VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Kishore Chandra N (1BM23CS154)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)

BENGALURU-560019
Sep-2024 to Jan-2025

B.M.S. College of Engineering,

Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "Object Oriented Java Programming (23CS3PCOOJ)" carried out by **Kishore Chandra N (1BM23CS154),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

Ambuja K Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link: https://github.com/Kishore868/OOJ-Lab-Programs

<u>Program 1</u>

Implement Quadratic Equation

Algorithm:

Lab Programs.

1. Develop Java program that prints all real solution to QE $ax^2+bx+c=0$. Read in a, b, c and use formula. If discriminant b^2-4ac = -ve, display message stating no real solutions.

A.
```
import static java.lang.Math.sqrt;
import java.util.Scanner;

public class QE
{       int a,b,c;
        double r1, r2, d;
```

```
void input()
{       Scanner sc = new Scanner (System.in);
        System.out.println ("Enter value of a : ");
        a = sc.nextInt();
        while (a==0)
        {       System.out.println ("Enter a non-zero number for a : ");
                a = sc.nextInt();
                d = b*b - 4*a*c;
        }
void display()
{       if (d==0) {
            r1 = -b/(2.0*a);
            System.out.println ("Roots are real and equal");
            System.out.println ("Root : "+r1);
        }
        else if (d>0) {
            r1 = (-b+ sqrt(d))/(2.0*a);
            r2 = (-b- sqrt(d))/(2.0*a);
            System.out.println ("Roots are real and distinct");
            System.out.println (" r1= "+r1+ ", r2 = "+r2);
        }
        else {
            r1 = -b/(2.0*a);
            r2 = sqrt(-d)/(2.0*a);
```

```
System.out.println("Roots are imaginary");
System.out.println("r1 = " + r1 + "+" + r2 + "i");
System.out.println("r2 = " + r1 + "- " + r2 + "i");
}
}
public static void main (String[] args) {
    QE qe = new QE();
    qe.input();
    qe.display();
  }
}
```

Output :

Enter value of a : 1
Enter value of b : -4
Enter value of c : 4
Roots are real and equal
Root : 2.0

Enter value of a : 2
Enter value of b : 2
Enter value of c : 2
Roots are imaginary
r1 = -0.5 + 0.8660i
r2 = -0.5 - 0.8660i.

Enter value of a : 1
Enter value of b : -9
Enter value of c : 10
Roots are real and different
r1 = 7.701 , r2 = 1.2984

Code:

```
import static java.lang.Math.sqrt;
import java.util.Scanner;

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

   void input() {
   Scanner sc = new Scanner(System.in);
   System.out.print("Enter value of a: ");
      a = sc.nextInt();

      while (a == 0) {
         System.out.println("Enter a non-zero number for a:");
         a = sc.nextInt();
      }
     System.out.print("Enter value of b: ");
      b = sc.nextInt();
      System.out.print("Enter value of c: ");
      c = sc.nextInt();
```

```
        d = b * b - 4 * a * c;
    }
void display() {
        if (d == 0) {
            r1 = -b / (2.0 * a);
            System.out.println("Roots are real and equal");
            System.out.println("Root: " + r1);
System.out.println("1BM23CS154");
            System.out.println("Kishore Chandra N");
        } else if (d > 0) {
            r1 = (-b + sqrt(d)) / (2.0 * a);
            r2 = (-b - sqrt(d)) / (2.0 * a);
            System.out.println("Roots are real and different");
            System.out.println("r1 = " + r1 + ", r2 = " + r2);
System.out.println("1BM23CS154");
            System.out.println("Kishore Chandra N");
        } else {
            r1 = -b / (2.0 * a);
            r2 = sqrt(-d) / (2.0 * a);
            System.out.println("Roots are imaginary");
            System.out.println("r1 = " + r1 + " + " + r2 + "i");
            System.out.println("r2 = " + r1 + " - " + r2 + "i");
                System.out.println("1BM23CS154");
            System.out.println("Kishore Chandra N");
        }
    }
  public static void main(String[] args) {
        QE qe = new QE();
        qe.input();
        qe.display();
    }
}
```

Output:

```
D:\1BM23CS154>java QE.java
Enter value of a: 1
Enter value of b: -9
Enter value of c: 10
Roots are real and different
r1 = 7.701562118716424, r2 = 1.2984378812835757
1BM23CS154
Kishore Chandra N
```

```
D:\1BM23CS154>java QE.java
Enter value of a: 2
Enter value of b: 2
Enter value of c: 2
Roots are imaginary
r1 = -0.5 + 0.86602540378443864i
r2 = -0.5 - 0.86602540378443864i
1BM23CS154
Kishore Chandra N
```

```
D:\1BM23CS154>java QE.java
Enter value of a: 1
Enter value of b: -4
Enter value of c: 4
Roots are real and equal
Root: 2.0
1BM23CS154
Kishore Chandra N
```

Program 2

Develop a Java program to create a class Student with members usn, name, an array credits and an array marks.

Include methods to accept and display details and a method to calculate SGPA of a student.

Algorithm:

```
2. WAP to create class Student with members 7-10-2024
usn, name, an array credits and array marks. Include
methods to accept & display details and method
to calculate SGPA of a student.
A. import java.util.Scanner;
public class Student {
    String name, usn;
    double SGPA;
    int[] marks = new int[4];
    int[] credits = new int[4];
    double[] gradepoints = new double[4];
    double total = 0, credittotal = 0;
    Scanner sc = new Scanner(System.in);
    void getStudentDetails() {
        System.out.println("Enter name: ");
        name = sc.nextline();
        System.out.println("Enter USN: ");
        usn = sc.nextline();
    }
    void getMarks() {
        for (int j=0; j<4; j++)
        {
            System.out.println("Enter "+(j+1)+" subject marks:");
            marks[j] = sc.next_Int();
            System.out.println("Enter credits for subject "
                +(j+1)+ " : ");
            credits[j] = sc.nextInt();
            gradepoints[j] = (marks[j]/10)+1;
            if (gradepoints[j]>10) {
                gradepoints[j] = 10;
            }
        }
    }
    void computeSGPA() {
        for (int j=0; j<4; j++) {
            total += gradepoints[j] * credits[j];
            credittotal += credits[j];
        }
    }
    void display() {
        System.out.println("Name: "+name);
        System.out.println("USN: "+usn);
        System.out.println("SGPA: "+SGPA);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of students");
        int nos = sc.nextInt();
        Student[] students = new Student[nos];
```

```
    for (int i=0; i<nos; i++) {
        students[i]=new Student();
        students[i].getStudentDetails();
        students[i].getMarks();
        Students[i].computeSGPA();
        students[i].display();
    }
    System.
  }
}
```

```
Output: Enter number of students:
1
Enter name:
KCN
Enter USN:
1BM23CS001
Enter 1 subject marks:
80
Enter credits for subject 1:
3
Enter 2 subject marks:
67
Enter credits for subject 2:
2
Enter 3 subject marks:
90
Enter credits for subject 3:
1
Enter 4 subject marks:
90
Enter credits for subject 4:
4
```

```
Name: KCN
USN: 1BM23CS001
SGPA: 9.24
```

