VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

KARTHIK O HAVALAGHATTA (1BM23CS140)

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 **KARTHIK O HAVALAGHATTA (1BM23CS140),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

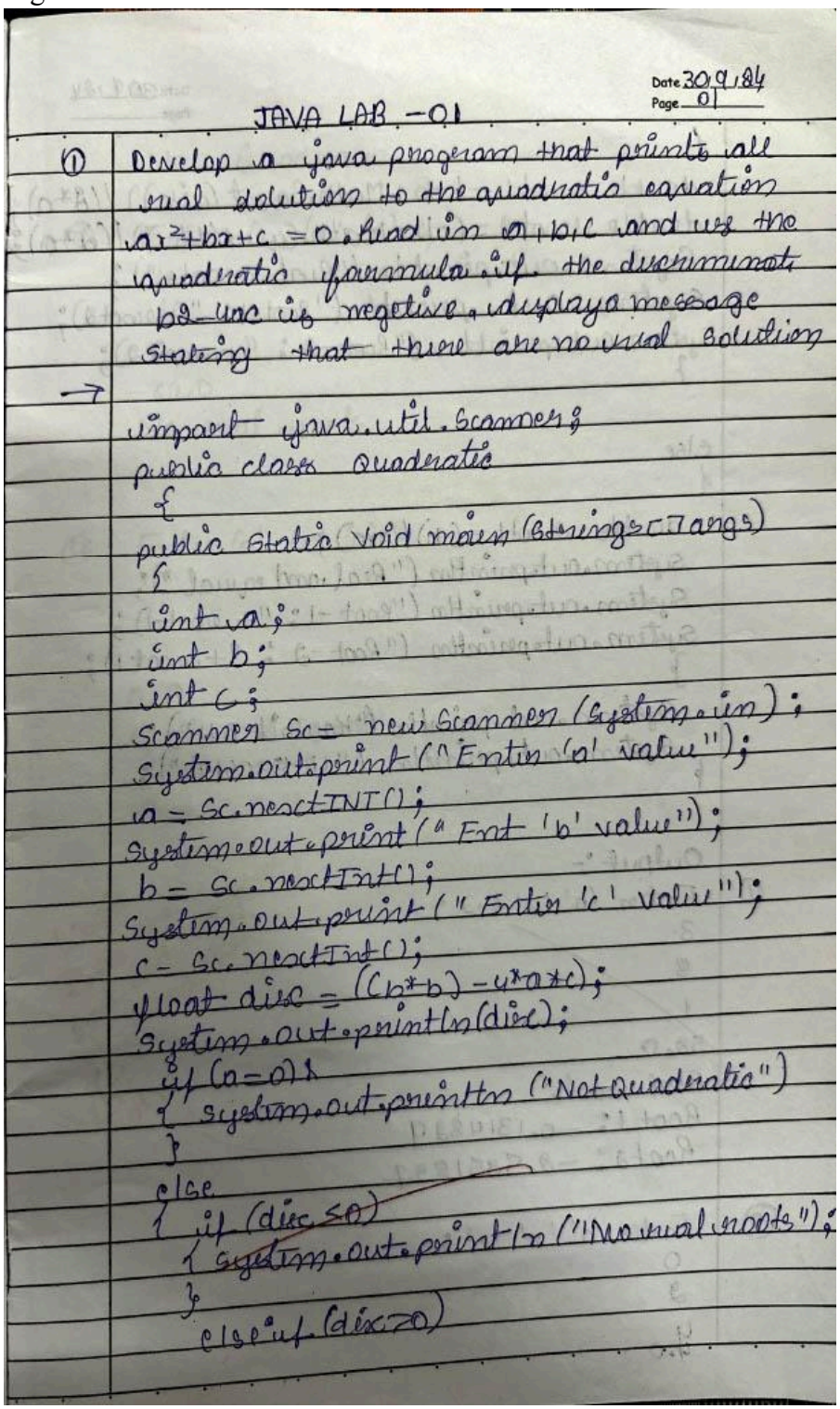
Lab faculty Incharge Name Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link:
KartikO2208/OOJ-LAB-140

Program 1
Implement Quadratic Equation

Algorithm:



JAVA LAB - 01

Date 30/9/24
Page 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 the discriminant b^2-4ac is negative, display a message stating that there are no real solutions

→

```
import java.util.Scanner;
public class Quadratic
{
public static void main (String[] args)
{
int a;
int b;
int c;
Scanner sc = new Scanner (System.in);
System.out.print ("Enter 'a' value");
a = sc.nextINT();
System.out.print ("Ent 'b' value");
b = sc.nextInt();
System.out.print ("Enter 'c' value");
c = sc.nextInt();
float disc = (b*b) - 4*a*c;
System.out.println(disc);
if (a==0)
{ System.out.println ("Not Quadratic")
}
else
{ if (disc<0)
{ System.out.println ("No real roots");
}
else if (disc>0)
```

Date 30/9/24
Page

```
{
double root1 = (-b+Math.Sqrt(dis))/(2*a);
double root2 = (-b+Math.Sqrt(dis))/(2*a);
System.out.println("Real roots");
System.out.println("Root-1: "+root2);
System.out.println("Root-2: "+root2);
}

else
{
double root1 = (-b)/(2*a);
System.out.println("Real and equal");
System.out.println("Root-1: "+root1);
System.out.println("Root-2: "+root1);
}
System.out.println("Karthik D H");
System.out.println("1BM23CS140");
}
```

Output:-

(1) Enter 'a' 'b' 'c':
3
8
1
58.0
Real roots:
Root-1: -0.1314829
Root-2: -2.5351837

(2) Enter 'a' 'b' 'c':
0
2
4.0

Date 30/9/24
Page

Not Quadratic

③ Enter 'a' 'b' 'c' :
10
4
9
-64.0
No real roots

④ Enter 'a' 'b' 'c' :
4
4
1
0.0
Real and Equal
Root 1 : 0.0
Root 2 : 0.0

Write Roots are real and imaginary condition

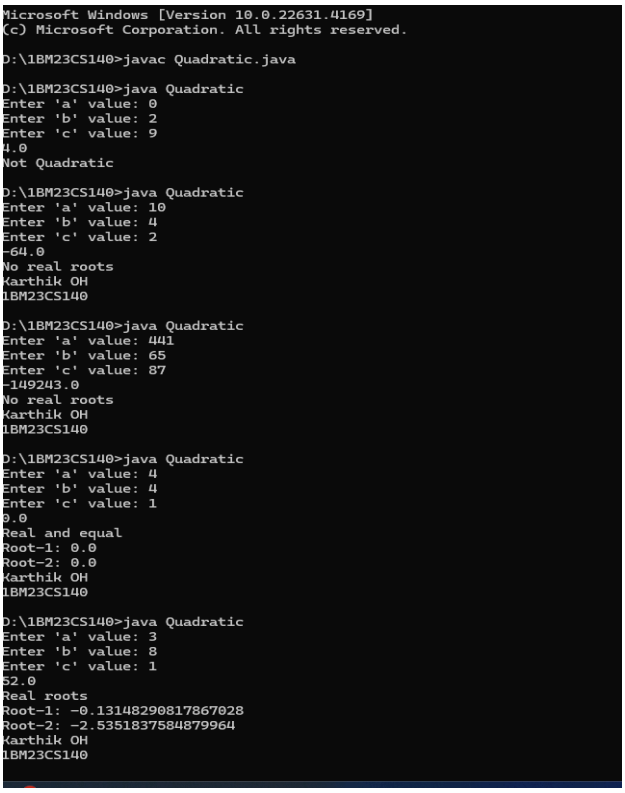
07.10

Code:

```
import java.util.Scanner;
public class Quadratic

{
public static void main(String[] args)
{
int a;
int b;
int c;
Scanner sc = new Scanner(System.in);
System.out.print("Enter 'a' value: ");
a= sc.nextInt();
System.out.print("Enter 'b' value: ");
b=sc.nextInt();
  System.out.print("Enter 'c' value: ");
  c=sc.nextInt();
   float disc = ((b*b)-4*a*c);
  System.out.println(disc);
  if (a==0)
    {
     System.out.println("Not Quadratic");
     }
   else
   {
    if (disc<0)
      {
      System.out.println("No real roots ");
       }
      else if (disc>0)
           {
            double root1= (-b + Math.sqrt(disc))/(2*a);
           double root2= (-b - Math.sqrt(disc))/(2*a);
           System.out.println("Real roots ");
           System.out.println("Root-1: "+root1);
           System.out.println("Root-2: "+root2);
           }
           else
           {
           double root1=(-b)/(2*a);
              System.out.println("Real and equal");

           System.out.println("Root-1: "+root1);
           System.out.println("Root-2: "+root1);


           }
```

```
            System.out.println("Karthik OH");
            System.out.println("1BM23CS140");

        }

    }

}
```

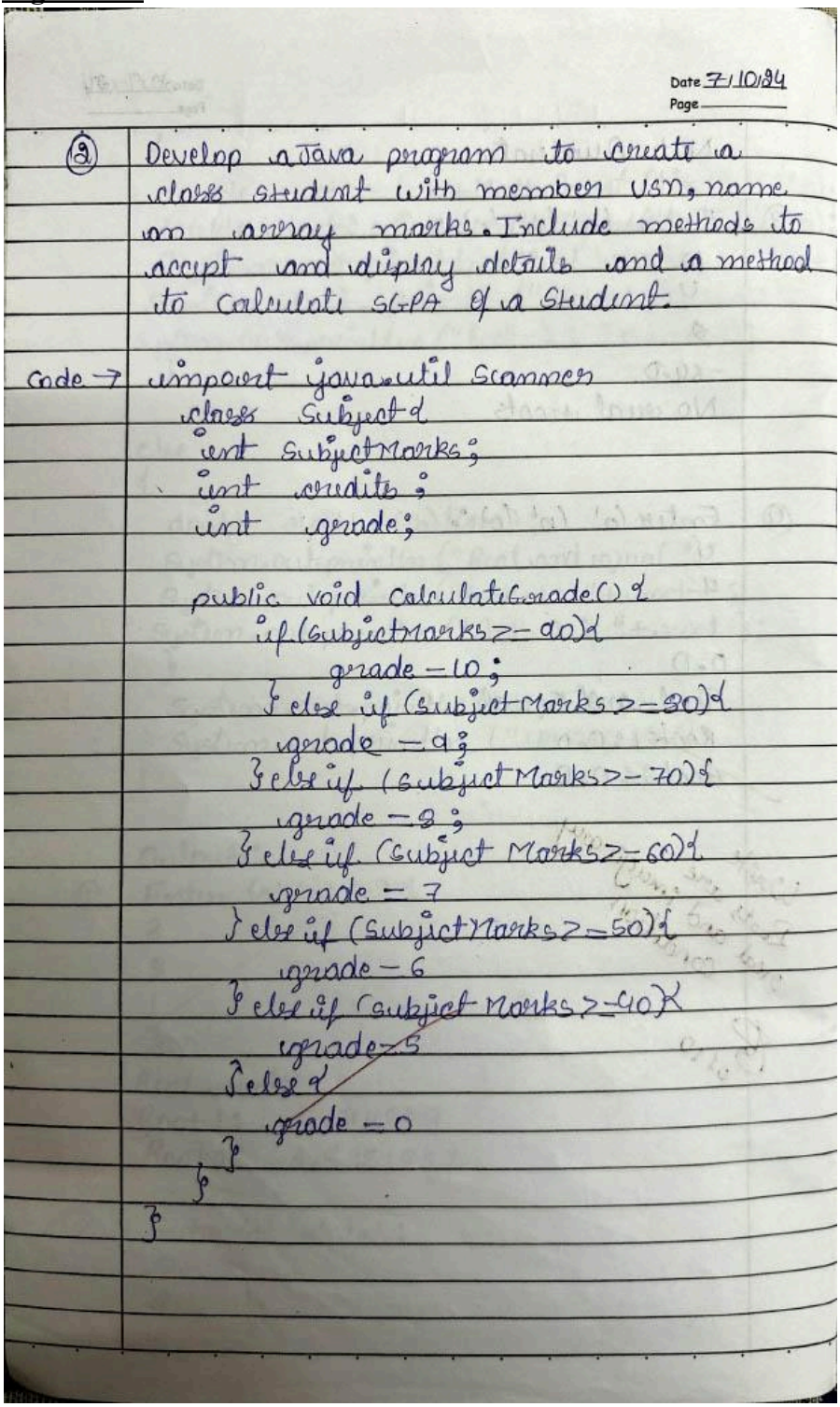
Output:



Program 2

SGPA Calculator

Algorithm:



Date 7/10/24
Page

(2) Develop a Java program to create a class student with member USN, name, an array marks. Include methods to accept and display details and a method to calculate SGPA of a Student.

