**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

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



# LAB REPORT on

**Object Oriented Java Programming**

**(23CS3PCOOJ)**

***Submitted by*  Anu Sai Shree R (1BM23CS045)**

***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 **Anu Sai Shree R (1BM23CS045),** 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 | Sheetal V A  Assistant Professor  Department of CSE, BMSCE |

# Index

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 | 09/10/2024 | Implement Quadratic Equation | 4-6 |
| 2 | 16/10/2024 | Calculation Of Student SGPA | 7-11 |
| 3 | 23/10/24 | Demonstration of Array of objects of each book type | 12-15 |
| 4 | 23/10/2024 | Demonstration Of Abstract Class | 16-19 |
| 5 | 30/10/2024 | Implementation of Bank | 20-27 |
| 6 | 13/11/2024 | Demonstration Of Package | 28-35 |
| 7 | 20/11/2024 | Demonstration Of Exception Handling | 36-40 |
| 8 | 27/11/2024 | Demonstration Of Threads | 41-43 |
| 9 | 27/11/2024 | User Interface for division of 2 Numbers(Open Ended Exercise) | 44-48 |
| 10 | 27/11/2024 | Demonstrate Inter process Communication and deadlock(Open Ended Exercise) | 49-58 |

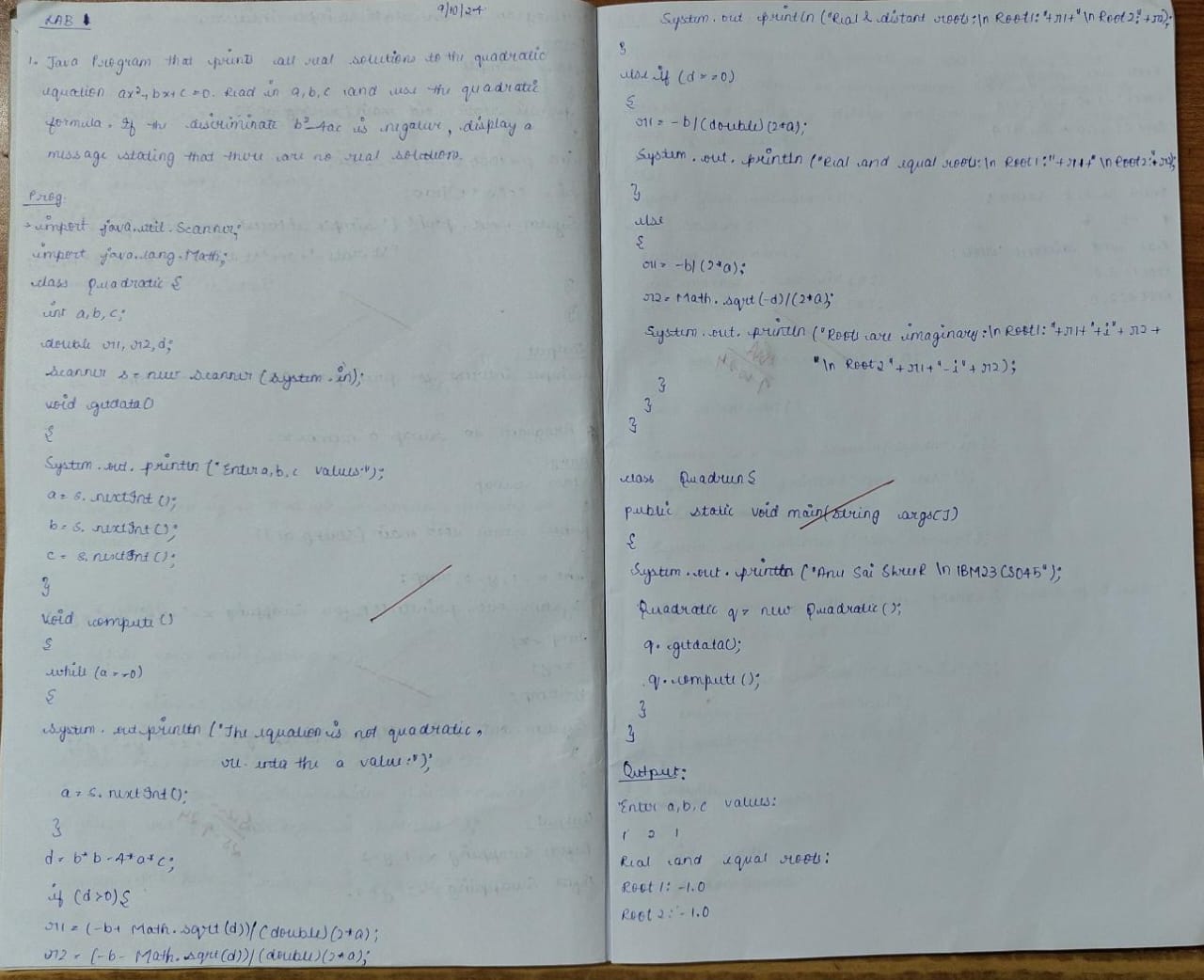
# Github Link:

