance();

scanner.close();

}

}

Output:

A screenshot of a computer

Description automatically generated

Program 6

Final Marks

Algorithm:

A close up of a paper

Description automatically generated

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

A notebook with writing on it

Description automatically generated

Code:  
import CIE.Internals;

import SEE.Externals;

import CIE.Student;

import java.util.Scanner;

class PrintInfo {

static void print() {

System.out.println("Name: Aswath Pavith Reddy");

System.out.println("USN: 1BM19CS028");

}

}

public class Main {

public static void main(String[] args) {

PrintInfo.print();

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number of students: ");

int n = sc.nextInt();

Internals[] cieStudents = new Internals[n];

Externals[] seeStudents = new Externals[n];

Student[] students = new Student[n];

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

System.out.println("\nEnter details for Student " + (i + 1) + ":");

cieStudents[i] = new Internals();

seeStudents[i] = new Externals();

students[i] = new Student();

students[i].inputStudentDetails();

cieStudents[i].inputCIEmarks();

seeStudents[i].inputSEEmarks();

}

System.out.println("\nFinal Results:");

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

students[i].calcFinalMarks(cieStudents[i], seeStudents[i]);

}

}

}

package SEE;

import CIE.Student;

import java.util.Scanner;

public class Externals extends Student {

public int[] seeMarks = new int[5];

public void inputSEEmarks() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter SEE marks for 5 subjects:");

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

System.out.print("Subject " + (i + 1) + ": ");

seeMarks[i] = sc.nextInt();

}

package CIE;

import SEE.Externals;

import java.util.Scanner;

public class Student {

public String usn;

public String name;

public int sem;

public void inputStudentDetails() {

Scanner sc = new Scanner(System.in);

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

usn = sc.nextLine();

System.out.print("Enter Name: ");

name = sc.nextLine();

System.out.print("Enter Semester: ");

sem = sc.nextInt();

}

public void displayStudentDetails() {

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

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

System.out.println("Semester: " + sem);

}

public void calcFinalMarks(Internals cie, Externals see) {

displayStudentDetails();

System.out.println("Final Marks for 5 subjects:");

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

int finalMarks = cie.cieMarks[i] + (see.seeMarks[i] / 2);

System.out.println("Subject " + (i + 1) + ": CIE = " + cie.cieMarks[i] +

", SEE = " + see.seeMarks[i] +

", Final = " + finalMarks);

}

}}

package CIE;

import java.util.Scanner;

public class Internals extends Student {

public int[] cieMarks = new int[5];

public void inputCIEmarks() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter CIE marks for 5 subjects:");

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

System.out.print("Subject " + (i + 1) + ": ");

cieMarks[i] = sc.nextInt();

}

}

}

Output:

A screenshot of a computer

Description automatically generated

Program 7

Exception

Algorithm:

A piece of lined paper with writing

Description automatically generated

A close-up of a notebook

Description automatically generated

A piece of lined paper with writing

Description automatically generated

Code:

import java.util.Scanner;

class WrongAge extends Exception {

public WrongAge() {

super("Age Error");

}

public WrongAge(String message) {

super(message);

}

}

class InputScanner {

Scanner s = new Scanner(System.in);

public int getInput() {

return s.nextInt();

}

}

class Father extends InputScanner {

protected int fatherAge;

public Father() throws WrongAge {

System.out.print("Enter father's age: ");

fatherAge = getInput();

if (fatherAge < 0) {

throw new WrongAge("Age cannot be negative");

}

}

public void displayFatherAge() {

System.out.println("Father's age: " + fatherAge);

}

}

class Son extends Father {

private int sonAge;

public Son() throws WrongAge {

super(); // Call Father's constructor

System.out.print("Enter son's age: ");

sonAge = getInput(); // Read son's age

if (sonAge >= fatherAge) {

throw new WrongAge("Son's age cannot be greater than or equal to father's age");

} else if (sonAge < 0) {

throw new WrongAge("Age cannot be negative");

}

}

public void displaySonAge() {

System.out.println("Son's age: " + sonAge);

}

}

public class Main {

public static void main(String[] args) {

try {

Son son = new Son();

son.displayFatherAge();

son.displaySonAge();

} catch (WrongAge e) {

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

}

System.out.println("Aswath Reddy\n1BM19CS028");

}

}

Output:

A computer screen shot of a black screen

Description automatically generated

Program 8

Multithreading

Algorithm:

A notebook with writing on it

Description automatically generated

A close-up of a list of tasks

Description automatically generated

Code:

class CollegeThread extends Thread {

@Override

public void run() {

try {

while (true) {

System.out.println("BMS College of Engineering");

Thread.sleep(10000);

}

} catch (InterruptedException e) {

System.out.println("CollegeThread interrupted: " + e.getMessage());

}

}

}

class DepartmentThread extends Thread {

@Override

public void run() {

try {

while (true) {

System.out.println("CSE");

Thread.sleep(2000);

}

} catch (InterruptedException e) {

System.out.println("DepartmentThread interrupted: " + e.getMessage());