Code →

```
import java.util.Scanner
class Subject{
int SubjectMarks;
int credits;
int grade;

public void calculateGrade(){
    if(SubjectMarks>=90){
        grade=10;
    }else if(SubjectMarks>=80){
        grade=9;
    }else if(SubjectMarks>=70){
        grade=8;
    }else if(SubjectMarks>=60){
        grade=7
    }else if(SubjectMarks>=50){
        grade=6
    }else if(SubjectMarks>=40){
        grade=5
    }else{
        grade=0
    }
}
}
```

Date __/__/__
Page _____

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

Student() {
    int i;
    Subject = new Subject[8];
// Assuming there are 8 Subjects
    for(i=0; i<8; i++){
        Subject[i] = new Subject();
    }
    s = new Scanner(System.in);
}

public void getStudentDetails() {
    System.out.print("Enter Student name");
    name = s.nextLine();
    System.out.print("Enter Student USN");
    usn = s.nextLine();
}

public void getmarks() {
    for(int i=0; i<8; i++) {
        System.out.println("Enter the marks for sub"
                + (i+1) + ":");
        Subject[i].SubjectMarks = s.nextInt();
        System.out.print("Enter credits for Sub
                + (i+1) + ":");
        Subject[i].credits = s.nextInt();
        Subject[i].CalculateGrade();
```

```
if (Subject[i].subjectMarks>100){
  System.out.print("Invalid marks, should
  not exceed 100");
  Subject[i].subjectMarks=0;
  }
  else if (Subject[i].subjectMarks<0){
      System.out.print("Invalid input,
        marks should not be in negative");
      Subject[i].subjectMarks=.0;
    }
  }
}

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

  for(Subject subj: Subject){
      totalGradePoints+=(subj.grade*
                         subj.credits);
    totalCredits+=subj.credits;
  }
  if(totalCredits>0){
    SGPA=totalGradePoints/totalCredits;
    }else{
      SGPA=0.0;
    }
  }
```

Date __/__/__
Page _____

Enter the marks for subject 7: 99
Enter the credits for subject 7: 3
Enter the marks for subject 8: 94
Enter the credit for subject 8: 1
Name: Karthik
USN: 1BM23CS140
SGPA: 9.27

14.10

Code:

```
import java.util.Scanner;

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

System.out.print("Enter the number of students: ");
int numStudents = sc.nextInt();
sc.nextLine();

String[] names = new String[numStudents];
String[] usns = new String[numStudents];
int[][] creditsArray = new int[numStudents][];
int[][] marksArray = new int[numStudents][];
double[] sgpas = new double[numStudents];


for (int s = 0; s < numStudents; s++) {
System.out.println("Enter details for student " + (s + 1) + ":");


System.out.print("Enter your name: ");
names[s] = sc.nextLine();
System.out.print("Enter your USN: ");
usns[s] = sc.nextLine();


System.out.print("Enter the number of subjects: ");
int numSubjects = sc.nextInt();


int[] credits = new int[numSubjects];
System.out.println("Enter the credits for each subject:");
for (int i = 0; i < numSubjects; i++) {
credits[i] = sc.nextInt();
}
creditsArray[s] = credits;

int[] marks = new int[numSubjects];
System.out.println("Enter the marks for each subject out of 100:");
for (int i = 0; i < numSubjects; i++) {
marks[i] = sc.nextInt();
}
marksArray[s] = marks;
```

```
int[] gradePoints = new int[numSubjects];
int[] resultArray = new int[numSubjects];
for (int i = 0; i < numSubjects; i++) {
gradePoints[i] = (marks[i] / 10) + 1;
resultArray[i] = credits[i] * gradePoints[i];
}

int totalCredits = sum(credits);
int totalResult = sum(resultArray);
if (totalCredits > 0) {
sgpas[s] = (double) totalResult / totalCredits;
} else {
sgpas[s] = 0.0;
}
}

System.out.println("\n--- Results ---");
for (int s = 0; s < numStudents; s++) {
System.out.println("Student " + (s + 1) + " (" + names[s] + ", " + usns[s] + "):");
System.out.print("Credits: ");
for (int credit : creditsArray[s]) {
System.out.print(credit + " ");
}
System.out.println();
System.out.print("Marks: ");
for (int mark : marksArray[s]) {
System.out.print(mark + " ");
}
System.out.println();
System.out.println("SGPA: " + sgpas[s]);
System.out.println();
}}
static int sum(int[] array) {
int sum = 0;
for (int value : array) {
sum += value;
}
return sum;
```

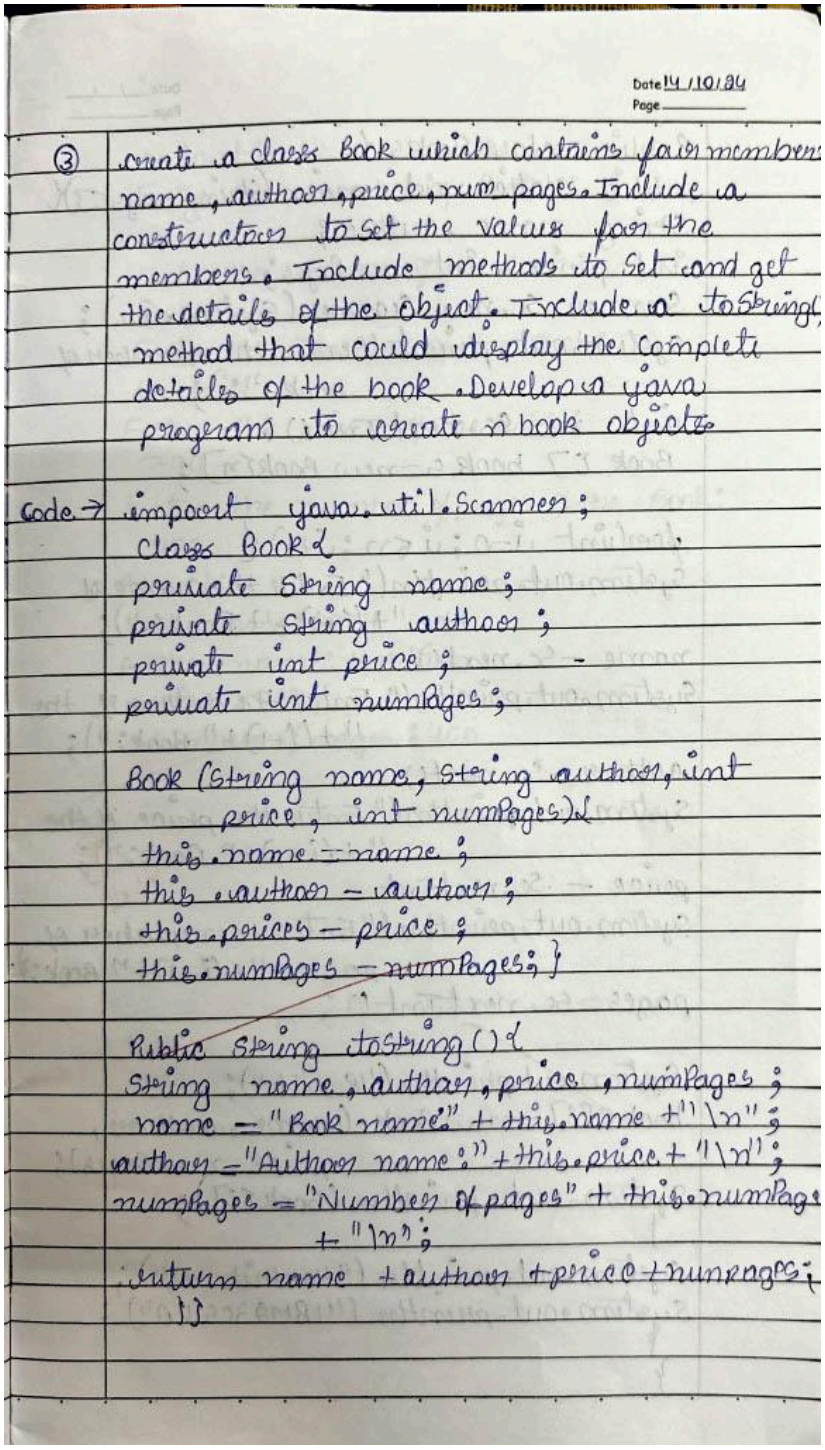
Output:

```
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

D:\1BM23CS140>javac student.java

D:\1BM23CS140>java student
Enter the number of students: 1
Enter details for student 1:
Enter your name: karthik o
Enter your USN: 1BM23CS140
Enter the number of subjects: 8
Enter the credits for each subject:
4
4
3
3
3
1
1
1
Enter the marks for each subject out of 100:
94
92
86
78
68
99
99
99

--- Results ---
Student 1 (karthik o, 1BM23CS140):
Credits: 4 4 3 3 3 1 1 1
Marks: 94 92 86 78 68 99 99 99
SGPA: 9.1
```

Program 3

Book Class

Algorithm:



Date 14/10/24
Page

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

Code →

```
import java.util.Scanner;
Class Book{
private String name;
private String author;
private int price;
private int numPages;

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

Public String toString(){
String name, author, price, numPages;
name = "Book name" + this.name + "\n";
author = "Author name:" + this.price + "\n";
numPages = "Number of pages" + this.numPages
          + "\n";
return name + author + price + numPages;
}}
```

Date __/__/__
Page _____

```
Public class Books {
public static void main (String args[]){
String name, author;
int price, int numPages;
Scanner sc = new Scanner(System.in);
System.out.print("enter the number of
                    books");
int n = sc.nextInt();
Book [] books = new Book[n];

for(int i=0; i<n; i++){
System.out.println("Enter the name of
        the "+(i+1)+" Book:");
name = sc.next()
System.out.println("Enter the author of the
                "+(i+1)+" Book:");
author = sc.next()
System.out.println("Enter the price of the
                "+(i+1)+" Book:");
price = sc.nextInt
System.out.println("Enter the number of
                pages "+(i+1)+" Book:");
pages = sc.nextInt();