Code:

```
import java.util.Scanner;

public class Student {
    String name, usn;
    double SGPA;

    int[] marks = new int[4];
    int[] credits = new int[4];
    double[] gradepoints = new double[4];
    double total = 0, credittotal = 0;

    Scanner sc = new Scanner(System.in);

    void getStudentDetails() {
        System.out.println("Enter name:");
        name = sc.nextLine();
        System.out.println("Enter USN:");
        usn = sc.nextLine();
    }

    void getMarks() {
        for (int j = 0; j < 4; j++) {
```

```
            System.out.println("Enter " + (j + 1) + " subject marks:");
            marks[j] = sc.nextInt();
            System.out.println("Enter credits for subject " + (j + 1) + ":");
            credits[j] = sc.nextInt();

            gradepoints[j] = (marks[j] / 10.0) + 1;
            if (gradepoints[j] > 10) {
                gradepoints[j] = 10;
            }
        }
        sc.nextLine();
    }

    void computeSGPA() {
        total = 0;
        credittotal = 0;
        for (int j = 0; j < 4; j++) {
            total += gradepoints[j] * credits[j];
            credittotal += credits[j];
        }
        SGPA = total / credittotal;
    }

    void display() {
        System.out.println("Name: " + name);
        System.out.println("USN: " + usn);
        System.out.println("SGPA: " + SGPA);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of students:");
        int numberOfStudents = sc.nextInt();
        sc.nextLine();

        Student[] students = new Student[numberOfStudents];

        for (int i = 0; i < numberOfStudents; i++) {
            students[i] = new Student();
            students[i].getStudentDetails();
            students[i].getMarks();
            students[i].computeSGPA();
            students[i].display();
        }
                    System.out.println("1BM23CS154 Kishore Chandra");


    }
}
```

Output:

```
Enter name:
XYZ
Enter USN:
456
Enter 1 subject marks:
67
Enter credits for subject 1:
3
Enter 2 subject marks:
79
Enter credits for subject 2:
4
Enter 3 subject marks:
90
Enter credits for subject 3:
3
Enter 4 subject marks:
78
Enter credits for subject 4:
4
Name: XYZ
USN: 456
SGPA: 8.85
1BM23CS154 Kishore Chandra
```

```
D:\1BM23CS154>javac Student.java

D:\1BM23CS154>java Student.java
Enter number of students:
2
Enter name:
KCN
Enter USN:
1223
Enter 1 subject marks:
80
Enter credits for subject 1:
3
Enter 2 subject marks:
67
Enter credits for subject 2:
2
Enter 3 subject marks:
90
Enter credits for subject 3:
1
Enter 4 subject marks:
90
Enter credits for subject 4:
4
Name: KCN
USN: 1223
SGPA: 9.24
```

Program 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.

Algorithm:

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

A.
```
import java.util.Scanner;
class Books
{
    String name;
    String author;
    int price, numPages;
Books (String name, String author, int price, int numPages)
{
    this.name = name;
    this.author = author;
    this.price = price;
    this.numPages = numPages;
}
public String toString()
{
    String name, author, price, numPages;
    name = "Book name : " + this.name + "\n";
    author = " Author name : " + this.author + "\n";
    price = " Price : " + this.price + "\n";
    numPages = " Number of pages : " + this.numPages + "\n";
    return name + author + price + numPages;
}
public static void main (String args[])
{
    Scanner sc = new Scanner(System.in);
    int n;
    String name;
    String author;
    int price;
    int numPages;
    System.out.println ("Enter no. of books");
    n = sc.nextInt();
    Books b[];
    b = new Books[n];
    System.out.println ("Enter book details");
    for (int i=0; i<n; i++)
    {
        System.out.println ("Enter book " + (i+1) + " name");
        name = sc.next();
        System.out.println ("Enter book " + (i+1) + " author");
        author = sc.next();
        System.out.println ("Enter book " + (i+1) + " price");
        price = sc.nextInt();
        System.out.println (" Enter book " + (i+1) + " pages");
        numPages = sc.nextInt();
    b[i] = new Books (name, author, price, numPages);
    }
    for (int i=0; i<n; i++)
    {
```

```
        b[i].toString();
        System.out.println(b[i]);
    }
}
}
```

Output :

```
Enter no. of Books
1
Enter book details
Enter book 1 name
Bharath
Enter book1 author
Bhasker
Enter book 1 price
35
Enter book1 pages
72
Book name : Bharath
Author name : Bhasker
Price : 35
Number of pages : 72
```

Code:

```
import java.util.Scanner;
class Books
{
        String name;
        String author;
        int price,numPages;
Books(String name,String author,int price,int numPages)
{       this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
}
public String toString()
{
String name, author, price, numPages;
name = "Book name: " + this.name + "\n";
author = "Author name: " + this.author + "\n";
price = "Price: " + this.price + "\n";
numPages = "Number of pages: " + this.numPages + "\n";
return name + author + price + numPages;
}
public static void main(String args[])
{
        Scanner sc=new Scanner(System.in);
         int n;
        String name;
```

```
        String author;
        int price;
        int numPages;
        System.out.println("Enter no. of books");
        n=sc.nextInt();
        Books b[];
        b=new Books[n];
        System.out.println("Enter book details");
        for(int i=0;i<n;i++)
{
        System.out.println("Enter book "+(i+1)+" name");
        name=sc.next();
        System.out.println("Enter book "+(i+1)+" author");
        author=sc.next();
        System.out.println("Enter book "+(i+1)+" price");
        price=sc.nextInt();
        System.out.println("Enter book "+(i+1)+" pages");
        numPages=sc.nextInt();
        b[i]=new Books(name,author,price,numPages);
}
        for(int i=0;i<n;i++)
{
        b[i].toString();
        System.out.println(b[i]);

}
        System.out.println("Kishore Chandra N");
        System.out.println("1BM23CS154");
}
}
```

