USN : 2028BMS02541

I N D E X

NAME : Shreya B. Patil STD.: III Sem SEC.: 3E ROLL NO.: _____ SUB.: JAVA

S. No.	Date	Title	Page No.	Teacher's Sign/ Remarks
1	12/12/23	Quadratic Equation	1-4	
2	19/12/23	Calculate SGPA	15-19	
3	26/12/23	N Book Objects	25-28	
4	02/01/24	Abstract Class	29-32	
5	09/01/24	Inheritence	33-41	
6	23/01/24	Packages	49-54	
7	30/01/24	Exception	55-58	
8	06/02/24	Multi Threading	59-60	
9	06/02/24	IPC & Deadlock	61-67	
10	20/02/24	AWT	68-73	

Lab: 1

Develop a Java program that prints all real solutions to the quadratic equation $ax^2+bx+c=0$. Read 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 coefficient
    of a,b,c");
     a= s.nextInt();
     b = s.nextInt();
     c = s.nextInt();
  }
  void compute()
  {
    while (a==0)
    {
      System.out.println ("Not a quadratic
            equation");
      System.out.println ("Enter a non zero
            value for a:");
```

```
Scanner s = new Scanner (System.in);
a = s.nextInt();
}
d = b*b - 4*a*c;
if(d==0)
{
  r1 = (-b)/(2*a)
  System.out.println("Roots are real and
                equal");
  System.out.println("Root1 = Root2 =" +r1);
}
else if (d>0)
{
  r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
  r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a);
  System.out.println("Roots are real and
      distinct");
  System.out.println("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[])
  {
    Qudratic q = new Qudratic();
    q.get();
    q.compute();
  }
}
```

Output:

```
C:\Users\Admin\Desktop\2023bms02541>
  javaC Quadratic.java
C:\Users\Admin\Desktop\2023bms02541>
  java Quadratic
  Enter the coefficients of a,b,c
  2
  5
  3
Roots are real and distinct
Root1 = -1.0  Root = -1.5
```

URA 9-12-23

DOMS | Page No. 15
Date 19 / 12 / 2023

Lab : 2

Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include method to accept and display details and a method to calculate sgpa of a student.

```
import java.util.Scanner;
class Student {
    String name;
    String usn;
    double SGPA;
    Subject subject[];
    Scanner s;

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

    void getStudentDetails() {
        System.out.println("Enter your Name:");
        name = s.next();
        System.out.println("Enter your USN:");
        usn = s.next();
    }
```

```
void getMarks() {
    for (int i=0; i<9; i++) {
        System.out.println("Enter Marks for
            subject "+(i+1)+" : ");
        subject[i].subjectMarks = s.nextInt();
        System.out.println("Enter credits for
                subject "+(i+1)+": ");
        subject[i].credits = s.nextInt();

        subject[i].grade = (subject[i].subject-
                -Marks/10)+1;

        if (subject[i].grade == 11) {
            subject[i].grade = 10;
        }
        if (subject[i].grade <= 4) {
            subject[i].grade = 0;
        }
    }
}

void computeSGPA() {
    int effectiveScore = 0;
    int totalCredits = 0;

    for (int i=0; i<9; i++) {
        effectiveScore += (subject[i].
            grade * subject[i].credits);
```

```
        totalCredits += subject[i].credits;
    }

    SGPA = (double) effectiveScore /
           (double) totalCredits;
  }
}

class Subject {
  int subjectMarks;
  int credits;
  int grade;
}

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

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

Output:

```
C:\Users\Admin\Desktop\2023BMS02541>
        javac Main.java

C:\Users\Admin\Desktop\2023BMS02541>
        java Main

Enter your Name:
Shreya
Enter your USN:
2023BMS02541
Enter marks for subject 1:
98
Enter credits for subject 1:
3
Enter marks for subject 2:
78
Enter credits for subject 2:
2
Enter marks for subject 3:
78
Enter credits for subject 3:
3
Enter marks for subject 4:
98
Enter credits for subject 4:
4
Enter marks for subject 5:
85
```

Enter credits for subject 5:
1
Enter marks for subject 6:
56
Enter ~~marks~~ credits for subject 6:
1
Enter marks for subject 7:
85
Enter credits for subject 7:
3
Enter marks for subject 8:
78
Enter credits for subject 8:
2
Enter marks for subject 9:
96
Enter credits for subject 9:
4
Name: Shreya
USN: 2023BMS02541
SGPA: 9.043478260869565

