VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Shashank U (1BM23CS314)

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 Lab (23CS3PCOOJ)carried out by **Shashank U(1BM23CS314),** 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 Object Oriented Java Programming Lab (23CS3PCOOJ) work prescribed for the said degree.

Spoorthi D M Assistant Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor &HOD Department of CSE, BMSCE

Index

Sl. No.	Date	Experiment Title	Page No.
1	26/09/2024	Quadratic Equation Solution	1-2
2	03/10/2024	SGPA Calculation	3-5
3	19/10/2024	Library program: demonstration of toString() method	6-9
4	24/10/2024	Abstract Class demonstration program	10-11
5	07/11/2024	Inheritance demonstration program	12-16
6	14/11/2024	Packages in java demonstration	17-21
7	21/11/2024	Exception handling	22-24
8	28/11/2024	Multithreaded Programming	25-26
9	19/12/2024	Open ended exercise-1: Event Handling	27-28
10	19/12/2024	Open ended exercise-2: IPC and Deadlock	29-32

Github Link:
https://github.com/Shashank-u803/OOJ-lab

Lab Programme - 01

1) Develop a Java program that prints all real solutions to the quadratic equation $ax^2+bx+c=0$. Read in a, b, c and use the quadratic formula. if discriminant $b^2-4ac<0$, display a message stating that there are no real solutions

A:

```
import java.util.Scanner;
public class Quad_solver {
    public static void main (String[] args) {
        Scanner sx = new Scanner (System.in);
        System.out.println ("Enter a, b, c values");
        double a = sx.nextDouble();
        double b = sx.nextDouble();
        double c = sx.nextDouble();
        double discriminant = (b*b) - (4*a*c);

        if (discriminant >0) {
            double r1 = (-b + Math.sqrt(discriminant)) / (2*a);
            double r2 = (-b - Math.Sqrt(discriminant)) / (2*a);
        System.out.Println ("The Eqn has real & distinct roots");
        System.out.println ("Root 1 " + r1 + "Root 2 " + r2);
        }

        else if (discriminant == 0){
            double r = (-b) / (2*a);
        System.out.println ("Real & equal roots i.e " + r);
        }
        else {
        System.out.println ("The Equation has no real solution.");
        }
        sx.close();
    }
}
```

Date __/__/__
Page _____

	o/p
Case1 :	Enter a,b,c values 1 -3 2 The Equation has real & distinct roots Root1 : 2.0 Root2 : 1.0
Case 2 :	Enter a,b,c values 1 -2 1 Real & equal roots i.e 1.0
Case 3 :	Enter a,b,c values 1 2 5 The Equation has no real solution.

24/10/24

```
import java.util.Scanner;

class Quad_Eq_cal{
   public static void main(String [] args){
      int y=0;
      Scanner sc=new Scanner(System.in);
      System.out.println("General form  of a quadratic equation is ax^2+bx+c=0");
      do{
         System.out.print("\nEnter value of a=");
         int a=sc.nextInt();
         System.out.print("Enter value of b=");
         int b=sc.nextInt();
         System.out.print("Enter value of c=");
         int c=sc.nextInt();
         float d=(float)(Math.pow(b,2)-4*a*c);
         if(d<0){
            System.out.println("There are no real solutions");
         }
         else if(d==0){
            System.out.println("It has one repeated root(2 equal roots):");
            float r=-b/(2.0f*a);
            System.out.println("x="+r);
         }
         else{
            System.out.println("It has two distinct roots:");
            double r1=((-b+Math.sqrt(d))/(2*a));
            System.out.println("x1="+r1);
            double r2=((-b-Math.sqrt(d))/(2*a));
            System.out.println("x2="+r2);
         }
         System.out.println("\nDo you want to calculate again?(yes=0 and no=1): ");
```

```
        y=sc.nextInt();
      }while(y==0);
   }
}
```

```
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> java Quad_Eq_cal
General form  of a quadratic equation is ax^2+bx+c=0

Enter value of a=1
Enter value of b=2
Enter value of c=3
There are no real solutions

Do you want to calculate again?(yes=0 and no=1):
0

Enter value of a=1
Enter value of b=-2
Enter value of c=1
It has one repeated root(2 equal roots):
x=1.0

Do you want to calculate again?(yes=0 and no=1):
0

Enter value of a=1
Enter value of b=7
Enter value of c=10
It has two distinct roots:
x1=-2.0
x2=-5.0
```

Lab-Program -2

Date __/__/__
Page

Q] 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.

A:

```
import java.util.Scanner;

class Student {
    String usn;
    String name;
    int[] credits;
    int [] marks;
    int numsubjects;

Public void acceptdetails() {
    Scanner sx = new Scanner (System.in);
    System.out.println ("Enter USN:");
     usn = sx.nextline();
    System.out.println ("Enter Name:");
     name = sx.nextline();
    System.out.println ("Enter no.of subjects:);
     numsubjects = sx.nextInt();
     credits = new int[numsubjects];
     marks = new int[numSubjects];
  for(int i=0; i<numsubjects; i++){
     System.out.println("Enter credits for sub" + (i+1)+":");
      credits[i] = sx.nextInt();
     System.out.println("Enter ~~credits~~ marks for sub" + (i+1) +":");
      marks[i] = sx.nextInt();
    }
}
```

Date __/__/__
Page

```
public void Display (){
  System.out.println ("Student USN:" +usn);
  System.out.println ("Student Name:" + name);
  For (int i=0; i <numSubjects ; i++) {
    System.out.println ( "subject " + (i+1) + ": Credits =4
      + credits[i] + "; Marks = " + marks[i]);
  }
  System.out.println ("SGPA:" + calculateSGPA ());
}

public double calculateSGPA (){
    int t_credits =0;
    int t_gradepoints =0;
  For (int i=0; i<numSubjects ; i++) {
    int gradepoint = getGradePoint (marks[i]);
      t_gradepoints += gradepoint * credits[i];
      t_credits += credits[i];
  }
  return (double) t_gradepoints /t_credits;

public int getGradePoint (int marks) {
    if (marks>=90) return 10;
  else if (marks >=80) return 9;
  else if (marks >=70) return 8;
  else if (marks >=60) return 7;
  else if (marks >=50) return 6;
  else if (marks >=40 ) return 5;
  else return 0;
  }
} // End of class Student
```