System.out.println("Result");
books[i] = new Book (name, author,
                price, numPages);
System.out.println(book[i]);
}
System.out.println("Kanthik 0");
System.out.println("1BM23CS140");
}
}
```

Date __/__/__
Page _____

output:-
Enter the number of Books:
2
Enter the name of the Book:
GOTH
Enter the authour of the Book:
Karthik
Enter the price of the Book:
799
Enter the number of pages of the Book:
400
Result:
Book name: GOTH
Authour name: Karthik
price: 799
Number of pages: 400

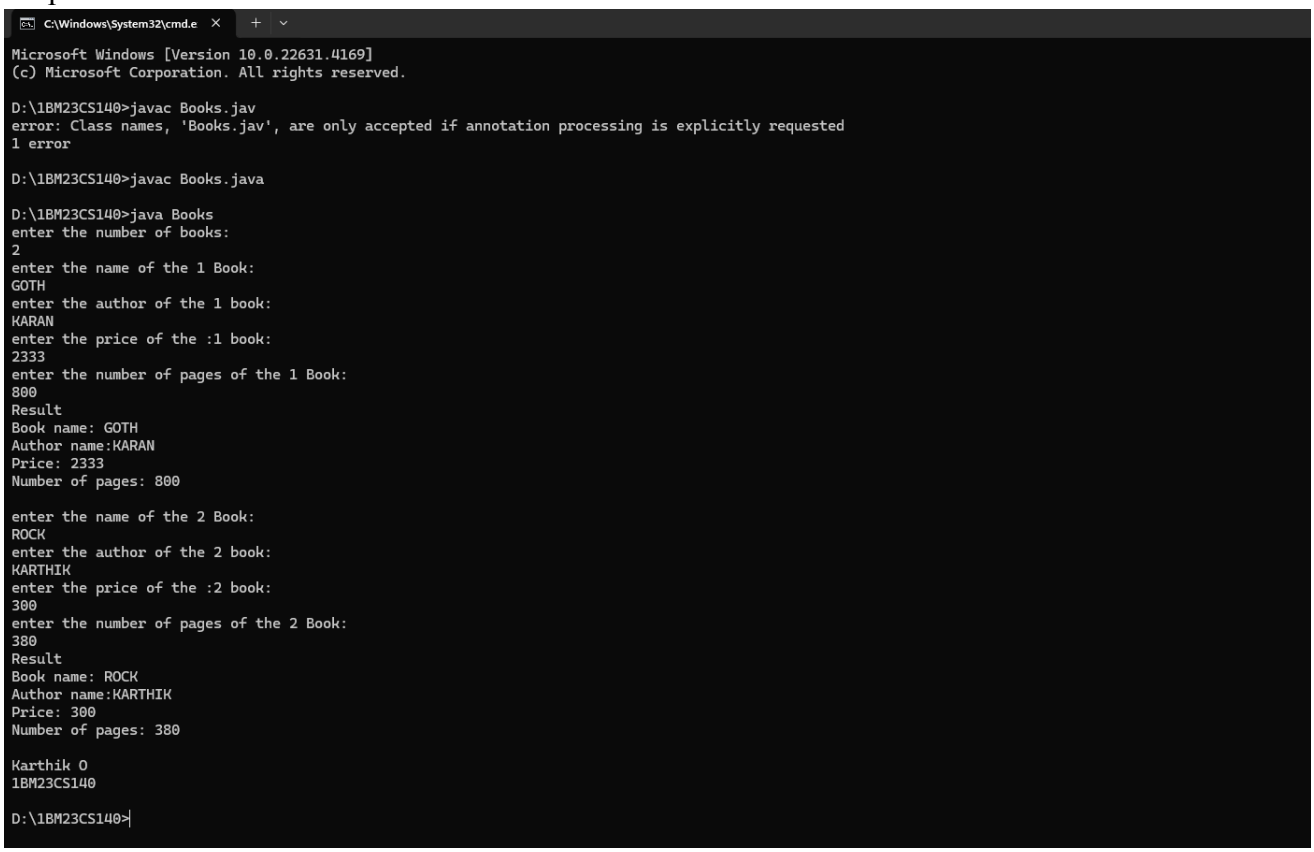
11.10

Code:

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

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

public String toString(){
String 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 class Books{
public static void main(String args[]){
String name, author;
int price, numPages;
Scanner sc=new Scanner(System.in);
System.out.print("enter the number of books:");
int n =sc.nextInt();
Book [] books= new Book[n];


for(int i=0;i<n;i++){
System.out.println("enter the name of the "+(i+1)+" Book:");
name =sc.next();
System.out.println("enter the author of the "+(i+1)+" book:");
author= sc.next();
System.out.println("enter the price of the :"+(i+1)+" book:");
price= sc.nextInt();
System.out.println("enter the number of pages of the "+(i+1)+" Book:");

numPages = sc.nextInt();

System.out.println("Result");
```

```
books[i]= new Book(name, author, price, numPages);
System.out.println(books[i]);

}

System.out.println("Karthik O");
System.out.println("1BM23CS140");
}
}
```

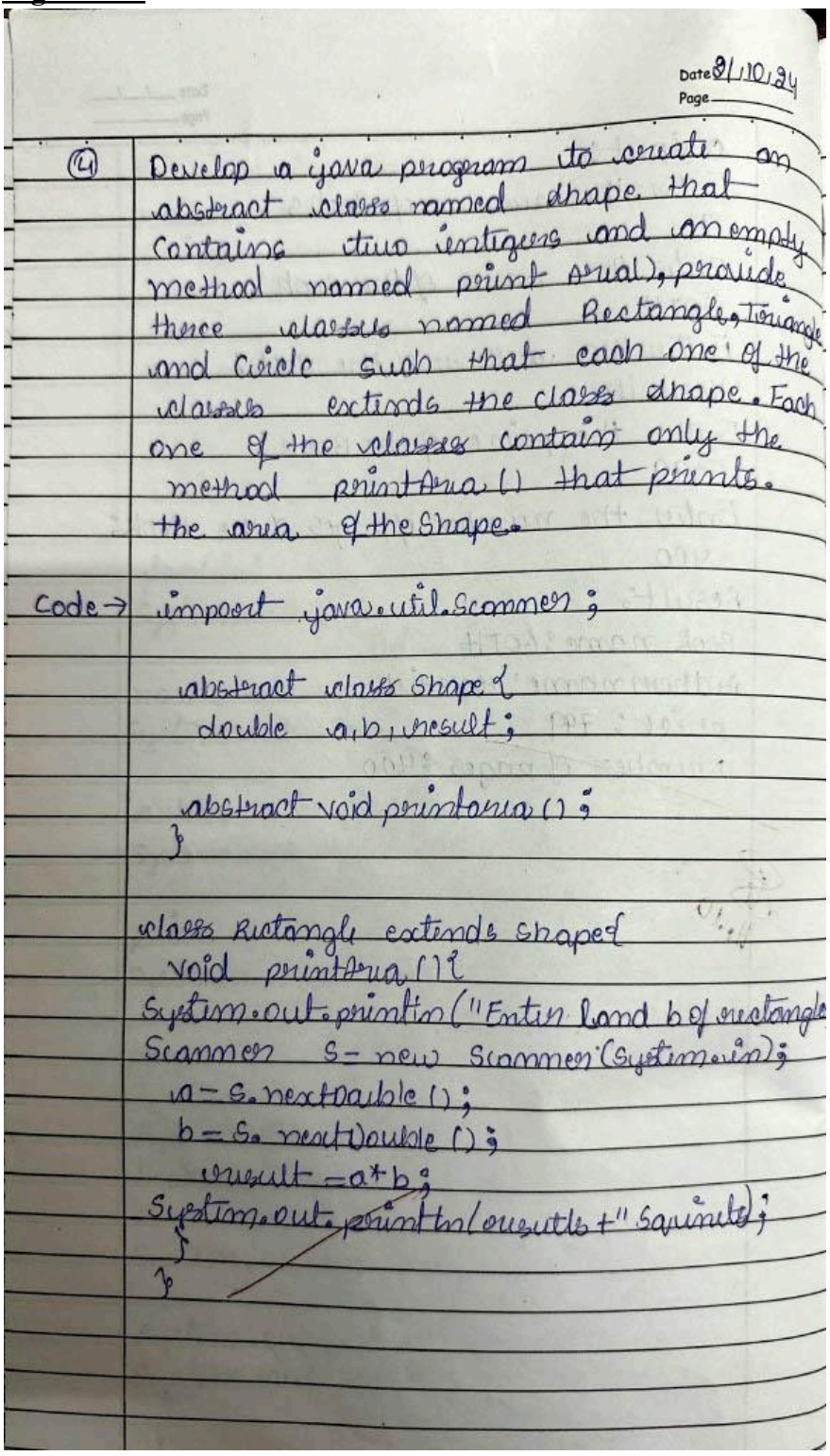
Output:



Program 4

Print Area Of Rectangle,triangle and Circle

Algorithm:



Date 31.10.24
Page

④ Develop a java program to create an abstract class named shape that contains two integers and an empty method named printArea(), provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class shape. Each one of the classes contain only the method printArea() that prints the area of the Shape.

Code →

```
import java.util.Scanner;

abstract class Shape{
  double a,b, result;

  abstract void printArea();
}

class Rectangle extends Shape{
  void printArea(){
System.out.println("Enter l and b of rectangle");
Scanner S= new Scanner(System.in);
  a=S.nextDouble();
  b=S.nextDouble();
    result=a*b;
System.out.println(result+" Squnits");
  }
}
```

Date __/__/__
Page _____

```
class Triangle extends Shape{
  void printArea(){
System.out.println("Enter b and h of
                    triangle:");
Scanner s=new Scanner(System.in);
a=s.nextDouble();
b=s.nextDouble();
result=a*b/2;
System.out.println(result+"Sq.units");
  }
}

class Circle extends Shape{
  void printArea(){
System.out.println("Enter radius of circle");
Scanner s=new Scanner(System.in);
a=s.nextDouble();
b=s.nextDouble();
result=3.142*a*a;
System.out.println(result+Sq.unit");

class printArea{
Public static void main(String args[]){
  Rectangle r=new Rectangle();
  Triangle t=new Triangle();
  Circle c=new circle();
  r.printArea();
  t.printArea();
  c.printArea();
}
}
```

Date __/__/__
Page _____

output :