Output:

```
D:\1BM23CS154>javac Books.java

D:\1BM23CS154>java Books
Enter no. of books
1
Enter book details
Enter book 1 name
Bharath
Enter book 1 author
Bhasker
Enter book 1 price
35
Enter book 1 pages
72
Book name: Bharath
Author name: Bhasker
Price: 35
Number of pages: 72

Kishore Chandra N
1BM23CS154
```

Program 4

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

Algorithm:

4. Develop program to create abstract class named Shape that contains 2 Integers and empty method named printArea( ). Provide 3 classes named Rectangle, Triangle and Circle such that each one of classes extends class Shape. Each one of classes contain only method printArea( ) that prints area of given shape.

A.

```
import java.util.Scanner;
abstract class Shape {
    protected int dimension1;
    protected int dimension2;
    public abstract void printArea();
}
class Rectangle extends Shape {
    public Rectangle (int length, int width) {
        this.dimension1 = length;
        this.dimension2 = width;
    }
    public void printArea() {
        int area = dimension1 * dimension2;
        System.out.println ("Area of Rectangle:"+area);
    }
}
class Triangle extends Shape {
    public Triangle (int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
    }
    public void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println ("Area of Triangle:"+area);
    }
}
class Circle extends Shape {
    public Circle (int radius) {
        this.dimension1 = radius;
    }
    public void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println ("Area of circle: \n"+area);
    }
}
public class ShapeTest {
    public static void main (String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println ("Enter length & breadth of rectangle");
        int length = sc.nextInt();
        int breadth = sc.nextInt();
        Shape rectangle = new Rectangle (length, breadth);
        rectangle.printArea();
```

```
System.out.println("Enter base & height of triangle");
int base = sc.nextInt();
int height = sc.nextInt();
Shape triangle = new Triangle(base, height);
triangle.printArea();

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

Output:

```
Enter length & breadth of rectangle
3 2
Area of Rectangle : 6
Enter base & height of triangle
2 4
Area of Triangle : 4.0
Enter radius of circle :
4
Area of circle : 50.265
```

Code:

```
import java.util.Scanner;

abstract class Shape {
   protected int dimension1;
   protected int dimension2;

   public abstract void printArea();
}

class Rectangle extends Shape {
   public Rectangle(int length, int width) {
      this.dimension1 = length;
      this.dimension2 = width;
   }
     public void printArea() {
      int area = dimension1 * dimension2;
      System.out.println("Area of Rectangle: " + area);
   }
}
class Triangle extends Shape {
   public Triangle(int base, int height) {
      this.dimension1 = base;
      this.dimension2 = height;
   }
```

```
        public void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Area of Triangle: " + area);
    }
}
class Circle extends Shape {
    public Circle(int radius) {
        this.dimension1 = radius;
    }
    public void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Area of Circle: \n" + area);
            System.out.println("1BM23CS154 Kishore Chandra");
    }
}
public class ShapeTest {
    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();
        Shape 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();
        Shape triangle = new Triangle(base, height);
        triangle.printArea();
        System.out.print("Enter radius of circle: ");
        int radius = scanner.nextInt();
        Shape circle = new Circle(radius);
        circle.printArea();

        scanner.close();
    }
}
```

Output:

```
D:\1BM23CS154>java ShapeTest
Enter length of rectangle: 3
Enter width of rectangle: 2
Area of Rectangle: 6
Enter base of triangle: 2
Enter height of triangle: 4
Area of Triangle: 4.0
Enter radius of circle: 4
Area of Circle:
50.26548245743669
1BM23CS154 Kishore Chandra
```

Program 5

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

a) Accept deposit from customer and update the balance.

b) Display the balance.

c) Compute and deposit interest

d) Permit withdrawal and update the balance

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

Algorithm:

28/10/2024

5. WAP to create class Bank that maintains two kinds of account for its customers, one called savings account and other current account. Savings account provides compound interest and withdrawal facilities but no cheque book facility. Current account provides cheque book facility but not interest. Current account holders should also maintain minimum balance and if balance falls below this level, service charge is imposed

- Create class Account that stores customer name, account number and type of account. Derive class Cur-acct and Sav-acct to make them more specific.

a. Accept deposit from customer and update balance.

b. Display balance.

c. Compute and deposit interest.

d. Permit withdrawal and update balance

e. Check for minimum balance, impose penalty.

```
A. Import java.util.Scanner;

class Account
{
    ~~protected~~ String customerName;
    ~~protected~~ String accType;
    ~~protected~~ double balance;
    int accountNumber;

    Account (String name, int accNumber, String accType)
    {
        customerName = name;
        accountNumber = accNumber;
        accountType = accType;
        balance = 0;
    }
    public void deposit (double amount)
    {
        balance += amount;
        System.out.println ("Deposited : "+amount+" Updated
                                balance : "+balance);
    }
    public void displayBalance()
    {
        System.out.println ("Account Balance : "+balance);
    }
    public void withdraw (double amount)
    {
        System.out.println ("Operation is specific to
                                savings account");
    }
}
class SavAccount extends Account
{
    double interestRate = 0.08;
    SavAccount (String name, int accNumber)
    {
        super (name, accNumber, "Savings");
    }
```

```
public void computeInterest()
{
    double interest = balance * interestRate;
    balance += interest;
    System.out.println("Interest added: "+interest+
                    " Updated balance: "+balance);
}

public void withdraw(double amount)
{
    if (balance >= amount)
    {
        balance -= amount;
        System.out.println("Withdrawn: "+amount+
                    " Updated Balance "+balance);
    }
    else
    {
        System.out.println("Insufficient balance");
    }
}
}
class CurAccount extends Account
{
    double minBalance = 500.0;
    double serviceCharge = 50.0;

    CurAccount(String name, int accNumber)
    {
        super(name, accNumber, "Current");
    }
```