RA 26-12-2

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

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

    Book(String name, String author, int price
        , int num_pages) {
    this.name = name;
    this.author = author;
    this.price = price;
    this.num_pages = num_pages;
    }

    String toString() {
        String name, author, price, num_pages;
        name = " Book name: " + this.name + "\n";
        author = " Book author: " + this.author +
                 "\n";
```

```
price = "Price : " + this.price + "\n";
num_pages = "Number of pages : " + this.num_
            pages + "\n";
return (name + author + price + num_pages);
}

public static void main(String args[]){
    Scanner s = new Scanner (System.in);
    int n;
    String name;
    String author;
    int price;
    int num_pages;

    System.out.println("Enter the no. of book
                -s : ");
    n = nextInt();

    Book b[] = new Book[n];

    for (int i = 0; i<n; i++) {
      System.out.print("Enter name of book:");
      ~~scanner.n~~ s.nextLine();
      name = s.nextLine();

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

```
System.out.print("Enter ~~auth~~ the price of the
        book:");
price = s.nextInt();

System.out.print("Enter the number of
  pages of book:");
num_page = s.nextInt();

b[i] = new Book(name, author, price, num_pa
        -ge);
}

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

Output:

```
c:\Users\Admin\Desktop\2023BMS02541>javac
                        Book.java
  >P java Book
Enter the no. of books:
2
Enter name of book:
```

The god of small things
Enter author of a book:
Arundhati Roy
Enter the price of book:
899
Enter the no. of pages of book:
450
Enter name of book:
The Guide
Enter author of a book:
R. K. Narayan
Enter the price of book:
999
Enter the no. of pages of book:
650
Book Details:
Book 1:
Book Name: The god of small things
Author name: Arundhati Roy
Price: 899
Number of pages: 450
Book 2:
Book name: The Guide
Author name: R. K. Narayan
Price: 999
Number of pages: 650

Lab: 4

WAP-1-23p

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

abstract class Shape extends InputScanner{
    double a,b;
    abstract void displayArea();
    abstract void getInput();
}

class Rectangle extends Shape {
    void getInput() {
        System.out.println("Enter the dimen-
        -sion of the rectangle (length and
        breadth):");
```

```
        a = s.nextDouble();
        b = s.nextDouble();
    }
    void displayArea(){
        System.out.println("Area of rectangle
            = " + (a * b));
    }
}

class Triangle extends Shape {
    void getInput(){
        System.out.println("Enter the dimension
            of the triangle (base and height):");
        a = s.nextDouble();
        b = s.nextDouble();
    }
    void displayArea(){
        System.out.println("Area of triangle="
            +((a*b)/2));
    }
}

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

```
    void displayArea(){
        System.out.println("Area of circle =" 
                + (3.14 * a * a));
    }
}

class MainMethod{
    public static void main(String args[]){
        Rectangle R = new Rectangle();
        R.getInput();
        R.displayArea();

        Triangle T = new Triangle();
        T.getInput();
        T.displayArea();

        Circle C = new Circle();
        C.getInput();
        C.displayArea();
    }
}
```

Output:

```
Enter the dimension of the rectangle (length
 and breadth:
4 5
Area of rectangle = 20.0
Enter the dimension of the triangle (base
```

and height:
9 6
Area of triangle = 27.0
Enter the dimension of the circle(radius):
5
Area of Circle = 78.5

Shreya Patil
2023BMS02541

Unit: 2 Lab: 5

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account pro-vides compound interest and withdrawal fa-cilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders sha-ld also maintain a minimum balance and if the balance falls below this level, a servia 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 balan

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

```
import java.util.Scanner;
public class Account {
    String customer_name;
```

```
int account_no;
String type_acc;
double balance = 0;
Scanner sc;

Account (String customer_name, int account
  no, String type_acc) {
    this.customer_name = customer_name;
    this.account_no = account_no;
    this.type_acc = type_acc;
  sc = new Scanner(System.in);
}

void diposite(){
  System.out.println("Enter the deposit
    amount:");
  int depo = sc.nextInt();
  balance += depo;
}

void withdrawl() {
  System.out.println("Enter the withdrawl
    amount :");
  int with = sc.nextInt();
  if(with > balance) {
      System.out.println("Insufficient
        Balance");
  }
```