Enter l and b of rectangle :
20
30
600.0 Sq units

Enter b and h of triangle :
20
50
500.0 Sq units

Enter radius of circle
30
2827.799 Sq units

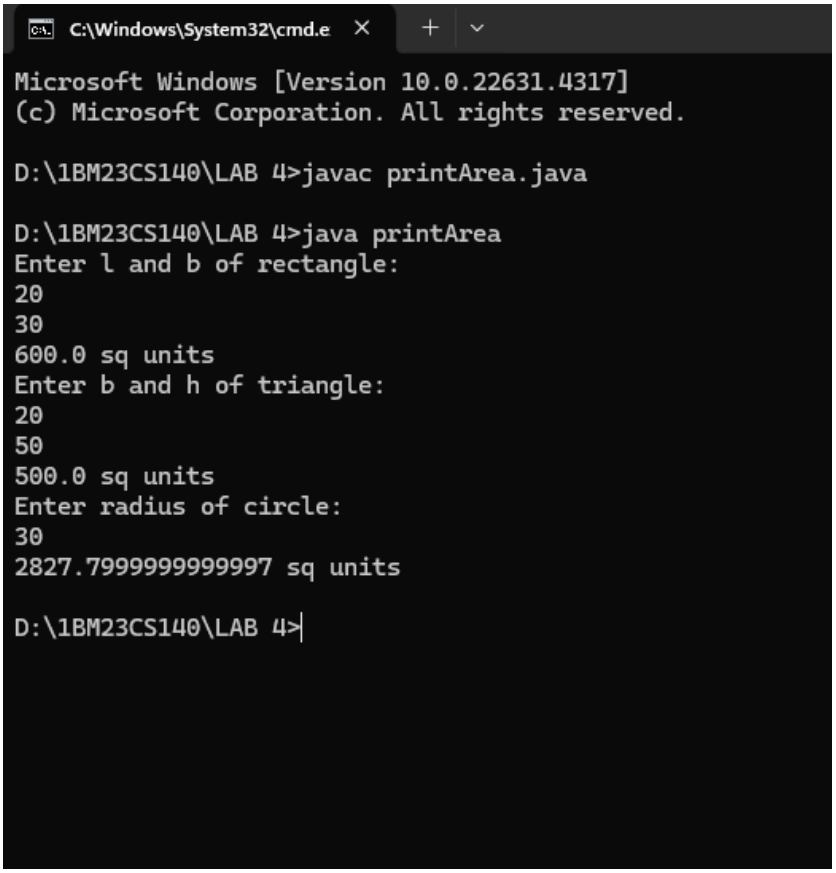
24.10

Code:

```
import java.util.Scanner;

   abstract class Shape{
    double a,b,result;

    abstract void printArea();
    }

class Rectangle extends Shape{
void printArea(){
System.out.println("Enter l and b of rectangle:");
Scanner s=new Scanner(System.in);
a=s.nextDouble();
b=s.nextDouble();
result=a*b;
System.out.println(result+" sq units");
}
}


class Triangle extends Shape{
void printArea(){
System.out.println("Enter b and h of triangle:");
Scanner s=new Scanner(System.in);
a=s.nextDouble();
b=s.nextDouble();
result=a*b/2;
System.out.println(result+" sq units");
}
}



class Circle extends Shape{
void printArea(){
System.out.println("Enter radius of circle:");
Scanner s=new Scanner(System.in);
a=s.nextDouble();
result=3.142*a*a;
System.out.println(result+" sq units");
}
}
```

```
class printArea{
public static void main(String args[]){
Rectangle r=new Rectangle();
Triangle t=new Triangle();
Circle c=new Circle();
r.printArea();
t.printArea();
c.printArea();

}
}
```

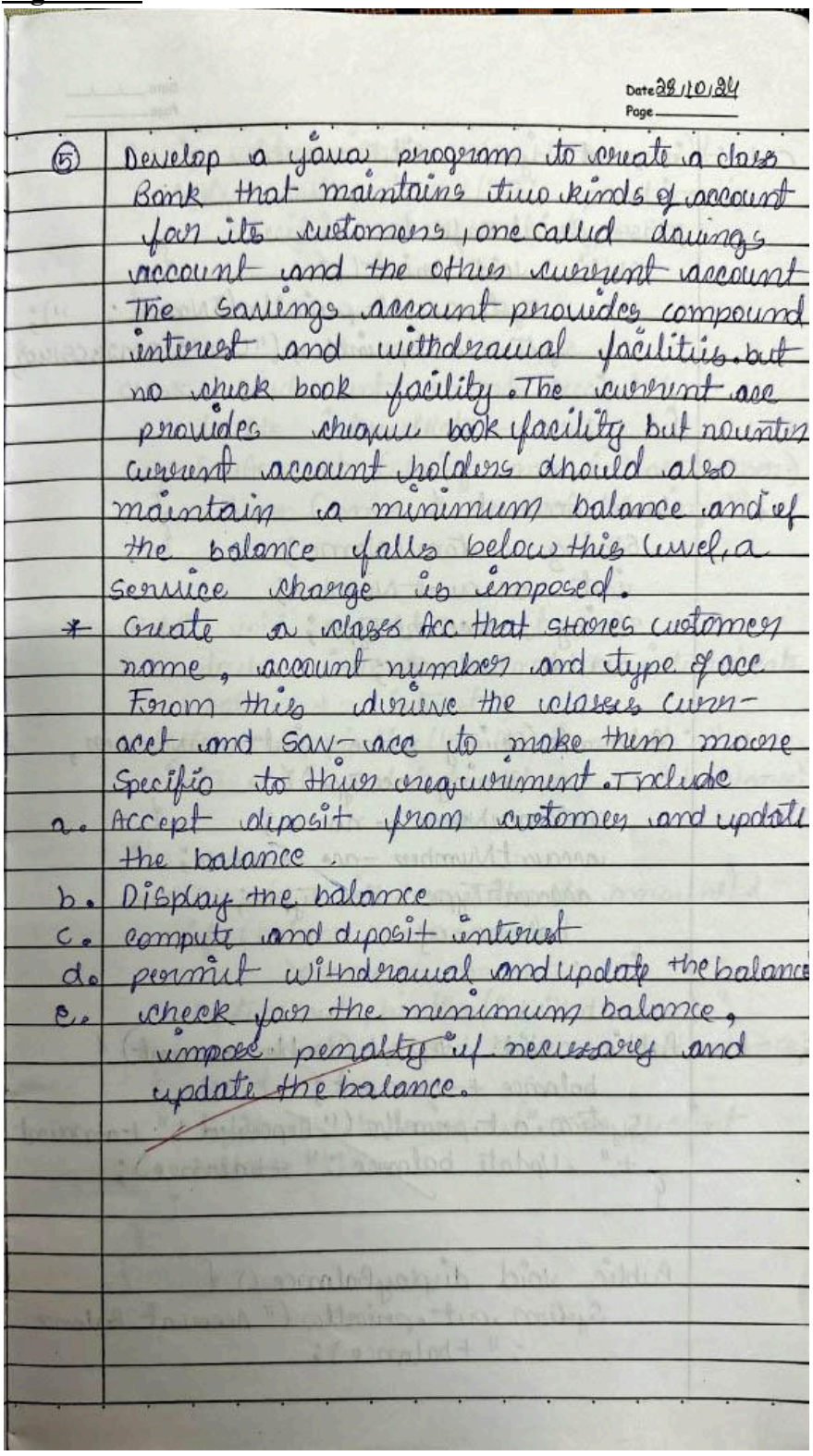
Output:



Program 5

Bank Account

Algorithm:



Date 28/10/24
Page

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 check book facility. The current acc provides check book facility but maintain current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

* Create a classes Acc that stores customer name, account number and type of acc. From this derive the classes curr-acct and sav-acc to make them more specific to their requirement. Include

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

Date __/__/__
Page _____

Code :

```
import java.util.Scanner

class PrintInfo {
    static void print() {
        System.out.println("Name : ");
        System.out.println("USN : 1BM23CS140");
    }
}

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

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

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

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

Date __/__/__
Page _____

```
Public void withdraw(double amount){
    System.out.println("This operation is
        specific to account type.");
    }
}

class SavAccount extends Account{
    double interestRate = 0.04;
    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.");
    }
  }
}
```

Date __/__/__
Page _____

```
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:" +amou
+ ". Updated balance:" +balance);
    checkMinBalance();
} else {
    System.out.println("Insufficient
    balance");
    }
}
}
```

Date __/__/__
Page _____

```
Public class Bank {
  Public static void main (String [] args) {
    printInfo.print();
    Scanner sc = new Scanner (System.in);
    System.out.println("Enter customer name");
      String name = sc.next()
    System.out.println("Enter account num");
    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 num");
       int accountnumber1 = sc.nextInt();
     CurAccount currentaccount = new CurAccount
    (name1, account number1);


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

    System.out.print("Enter the type of account
      Strin. accType = Sc.next();
```

Date / /
Page

```
if (accType.equals ("savings")) {
    Switch (choice) {
    Case 1:
    System.out.println ("Enter the deposit
        amount");
    double depositAmount = sc.nextDouble();
    savingsAccount.deposit (depositAmount);
        break;

    Case 2:
    System.out.println ("Enter withdrawal
        amount");
    double withdrawalAmount = sc.nextDouble();
    SavingsAccount.withdrawal (withdrawal);
        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);
    Savings Account.displayBalance();
        break;
```

```
System.out.println("Account number:"
+currentAccount.accountNumber);
System.out.println("Type of Account:"+
currentAccount.accountType);
break;

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

Date / /
Page

output:

```
Name : Karthik.O.Havalaghatta
USN : 1BM23CS140
Enter customer name:
Karan
Enter account number :
458
Enter Customer name:
Karthik
Enter account number:
965
---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 (savings/current):
Enter the deposit amount : 100000
Deposited : 10000 : updated balance : 1000000.


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 (savings/current) : current
Enter withdrawal amount : 15000
withdrawn : 15000 , updated balance : 93000
```

28.10

Code:

```
import java.util.Scanner;

class PrintInfo {
  static void print() {
      System.out.println("Name:Karthik O Havalaghatta ");
      System.out.println("USN: 1BM23CS140");
   }
}

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);
    }

    @Override
    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);
        }
    }

    @Override
    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) {
        PrintInfo.print();
        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.");
    }
  }
}
```