Date __/__/__
Page _____

```
public class StudentDetails {
    public static void main (String[] args) {
        Student S1 = new Student();
        S1.acceptDetails();
        S1.displayDetails();
    }
}
```

o/p

```
Enter USN : 1BM23IS007
Enter Name : Eshwar Mayank
Enter number of subjects: 3
Enter credits for subject 1: 4
Enter marks for subject 1: 85
Enter credits for subject 2: 3
Enter marks for subject 2: 75
Enter credits for subject 3: 3
Enter marks for subject 3: 65

Student USN : 1BM23IS007
Student Name : Eshwar Mayank
Subject 1: Credits = 4, Marks = 85
Subject 2: Credits = 3, Marks = 75
Subject 3: Credits = 3, Marks = 65
SGPA : 8.0
```

```
import java.util.Scanner;

class Subject {
    int subM;
    int cred;
    int grade;

    void setSubDet(int marks, int cred) {
        this.subM = marks;
        this.cred = cred;

        if (subM >= 90) {
            grade = 10;
        } else if (subM >= 80) {
            grade = 9;
        } else if (subM >= 70) {
            grade = 8;
        } else if (subM >= 60) {
            grade = 7;
        } else if (subM >= 50) {
            grade = 6;
        } else if (subM >= 40) {
            grade = 5;
        } else {
            grade = 0;
        }
    }
}

class Student {
```

```
Scanner s = new Scanner(System.in);
Subject[] subjects = new Subject[8];

Student() {
   for (int i = 0; i < subjects.length; i++) {
      subjects[i] = new Subject();
   }
}

void getMarks() {
   for (int i = 0; i < subjects.length; i++) {
      System.out.print("Enter marks for subject " + (i + 1) + ": ");
      int marks = s.nextInt();
      System.out.print("Enter credit for subject " + (i + 1) + ": ");
      int cred = s.nextInt();
      subjects[i].setSubDet(marks, cred);
   }
}

double calSGPA() {
   double Score = 0;
   int totalCred = 0;
   double SGPA = 0.0;

   for (Subject subject : subjects) {
      Score += (subject.grade * subject.cred);
      totalCred += subject.cred;
   }

   if (totalCred > 0) {
      SGPA = Score / totalCred;
```

```
        } else {
            SGPA = 0;
        }
        return SGPA;
    }
}

public class StudentDetails {

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



        System.out.print("Enter number of semesters: ");
        int numSems = sc.nextInt();

        Student[] students = new Student[numSems];
        double cumulativeSGPA = 0.0;



        System.out.print("Enter USN: ");
        String usn = sc.next();

        System.out.print("Enter Name: ");
        String name = sc.next();



        for (int i = 0; i < numSems; i++) {
            System.out.println("Enter details for semester " + (i + 1));
            students[i] = new Student();
            students[i].getMarks();
```

```
            double semSGPA = students[i].calSGPA();
            cumulativeSGPA += semSGPA;
        }


        for (int i = 0; i < numSems; i++) {
            System.out.println("USN: " + usn);
            System.out.println("Name: " + name);
            System.out.println("SGPA for sem " + (i + 1) + ": " + students[i].calSGPA());
        }


        double CGPA = cumulativeSGPA / numSems;
        System.out.println("CGPA: " + CGPA);
    }
}
```

```
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> java StudentDetails
Enter number of semesters: 1
Enter USN: 1BM22IS007
Enter Name: Dikshith
Enter details for semester 1
Enter marks for subject 1: 94
Enter credit for subject 1: 4
Enter marks for subject 2: 90
Enter credit for subject 2: 4
Enter marks for subject 3: 86
Enter credit for subject 3: 3
Enter marks for subject 4: 91
Enter credit for subject 4: 3
Enter marks for subject 5: 89
Enter credit for subject 5: 3
Enter marks for subject 6: 75
Enter credit for subject 6: 1
Enter marks for subject 7: 90
Enter credit for subject 7: 1
Enter marks for subject 8: 80
Enter credit for subject 8: 1
USN: 1BM22IS007
Name: Dikshith
SGPA for sem 1: 9.55
CGPA: 9.55
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs>
```

Date __/__/__
Page

Lab_Program-3

Q] 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.

A:

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

    public Book(String name, String author, double price
    int pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.pages = pages;
    }

    public void setName(String name) {
        this.name = name;
    }
    public void setAuthor(String author) {
        this.price = price;
                author  author
    }
    public void setPages(int pages) {
        this.pages = pages;
    }
```

Date __/__/__
Page _____

```
public void set pao setprice (double price) {
        this. price = price;
}

public String getname() { return name; }
public String get author () { return author ; }
public double get price () { return price ; }
pubic int getpage () { return pages ; }

public String to string () {
    return "Book Name :" + name + "\n Author : "
    + author + "\n Price : " + price + "\n Pages : "
    + pages ;
}
} // End of "Book" class

public class Bookdetails {
    public static void main (String [] args) {
        Scanner sx = new Scanner (System.in);
System.out.println (" Enter no. of Books : ");
    int n = sx.nextInt();
    Book [] books = new Book [n];
For (int i = 0; i<n; i++) {
    System.out.Println ("\n Enter details for Book " + (i+1)
        + ":");
    System.out.Println (" Book name : ");
    String name = sx.nextline();
    System.out.Println (" Author name : ");
    String author = sx.nextline();
    System.out.println (" Price : ");
    double price = sx.nextDouble();
    System.out.println (" Pages : ");
    int pages = sx.nextInt;
```

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

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

o/p