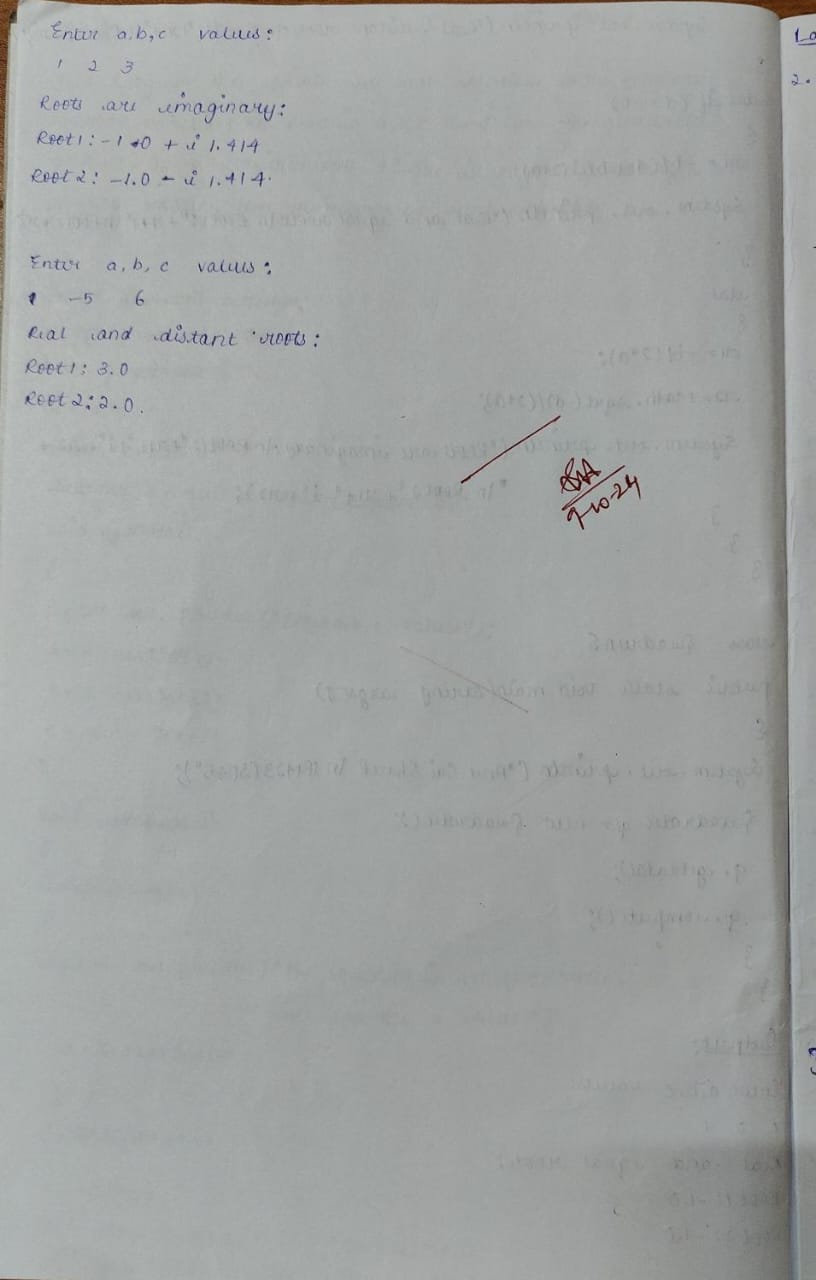
<https://github.com/Anu1BM23CS045/Java-Lab-OOJ>