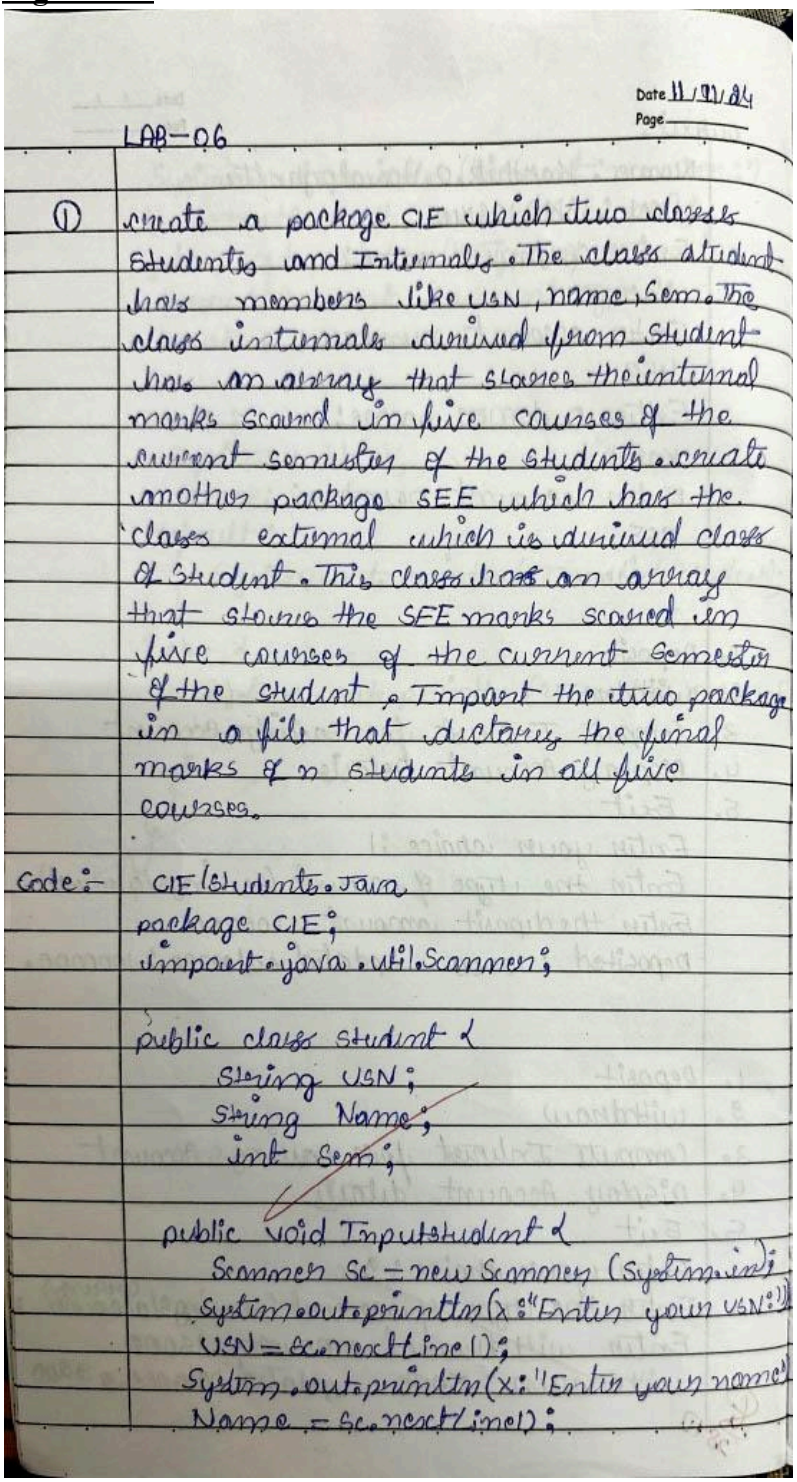
Output:

```
D:\1BM23CS140\LAB 5>java Bank
Name:Karthik O Havalaghatta
USN: 1BM23CS140
Enter customer name:
karan
Enter account number:
458
Enter customer name:
karthik
Enter account number:
965

-----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: 100000
Deposited: 100000.0. Updated balance: 100000.0

-----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: 50000
Deposited: 50000.0. Updated balance: 50000.0

-----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: 20000
Withdrawn: 20000.0. Updated balance: 80000.0

-----MENU-----
```

Program 6

CIE and SEE Packages

Algorithm:



Date 11/12/24
Page

LAB-06

① create a package CIE which has two classes Students and Internals. The class student has members like USN, name, Sem. The class internals derived from student has an array that stores the internal marks scored in five courses of the current semester of the students. create another package SEE which has the class external which is 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 package in a file that declares the final marks of n students in all five courses.

Code:- CIE/Students.java

```
package CIE;
import java.util.Scanner;

public class Student {
    String USN;
    String Name;
    int Sem;

    public void InputStudent {
        Scanner Sc = new Scanner (System.in);
        System.out.println(x:"Enter your USN:");
        USN = Sc.nextLine();
        System.out.println(x:"Enter your name");
        Name = Sc.nextLine();
```

Date __/__/__
Page _____

```
System.out.println("Enter your sem");
Sem = Sc.nextLine();
}
public void DisplayStudent(){
    System.out.println("Name:"+Name);
    System.out.println("USN:"+USN);
    System.out.println("Sem:"+Sem);
}
}
```

CIE/Internals.java

```
package CIE;
import java.util.Scanner;

public class Internal extends Students{
    int[] marks = new Scanner[5];
    public void InputCIEMarks(){
        Scanner Sc = new Scanner(System.in);
        System.out.println("Enter your CIE marks for 5 Subjects");
        for(int i=0; i<5; i++){
            System.out.println("Subjects"+(i+1)+":");
            marks[i] = Sc.nextInt();
        }
    }
}
```

Date / /
Page

SEE/External.java

```
package SEE;
import java.util.Scanner;
import CIE.Internal;
public class External extends Internal {
    int[] marks = new int[5];
    int[] finalmarks = new int[5];

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

Public void calculatefinalmarks() {
    for(int i=0; i<5; i++) {
        finalmarks[i] = this.marks[i]/2
                        +super.marks[i];
    }
}
public void Displayfinalmarks() {
    DisplayStudent();
    System.out.println("The final marks
                        for 5 subject");
    for (int i=0; i<5; i++) {
        System.out.println("Subject"+(i+1)+":"
                        +finalmarks[i]);
    }
}
}
```

Date __/__/__
Page _____

SEE/main.java:

```
import SEE.External;
import java.util.Scanner

public class main {
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.println("x: Enter the number
                of students :")
        int n = sc.nextInt();
        External [] student = new External [n];
        for(int i=0 ; i<n; i++){
        System.out.println("\n Enter the details
                of student no: "+(i+1)+" :");
        student[i] = new External();
        Student[i].InputStudent();
        Student[i].InputCIEMarks();
        Student[i].InputSEEMarks();
        Student[i].Calculatifinal marks();
    }

    System.out.println("\n Final marks of
                Student :");
    for(int i=0 ; i<5 ; i++){
        System.out.println("\n Student "+(i+1)+":
        Student[i].Displaymarks();
    }
    sc.close();
    }
}
```

Code:

//Externals

```
package SEE;
import java.util.Scanner;
import CIE.Internal;
public class External extends Internal{
    int[] marks=new int[5];
    int[] finalmarks = new int[5];


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

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

    public void Displayfinalmarks(){
        DisplayStudent();
        System.out.println(x:"The final marks for 5 subjects");
        for(int i=0;i<5;i++){
            System.out.print("Subjects"+(i+1)+":"+finalmarks[i]);
        }

    }

}

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

public class Internal extends Student{
    int[]marks=new int[5];
    protected int[]ciemarks=new int[5];
    public void InputCIEMarks(){
        Scanner sc = new Scanner(System.in);
        System.out.println(x:"Enter your CIE marks for 5 subjects");
```

```
        for(int i=o;i<5;i++){
            System.out.println("Subject"+(i+1)+":");
            ciemarks[i]=sc.nextInt();
        }
    }
}



//Student
package CIE;
import.java.util.Scanner;

public class student{
    String USN;
    String Name;
    int Sem;




    public void InputStudent{
        Scanner sc=new Scanner(System.in);
        System.out.println(x:"Enter your usn:");
        USN=sc.nextLine();
        System.out.println(x:"Enter your name");
        Name=sc.nextLine();
        System.out.println(x:"Enter your Sem");
        Sem=sc.nextLine();
}



public void DisplayStudent(){
    System.out.println("Name:"+Name);
    System.out.println("USN:"+USN);
    System.out.println("Sem:"+Sem);

}
}
//Main
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("x:Enter the number of students:");
        int n=sc.nextInt();
        Externals[]student = new Externals[n];
        for(int i=0;i<n;i++){
          System.out.println("\n Enter the details of student here:"+(i+1)+":");
          student[i]= new Externals();
          student[i].InputStudent();
          student[i].InputCIEMarks();
          student[i].InputSEEMarks();
          student[i].Calculatefinalmarks();
        }

        System.out.println("\n Finalmarks of student:");
        for(int i=0;i<5;i++){
          System.out.println("\nStudent"+(i+1)+":");
          student[i].Displayfinalmarks();

       }
       sc.close();
        }


}
```

Output:

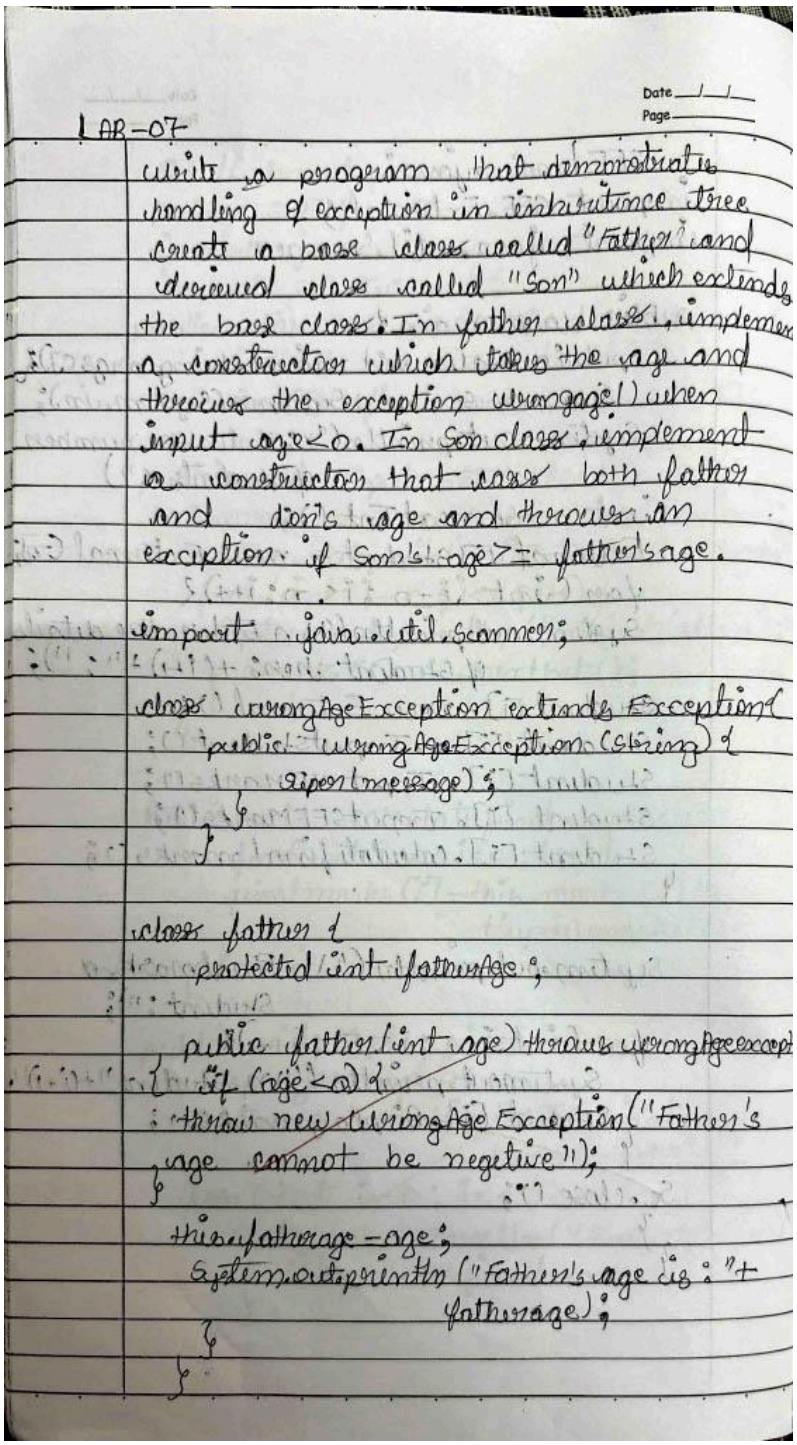
```
PS D:\1BM23CS139> cd "d:\1BM23CS139\" ; if ($?) { javac main.java } ; if ($?) { java main }

Enter the number of students:
2

Enter details for Students here1:
Enter your usn here:
139
Enter your name here:
Karna
Enter your semester here:
3
Enter your CIE marks for the 5 subjects here:
Subject1:98
Subject2:99
Subject3:100
Subject4:100
Subject5:97
CIE marks are as follows:

Subject1:98
Subject2:99
Subject3:100
Subject4:100
Subject5:97
Enter the 5 SEE Marks here:

Subject1:100

Subject2:100

Subject3:99

Subject4:98

Subject5:95
```

```
Enter details for Students here2:
Enter your usn here:
140
Enter your name here:
Karthik
Enter your semester here:
3
Enter your CIE marks for the 5 subjects here:
Subject1:99
Subject2:98
Subject3:99
Subject4:95
Subject5:91
CIE marks are as follows:

Subject1:99
Subject2:98
Subject3:99
Subject4:95
Subject5:91
Enter the 5 SEE Marks here:

Subject1:100

Subject2:100

Subject3:92

Subject4:91

Subject5:98

Finalmarks of students:

Student1:
Name:Karna
USN:139
Semester:3

The final marks of the 5 subjects are:

Subject1:99
Subject2:99
Subject3:99
Subject4:99
Subject5:96
```

Program 7

Father and Son age

Algorithm:



LAB-07

Write a program that demonstrates handling of exception 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 input age<0. In Son class, implement a constructor that takes both father and son's age and throws an exception if Son's age >= father's age.