```
else {
    balance -= with;
  }
}

void display() {
  System.out.println("Customer name: "
                  + customer_name);
  System.out.println("Account number: "
                  + account_no);
  System.out.println("Type of Account: "
                  + type_acc);
  System.out.println("Balance: "+balance
}

void applyinterest() { }
}

public class Cur_acct extends Account{
  Cur_acct(String customer_name, int account_
      no, String type_acc) {
    super(customer_name, account_no,
          type_acc);
  }

  void withdrawl() {
    System.out.println("Enter the withdrawl
        amount: ");
```

```
int with = sc.nextInt();
if(balance <= 2000){
    double pen = balance/(0.06);
    System.out.println("Insufficient ba-
    -lance penalty to be paid :"+pen);
    balance += pen;
  }
  else{
    balance += with;
  }
 }
}

public class Sav_acct extends Account{
  Sav_acct(String customer_name, int account_
      no, String type_acc){
       super(customer_name, account_no,
          type_acc);
       }

  void applyinterest(){
      System.out.println("Enter the interest
          rate:");
      int rate = sc.nextInt();
      double interest = balance*(rate/100);
      balance += interest;
      System.out.println("Balance after
        interest :"+balance);
    }
}
```

```
import java.util.Scanner;
public class Bank {
  public static void main (String args[]) {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter customer name:
      ");
    String cus_name = sc.nextLine();
    System.out.println("Enter account number
      :");

    Account ca = new Cur_acct(cus_name,
        acc_no, "Current Account");
    Account sa = new Sav_acct(cus_name,
        acc_no, "Saving Account");

    int choice;
    while (true) {
      System.out.println("----MENU----");
      System.out.println("1. Deposite \n 2.
      Withdrawl \n3. Compute interest for
      saving account \n 4. Display account
      details \n 5. Exit");
      choice = sc.nextInt();

      switch (choice) {
      case 1:
        System.out.println("Enter the type
        of account: \n1. Saving Account \n
        2. Current Account");
```

```
int acc = sc.nextInt();
if (acc==1){
    sa.diposite();
}
else{
    ca.diposite();
}
break;
case 2:
    System.out.println("Enter the type of
    account:\n1. Saving Account \n2. Current
    Account");
    int acc1= sc.nextInt();
    if(acc1==1){
        sa.withdrawl();
    }
    else{
        ca.withdrawl();
    }
    break;
case 3:
    sa.applyinterest();
    break;
case 4:
    System.out.println("Enter the type of
    account:\n1. Saving Account \n2. Curr-
    -ent Account");
    int acc2 = sc.nextInt();
```

```
if (acc2 == 1) {
    sa.display();
  }
else {
    ca.display();
  }
  break;
  case 5:
      break;
  default:
      System.out.println("Invalid choice");
      break;
  }
if(choice == 5){
    break;
  }
 }
}
```

Output :
Enter custome name :
Shreya
Enter account number :
22
--MENU--
1. Deposite
2. Withdrawl
3. Compute interest for Saving account

4. Display account details
5. Exit
1
Enter the type of account:
1. Saving Account
2. Current Account
1
Enter the deposit amount:
5000
2
Enter the type of account:
1. Saving Account
2. Current Account
1
Enter the withdrawal amount:
2000
4
Enter the type of account:
1. Saving Account
2. Current Account
1.
Customer name: Shreya
Account Number: 22
Type of Account: Saving Account
Balance: 3000.0
2.
Enter the type of account:
1. Saving Account
2. Current Account

2.
Enter the withdrawal amount :
6000.
Insuffiecient balance. Penalty to be paid:
8333.3334

Shreya Patil
2023BMS02541

9/11/24

Lab

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.

```
package CIE;
import java.util.Scanner;
public class Student{
    public String usn, name;
    public int sem;

    public void inputStudentDetails(){
        Scanner sc = new Scanner(System.in);
        S.o.p("Enter the student usn:");
        usn = sc.nextLine();
        S.o.p("Enter the student name:");
        name = sc.nextLine();
        S.o.p("Enter student semester:");
        sem = sc.nextInt();
    }
```

```
    public void display(){
        S.o.p("Student USN: " + usn);
        S.o.p("Student Name:" + name);
        S.o.p("Student Sem: " + sem);
    }
}
```