```
Enter the number of Books: 2

Enter details for Book 1:
Name: The Alchemist
Author: Paulo Coelho
Price: 299.99
Pages: 208
Enter details for Book 2:
Name: To kill a Mockingbird
Author: Harper lee
Price: 399.5
Pages: 3241
Book Details
Book 1:
Name: The Alchemist
Author: Paulo Coelho
Price: 299.99
Pages: 208
```

Date __/__/__
Page _____

Book 2 :

Name: To Kill a Mocking bird

Author: Harper lee

Price: 399.5

Pages: 324

—x—

24/10/24

```
import java.util.Scanner;
class Book {
   String name, author;
   double price;
   int noPage;
   Book() {}
   Book(String name, String author, double price, int noPage) {
      this.name = name;
      this.author = author;
      this.price = price;
      this.noPage = noPage;
   }
   void setDetails() {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter name of book: ");
      name = sc.nextLine();
      System.out.println("Enter author name: ");
      author = sc.nextLine();
      System.out.println("Enter price of book: ");
      price = sc.nextDouble();
      System.out.println("Enter number of pages: ");
      noPage = sc.nextInt();
   }
   void getDetails() {
      System.out.println("Name of book: " + name);
      System.out.println("Author: " + author);
      System.out.println("Price: " + price);
      System.out.println("Number of pages: " + noPage);
   }
   public String toString() {
```

```
        return "Book name: " + name + "\n" + "Author: " + author + "\n" + "Price: " + price + "\n" +
"Number of pages: " + noPage + "\n";
    }
}
class MyBook {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of books: ");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            books[i] = new Book();
            System.out.println("Enter details for book " + (i + 1));
            books[i].setDetails();
            books[i].getDetails();
        }
        System.out.println("All book details: ");
        for (Book book : books) {
            System.out.println(book);
        }
    }
}
```

```
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> java MyBook
Enter number of books:
2
Enter details for book 1
Enter name of book:
The Life of an atheist
Enter author name:
Shakespeare
Enter price of book:
450
Enter number of pages:
250
Name of book: The Life of an atheist
Author: Shakespeare
Price: 450.0
Number of pages: 250
Enter details for book 2
Enter name of book:
The Alchemist
Enter author name:
Robert blair
Enter price of book:
200
Enter number of pages:
140
Name of book: The Alchemist
Author: Robert blair
Price: 200.0
Number of pages: 140
All book details:
Book name: The Life of an atheist
Author: Shakespeare
Price: 450.0
Number of pages: 250

Book name: The Alchemist
Author: Robert blair
Price: 200.0
Number of pages: 140

PS C:\Users\Shashank U\Desktop\OOJ\Lab programs>
```

Lab program - 4

Date __/__/__
Page

① Develop a java program to create an abstract class named shape that contains two integers and an empty method named printArea(), provide 3 classes named □, △, ○ such that each of of them extend the class Shape. Each of the class contain only one method named printArea that prints area of the given shape

```
abstract class shape{
  int a, b;
  abstract double printarea();
}

class rectangle extends shape{
  rectangle (int x, int y){
    this.a = x;
    this.b = y;
  }
  double printarea (){
      return a*b;
  }
}

class triangle extends shape {
  triangle (int x, int y){
      this.a = x;
      this.b = y;
  }
  double printarea (){
      return 0.5 * a * b;
  }
}
```

```
class circle extends Shape {
final double pi = 3.14159;
circle(int r) {
    this.a = r;
} :
double printarea () {
    return pi * a * a;
}
}

class AbstractDemo {
public static void main (String args[]) {
    rectangle o1 = new rectangle (7,25);
    triangle o2 = new triangle (15,20);
    circle o3 = new circle (5);
    Shape sh;
    sh = o1;
    double a1 = sh.printarea();
System.out.println ("Area of rectangle:" + a1);
    sh = o2;
    double a2 = sh.printarea();
System.out.println ("Area of triangle:" + a2);
    sh = o3;
    double a3 = sh.printarea();
    System.out.println ("Area of circle:" + a3);
}
}
```

o/p

```
Area of rectangle : 175.0
Area of triangle : 150.0
Area of circle : 78.53975
```

```
import java.util.Scanner;


abstract class Shape {
    int dimension1;
    int dimension2;


    abstract void printArea();
}

class Rectangle extends Shape {


    public Rectangle(int length, int width) {
        this.dimension1 = length;
        this.dimension2 = width;
    }


    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {


    public Triangle(int base, int height) {
        this.dimension1 = base;
```

```
        this.dimension2 = height;
    }




    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Triangle Area: " + area);
    }
}

class Circle extends Shape {
    private final double pi = 3.14159;



    public Circle(int radius) {
        this.dimension1 = radius;
        this.dimension2 = 0;
    }



    void printArea() {
        double area = pi * dimension1 * dimension1;
        System.out.println("Circle Area: " + area);
    }
}
public class Abstract_ {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
```

```
        System.out.print("Enter length of rectangle: ");
        int length = scanner.nextInt();
        System.out.print("Enter width of rectangle: ");
        int width = scanner.nextInt();
        Rectangle rectangle = new Rectangle(length, width);
        rectangle.printArea();