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

class father{
    protected int fatherAge;

    public father(int age) throws wrongAgeexcep
    {
        if (age<0){
        throw new wrongAgeException("Father's
    age cannot be negetive");

    this.fatherage=age;
    System.out.println("Father's age is: "+
                        fatherage);
    }
}
```

Date __/__/__
Page _____

```
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 negetive");
}
if (SonAge >= fatherAge) {
  throw new wrongAgeexception ("Son's
age cannot be geretion than or equal to
father's age !");
}
this.SonAge = SonAge;
System.out.println ("Son's age is :" +SonAge)
}
}

Public class ExceptionInInheritance {
  public static void main (String [] Jargs
{ Scanner scanner = new scanner (System.in)
  try {
    System.out.println ("Enter father's age");
    int father age = scanner.nextInt ();
  System.out.println ("Enter Son's age :");
  int SonAge = Scanner.nextInt () ;
    Son Son = new Son (fatherage, SonAge);
} catch (wrongAge exception e) {
      System.err.exception " + e.getMessage());
} catch (Exception e) {
    System.err.println ("Invalid input
      please enter integer");
```

Date __/__/__
Page _____

```
finally {
    Scanner.close();
}
System.out.println("Karthik, 1BM23CS140");
}
```

Output:

1. Enter fathers age :- 1
Enter Son's age : 20
Exception : Father's age cannot be negetive
Karthik 1BM23CS140

2. Enter father's age : 47
Enter Son's age : 20
Fathers age is : 47
Son's age is 20
Karthik 1BM23CS140

Code:

```
import java.util.Scanner;

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

class Father {
    protected int fatherAge;

    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age cannot be negative!");
        }
        this.fatherAge = age;
        System.out.println("Father's age is: " + fatherAge);
    }
}

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

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

        try {
            System.out.print("Enter father's age: ");
            int fatherAge = scanner.nextInt();
            System.out.print("Enter son's age: ");
            int sonAge = scanner.nextInt();
```

```
            Son son = new Son(fatherAge, sonAge);
        } catch (WrongAgeException e) {
            System.err.println("Exception: " + e.getMessage());
        } catch (Exception e) {
            System.err.println("Invalid input! Please enter integers.");
        } finally {
            scanner.close();
        }
        System.out.println("KARTHIK,1BM23CS140") ;
    }
}
```

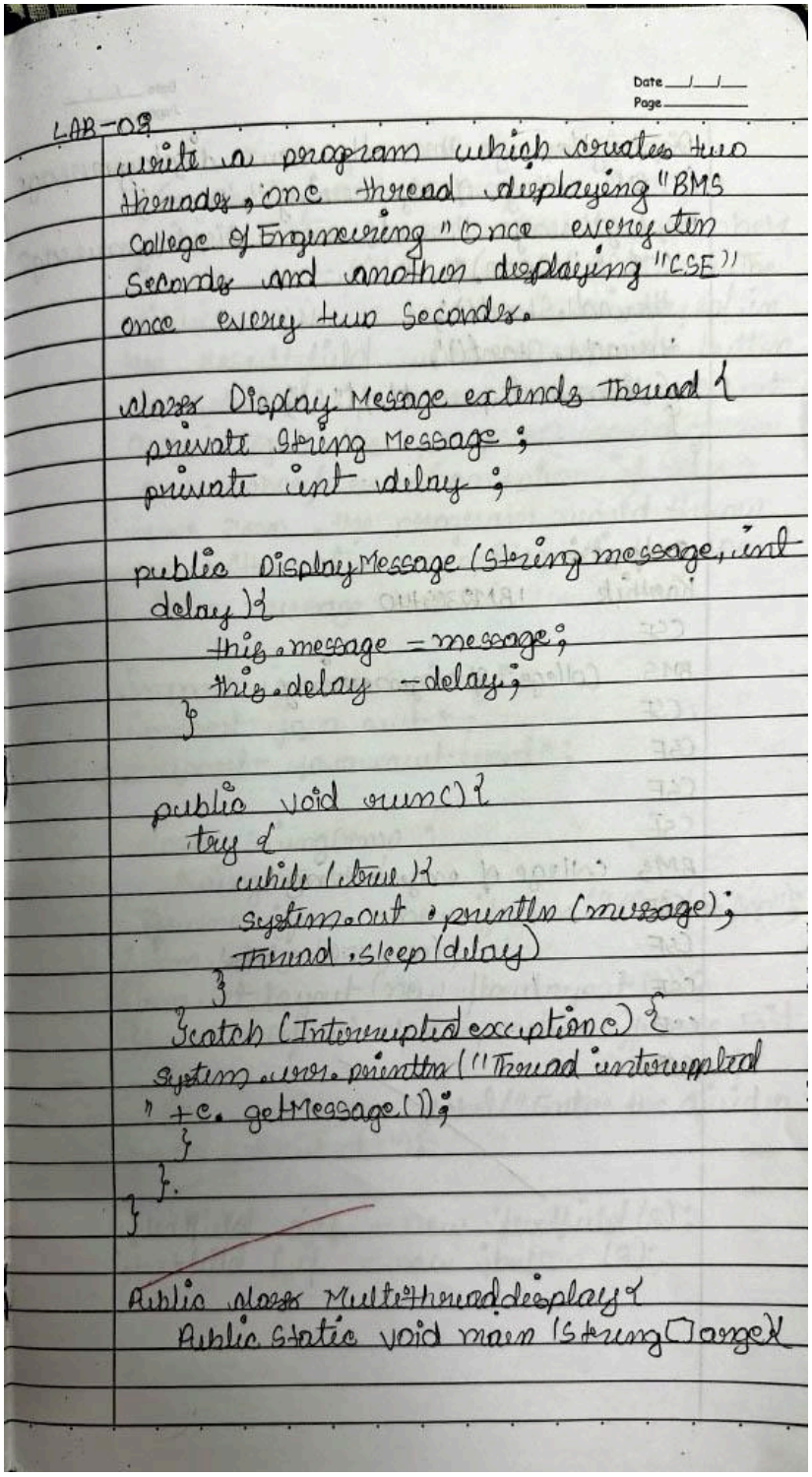
Output:

```
Microsoft Windows [Version 10.0.26100.2314]
(c) Microsoft Corporation. All rights reserved.

C:\Users\karth\OneDrive\Documents\JAVA>javac ExceptionInInheritance.java

C:\Users\karth\OneDrive\Documents\JAVA>java ExceptionInInheritance.java
Enter father's age: 40
Enter son's age: 60
Father's age is: 40
Exception: Son's age cannot be greater than or equal to father's age!
KARTHIK,1BM23CS140

C:\Users\karth\OneDrive\Documents\JAVA>java ExceptionInInheritance.java
Enter father's age: -40
Enter son's age: 20
Exception: Father's age cannot be negative!
KARTHIK,1BM23CS140

C:\Users\karth\OneDrive\Documents\JAVA>java ExceptionInInheritance.java
Enter father's age: 40
Enter son's age: 20
Father's age is: 40
Son's age is: 20
KARTHIK,1BM23CS140

C:\Users\karth\OneDrive\Documents\JAVA>
```

Program 8

Multi Threading

Algorithm:



LAB-09

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 DisplayMessage extends Thread {
    private String message;
    private int delay;

    public DisplayMessage(String message, int delay) {
        this.message = message;
        this.delay = delay;
    }

    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(delay);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread interrupted " + e.getMessage());
        }
    }
}

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

Date / /
Page

```
DisplayMessage thread1 = new displaymessage
("BMS college of engineering", 6000);
DisplayMessage thread2 = new DisplayMessage
("CSE", 3000);
thread1.start();
thread2.start();
System.out.println("");
}
}
```

Output:-

```
Kanthik    1BM23CS140
CSE
BMS College of engineering
CSE
CSE
CSE
CSE
BMS College of engineering
CSE
CSE
CSE
CSE
CSE
```

Code:

```
class DisplayMessage extends Thread {
    private String message;
    private int delay;

    public DisplayMessage(String message, int delay) {
        this.message = message;
        this.delay = delay;
    }

    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(delay);
            }
        } catch (InterruptedException e) {
            System.err.println("Thread interrupted: " + e.getMessage());
        }
    }
}

public class MultiThreadDisplay {
    public static void main(String[] args) {
        DisplayMessage thread1 = new DisplayMessage("BMS College of Engineering", 10000);
        DisplayMessage thread2 = new DisplayMessage("CSE", 2000);
        thread1.start();
        thread2.start();
        System.out.println("") ;
    }
}
```

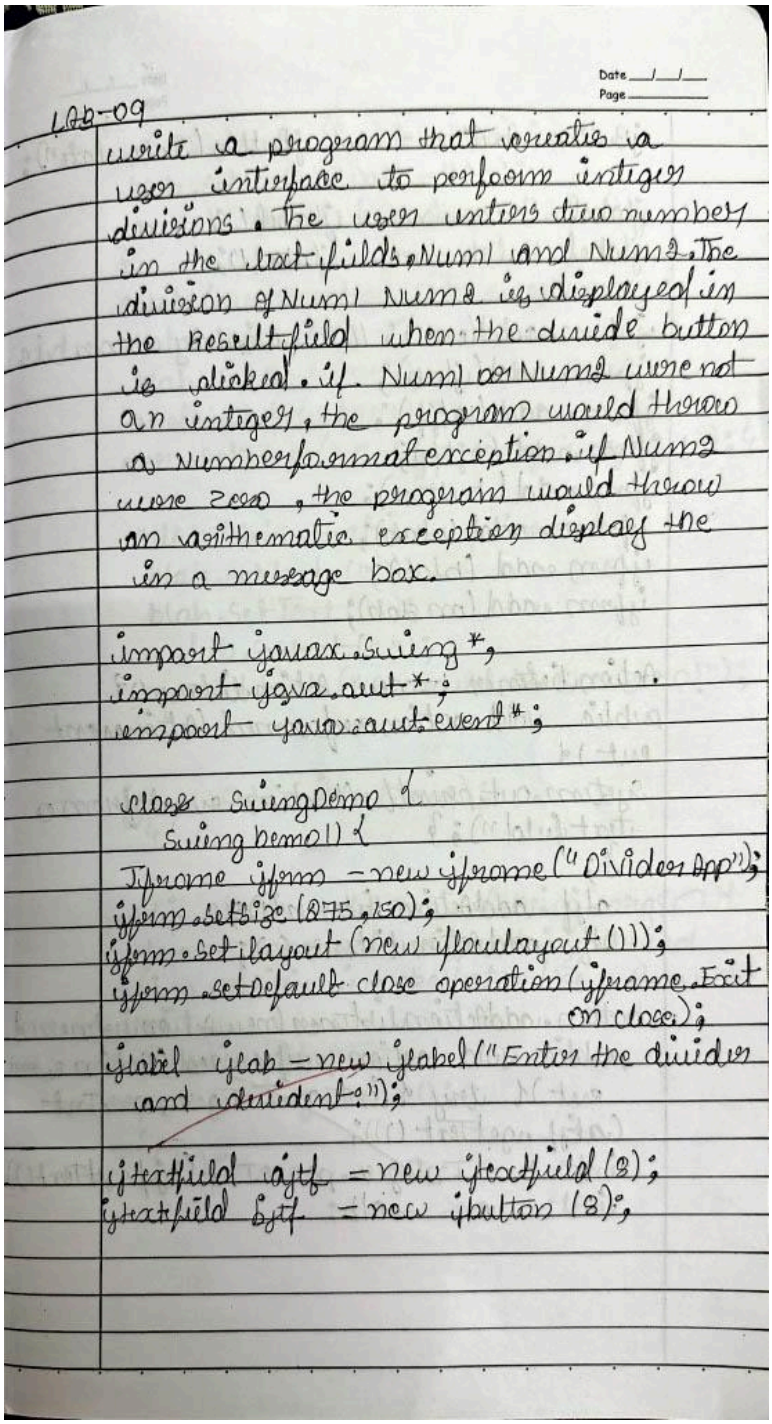
Output:

```
Microsoft Windows [Version 10.0.26100.2314]
(c) Microsoft Corporation. All rights reserved.

C:\Users\karth\OneDrive\Documents\JAVA>javac MultiThreadDisplay.java

C:\Users\karth\OneDrive\Documents\JAVA>java MultiThreadDisplay.java

BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
^C
C:\Users\karth\OneDrive\Documents\JAVA>
```

Program 9

Divider and Divisor:

Algorithm:



Date __/__/__
Page _____

LAB-09

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 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 in a message box.

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

class SwingDemo {
    SwingDemo() {
JFrame jfrm = new JFrame("Divider App");
jfrm.setSize(275, 150);
jfrm.setLayout(new FlowLayout());
jfrm.setDefaultCloseOperation(JFrame.EXIT_
                                ON_CLOSE);
JLabel jlab = new JLabel("Enter the divider
  and dividend:");

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

Date / /
Page

```
jButton button = new jButton("Calculate");

jlabel err = new jlabel();
jlabel alab = new jlabel();

jfrm.add(err); //to display error on this
jfrm.add(jlab);
jfrm.add(atf);
jfrm.add(btf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(ansLab);

ActionListener l = new ActionListener()
public void actionPerformed(ActionEvent
ext){
System.out.println("Action event from a
textfield");}
};
atf.addActionListner(l);
btf.addActionListner(l);

button.addActionListner(new ActionListner(
public void actionPerformed(ActionEvent
ext){ try{int a = Integer.parseInt
(atf.getText());
int b = Integer.parseInt(btf.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!");
}}

jFrame.SetVisible (true);
}

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

Date / /
Page

output :-

Enter only Intigers !
Enter the dividend and divisor
abc & calculate

Should be NON zero!
Enter the dividend and divisor
10 0 calculate

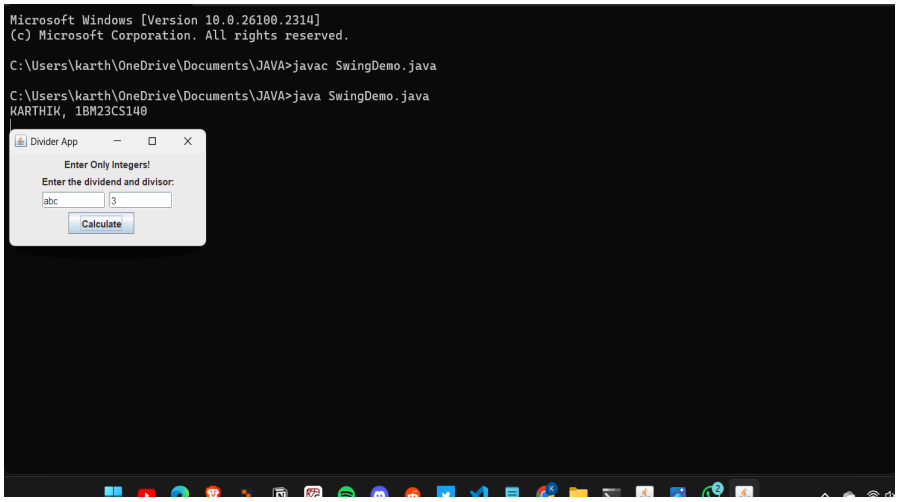
Enter the dividend and divisor
100 33 calculate
A=100 B=33 Ans=3

Code:

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

class SwingDemo {
   SwingDemo() {

      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 150);
      jfrm.setLayout(new FlowLayout());
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


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


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

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


    jfrm.setVisible(true);
  }

  public static void main(String args[]) {
    System.out.println("KARTHIK, 1BM23CS140") ;
    SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        new SwingDemo();
      }
    });
  }
}
```

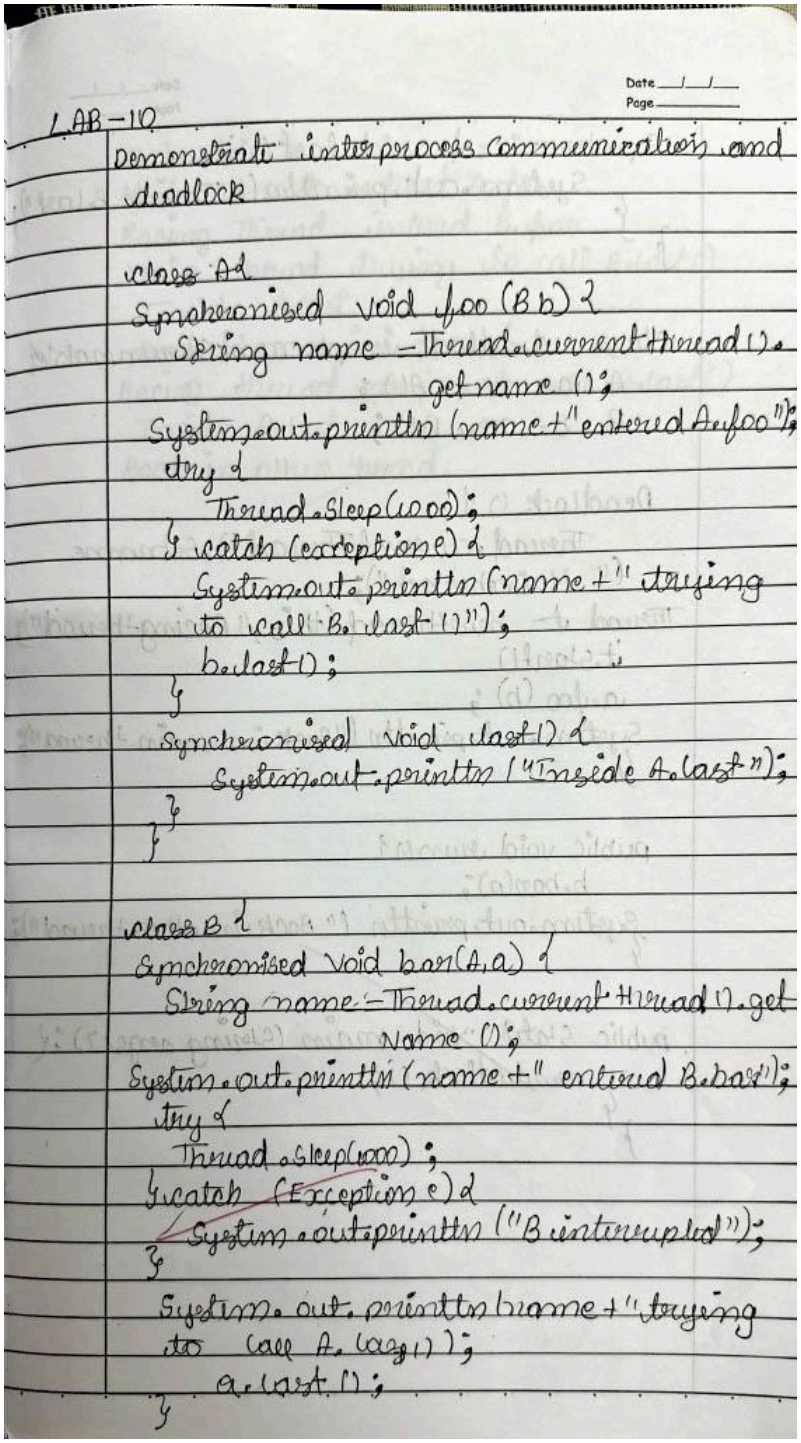
Output:



Program 10:

Demonstration of Deadlock

Algorithm:



LAB-10

Demonstrate interprocess communication and deadlock

```
class A{
Synchronised 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(name+" trying
    to call B.last()");
    b.last();
  }
  Syncheronised 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();
  }
```

Date / /
Page

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

Date __/__/__
Page _____

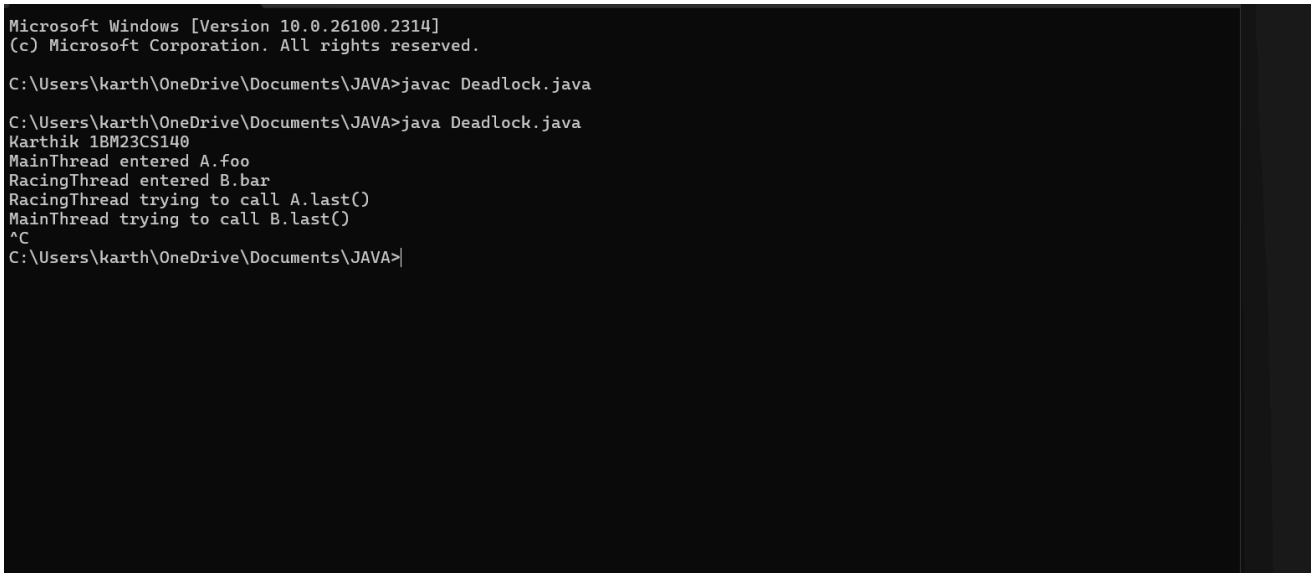
Output :-
Main thread entered A.foo
Racing Thread entered B.bar
Main thread trying to call B.last()
Inside A.last
Back in main thread
Racing thread trying to call A.last()
Inside A.last
Back in other thread.

Code:

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

   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); // Simulating some work
      } catch (Exception e) {
         System.out.println("B Interrupted");
      }
      System.out.println(name + " trying to call A.last()");
      a.last(); // Deadlock occurs here
   }

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

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

   Deadlock() {
      Thread.currentThread().setName("MainThread");
      Thread t = new Thread(this, "RacingThread");
      t.start();
      a.foo(b); // get lock on A in this thread
      System.out.println("Back in main thread");
```

```
    }

    public void run() {
        b.bar(a); // get lock on B in other thread
        System.out.println("Back in other thread");
    }

    public static void main(String args[]) {
        System.out.println("Karthik 1BM23CS140") ;
        new Deadlock();
    }
}
```