```
public void checkMinBalance()
{
    if (balance < minBalance)
    {
        balance -= serviceCharge;
        System.out.println("Balance below minimum.
                    Service charge "+serviceCharge+
                    " Updated balance: "+balance);
    }
}

public void withdraw(double amount)
{
    if (balance >= amount)
    {
        balance -= amount;
        System.out.println("Withdrawn: "+amount+" Updated balance "
                    + balance);
        checkMinBalance();
    }
    else
    {
        System.out.println("Insufficient Balance");
    }
}
}
public class Bank
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter customer name");
        String name = sc.next();
```

```
System.out.println("Enter account number");
int accountnumber = sc.nextInt();
SavAccount savingsAccount = new SavAccount(name,
                    accountnumber);
System.out.println("Enter customer name:");
String name1 = sc.next();
System.out.println("Enter account number");
int accountnumber1 = sc.nextInt();
CurAccount currentAccount = new CurAccount(name1, accountnumber1);
while (true)
{
    System.out.println("\n---MENU---");
    System.out.println("1.Deposit \n2.Withdraw \n3. Compute
                Interest \n 4. Display \n5.Exit");
    System.out.print("Enter your choice:");
    int choice = sc.nextInt();
    System.out.print("Enter type of account:");
    String accType = sc.next();
    if (accType.equals("saving"))
    {
        switch (choice)
        {
            case 1: System.out.print("Enter deposit
                    amount");
```

```
double depositAmount = sc.nextDouble();
savingsAccount.deposit(depositAmount);
break;
case 2: System.out.print("Enter withdrawal amount");
    double withdrawalAmount = sc.nextDouble();
    savingsAccount.withdraw(withdrawalAmount);
    break;
case 3: savingsAccount.computeInterest();
    break;
case 4: System.out.println("Customer name "+savingsAccount.
                customerName);
    System.out.println("Account number: "+savingsAccount.accountNumber);
    System.out.println("Type of Account: "+savingsAccount.accountType);
    savingsAccount.displayBalance();
    break;
case 5: System.exit(0);
    break;
}
} else if (accType.equals("current"))
    {
        switch (choice)
        {
            case 1: System.out.print("Enter deposit amount:");
                double depositAmt = sc.nextDouble();
                CurrentAccount.deposit(depositAmt);
                break;
```

```
case 2 : System.out.print("Enter withdrawal amount");
        double withdrawalAmt = sc.nextDouble();
        currentAccount.withdraw(withdrawalAmt);
        break;

case 3 : System.out.println("Current accounts do not
                earn interest");
        break;

case 4 : System.out.println("Name : "+currentAccount.customerName);
        System.out.println("Account Number : "+currentAccount.accountNumber
        System.out.println("Type of Account "+currentAccount.accountType)
        currentAccount.displayBalance();
        break;

case 5 : System.exit(0);
        break;
} else {
        System.out.print("Invalid account type");
    }
}
}
```

25/10/24

Output :

```
Enter customer Name :
Jishnu
Enter account number :
1234
Enter customer name :
Kiran
Enter account number :
890
---- MENU ----
1. Deposit
2. Withdraw
3. Compute Interest
4. Display
5. Exit
Enter your choice
1
Enter type of account : saving
Enter deposit amount : 1000
Deposited : 1000.0  Updated balance : 1000.0.
---- MENU ----
1. Deposit.
2. Withdraw
3. Compute Interest
4. Display
5. Exit
Enter your choice
2
Enter type of account : saving
Enter withdrawal amount : 200
Withdrawn : 200.0  Updated balance : 800.0
```

```
---- MENU ----
1. Deposit
2. Withdraw
3. Compute Interest
4. Display
5. Exit
Enter your choice : 3
Enter type of account : saving
Interest added : 32.0 . Updated balance = 832.0
```

Code:

```
import java.util.Scanner;

class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;


    Account(String name, int accNumber, String accType) {
        customerName = name;
        accountNumber = accNumber;
        accountType = accType;
        balance = 0;
    }


    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: " + amount + ". Updated balance: " + balance);
    }

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


    public void withdraw(double amount) {
        System.out.println("This operation is specific to account type.");
    }
}
class SavAccount extends Account {
    double interestRate = 0.04;  // 4% annual interest rate

    SavAccount(String name, int accNumber) {
        super(name, accNumber, "Savings");
    }
    public void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest added: " + interest + ". Updated balance: " + balance);
    }
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ". Updated balance: " + balance);
        } else {
            System.out.println("Insufficient balance.");
        }
```

```
    }
}
class CurAccount extends Account {
    double minBalance = 500.0;
    double serviceCharge = 50.0;

    CurAccount(String name, int accNumber) {
        super(name, accNumber, "Current");
    }

    public void checkMinBalance() {
        if (balance < minBalance) {
            balance -= serviceCharge;
            System.out.println("Balance below minimum. Service charge imposed: " + serviceCharge + ". Updated
balance: " + balance);
        }
    }
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ". Updated balance: " + balance);
            checkMinBalance();
        } else {
            System.out.println("Insufficient balance.");
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter customer name:");
        String name=sc.next();
        System.out.println("Enter account number:");
        int accountnumber=sc.nextInt();
        SavAccount savingsAccount = new SavAccount(name, accountnumber);
System.out.println("Enter customer name:");
        String name1=sc.next();
        System.out.println("Enter account number:");
        int accountnumber1=sc.nextInt();
        CurAccount currentAccount = new CurAccount(name1, accountnumber1);

        while (true) {
            System.out.println("\n-----MENU-----");
            System.out.println("1. Deposit\n2. Withdraw\n3. Compute Interest for Savings Account\n4. Display
Account Details\n5. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();

            System.out.print("Enter the type of account (saving/current): ");
            String accType = sc.next();
```