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

        System.out.print("Enter radius of circle: ");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius);
        circle.printArea();

        scanner.close();
    }
}
```

```
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> javac Abstract_.java
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> java Abstract_
Enter length of rectangle: 5
Enter width of rectangle: 4
Rectangle Area: 20
Enter base of triangle: 10
Enter height of triangle: 9
Triangle Area: 45.0
Enter radius of circle: 7
Circle Area: 153.93791
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs>
```

Date __/__/__
Page

Lab program - 5

Q] Develop a Java Program to create a class Bank that maintains two kinds of accounts for its customers, one is savings, other one is current account, the savings account provides compound interest & withdrawal facilities but no checkbook facility, while current account provide checkbook facility but no interest. Current acco holders should 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, Acc no and type of Account. From this derive the classes Curr-acct and Sav-acct to make the more specific to their requirements. Include the necessary methods in order to achieve the follo tasks.

a) Accept deposit from customer & update the balance

b) Display the balance

c) Compute & deposit interest

d) Permit withdrawal & update the balan

Check for minimum balance, impose penalty if necessary & update the balance

Date __/__/__
Page _____

A:

```
import java.util.Scanner;

class Account {
protected String customer_name;
protected int Acc_no;
protected double balance;

public Account (String customerName, int accountNumber, double balance){
    Customer_name = customerName;
    Acc_no = accountNumber;
    this.balance = balance;
}

public void deposit (double amount) {
    if (amount > 0) {
        balance += amount;
        System.out.println ("Deposited:" + amount);
    } else {
        System.out.println ("Invalid amount");
    }
}

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

class SavAcct extends Account {
    final double interest = 0.05;

public SavAcct (String customer_Name, int accountnumber, double balance)
    {
        super (customerName, accountnumber, balance);
```

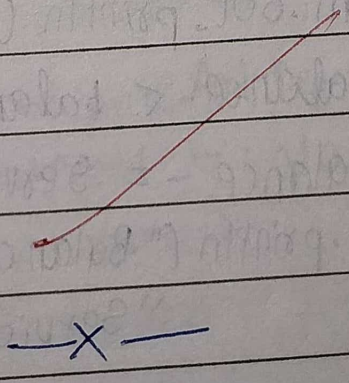
Date / /
Page

```
public void CompandIntrest () {
    int time, pamount;
    Scanner sx = new Scanner (System.in);
    System.out.println ("Enter Principle amount & time
    time = sx.nextInt ();
    pamount = sx.nextInt ();
    double c1 = ((Math.pow ((1+(intrest/12)), time*12)) *
     pamount;
    System.out.println ("Amount to pay monthly : " + c1);
}

public void computeAndDeposit Interest () {
    double Interest = balance * interestRate;
    balance += Interest;
    System.out.println ("Interest:" +Intrest +" Added!"
}

public void withdraw (double amount) {
    if (amount > 0 && amount <= balance) {
        balance -= amount;
        system.out.println ("withdrawn:" + amount);
    }
    else {
    System.out.println ("Insufficient balance / invalid
    }
}
} // End of SavingsAcc
```

```
class CurrAcc extends Account{
    final double balancereq = 500;
    final double serviceCharge = 50;

public CurrAcc(String customerName, int accountNumber, double balance){
    super(customerName, accountNumber, balance);
}

public void Checkbook(double amount){
    if(amount >0 && amount <= balance){
        balance -= amount;
        System.out.println("withdrawn :"+amount);
        if(balance < balancereq){
            balance -= serviceCharge;
        System.out.println("Balance below minimum." +
                           "Service charge 50 deducted");
        }
        else{
        System.out.println("Insufficient Balance or invalid amount");
        }
    }
}//End of Savings class

public class Bank{
    public static void main(String[] xx){
        Scanner Sx = new Scanner(System.in);

    CurAcct a1 = new CurrAcct("Rohit", 12345, 1000);
    System.out.println("Current Account:");
     a1.deposit(200);
     a1.displayBalancle();
     a1.withdraw(800);
     a1.displaybalance();
```

Date __/__/__
Page _____

```
SavAcct a2 = new SavAcct ("Sujay", 67890, 150
System.out.println("Savings Account");
    a2.deposit(500);
    a2.displayBalance();
    a2.withdraw(200);
    a2.displayBalance();
    a2.compoundInterest();
    a2.display
    a2.computeAndDepositInterest();
    a2.displayBalance();
    sx.close();
  }
}
```



```
import java.util.Scanner;

class Account {
    protected String customerName;
    protected String accountNumber;
    protected double balance;

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

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposit successful. Updated balance: " + balance);
        } else {
            System.out.println("Invalid deposit amount.");
        }
    }

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

class SavAcct extends Account {
    private double interestRate;
```

```
    public SavAcct(String customerName, String accountNumber, double initialBalance, double
interestRate) {
        super(customerName, accountNumber, initialBalance);
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest(int years) {
        double interest = balance * Math.pow((1 + interestRate / 100), years) - balance;
        balance += interest;
        System.out.println("Interest added: " + interest);
        System.out.println("Updated balance: " + balance);
    }

    public void withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawal successful. Updated balance: " + balance);
        } else {
            System.out.println("Insufficient funds or invalid withdrawal amount.");
        }
    }
}

class CurAcct extends Account {
    private double minimumBalance;
    private double penalty;

    public CurAcct(String customerName, String accountNumber, double initialBalance, double
minimumBalance, double penalty) {
        super(customerName, accountNumber, initialBalance);
        this.minimumBalance = minimumBalance;
        this.penalty = penalty;
```

```
    }

    public void withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            if (balance < minimumBalance) {
                balance -= penalty;
                System.out.println("Penalty imposed: " + penalty);
            }
            System.out.println("Withdrawal successful. Updated balance: " + balance);
        } else {
            System.out.println("Insufficient funds or invalid withdrawal amount.");
        }
    }
}

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

        SavAcct savingsAccount = new SavAcct("Alice", "S123", 1000.0, 5.0);
        CurAcct currentAccount = new CurAcct("Bob", "C123", 2000.0, 500.0, 50.0);

        System.out.println("1. Savings Account\n2. Current Account");
        System.out.print("Select account type: ");
        int accountType = scanner.nextInt();

        if (accountType == 1) {
            System.out.println("Savings Account Selected.");
            savingsAccount.displayBalance();
            savingsAccount.deposit(500);
```