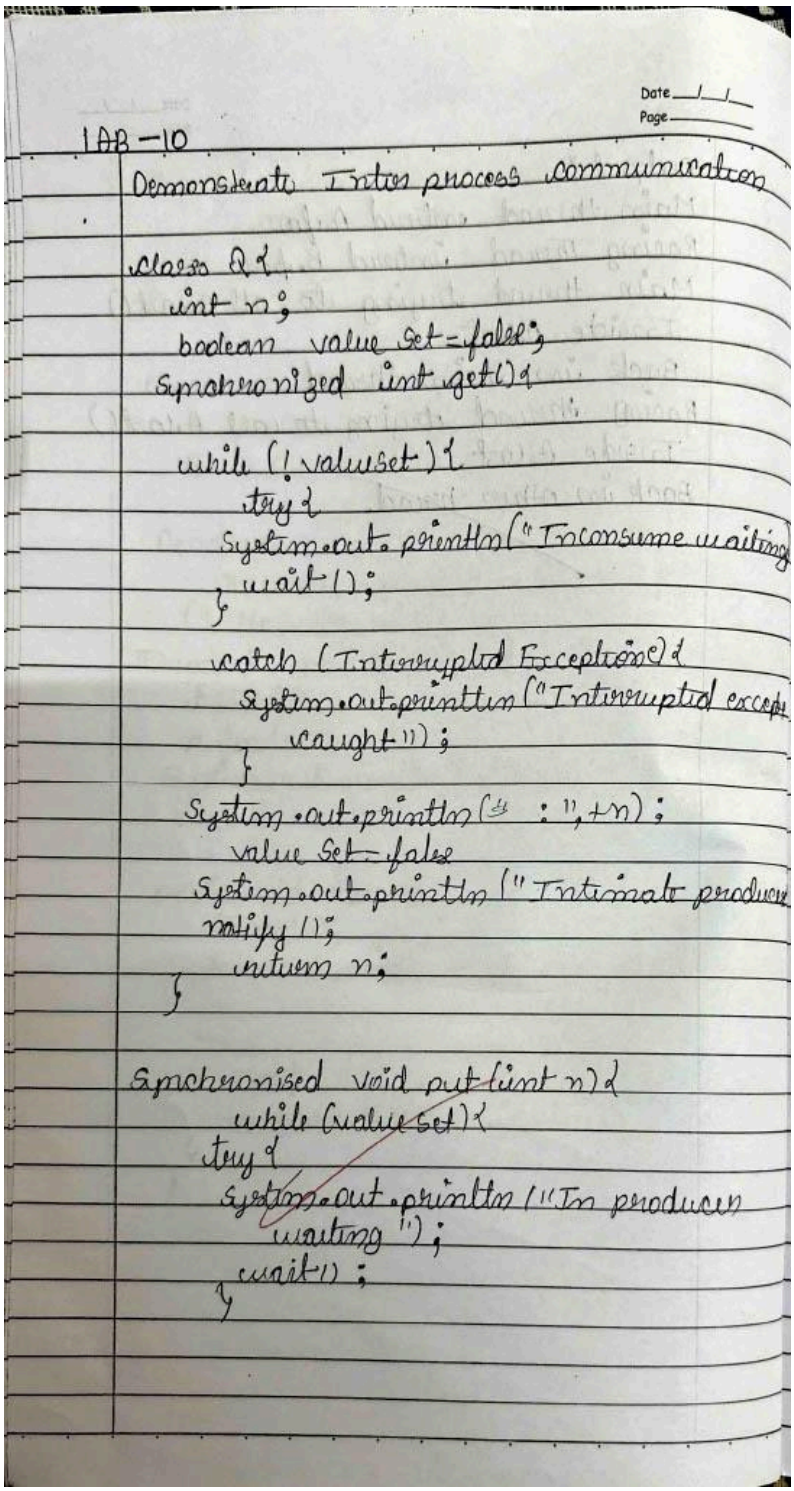
Output:



Program 11:

Demonstration of PCF

Algorithm:



LAB – 10

Demonstrate Inter process communication

```
class Q {
  int n;
  boolean valueSet = false;
  synchronized int get() {
    while (!valueSet) {
      try {
        System.out.println("\nConsume waiting");
        wait();
      }
      catch (InterruptedException e) {
        System.out.println("Interrupted exception
          caught");
      }
    System.out.println("Got : ", n);
      valueSet = false;
    System.out.println("\nIntimate producer");
    notify();
    return n;
  }

synchronised void put(int n) {
    while (valueSet) {
  try {
    System.out.println("\nProducer
      waiting");
    wait();
  }
```

```
catch (InterruptedException e){
    System.out.println("Put: "+n);
    System.out.println("Put: "+n);
    System.out.println("\n Intimate consumer");
    notify();
}

class Producer implements runnable {
    Qq;
    producer(Qq) {
        this.q = q;
        new Thread(this, "producer").start();
    }
    public void run(){
        int i=0;
        while(i<15){
            q.put(i++);
        }
    }
}

class Consumer implements runnable {
    Qq;
    Consumer(Qq) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }

    public void run(){
        int i=0;
        while(i<15){
            int n=q.get();
            System.out.println("Consumed: "+n);
            i++;
        }
    }
}
```

Date / /
Page

```
class PC_Fixed {
    public static void main(String args[])
    {
        Q q = new Q();
        new producer(q);
        new Consumer(q);
        System.out.println("Press control to
            Stop");
    }
}
```

Output:

```
Put: 1
Got: 1
put: 2
Got: 2
Put : 3
Got : 3
put : 4
Got : 4
Put : 5
Got : 5
```

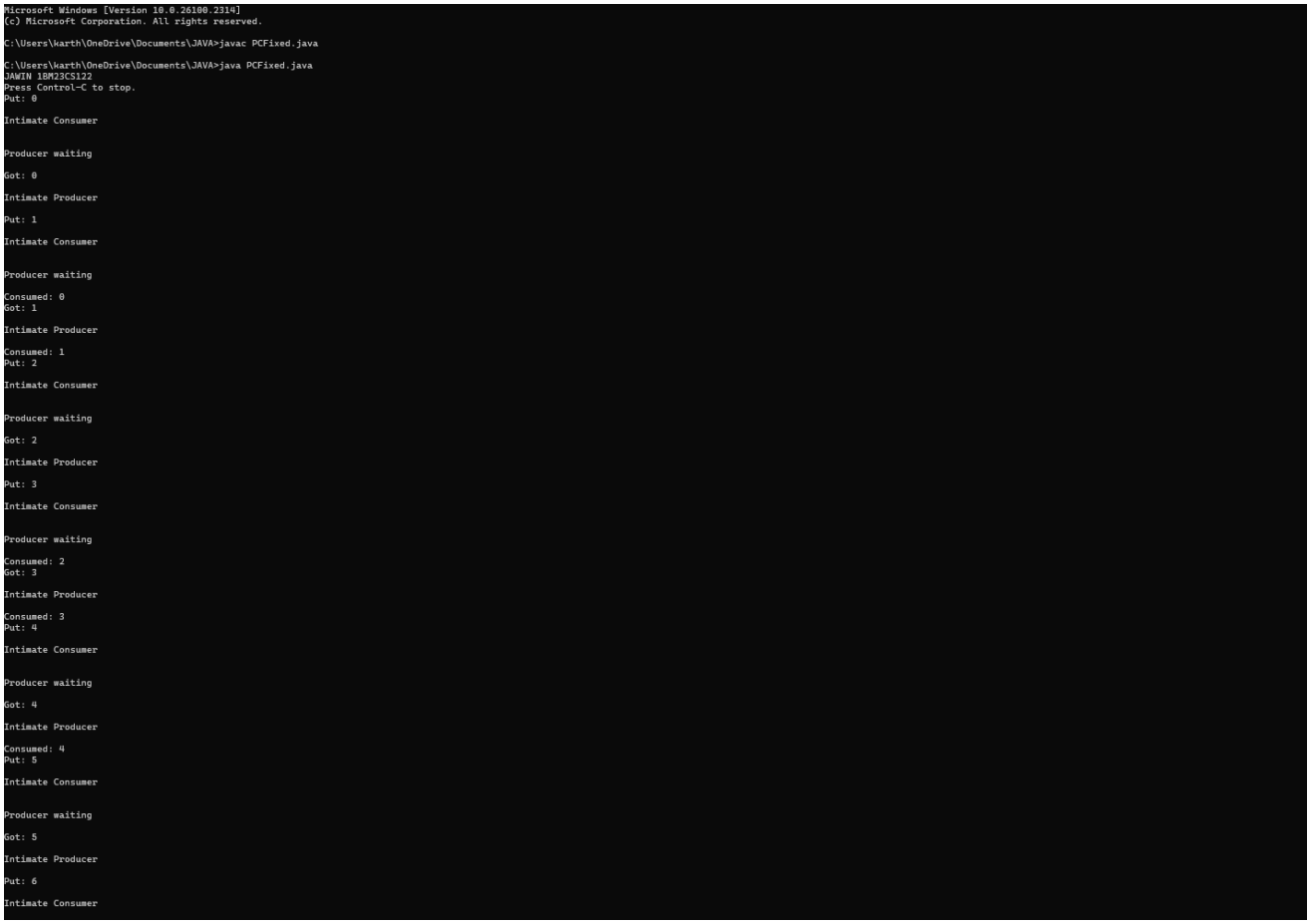
Dz: 12

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++;
        }
    }
}

public class PCFixed {
    public static void main(String args[]) {
        System.out.println("JAWIN 1BM23CS122") ;
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
    }
}
```

Output:


Microsoft Windows [Version 10.0.26100.2314]
(c) Microsoft Corporation. All rights reserved.
C:\Users\karth\OneDrive\Documents\JAVA>javac PCFixed.java
C:\Users\karth\OneDrive\Documents\JAVA>java PCFixed.java
JAWIN 1BM23CS122
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
Consumed: 3
Put: 4
Intimate Consumer
Producer waiting
Got: 4
Intimate Producer
Consumed: 4
Put: 5
Intimate Consumer
Producer waiting
Got: 5
Intimate Producer
Put: 6
Intimate Consumer