```
if (accType.equals("saving")) {
    switch (choice) {
        case 1:
            System.out.print("Enter the deposit amount: ");
            double depositAmount = sc.nextDouble();
            savingsAccount.deposit(depositAmount);
            break;
        case 2:
            System.out.print("Enter the withdrawal amount: ");
            double withdrawalAmount = sc.nextDouble();
            savingsAccount.withdraw(withdrawalAmount);
            break;
        case 3:
            savingsAccount.computeInterest();
            break;
        case 4:
            System.out.println("Customer name: " + savingsAccount.customerName);
            System.out.println("Account number: " + savingsAccount.accountNumber);
            System.out.println("Type of Account: " + savingsAccount.accountType);
            savingsAccount.displayBalance();
            break;
        case 5:
            System.exit(0);
            break;
        default:
            System.out.println("Invalid choice.");
    }
} else if (accType.equals("current")) {
    switch (choice) {
        case 1:
            System.out.print("Enter the deposit amount: ");
            double depositAmount = sc.nextDouble();
            currentAccount.deposit(depositAmount);
            break;
        case 2:
            System.out.print("Enter the withdrawal amount: ");
            double withdrawalAmount = sc.nextDouble();
            currentAccount.withdraw(withdrawalAmount);
            break;
        case 3:
            System.out.println("Current accounts do not earn interest.");
            break;
        case 4:
            System.out.println("Customer name: " + currentAccount.customerName);
            System.out.println("Account number: " + currentAccount.accountNumber);
            System.out.println("Type of Account: " + currentAccount.accountType);
            currentAccount.displayBalance();
            break;
        case 5:
            System.exit(0);
            break;
        default:
```

```
                System.out.println("Invalid choice.");
            }
        } else {
            System.out.println("Invalid account type.");
        }
    }
  }
}
```

Output:

```
D:\1BM23CS154>java Bank
Enter customer name:
Kishore
Enter account number:
1234
Enter customer name:
Kishen
Enter account number:
890

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): saving
Enter the deposit amount: 1000
Deposited: 1000.0. Updated balance: 1000.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account (saving/current): saving
Enter the withdrawal amount: 200
Withdrawn: 200.0. Updated balance: 800.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 3
Enter the type of account (saving/current): saving
Interest added: 32.0. Updated balance: 832.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 4
Enter the type of account (saving/current): saving
Customer name: Kishore
Account number: 1234
Type of Account: Savings
Account Balance: 832.0
```

```
-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): current
Enter the deposit amount: 2000
Deposited: 2000.0. Updated balance: 2000.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account (saving/current): current
Enter the withdrawal amount: 200
Withdrawn: 200.0. Updated balance: 1800.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 3
Enter the type of account (saving/current): current
Current accounts do not earn interest.

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 4
Enter the type of account (saving/current): current
Customer name: Kishen
Account number: 890
Type of Account: Current
Account Balance: 1800.0
```

Program 6

Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

Algorithm:

6. Create a package CIE which has 2 classes Students & Internals. The class Student has members like USN, name, sem. Class internals derived from Student has an array that stores internal marks scored in 5 courses of current semester of student. Create another package SEE which has to class External which is a derived class of Student. Class has an array that stores SEE marks scored in 5 courses of current sem of student. Import 2 packages in a file that declares final marks of "n" students in all five courses

A \\ Student.java

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

    public ~~stat~~ void inputStudentDetails()
    {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Enter USN:");
        usn = sc.nextline();
        System.out.println ("Enter Name:");
        name = sc.nextline();
        System.out.println ("Enter semester:");
        sem=sc.nextInt();
    }
    public void displayStudent()
    {
        System.out.println ("USN: "+usn);
        System.out.println ("Name: "+name);
        System.out.println ("Semester: "+sem);
    }
}

\\ Internals.java
package CIE;
import java.util.Scanner;
public class Internals extends student
{
    protected int marks[] = new int[5];

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

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

    public Externals()
    {
        marks = new int[5];
        finalMarks = new int[5];
    }
    public void inputSEEMarks()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter SEE marks for 5 courses");
        for(int i=0; i<5; i++)
        {
            System.out.print("Enter SEE marks for course "+(i+1)+": ");
            marks[i] = sc.nextInt();
        }
    }
public void calculateFinalMarks()
{
    for(int i=0; i<5; i++)
    {
        finalMarks[i] = marks[i] + (this.marks[i])/2;
    }
}
public void displayFinalMarks()
{
    displayStudentDetails();
    System.out.println("Final Marks : ");
    for(int i=0; i<5; i++)
    {
        System.out.println("Course "+(i+1)+": "+finalMarks[i]
    }
}
}
// Main.java
import SEE.Externals;
import java.util.Scanner;
class Main
{
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter no. of students");
        int n = sc.nextInt();
        Externals[] s = new Externals[n];
        for(int i=0; i<n; i++)
        {
            s[i] = new Externals();
            System.out.println("Enter details of student "+(i+1));
            s[i].inputStudentDetails();
            s[i].inputCIEMarks();
            s[i].inputSEEMarks();
            s[i].calculateFinalMarks();
        }
        System.out.println("\nDisplaying final marks");
        for(int i=0; i<n; i++)
        {
            System.out.println("\nStudent "+(i+1)+":");
            s[i].displayFinalMarks();
        }
    }
}
```

Output :

```
Enter no. of students : 1
Enter details for student 1
Enter USN : 1BM23CS154
Enter Name : Kishore Chandra
Enter semester : 3
Enter Internal Marks for 5 courses :
Enter marks for course 1 : 30
Enter marks for course 2 : 38
Enter marks for course 3 : 39
Enter marks for course 4 : 37
Enter marks for course 5 : 39
Enter SEE marks for 5 courses :
70
70
80
90
90
Displaying final marks for all students
Student 1 :
USN : 1BM23CS154
Name : Kishore Chandra N
Semester : 3
Final Marks : (Internal + External)
Course 1 : 140
Course 2 : 140
Course 3 : 160
Course 4 : 180
Course 5 : 196
```

Code:

```
package SEE;

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

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

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

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

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

    public void displayFinalMarks() {
        displayStudentDetails();
        System.out.println("Final Marks (Internal + External):");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}

package CIE;

import java.util.Scanner;

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

    public void inputCIEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Internal Marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for course " + (i + 1) + ": ");
```

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

package CIE;

import java.util.Scanner;

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

    public void inputStudentDetails() {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = s.nextLine();
        System.out.print("Enter Name: ");
        name = s.nextLine();
        System.out.print("Enter Semester: ");
        sem = s.nextInt();
    }

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

import SEE.Externals;
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = scanner.nextInt();

        Externals[] students = new Externals[n];

        for (int i = 0; i < n; i++) {
            students[i] = new Externals();
            System.out.println("\nEnter details for student " + (i + 1));


            students[i].inputStudentDetails();

            students[i].inputCIEmarks();

            students[i].inputSEEmarks();
```

```
            students[i].calculateFinalMarks();
        }

        System.out.println("\nDisplaying final marks for all students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            students[i].displayFinalMarks();
        }
    }
}
```

Output:

```
Enter the number of students: 2

Enter details for student 1
Enter USN: 1BM23CS154
Enter Name: Kishore Chandra N
Enter Semester: 3
Enter Internal Marks for 5 courses:
Enter marks for course 1: 30
Enter marks for course 2: 38
Enter marks for course 3: 39
Enter marks for course 4: 37
Enter marks for course 5: 39
Enter SEE Marks for 5 courses:
Enter SEE marks for course 1: 70
Enter SEE marks for course 2: 70
Enter SEE marks for course 3: 80
Enter SEE marks for course 4: 90
Enter SEE marks for course 5: 98

Enter details for student 2
Enter USN: 1BM23CS155
Enter Name: Kishen
Enter Semester: 3
Enter Internal Marks for 5 courses:
Enter marks for course 1: 40
Enter marks for course 2: 34
Enter marks for course 3: 36
Enter marks for course 4: 37
Enter marks for course 5: 38
Enter SEE Marks for 5 courses:
Enter SEE marks for course 1: 90
Enter SEE marks for course 2: 98
Enter SEE marks for course 3: 89
Enter SEE marks for course 4: 97
Enter SEE marks for course 5: 80

Displaying final marks for all students:

Student 1:
USN: 1BM23CS154
Name: Kishore Chandra N
Semester: 3
Final Marks (Internal + External):
Course 1: 140
Course 2: 140
Course 3: 160
Course 4: 180
Course 5: 196

Student 2:
USN: 1BM23CS155
Name: Kishen
Semester: 3
Final Marks (Internal + External):
Course 1: 180
Course 2: 196
Course 3: 178
Course 4: 194
Course 5: 160
```

Program 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 uses both father and son's age and throws an exception if son's age is >=father's age.

Algorithm:

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

A.
```
import java.util.Scanner;
class WrongAge (String message)
{       super (message);
  }
}
class Father
{ int age;
  public Father (int age) throws WrongAge
  {     if (age > 0)
        {       throw new WrongAge("Father's age cannot
                                    be negative");
        }
        this.age = age;
        System.out.println("Father's age: "+age);
  }
}
class Son extends Father
{     int sonAge;
  public Son (int fatherAge, int sonAge) throws WrongAge
  {     super (fatherAge);
        if (sonAge >= fatherAge)
        {       throw new WrongAge("Son's age cannot be
                                    greater than Father's
                                    age");
        }
        this.sonAge = sonAge;
        System.out.println("Son's age: "+sonAge);
  }
}
public class Main
{     public static void main (String[] args)
  {     Scanner sc = new Scanner (System.in);
        try
        {       System.out.println("Enter Father's age
                                    for case 1");
                int fatherAge1 = sc.nextInt();
                System.out.println("Enter Son's age for
                                    case 1");
        } catch (WrongAge e) {
                System.out.println("Exception in Test case 1: "
                                    + e.getMessage());
  }
```

```
try {
    System.out.println("Enter father age for Test case 2: ");
    int fatherAge2 = sc.nextInt();
    Father father2 = new Father(fatherAge2);
} catch (WrongAge e) {
    System.out.println("Exception in Test case 2: "
                        + e.getMessage());

}
try {
    System.out.println("Enter father's age for
                        Test case 3: ");

    int fatherAge3 = sc.nextInt();
    System.out.println("Enter son age ");
    int sonAge3 = sc.nextInt();
    Son son3 = new Son(fatherAge3, sonAge3);
} catch (WrongAge e) {
    System.out.println("Exception: "+e.getMessage())
} finally {
    sc.close();
}
}
}
```

```
Output:
Enter Father's age for Test Case 1:
23
Enter son's age for Test Case 2:
12
Father's age: 23
Son's age : 12
Enter Father's age :
-9
Exception: Father's age cannot be negative
Enter Father's age : 30
Enter son's age : 95
Exception: Son's age cannot be greater than Father's age.
```

Code:

```
import java.util.Scanner;

class WrongAge extends Exception {

    public WrongAge(String message) {

        super(message);

    }

}

class Father {

    int age;

public Father(int age) throws WrongAge {

        if (age < 0) {

            throw new WrongAge("Father's age cannot be negative.");
```

```
        }
        this.age = age;
        System.out.println("Father's age: " + age);
    }
}
class Son extends Father {
    int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAge {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to Father's age.");
        }
        this.sonAge = sonAge;
        System.out.println("Son's age: " + sonAge);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
 try {
            System.out.println("Enter Father's age for Test Case 1:");
            int fatherAge1 = scanner.nextInt();
            System.out.println("Enter Son's age for Test Case 1:");
            int sonAge1 = scanner.nextInt();
            Son son1 = new Son(fatherAge1, sonAge1);
```

```
        } catch (WrongAge e) {
            System.out.println("Exception in Test Case 1: " + e.getMessage());
        }
        try {
            System.out.println("\nEnter Father's age for Test Case 2:");
            int fatherAge2 = scanner.nextInt();
            Father father2 = new Father(fatherAge2);
        } catch (WrongAge e) {
            System.out.println("Exception in Test Case 2: " + e.getMessage());
        }
    try {
            System.out.println("\nEnter Father's age for Test Case 3:");
            int fatherAge3 = scanner.nextInt();
            System.out.println("Enter Son's age for Test Case 3:");
            int sonAge3 = scanner.nextInt();
            Son son2 = new Son(fatherAge3, sonAge3);
        } catch (WrongAge e) {
            System.out.println("Exception in Test Case 3: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

```
D:\1BM23CS154>java Main
Enter Father's age for Test Case 1:
23
Enter Son's age for Test Case 1:
12
Father's age: 23
Son's age: 12

Enter Father's age for Test Case 2:
-9
Exception in Test Case 2: Father's age cannot be negative.

Enter Father's age for Test Case 3:
30
Enter Son's age for Test Case 3:
45
Father's age: 30
Exception in Test Case 3: Son's age cannot be greater than or equal to Father's age.
```

Program 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.