```
            savingsAccount.computeAndDepositInterest(2);
            savingsAccount.withdraw(300);
        } else if (accountType == 2) {
            System.out.println("Current Account Selected.");
            currentAccount.displayBalance();
            currentAccount.deposit(1000);
            currentAccount.withdraw(1800);
        } else {
            System.out.println("Invalid account type.");
        }

        scanner.close();
    }
}
```

```
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> javac Bank.java
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> java Bank
1. Savings Account
2. Current Account
Select account type: 1
Savings Account Selected.
Account Balance: 1000.0
Deposit successful. Updated balance: 1500.0
Interest added: 153.75
Updated balance: 1653.75
Withdrawal successful. Updated balance: 1353.75
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> java Bank
1. Savings Account
2. Current Account
Select account type: 2
Current Account Selected.
Account Balance: 2000.0
Deposit successful. Updated balance: 3000.0
Withdrawal successful. Updated balance: 1200.0
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs>
```

Lab program - 6

Date 14/11/24
Page

Q] Create a package CIE which has two classes. ~~Personal~~ (Student) and Internals. The class Personal has members like USN, Name, Sem. The class internals has an array that stores the internal marks stored for 5 courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of ~~Personal~~ Student. This class has an array that stores the SEE Marks scored in 5 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 5 courses.

→ CIE/Internals.java

```
package CIE;
import java.util.Scanner;
public class Internals {
    public int marksCIE[] = new Int[5];
    public void getMarks() {
        for(int i=0; i<5; i++){
            Scanner sc = new Scanner(System.in);
            System.out.println("Enter CIE Marks out of
            50 in subject " + (i+1));
            marksCIE[i] = sc.nextInt();
        }
    }
    public int returnCIEMarks(int i){
        return marksCIE[i];
    }
}
```

Date __/__/__
Page

CIE / Student.java

```
package CIE;
import java.util.Scanner;
public class Student{
    String usn;
    String name;
    int sem;
    public void getId(){
     Scanner sc = new Scanner(System.in);
     System.out.println("Enter Student USN");
     usn = sc.nextLine();
     System.out.println("Enter Student name");
     name = sc.nexline();
     System.out.println("Enter semester");
     sem = sc.nextInt();
    }
public void display(){
  System.out.println();
  System.out.println("Student USN:" + usn);
  System.out.println("Student name:" + name);
  System.out.println("Semester:" + sem);
  System.out.println();
}
}
```

→ SEE/Externals.java

```
package SEE;
import CIE.Student;
import CIE.Internals;
import java.util.Scanner;
```

```
public class Externals extends Student{
    int marksSEE[] = new int[5];
    public void getmarks(){
        for(int i=0; i<5; i++){
            Scanner sc = new Scanner(System.in);
            System.out.println("Enter SEE marks in
            subject " + (i+1));
            marksSEE[i] = sc.nextInt();
        }
    }
    public void calcTotalMarks(Internals il){
        for(int i=0; i<5; i++){
            System.out.println("subject " + (i+1) + " : "
            + (il.returnCIEMarks[i] + (marksSEE[i]/2));
        }
        System.out.println();
    }
}
```

Main.java

```
import CIE.*;
import SEE.Externals;
import java.util.Scanner;
public class Main{
    public static void main(String args[]){
        Scanner sx = new Scanner(System.in);
        System.out.println("Enter the no. of students");
        int n = sx.nextInt();
        Internals[] il = new Internals[n];
        Externals[] el = new Externals[n];
```

→

```
for (int i=0; i<n; i++) {
    System.out.println("Student " + (i+1) + "deta
  e1[i] = new Externals();
  i1[i] = new Internals();
  e1[i].getId();
  e1[i].getMarks();
  i1[i].getmarks();
 }
for (int i=0; i<n; i++) {
  e1[i].display();
  e1[i].calcTotalMarks(i1[i]);
 }
}
}
```

o/p

→ Enter the no. of students
1

Student 1 details:
Enter student USN
1BM22IS007
Enter student Name:
DIKSHITH
Enter Semester:
2
Enter CIE marks in subject 1
90
Enter CIE marks in subject 2
85
Enter CIE marks in subject 3
83
Enter CIE marks in subject 4
90

Enter CIE Marks in Subject 5
86
~~Enter CIE~~
Enter SEE marks in Subject 1
98
Enter SEE marks in Subject 2
96
Enter SEE marks in Subject 3
86
Enter SEE marks in Subject 4
80
Enter SEE marks in Subject 5
84

Student USN : 1BM22IS007
Student name : Dikshith
Semester : 2

Subject 1 : 139
Subject 2 : 133
Subject 3 : 126
Subject 4 : 130
Subject 5 : 128

o/p seen
21/11/24

```
package SEE;

import CIE.Internals;
import CIE.Student;
import java.util.Scanner;

public class Externals extends Student {
  int[] marksSee = new int[5];

  public Externals() {
  }

  public void getMarks() {
    for(int var1 = 0; var1 < 5; ++var1) {
      Scanner var2 = new Scanner(System.in);
      System.out.println("Enter SEE marks in subject " + (var1 + 1));
      this.marksSee[var1] = var2.nextInt();
    }

  }

  public void calcTotalMarks(Internals var1) {
    for(int var2 = 0; var2 < 5; ++var2) {
      System.out.println("Subject " + (var2 + 1) + ": " + (var1.returnCieMarks(var2) +
this.marksSee[var2] / 2));
    }