}

}

}

public class Main1 {

public static void main(String[] args) {

CollegeThread collegeThread = new CollegeThread();

DepartmentThread departmentThread = new DepartmentThread();

collegeThread.start();

departmentThread.start();

}

}

Output:

A black screen with white text

Description automatically generated

Program 9

Jframe app

Algorithm:

A close-up of a notebook

Description automatically generated

A close-up of a notebook

Description automatically generated

A lined notebook with a note

Description automatically generated with medium confidence

Code:  
import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

class SwingDemo{

SwingDemo(){

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

jfrm.setSize(275, 150);

jfrm.setLayout(new FlowLayout());

jfrm.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

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

JTextField ajtf = new JTextField(8);

JTextField bjtf = new JTextField(8);

JButton button = new JButton("Calculate");

JLabel err = new JLabel();

JLabel alab = new JLabel();

JLabel blab = new JLabel();

JLabel anslab = new JLabel();

jfrm.add(err);

jfrm.add(jlab);

jfrm.add(ajtf);

jfrm.add(bjtf);

jfrm.add(button);

jfrm.add(alab);

jfrm.add(blab);

jfrm.add(anslab);

ActionListener l = new ActionListener() {

public void actionPerformed(ActionEvent evt) { System.out.println("Action event from a text field"); } };

ajtf.addActionListener(l);

bjtf.addActionListener(l);

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

alab.setText("\nA = " + a);

blab.setText("\nB = " + b);

anslab.setText("\nAns = "+ ans);

}

catch(NumberFormatException e){

alab.setText("");

blab.setText("");

anslab.setText("");

err.setText("Enter Only Integers!"); }

catch(ArithmeticException e){

alab.setText("");

blab.setText("");

anslab.setText("");

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

});

jfrm.setVisible(true);

}

public static void main(String args[]){

SwingUtilities.invokeLater(new Runnable(){ public void run(){

new SwingDemo();

}

});

}

}

Output:  
A screenshot of a computer screen

Description automatically generated

Program 10

Deadlock

Algorithm:

A close-up of a notebook

Description automatically generated

A close-up of a list of items

Description automatically generated

A close up of a notebook

Description automatically generated

Code:

class Resource1 {

synchronized void lock(Resource2 res2) {

System.out.println(Thread.currentThread().getName() + " locked Resource1");

try {

Thread.sleep(100);

} catch (InterruptedException e) {

System.out.println(e);

}

System.out.println(Thread.currentThread().getName() + " trying to lock Resource2...");

res2.method2();

}

synchronized void method1() {

System.out.println(Thread.currentThread().getName() + " is working with Resource1");

}

}

class Resource2 {

synchronized void lock(Resource1 res1) {

System.out.println(Thread.currentThread().getName() + " locked Resource2");

try {

Thread.sleep(100);

} catch (InterruptedException e) {

System.out.println(e);

}

System.out.println(Thread.currentThread().getName() + " trying to lock Resource1...");

res1.method1();

}

synchronized void method2() {

System.out.println(Thread.currentThread().getName() + " is working with Resource2");

}

}

public class DeadlockDemo {

public static void main(String[] args) {

Resource1 res1 = new Resource1();

Resource2 res2 = new Resource2();

Thread t1 = new Thread(() -> res1.lock(res2), "Thread-1");

Thread t2 = new Thread(() -> res2.lock(res1), "Thread-2");

t1.start();

t2.start();

}

}

import java.util.concurrent.locks.Lock;

import java.util.concurrent.locks.ReentrantLock;

class SafeResource1 {

private final Lock lock = new ReentrantLock();

public boolean tryLockBoth(SafeResource1 otherResource) {

while (true) {

boolean gotFirstLock = lock.tryLock();

boolean gotSecondLock = otherResource.tryLock();

if (gotFirstLock && gotSecondLock) {

System.out.println(Thread.currentThread().getName() + " acquired both locks");

return true;

}

if (gotFirstLock) lock.unlock();

if (gotSecondLock) otherResource.unlock();

}

}

public boolean tryLock() {

return lock.tryLock();

}

public void unlock() {

lock.unlock();

}

}

public class DeadlockResolved {

public static void main(String[] args) {

SafeResource1 res1 = new SafeResource1();

SafeResource1 res2 = new SafeResource1();

Thread t1 = new Thread(() -> {

if (res1.tryLockBoth(res2)) {

System.out.println("Thread-1 completed safely");

res1.unlock();

res2.unlock();

}

}, "Thread-1");

Thread t2 = new Thread(() -> {

if (res2.tryLockBoth(res1)) {

System.out.println("Thread-2 completed safely");

res2.unlock();

res1.unlock();

}

}, "Thread-2");

t1.start();

t2.start();

}

}

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[]) {

System.out.println("Name: Harsha, USN: 1BM23CS136");

Q q = new Q();

new Producer(q);

new Consumer(q);

System.out.println("Press Control-C to stop.");

}

}

Output:

A screen shot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screenshot of a computer screen

Description automatically generated