```
package CIE;
import java.util.Scanner;
public class Internals extends Student {
    public int marks[] = new int[5];
    public void inputCIEmarks(){
        Scanner sc = new Scanner(System.in);
        for(int i=0; i<5; i++){
            S.o.p("Enter the marks for subje-
            -ct "+(i+1)+" : ");
            marks[i] = sc.nextInt();
        }
    }
}
```

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends CIE.Internals{
    public int marks[] = new int[5];
    public int finalMarks[] = new int[5];

    public void getSEEMarks(){
```

```
Scanner sc = new Scanner(System.in);
for(int i=0; i<5; i++){
  System.out.println("Enter "+(i+1)+" marks:");
  marks[i] = sc.nextInt();
  }
}

public void calculateFinalMarks(){
  for(int i=0; i<5; i++){
    finalMarks[i] = marks[i]/2 + super.marks
  }
}

public void displayFinalMarks(){
  for(int i=0; i<5; i++){
    System.out.println("Marks of "+(i+1)+
                "subject : "+ finalMarks[i]);
  }
}
}

import SEE.Externals;
class PackageMain {
  public static void main(String args[]){
    int numOfStudent = 2;
  Externals finalMarks[] = new Externals[numO
                Student],
```

```
for(int i=0; i<numOfStudent; i++){
    finalMarks[i] = new Externals();
    finalMarks[i].getStudentInfo();
    System.out.println("Enter CIE Marks:");
    finalMarks[i].getCIEMarks();
    System.out.println("Enter SEE Marks:");
    finalMarks[i].getSEEMarks();
}

System.out.println("Displaying data:");
for(int i=0; i<numOfStudent; i++){
    finalMarks[i].display();
    finalMarks[i].calculateFinalMarks();
    finalMarks[i].displayFinalMarks();
  }
 }
}
```

Output:

```
E:\Engineering\3sem\OOPS>javac CIE/Student.java
E:\Engineering\3sem\OOPS>javac CIE/Internals.java
E:\Engineering\3sem\OOPS>javac SEE/Externals.java
E:\Engineering\3sem\OOPS>javac PackageMain.java
E:\Engineering\3sem\OOPS>java PackageMain
Enter student usn: 2023BMS02541
Enter student name: Shreya Patil
Enter student semester: 3
Enter CIE Marks:
```

Enter 1 marks : 45
Enter 2 marks : 44
Enter 3 marks : 46
Enter 4 marks : 42
Enter 5 marks : 49
Enter SEE Marks :
Enter 1 marks : 98
Enter 2 marks : 95
Enter 3 marks : 96
Enter 4 marks : 88
Enter 5 marks : 85
Enter student usn : 4SFCS22007I
Enter student name : Shravan Patil
Enter student semester : 3
Enter CIE Marks :
Enter 1 marks : 50
Enter 2 marks : 49
Enter 3 marks : 48
Enter 4 marks : 45
Enter 5 marks : 45
Enter SEE Marks :
Enter 1 marks : 98
Enter 2 marks : 96
Enter 3 marks : 95
Enter 4 marks : 98
Enter 5 marks : 100

Displaying Data :

Student Name: Shreya Patil
Student USN: 2023BMS02541
Semester: 3
Marks of 1 subject: 94
Marks of 2 subject: 91
Marks of 3 subject: 94
Marks of 4 subject: 86
Marks of 5 subject: 91

Student Name: Shravan Patil
Student USN: 457CS22007L
Semester: 3
Marks of 1 subject: 99
Marks of 2 subject: 97
Marks of 3 subject: 95
Marks of 4 subject: 94
Marks of 5 subject: 95

30/12/2024

Lab

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.

```
class WrongAge extends Exception {
    WrongAge(String msg){
        super(msg);
    }
}

import java.util.Scanner;
class InputScanner {
    Scanner sc;
    InputScanner(){
        sc = new Scanner(System.in);
    }
}

class Father extends InputScanner {
    int fatherAge;
```

```
Father() throws WrongAge {
    System.out.println("Enter father's
            age: ");
    fatherAge = sc.nextInt();
    if (fatherAge < 0){
     throw new WrongAge("Age cannot be
      negative");
    }
  void display(){
      System.out.println("Father's age:"
                + fatherAge);
  }
}

class Son extends Father{
   int sonAge;
   Son() throws WrongAge{
      System.out.println("Enter Son's age:");
      sonAge = sc.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: "+sonAge);
}

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

Output:

Enter father's age:
-12
WrongAge: Age cannot be negative

Enter father's age:
25
Enter Son's age:
45
WrongAge: Son's age cannot be greater than father's age

Enter father's age:
45
Enter Son's age:
22
Father's age : 45
Son's age : 22

30/1/24

Shreya Patil
2023BMS02541

Lab : 8

Write a program which creates two thread displaying "BMS College of Engineering" once every then seconds and another displaying "CSE" once every two seconds.

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

class CSE implements Runnable{
    public void run() {
        while(true){
            try {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
            catch(InterruptedException e){
```

```
        e.printStackTrace();
    }
  }
}
```

```
public class ThreadMain{
    public static void main(String args[]){
        Thread t1 = new Thread(new
                    BMS_College_of_Engineering());
        Thread t2 = new Thread(new CSE());
        t1.start();
        t2.start();
    }
}
```