Algorithm

8. WAP which creates two threads, one thread displaying "BMS College of Engineering" one every ten seconds and another displaying "CSE" once every 2 seconds.

```
A. class CollegeThread extends Thread
{
    public void run()
    {
        while (true)
        {
            try {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            } catch (InterruptedException e)
            {
                System.out.println(e);
            }
        }
    }
}
class CSEThread extends Thread
{
    public void run()
    {
        while (true)
        {
            try {
                System.out.println("CSE");
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}
```

```
public class ThreadPrint
{
    public static void main (String[] args)
    {
        CollegeThread collegeThread = new CollegeThread();
        CSEThread cseThread = new CSEThread();
        collegeThread.start();
        cseThread.start();
    }
}
```

Output:

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

Code:

```
class CollegeThread extends Thread {
    public void main{
        while (true) {
            try {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}
class CSEThread extends Thread {
   public void run() {
        while (true) {
           try {
                System.out.println("CSE");
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}
```

```
public class ThreadPrint {

    public static void main(String[] args) {

        CollegeThread collegeThread = new CollegeThread();

        CSEThread cseThread = new CSEThread();

        collegeThread.start();

        cseThread.start();

    }

}
```

Output:

```
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
CSE
CSE
CSE
```

Program 9

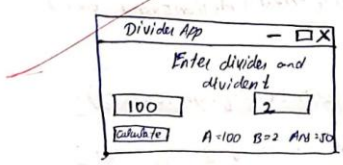
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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

Algorithm:

9. WAP to create user interface to perform integer divisions. Division of Num1 & Num2 is displayed in Result field, If Num1 & or Num2 is not an integer, program would throw a NumberFormatException. If Num2 were zero, program throws an Arithmetic Exception.

```
A. import javax.swing.*;
   import java.awt.*;
   import java.awt.event.*;
   class SwingDemo {
   SwingDemo() {
   JFrame jfrm = new JFrame("Divider App");
   jfrm.setSize(275,150);
   jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   JLabel jlab = new JLabel("Enter the divider and
                                 dividant:");

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

   JButton button = new JButton("Calculate");
   JLabel err = new JLabel();
   JLabel alab = new JLabel();
   JLabel blab = new JLabel();
   JLabel anslab = new JLabel();

   jfrm.add(err);
   jfrm.add(jlab);
   jfrm.add(ajtf);
   jfrm.add(bjtf);
   jfrm.add(button);
   jfrm.add(alab);
   jfrm.add(blab);
   jfrm.add(anslab);
   ActionListener l = new ActionListener()
   {   public void actionPerformed(ActionEvent evt)
   {      System.out.println("Action event from 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("B should be non zero!"); }
        }
    });
    jfrm.setVisible(true);
  }
  public static void main(String args[])
  {     SwingUtilities.invokeLater(new Runnable()
    {       public void run()
      {         new SwingDemo();
        }
      });
    }
}
```



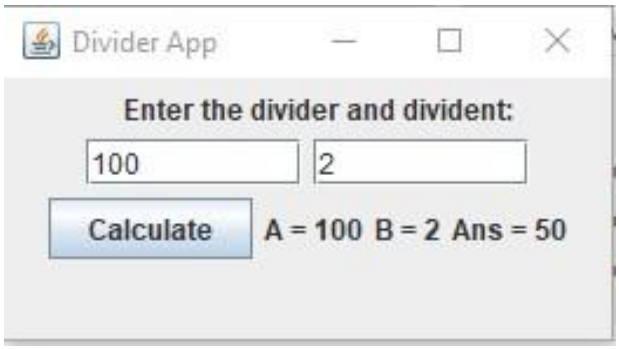
Code:

```
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();

}

});

}

}
```

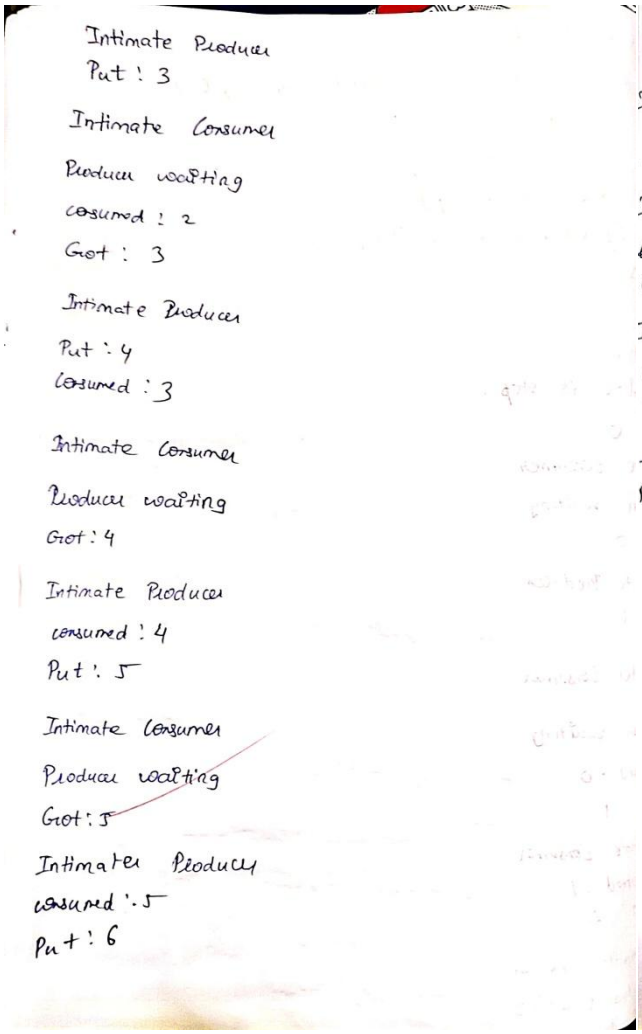


Program 10

Demonstrate Inter process Communication and deadlock

Algorithm:

```
10a. Inter Process Communication
A. class Q {
    int n;
    boolean valueSet = false;
    synchronized int get()
    {       while (!valueSet)
        try {
        System.out.println("\n Consumer waiting\n");
        wait();
        } catch (InterruptedException e) {
        System.out.println("InterruptedException caught");
        }
        System.out.println("Got: "+n);
        valueSet = false;
        System.out.println("\n Intimate Producer\n");
        notify();
        return n;
    }
    synchronized void put(int n)
    {   while (valueSet)
        try {
        System.out.println("Producer waiting\n");
        wait();
        } catch (InterruptedException e) {
        System.out.println("InterruptedException caught");
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: "+n);
        System.out.println("\n Intimate Consumer\n");
        notify();
    }
}
class Producer implements Runnable {
    Q q;
    Consumer (Q q)
    {       this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void run()
    {       int i = 0;
        while (i<15)
    {       int r = q.get();
        System.out.println("Consumed: "+r);
        i++;
    }
}
}
```