    System.out.println();
  }
}
```

```
package CIE;
import java.util.Scanner;
public class Internals {
	public int marksCie[] = new int[5];
	public void getMarks() {
		for(int i=0;i<5;i++) {
			Scanner sc = new Scanner(System.in);
			System.out.println("Enter CIE marks in subject "+(i+1));
			marksCie[i]=sc.nextInt();
		}
	}
	public int returnCieMarks(int i) {
		return marksCie[i];
	}
}

import CIE.Student;
import CIE.Internals;
import SEE.Externals;
import java.util.Scanner;
public class Main {
	public static void main(String args[]) {
		Scanner sc = new Scanner(System.in);
		System.out.println("Enter the number of students whose details you want to enter");
		int n = sc.nextInt();
		Internals[] i1 = new Internals[n];
		Externals[] e1 = new Externals[n];
		for(int i=0;i<n;i++) {
			System.out.println("Student "+(i+1)+" details:");
			e1[i] = new Externals();
			i1[i] = new Internals();
			e1[i].getd();
```

```
                        i1[i].getMarks();
                        e1[i].getMarks();
                }
                for(int i=0;i<n;i++) {
                        e1[i].display();
                        e1[i].calcTotalMarks(i1[i]);
                }

        }
}

package CIE;

import java.util.Scanner;

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

  public Student() {
  }

  public void getd() {
    Scanner var1 = new Scanner(System.in);
    System.out.println("Enter student USN");
    this.usn = var1.nextLine();
    System.out.println("Enter student name");
    this.name = var1.nextLine();
    System.out.println("Enter semester");
    this.sem = var1.nextInt();
  }
```

```
    public void display() {
      System.out.println();
      System.out.println("Student USN: " + this.usn);
      System.out.println("Student name: " + this.name);
      System.out.println("Semester: " + this.sem);
      System.out.println();
    }
  }
```

```
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> javac CIE/Internals.java CIE/Student.java
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> javac SEE/Externals.java
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> javac Main.java
PS C:\Users\Shashank U\Desktop\OOJ\Lab programs> java -cp . Main
>>
Enter the number of students whose details you want to enter
1
Student 1 details:
Enter student USN
1BM22IS007
Enter student name
Dikshith
Enter semester
2
Enter CIE marks in subject 1
90
Enter CIE marks in subject 2
85
Enter CIE marks in subject 3
83
Enter CIE marks in subject 4
90
Enter CIE marks in subject 5
86
Enter SEE marks in subject 1
98
```

```
Enter SEE marks in subject 1
98
Enter SEE marks in subject 2
96
Enter SEE marks in subject 1
98
Enter SEE marks in subject 2
96
Enter SEE marks in subject 3
Enter SEE marks in subject 2
96
Enter SEE marks in subject 3
Enter SEE marks in subject 3
86
Enter SEE marks in subject 4
86
Enter SEE marks in subject 4
80
Enter SEE marks in subject 5
84

Student USN: 1BM22IS007
Student name: Dikshith
Semester: 2

Subject 1: 139
Subject 2: 133
Subject 3: 126
Subject 4: 130
Subject 5: 128
```

Date __/__/__
Page

Lab program - 7

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

→

```
import java.util.Scanner;
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}

class Father {
    protected int age;
    public Father(int age) throws wrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age cannot
            be negative");
        }
    this.age = age;
}
}

class Son extends Father {
    private int sonAge;
```

```
public Son (int fatherAge, int sonAge) throws WrongAgeException {
    super (fatherAge);
    if (sonAge < 0) {
        throw new WrongAgeException ("Son's age
        cannot be negative:");
    }
    if (sonAge >= fatherage) {
        throw new WrongAgeException ("Son's age cannot
        be greater than or equal to father's age");
    }
    this.sonAge = sonAge;
}

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

public class Demo {
    public static void main (String [] args) {
        try {
            int fatherAge = 45;
            int sonAge = 20;

            Son son = new Son (fatherAge, sonAge);
            son.displayAges ();
        }
        catch (WrongAgeException e) {
            System.out.println ("Caught Exception:" + e.getMessage());
        }
    }
}
```

Date __/__/__
Page

O/P

when fatherage = 5 SonAge = 25
Caught Exception: Son's Age cannot be greater than or equal to Father's age.

when Father age = -1 Son Age = 20
Caught Exception: Father's age cannot be negative

when Fatherage = 45 son Age = -1
Caught Exception: Son's Age cannot be negative.

off seen
21/4/24

```
class WrongAgeException extends Exception {
    private String message;
    public WrongAgeException(String message) {
        this.message=message;
    }
    public String toString(){
            return message;
  }
}



class Father {
    protected int age;
     public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age cannot be negative.");
        }
        this.age = age;
    }
}

class Son extends Father {
    private int sonAge;
             public Son(int fatherAge, int sonAge) throws WrongAgeException {
        super(fatherAge);
        if (sonAge < 0) {
            throw new WrongAgeException("Son's age cannot be negative.");
        }
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age cannot be greater than or equal to Father's age.");
        }
        this.sonAge = sonAge;
    }
```

```
    public void displayAges() {
        System.out.println("Father's Age: " + age);
        System.out.println("Son's Age: " + sonAge);
    }
}

public class InheritanceExceptionDemo {
    public static void main(String[] args) {
        try {

            int fatherAge = 45;
            int sonAge = -5;

           Son son = new Son(fatherAge, sonAge);
           son.displayAges();

        } catch (WrongAgeException e) {
            System.out.println("Caught Exception: " + e);
        }
    }
}
```

```
PS C:\Users\Admin\Desktop\1BM23CS314> java InheritanceExceptionDemo
Father's Age: 45
Son's Age: 20
PS C:\Users\Admin\Desktop\1BM23CS314> javac InheritanceExceptionDemo.java
PS C:\Users\Admin\Desktop\1BM23CS314> java InheritanceExceptionDemo
Caught Exception: Son's age cannot be greater than or equal to Father's age.
PS C:\Users\Admin\Desktop\1BM23CS314> javac InheritanceExceptionDemo.java
PS C:\Users\Admin\Desktop\1BM23CS314> java InheritanceExceptionDemo
Caught Exception: Father's age cannot be negative.
PS C:\Users\Admin\Desktop\1BM23CS314> javac InheritanceExceptionDemo.java
PS C:\Users\Admin\Desktop\1BM23CS314> java InheritanceExceptionDemo
Caught Exception: Son's age cannot be negative.
PS C:\Users\Admin\Desktop\1BM23CS314>
```

Date 28/11/24
Page

Week-8

Q] 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 CollegeThread extends Thread {
    public void run () {
        try {
            while (true) {
                System.out.println ("BMS College of Engineering");
                Thread.sleep (10000);
            }
        } catch (InterruptedException e) {
            System.out.println ("CollegeThread interrupted");
        }
    }
}

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

Date __/__/__
Page ____