O/P

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

Lab : 10

Demonstrate Inter process Communication and deadlock.

```
class Q {
 int n;
 boolean valueSet = false;
 synchronized int get() {
      while (!valueSet){
          try{
           S.o.p("Consumer waiting\n");
           wait();
        }
          catch(InterruptedException e) {
              S.o.p(" Interrupted Exception
                        caught");
          }
        S.o.p("Got : " +n);
        valueSet = false;
        S.o.p("Intimate Producer\n");
        notify();
        return n;
      }
 }

 synchronized void put (int n) {
      while(valueSet) {
          try{
            S.o.p("Producer Waiting\n");
            wait();
          }
```

```
        catch (InterruptedException e) {
            S.o.p("InterruptedException
                    caught");
        }
        this.n=n;
        valueSet = true;
        S.o.p("Put: " + n);
        S.o.p("Intimate Consumer\n");
        notify();
    }
}
}
```

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

```
class Consumer implements Runnable {
    Q q
    Consumer(Q q){
        this.q=q;
        new Thread(this, "Consumer").start();
    }

    public void run() {
        int i=0;
        while (i<5) {
            int r = q.get();
            S.o.p("consumed : "+r);
            i++;
        }
    }
}

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

O/P:

```
Press Control-C to stop.
Put: 0
Intimate Consumer
Producer waiting
Got: 0
Intimate Consumer
Put: 1
Intimate Consumer
Producer waiting
consumed: 0
Got: 1
Intimate Consumer
Producer waiting
Got: 2
```

Shreya Patil
2023BMS02541

```
class A {
  synchronized void foo(B b) {
    String name = Thread.currentThread().
                        getName();
    S.o.p(name + " entered A.foo");
    try {
      Thread.sleep(1000);
    } catch(Exception e) {
      S.o.p("A Interrupted");
    }
    S.o.p(name + "trying to call B.last()");
    b.last();
  }
  void last() {
    S.o.p("Inside A.last");
  }
}

class B {
  synchronized void bar(A a) {
    string name = Thread.currentThread().
                        getName();
    S.o.p(name + " entered B.bar");
    try {
      Thread.sleep(1000);
    } catch(Exception e) {
      S.o.p("B Interrupted");
    }
```

```
    S.o.p(name + " trying to call A.last()");
    a.last();
    }
    void last() {
        S.o.p("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);
        S.o.p("Back in main Thread");
    }

public void run() {
        b.bar(a);
        S.o.p("Back in other thread");
    }

public static void main(String[] args){
        new DeadLock();
    }
}
```

O/P :

Main Thread entered A.foo
Racing Thread entered B.bar
Main Thread trying to call B.last()
Inside A.last
Back in main thread
Racing Thread trying to call A.last()
Inside A.last
Back in other thread

Shreya Patil
2023BMS02541
13.02.24

Lab

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(275, 150);
        jfrm.setLayout(new FlowLayout());

        //to terminate on close
        jfrm.setDefaultCloseOperation( JFram
                EXIT_ON_CLOSE);
```

```
// text label
JLabel jlab = new JLabel("Enter the divider
        and divident:");

// 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);
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 (1);
bjtf.addActionListener (1);

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 ("\n Ans = " + 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 threa

  SwingUtilities.invokeLater(new Runnable()
    {
      public void run(){
        new SwingDemo();
      }
    });
  }
}
```

Methods used in this program :

1. JFrame.setDefaultCloseOperation(int operation) : This method sets the operation that will happen by default when the user initiates a "close" on this frame. In this case, the frame will exit on close.

2. JFrame.add(component comp) : This method adds the specified component to this container. It is used to add the JLabel, JTextFields, and JButton to the JFrame.

3. JTextField.addActionListener(ActionListener l) : This method adds an ActionListener to the JTextField. The ActionListener is notified when an action occurs, which is when the user presses Enter.

4. JButton.addActionListener(ActionListener l): This method adds an actionListener to the Button. The actionListener is notified when the button is pressed.