Intimate Producer
Put : 3

Intimate Consumer
Producer waiting
consumed : 2
Got : 3

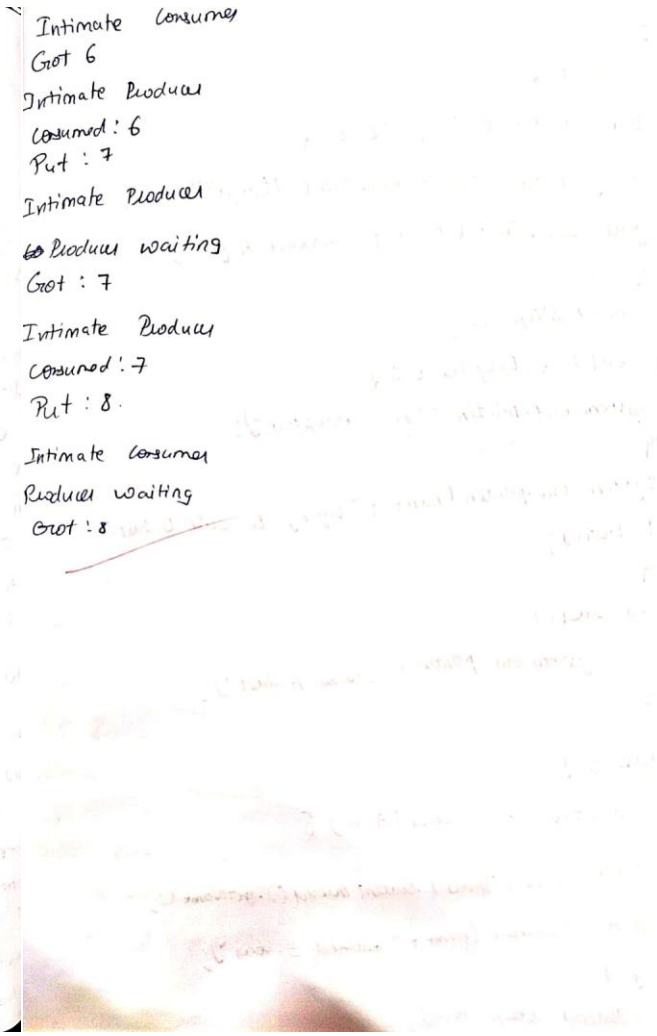
Intimate Producer
Put : 4
Consumed : 3

Intimate Consumer
Producer waiting
Got : 4

Intimate Producer
consumed : 4
Put : 5

Intimate Consumer
Producer waiting
Got : 5

Intimater Producer
consumed : 5
Put : 6



Intimate Consumer
Got 6
Intimate Producer
consumed : 6
Put : 7
Intimate Producer
Producer waiting
Got : 7
Intimate Producer
consumed : 7
Put : 8.
Intimate Consumer
Producer waiting
Got : 8

Code:

```
class Q {
int n;
boolean valueSet = false;
synchronized int get() {
while(!valueSet)
try {
System.out.println("\nConsumer waiting\n");
```

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

```
} }
class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
public void run() {
int i = 0;
while(i<15) {
q.put(i++);
} } }
class Consumer implements Runnable {
Q q;
Consumer(Q q) {
this.q = q;
new Thread(this, "Consumer").start();
}
public void run() {
int i=0;
while(i<15) {
int r=q.get();
System.out.println("consumed:"+r);
i++;
```

```
} } }

class PCFixed {

public static void main(String args[]) {

Q q = new Q();

new Producer(q);

new Consumer(q);

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

} }
```

Output:

```
D:\1BM23CS154>javac PCFixed.java

D:\1BM23CS154>java PCFixed
Press Control-C to stop.
Put: 0

Intimate Consumer


Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer


Producer waiting

consumed:0
Got: 1

Intimate Producer

consumed:1
Put: 2

Intimate Consumer


Producer waiting

Got: 2

Intimate Producer

Put: 3

Intimate Consumer


Producer waiting

consumed:2
Got: 3

Intimate Producer

Put: 4
consumed:3
```

```
Intimate Consumer


Producer waiting

Got: 4

Intimate Producer

consumed:4
Put: 5

Intimate Consumer


Producer waiting

Got: 5

Intimate Producer

consumed:5
Put: 6

Intimate Consumer

Got: 6

Intimate Producer

consumed:6
Put: 7

Intimate Consumer


Producer waiting

Got: 7

Intimate Producer

consumed:7
Put: 8

Intimate Consumer


Producer waiting

Got: 8
```

Deadlock

Algorithm:

```
b. Deadlock.
    class A {
    synchronized void foo(B b) {
    String name = Thread.currentThread().getName();
    System.out.println(name + " entered A.foo");
    try {
      Thread.sleep(1000);
    } catch (Exception e) {
    System.out.println("A Interrupted");
    }
    System.out.println(name + "trying to call B.last()");
    b.last();
    }
  void last()
  {     System.out.println("Inside A.last");
  }
}
class B {
Synchronized void bar(A a) {
  String name = Thread.currentThread().getName();
System.out.println(name + " entered B.bar");
try {
  Thread.sleep(1000);
} catch (Exception e) {
System.out.println("B interrupted");
}
System.out.println(name + " trying to call A.last()");
a.last();
}
void last()
{     System.out.println("Inside A.last");
}
}
class Deadlock implements Runnable
{
    A a = new A();
  B b = new B();
  Deadlock() {
  Thread.currentThread().setName("MainThread");
  Thread t = new Thread(this, "RacingThread");
  t.start();
  a.foo(b);
System.out.println("Back in main thread");
}
public void run()
{     b.bar(a);
System.out.println("Back in other thread");
}
```

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

Output:

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

Code:

```
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);

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

}

public static void main(String args[]) {

new Deadlock();

}

}
```

```
D:\1BM23CS154>java Deadlock
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
Inside A.last
RacingThread trying to call A.last()
Back in main thread
Inside A.last
Back in other thread
```