```
public class ThreadDemo {
    public static void main (String xx [])  {
        CollegeThread ct = new CollegeThread ();
        DepartmentThread dt = new DepartmentThread ();

        ct.start ();
        dt.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
:
:
:

o/p not sure

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

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

public class ThreadDemo {
    public static void main(String[] args) {
        CollegeThread ct = new CollegeThread();
        DepartmentThread dt = new DepartmentThread();

        // Start the threads
```

```
        ct.start();
        dt.start();
    }
}
```

```
PS C:\Users\Shashank U> cd desktop
PS C:\Users\Shashank U\desktop> javac ThreadDemo.java
PS C:\Users\Shashank U\desktop> java ThreadDemo
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
BMS College of Engineering
CSE
CSE
```

Week-9

Open Ended Exercise-1

Date 19/12/24
Page

Q] Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 & Num2. The division of Num1 & 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.

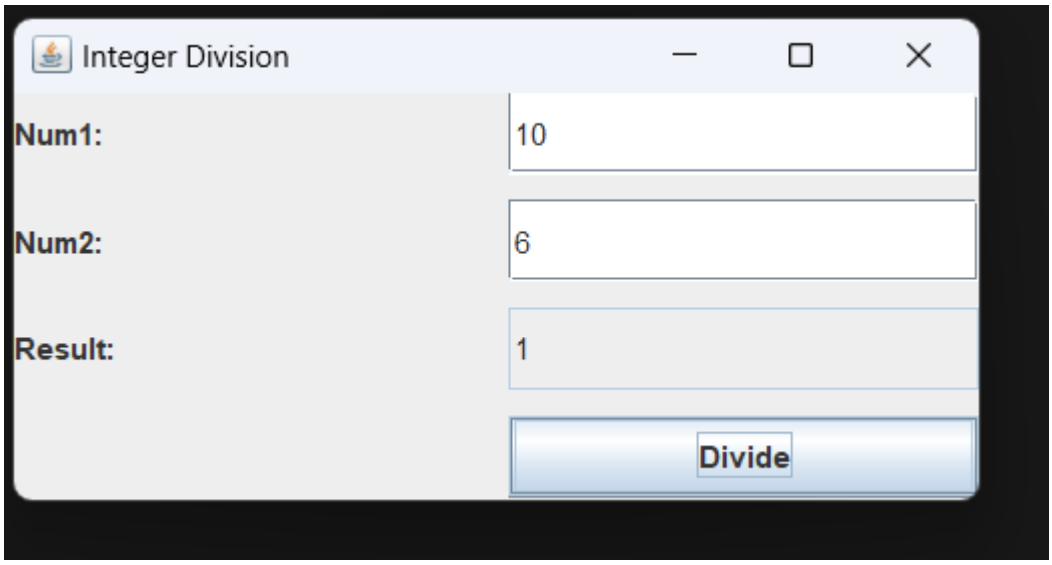
→
```
import java.swing.*;
import java.awt.*;
import java.awt.*;

public class IntegerDivisionApp {
    public static void main(String xx[]) {
        Jframe frame = new Jframe ("Integer Division");
        frame.SetDefaultCloseOperation(
            Jframe.EXIT_ON_CLOSE);
        Frame.setSize(400,200);
        frame.setLayout(new GridLayout(4,2,10,10));

        JLabel labelNum1 = new Jlabel("Num1:");
        JTextfield textNum1 = new JTextfield();
        JLabel labelNum2 = new Jlabel("Num2:");
        JTextField textNum2 = new JTextField();
        JLabel labelResult = new Jlabel("Result:");
        JTextfield textResult = new JTextField();
        textResult.setEditable(false);
        JButton divideButton = new JButton("Divide");
```

```
frame.add (labelNum1);
frame.add (textNum1);
frame.add (labelNum2);
frame.add (textNum2);
frame.add (labelResult);
frame.add (textResult);
frame.add (new JLabel());
frame.add (divideButton);

divideButton.addActionListener (new ActionListener()
    {
        public void actionPerformed (ActionEvent e) {
          try {
        int num1 = Integer.parseInt (textNum1.getText
        int num2 = Integer.parseInt (textNum2.getText()
        if (num2==0) {
          throw new ArithmeticException ("Cannot divide by zero
        }
        int res = num1/num2;
        textResult.setText (String.valueOf (result));
        }
        catch (NumberFormatException ex) {
        JoptionPane.showMessageDialog (frame, "Please Enter
        valid Integers for Num1 & Num2.", "Number
        Format Error", JoptionPane.ERROR_MESSAGE);
      } catch (ArithmeticException e) {
        JoptionPane.showMessageDialog (frame, ex.getM
        -essage(), "Arithmetic Error", JoptionPane.ERRO
        MESSAGE);
      }
    }
);
frame.setVisible (true);
}
}
```

```
import java.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class IntegerDivisionApp {
    public static void main(String[] args) {
        // Create the main frame
        JFrame frame = new JFrame("Integer Division");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 200);
        frame.setLayout(new GridLayout(4, 2, 10, 10));