5. Integer.parseInt(String s) : This method parses the specified string as an integer. It throws a NumberFormatException if the string cannot be parsed as an integer.

6. JLabel.setText(String Text): This method sets the text of the JLabel to the specified text.

7. ActionListener.actionPerformed(ActionEvent evt): This method is a part of ActionListener interface. It is implemented by the anonymous ActionListener class in this code. The action-Performed method is called when an action event occurs.

8. JFrame.setSize(int width, int height): This method sets the size of the JFrame to the specified width and height.

9. JFrame.setVisible(boolean b): This method makes the JFrame Visible or invisible.

10. JFrame.setTitle(String title): This method sets the title of the JFrame to the specified string.

20/2/24

LAB: 1

Develop a Java program that prints all real solutions to the quadratic equation $ax^2+bx+c = 0$. Read a, b, c and use the quadratic formula. If the discriminate b^2-4ac is negative, display a stating that there are no real solutions

```
import java.util.Scanner;
class Quadratic
{
	int a, b, c;
	double r1, r2, d;
	void getd()
	{
		Scanner s = new Scanner(System.in);
		System.out.println("Enter the coefficients of a,b,c");
		a = s.nextInt();
		b = s.nextInt();
		c = s.nextInt();
	}
	void compute()
	{
		while(a==0)
		{
			System.out.println("Not a quadratic equation");
			System.out.println("Enter a non zero value for a:");
			Scanner s = new Scanner(System.in);
			a = s.nextInt();
		}
		d = b*b-4*a*c;
		if(d==0)
		{
			r1 = (-b)/(2*a);
			System.out.println("Roots are real and equal");
			System.out.println("Roo1 = Root2 = " + r1);
		}
		else if(d>0)
		{
			r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
			r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a);
			System.out.println("Roots are real and distinct");
			System.out.println("Roo1 = " + r1 + " Root2 = " + r2);
		}
		else if(d<0)
		{
			System.out.println("Roots are imaginary");
			r1 = (-b)/(2*a);
			r2 = Math.sqrt(-d)/(2*a);
			System.out.println("Root1 = " + r1 + " + i"+r2);
			System.out.println("Root1 = " + r1 + " - i"+r2);
		}
	}
}
```

```
class QuadraticMain
{
	public static void main(String args[])
	{
		Quadratic q = new Quadratic();
		q.getd();
		q.compute();
	}
}
```

LAB: 2

Develop a Java program to create a class Student with members usn, name, an array credits and an array mark. Include methods to accept and display details and a method to calculate SGPA of a student.
//Develop a java program to create a class Student with members usn,name, an array creadits 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 Student{
   String name;
   String usn;
   double SGPA;
   Subject subject[];
   Scanner s;

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

   }

   void getStudentDetails(){
      System.out.println("Enter your Name:");
      name=s.next();
      System.out.println("Enter your USN:");
      usn=s.next();
   }

   void getMarks(){
      for(int i=0;i<9;i++){
         System.out.println("Enter marks for subject "+(i+1)+":");
         subject[i].subjectMarks=s.nextInt();
         System.out.println("Enter credits for subject "+(i+1)+":");
         subject[i].credits=s.nextInt();

         subject[i].grade=(subject[i].subjectMarks/10)+1;

         if (subject[i].grade==11){
            subject[i].grade=10;
         }
         if (subject[i].grade<=4){
            subject[i].grade=0;
         }

      }
   }

   void computeSGPA(){
      int effectiveScore=0;
      int totalCreadits=0;
```

```
      for(int i=0;i<9;i++){
         effectiveScore += (subject[i].grade*subject[i].credits);
         totalCreadits += subject[i].credits;
      }

      SGPA=(double)effectiveScore/(double)totalCreadits;
   }
}
class Subject{
         int subjectMarks;
         int credits;
         int grade;

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

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

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 Book{
   String name, author;
   int price,page_num;

   Book(String name, String author, int price, int page_num){
      this.name=name;
      this.author=author;
      this.price=price;
      this.page_num=page_num;
   }

   String toStrings(){
      String name,author,price,page_num;
      name="Book name: "+this.name+"\n";
      author="Author name: "+this.author+"\n";
      price="Price: "+this.price+"\n";
      page_num="Number of pages: "+this.page_num+"\n";

      return (name+author+price+page_num);
   }

   public static void main(String args[]){
      int n;
      String name,author;
      int price,page_num;

      Scanner sc=new Scanner(System.in);

      System.out.println("Enter the no. of books:");
      n=sc.nextInt();

      Book b[]=new Book[n];

      for (int i=0;i<n;i++){
         System.out.println("Enter name of book:");
         sc.nextLine();
         name=sc.nextLine();

         System.out.println("Enter author of a book:");
         author=sc.nextLine();

         System.out.println("Enter the price of book:");
         price=sc.nextInt();

         System.out.println("Enter the no. of pages of book:");
         page_num=sc.nextInt();
```

```
            b[i]=new Book(name,author,price,page_num);
        }

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

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;
public class InputScanner {
	Scanner s;
	InputScanner(){
		s=new Scanner(System.in);
	}
}
abstract public class Shape extends InputScanner{
	double a,b;
	abstract void getInput();
	abstract void displayArea();
}
public class Rectangle extends Shape {
	void getInput() {
		System.out.println("Enter the dimension of rectangle:");
		a=s.nextDouble();
		b=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of reactangle:"+(a*b));
	}
}
public class Circle extends Shape{
	void getInput() {
		System.out.println("Enter the dimension of circle:");
		a=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of circle:"+(3.14*a*a));
	}
}
public class Triangle extends Shape {
	void getInput() {
		System.out.println("Enter the dimension of triangle:");
		a=s.nextDouble();
		b=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of triangle:"+((a*b)/2));
	}
}
public class MainMethod {

	public static void main(String[] args) {
```

```
                Rectangle R=new Rectangle();
                R.getInput();
                R.displayArea();

                Triangle T=new Triangle();
                T.getInput();
                T.displayArea();

                Circle C=new Circle();
                C.getInput();
                C.displayArea();
        }

}
```

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.

```
import java.util.Scanner;
public class Account {
	String customer_name;
	int account_no;
	String type_acc;
	double balance=0;
	Scanner sc;

	Account(String customer_name,int account_no,String type_acc){
		this.customer_name=customer_name;
		this.account_no=account_no;
		this.type_acc=type_acc;
		sc=new Scanner(System.in);
	}

	void diposite() {
		System.out.println("Enter the deposit amount:");
		int dipo=sc.nextInt();
		balance+=dipo;
	}

	void withdrawal() {
		System.out.println("Enter the withdrawl amount:");
		int with=sc.nextInt();
		if(with>balance) {
			System.out.println("Insufficient Balance");
		}
		else{
			balance-=with;
		}
	}

	void display() {
		System.out.println("Customer name:"+customer_name);
		System.out.println("Account number:"+account_no);
		System.out.println("Type of Account:"+type_acc);
		System.out.println("Balance:"+balance);
	}
	void applyinterest() {}

}

public class Cur_acct extends Account {

	Cur_acct(String customer_name,int account_no,String type_acc){
		super(customer_name,account_no,type_acc);}
```

```
	void withdrawal() {
		System.out.println("Enter the withdrawl amount:");
		int with=sc.nextInt();
		if (balance<=2000) {
			double pen=balance/(0.06);
			System.out.println("Insuffiecient balance penalty to be paid:"+pen);
			balance+=pen;
		}
		else{
			balance-=with;
		}
	}
}

public class Sav_acct extends Account{
Sav_acct(String customer_name,int account_no,String type_acc){
super(customer_name,account_no,type_acc);
}

void applyinterest() {
System.out.println("Enter the interest rate:");
int rate=sc.nextInt();
double interest=balance*rate;
balance+=interest;
System.out.println("Balance after interest:"+balance);
}
}

import java.util.Scanner;
public class Bank {
	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter customer name:");
		String cus_name=sc.nextLine();
		System.out.println("Enter account number:");
		int acc_no=sc.nextInt();

		Account ca=new Cur_acct(cus_name,acc_no,"Current Account");
		Account sa=new Sav_acct(cus_name,acc_no,"Saving Account");

		int choice;
		while(true){
			System.out.println("----MENU----");
			System.out.println("1.Deposite\n2.Withdrawl\n3.Compute intereset for Saving
			account\n4.Display account details\n5.Exit");
			choice=sc.nextInt();

			switch(choice) {
			case 1:
			System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
			int acc=sc.nextInt();
			if(acc==1) {
```

```
                    sa.diposite();
                    }
                    else {
                    ca.diposite();
                    }
                    break;
                    case 2:
                    System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
                    int acc1=sc.nextInt();
                    if(acc1==1) {
                    sa.withdrawal();
                    }
                    else {
                    ca.withdrawal();
                    }
                    break;
                    case 3:
                    sa.applyinterest();
                    break;
                    case 4:
                    System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
                    int acc2=sc.nextInt();
                    if(acc2==1) {
                    sa.display();
                    }
                    else {
                    ca.display();
                    }
                    break;
                    case 5:
                    break;
                    default:
                    System.out.println("Invalid Choice");
                    break;
                    }

                    if(choice==5) {
                    break;
                    }
                }
        }
}
```

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.