        // Create and add components
        JLabel labelNum1 = new JLabel("Num1:");
        JTextField textNum1 = new JTextField();
        JLabel labelNum2 = new JLabel("Num2:");
        JTextField textNum2 = new JTextField();
        JLabel labelResult = new JLabel("Result:");
        JTextField textResult = new JTextField();
        textResult.setEditable(false);
        JButton divideButton = new JButton("Divide");

        // Add components to the frame
        frame.add(labelNum1);
        frame.add(textNum1);
        frame.add(labelNum2);
        frame.add(textNum2);
        frame.add(labelResult);
        frame.add(textResult);
        frame.add(new JLabel()); // Empty cell
        frame.add(divideButton);
```

```
    // Add action listener to the button
    divideButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            try {
                // Parse inputs
                int num1 = Integer.parseInt(textNum1.getText());
                int num2 = Integer.parseInt(textNum2.getText());

                // Perform division
                if (num2 == 0) {
                    throw new ArithmeticException("Division by zero is not allowed.");
                }

                int result = num1 / num2;
                textResult.setText(String.valueOf(result));
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(frame,
                        "Please enter valid integers for Num1 and Num2.",
                        "Number Format Error",
                        JOptionPane.ERROR_MESSAGE);
            } catch (ArithmeticException ex) {
                JOptionPane.showMessageDialog(frame,
                        ex.getMessage(),
                        "Arithmetic Error",
                        JOptionPane.ERROR_MESSAGE);
            }
        }
    });

    // Make the frame visible
    frame.setVisible(true);
}
```

}


Integer Division
Num1:
10
Num2:
6
Result:
1
Divide

Week - 10 : Open Ended Exercise II

Date 19/12/24
Page

Q] Demonstrate Inter process Communication & deadlock

→

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

class B {
    synchronised 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();
    }
```

```
synchronized void last() {
    System.out.println("Inside B.last");
  }
}

class Deadlock implements Runnable {
  A a = new A();
  B b = new B();
  Thread t;

  Deadlock() {
    Thread.currentThread().setName("Main Thread");
    t = new Thread(this, "Racing Thread");
  }

  void deadlockStart() {
    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 xx[]) {
    Deadlock d1 = new Deadlock();
    d1.deadlockStart();
  }
}
```

Date 19/12/24
Page

```
// Interprocess Communication

Class Q {
    int n; boolean valueSet = false;
    synchronized int get() {
        while(!valueSet) {
            try { wait(); }
            catch (InterruptedException e) {
            System.out.println(e.getMessage());
            }
        System.out.println("Got : " +n);
        valueSet = false;
        notify();
        return n;
    }
    Synchronized void put(int n) {
        while(valueSet) {
            try { wait(); }
            catch (InterruptedException e) {
                System.out.println("Int. excep. caught");
            }
        this.n = n;
        valueSet = true;
        System.out.println("Put : " +n);
        notify();
    }
}
class Producer implements Runnable {
    Q q; Thread t;
    Producer(Q q) {
        this.q = q;
        t = new Thread(this, "Producer");
    }
```

```
public void run() {
    int i=0;
    while (true) {
        q.put(i++);
    }
}

class Consumer implements Runnable {
    Q q; Thread t;
    Consumer (Q q) {
        this.q = q;
        t = new Thread (this, "Consumer");
    }
    public void run () {
        while (true) {
            q.get();
        }
    }
}

class PCFixed {
    public static void main (String xx[]) {
        Q q = new Q();
        Producer p = new Producer (q);
        Consumer c = new Consumer (q);
        p.t.start();
        c.t.start();
        System.out.println("Press Ctrl c to stop");
    }
}
```

```
class Q {
    int n;
    boolean valueSet = false;

    synchronized int get() {
        while (!valueSet) {
            try {
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        }
        System.out.println("Got: " + n);
        valueSet = false;
        notify();
        return n;
    }

    synchronized void put(int n) {
        while (valueSet) {
            try {
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        notify();
    }
}
```

```
class Producer implements Runnable {
   Q q;
   Thread t;

   Producer(Q q) {
      this.q = q;
      t = new Thread(this, "Producer");
   }

   public void run() {
      int i = 0;
      while (true) {
         q.put(i++);
      }
   }
}

class Consumer implements Runnable {
   Q q;
   Thread t;

   Consumer(Q q) {
      this.q = q;
      t = new Thread(this, "Consumer");
   }

   public void run() {
      while (true) {
         q.get();
      }
   }
}
```

```
class PCFixed {
    public static void main(String args[]) {
        Q q = new Q();
        Producer p = new Producer(q);
        Consumer c = new Consumer(q);

        // Start the threads.
        p.t.start();
        c.t.start();

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

```
Put: 23034
Got: 23034
Put: 23035
Got: 23035
Put: 23036
Got: 23036
Put: 23037
Got: 23037
Put: 23038
Got: 23038
Put: 23039
Got: 23039
Put: 23040
Got: 23040
Put: 23041
Got: 23041
Put: 23042
Got: 23042
Put: 23043
Got: 23043
Put: 23044
Got: 23044
Put: 23045
Got: 23045
Put: 23046
Got: 23046
Put: 23047
Got: 23047
Put: 23048
Got: 23048
Put: 23049
Got: 23049
Put: 23050
Got: 23050
```

DEADLOCK Program

```
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();
   }

   synchronized 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();
    }

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

class Deadlock implements Runnable {
    A a = new A();
    B b = new B();
    Thread t;

    Deadlock() {
        Thread.currentThread().setName("MainThread");
        t = new Thread(this, "RacingThread");
    }

    void deadlockStart() {
        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[]) {
        Deadlock dl = new Deadlock();
        dl.deadlockStart();
    }
```

```
}
```

```
PS C:\Users\Shashank U\Desktop\NOTES\Object oriented java programming> javac Deadlock.java
PS C:\Users\Shashank U\Desktop\NOTES\Object oriented java programming> java Deadlock
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
MainThread trying to call B.last()
```