```
package CIE;
import java.util.Scanner;
public class Student{
   public String usn,name;
   public int sem;
   public void inputStudentDetails(){
      Scanner sc=new Scanner(System.in);
      System.out.println("Enter the student usn:");
      usn=sc.nextLine();
      System.out.println("Enter the student name:");
      name=sc.nextLine();
      System.out.println("Enter student semester:");
      sem=sc.nextInt();
   }

   public void disply(){
      System.out.println("Student USN:"+usn);
      System.out.println("Student Name:"+name);
      System.out.println("Student Sem:"+sem);
   }
}

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

   public Externals(){
      marks=new int[5];
      finalMarks=new int[5];
   }
```

```
    public void inputSEEMarks(){
        Scanner sc=new Scanner(System.in);
        for(int i=0;i<5;i++){
            System.out.println("Enter subject "+(i+1)+" marks:");
            marks[i]=sc.nextInt();
        }
    }

    public void calculateFinalMarks(){
        for(int i=0;i<5;i++){
            finalMarks[i]=marks[i]/2+super.marks[i];
        }
    }

    public void displayFinalMarks(){
        inputStudentDetails();
        for(int i=0;i<5;i++){
            System.out.println("Subject "+(i+1)+" final marks:"+finalMarks[i]);
        }
    }
}
import SEE.Externals;
class PackageMain{
    public static void main(String args[]){
        int numOfStudent=2;
        Externals finalMarks[]=new Externals[numOfStudent];
        for(int i=0;i<numOfStudent;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<numOfStudent;i++){
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].disply();
            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<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.

```
public class WrongAge extends Exception{
   WrongAge(String msg){
      super(msg);
   }
}
import java.util.Scanner;
class InputScanner{
   Scanner sc;
   InputScanner(){
      sc=new Scanner(System.in);
   }
}
class Father extends InputScanner{
   int fatherAge;
   Father() throws WrongAge{
      System.out.println("Enter father's age:");
      fatherAge=sc.nextInt();
         if(fatherAge<0){
         throw new WrongAge("Age cannot be negative");
      }

   }
   void display(){
      System.out.println("Father's age:"+fatherAge);
   }
}
class Son extends Father{
   int sonAge;
   Son() throws WrongAge{
      System.out.println("Enter Son's age:");
      sonAge=sc.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:"+sonAge);
   }

   public static void main(String args[]){
```

```
        try{
            Son son=new Son();
            son.display();
        }
        catch(WrongAge e){
            System.out.println(e);

        }
    }
}
```

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

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

public class ThreadMain {
   public static void main(String[] args) {
      Thread t1 = new Thread(new BMS_College_of_Engineering());
      Thread t2 = new Thread(new CSE());
      t1.start();
      t2.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(275, 150);
      jfrm.setLayout(new FlowLayout());
      // to terminate on close
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

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

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

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

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

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

        // 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

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<5) {
			q.put(i++);
		}
	}
}
```

```
class Consumer implements Runnable {
	Q q;
	Consumer(Q q) {
		this.q = q;
		new Thread(this, "Consumer").start();
	}

	public void run() {
		int i=0;
		while(i<5) {
			int r=q.get();
			System.out.println("consumed:"+r);
			i++;
		}
	}

}

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

Deadlock:

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

	void last() {
		System.out.println("Inside A.last");
	}
}

class B {
	synchronized void bar(A a) {
		String name =Thread.currentThread().getName();
		System.out.println(name + " entered B.bar");
		try {
```

```
                Thread.sleep(1000);
            } catch(Exception e) {
                System.out.println("B Interrupted");
            }
            System.out.println(name + " trying to call A.last()");
            a.last();
        }
        void last() {
            System.out.println("Inside A.last");
        }
}

class Deadlock implements Runnable
{
        A a = new A();
        B b = new B();
        Deadlock() {
            Thread.currentThread().setName("MainThread");
            Thread t = new Thread(this,"RacingThread");
            t.start();
            a.foo(b); // 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();
        }

}
```