# Program 1

## Implement Quadratic Equation

Program that prints all real solutions to the quadratic equation ax2+bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2-4ac is negative, display a message stating that there are no real solutions.

Algorithm: 



Code:

import java.util.Scanner;

import java.lang.Math;

class Quadratic {

int a, b, c;

double r1, r2, d;

Scanner s = new Scanner(System.in);

void getdata() {

System.out.println("Enter a,b,c values:");

a = s.nextInt();

b = s.nextInt();

c = s.nextInt();

}

void compute() {

while (a == 0) {

System.out.println("The equation is not quadratic,re-enter the a value:");

a = s.nextInt();

}

d = b \* b - 4 \* a \* c;

if (d > 0) {

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

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

System.out.println("Real and distant roots:\nRoot1:" + r1 + "\nRoot2:" +r2);

} else if (d == 0) {

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

System.out.println("Real and equal roots:\nRoot1:" + r1 + "\nRoot2:" + r1);

} else {

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

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

System.out.println("Roots are imaginary:\nRoot1:" + r1 + "+i" + r2 + "\nRoot2:" + r1 + "-i" + r2);

}

}

}

class Quadrun {

public static void main(String args[]) {

Quadratic q = new Quadratic();

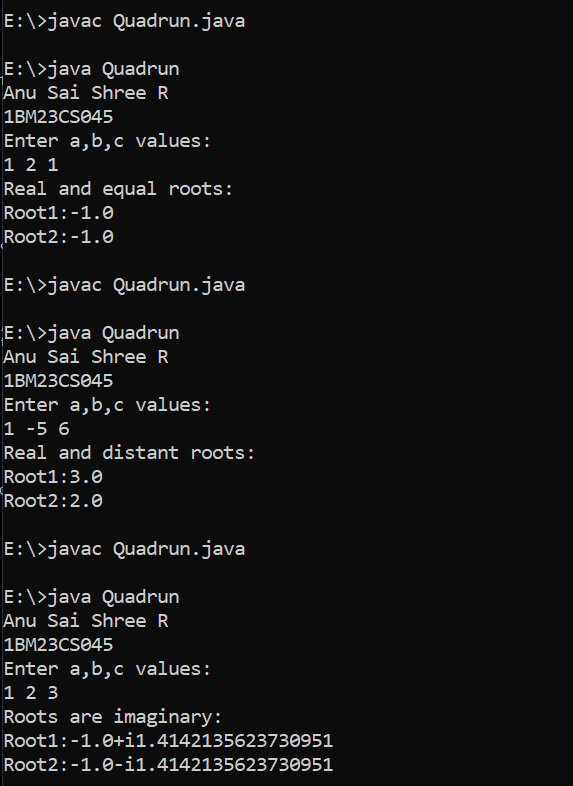
q.getdata();

q.compute();

}

}

Output:

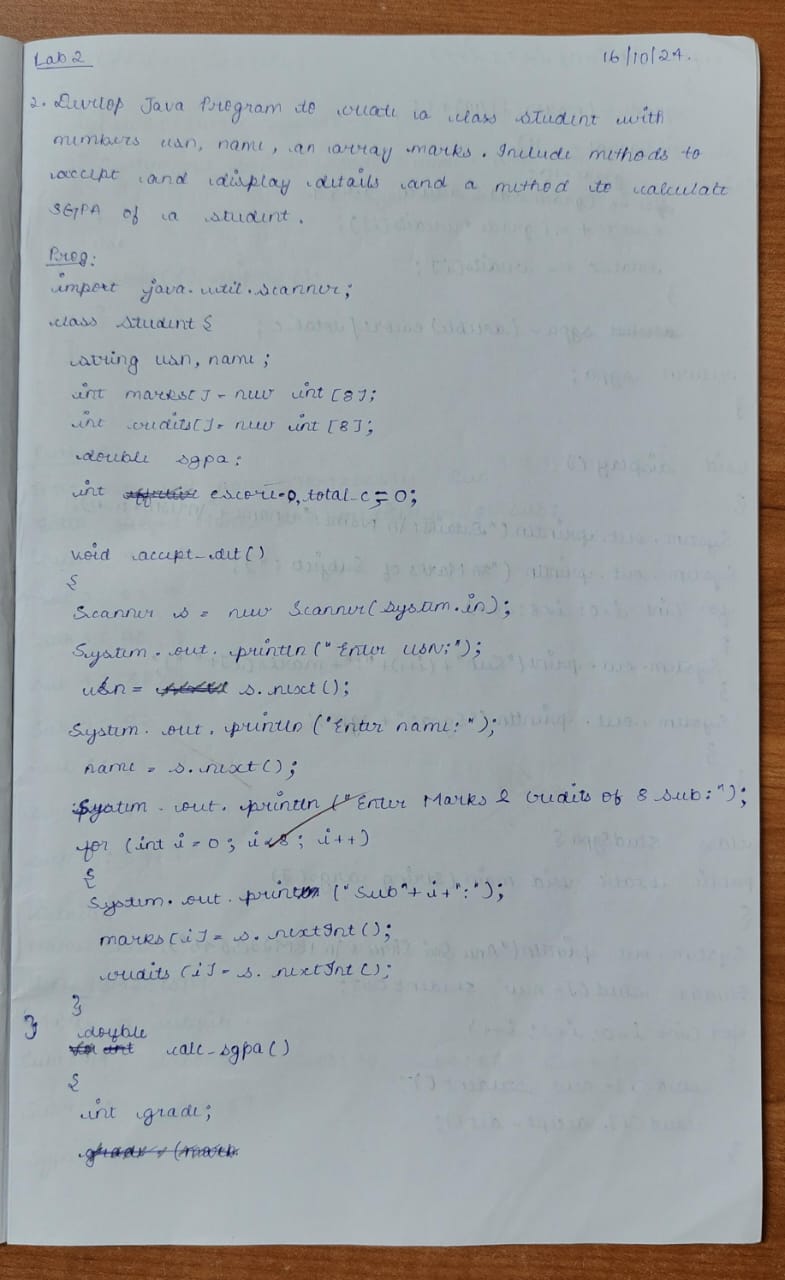


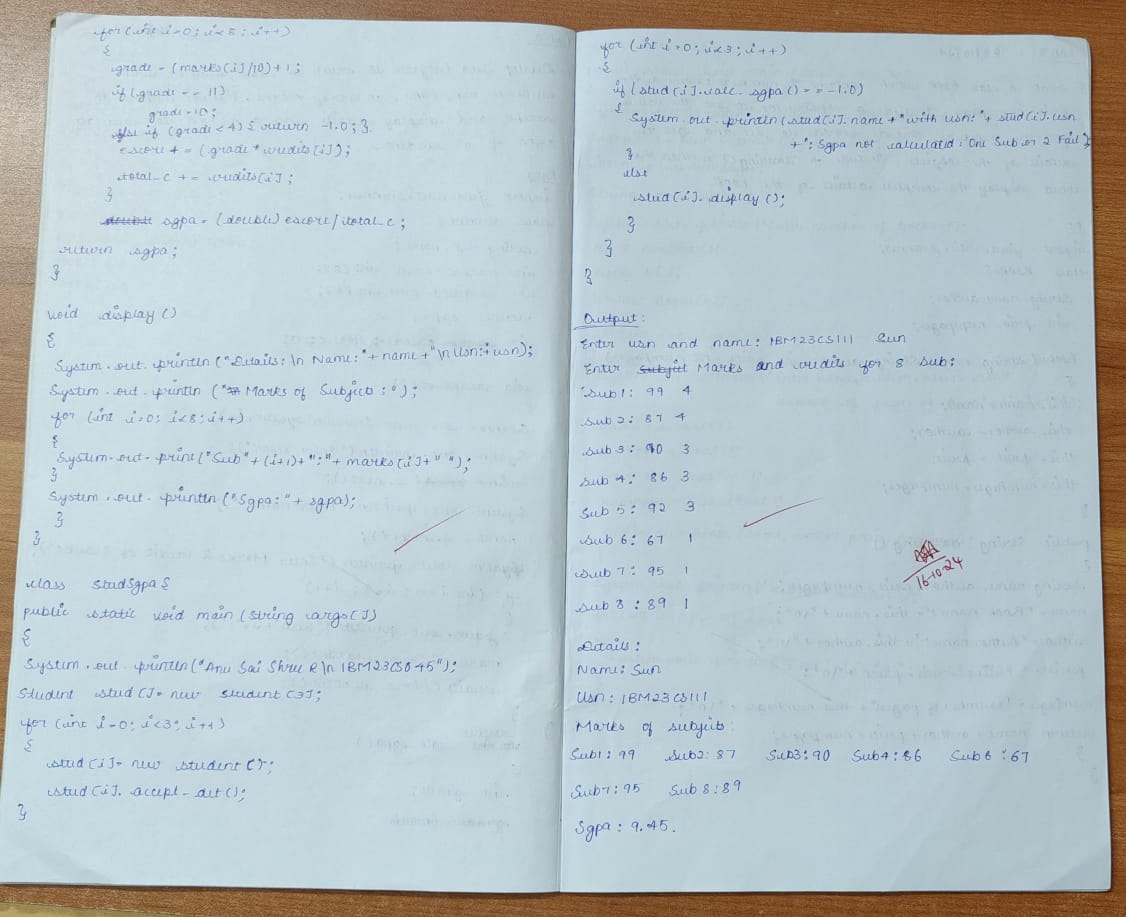
# Program 2

## Calculation Of Student SGPA

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:





Code:

import java.util.Scanner;

class Student {

int marks[] = new int[8];

int credits[] = new int[8];

String usn, name;

double sgpa;

int escore = 0, total\_c = 0;

void accept\_det() {

Scanner s = new Scanner(System.in);

System.out.print("Enter usn and name:");

usn = s.next();

name = s.next();

System.out.println("Enter Subject marks with it's respective credits:");

for (int i = 0; i < 8; i++) {

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

marks[i] = s.nextInt();

credits[i] = s.nextInt();

}

}

double calc\_sgpa() {

int grade;

for (int i = 0; i < 8; i++) {

grade = (marks[i] / 10) + 1;

if (grade == 11)

grade = 10;

else if (grade < 4)

return -1.0;

escore += (grade \* credits[i]);

total\_c += credits[i];

}

sgpa = (double) escore / total\_c;

return sgpa;

}

void display() {

System.out.println("Details:\nName:" + name + "\nUsn:" + usn + "\nMarks of subjects:");

for (int i = 0; i < 8; i++) {

System.out.print("Sub" + (i + 1) + ": " + marks[i] + " ");

}

System.out.println("\nSgpa:" + sgpa);

}

}

class StudSgpa {

public static void main(String args[]) {

System.out.println("Anu Sai Shree R\n1BM23CS045");

Student stud[] = new Student[3];

for (int i = 0; i < 3; i++) {

stud[i] = new Student();

stud[i].accept\_det();

}

for (int i = 0; i < 3; i++) {

if (stud[i].calc\_sgpa() == -1.0)

System.out.println(stud[i].name + "with usn:" + stud[i].usn + ":Sgpa cannot be calculated:Any one sub is Failed");

else

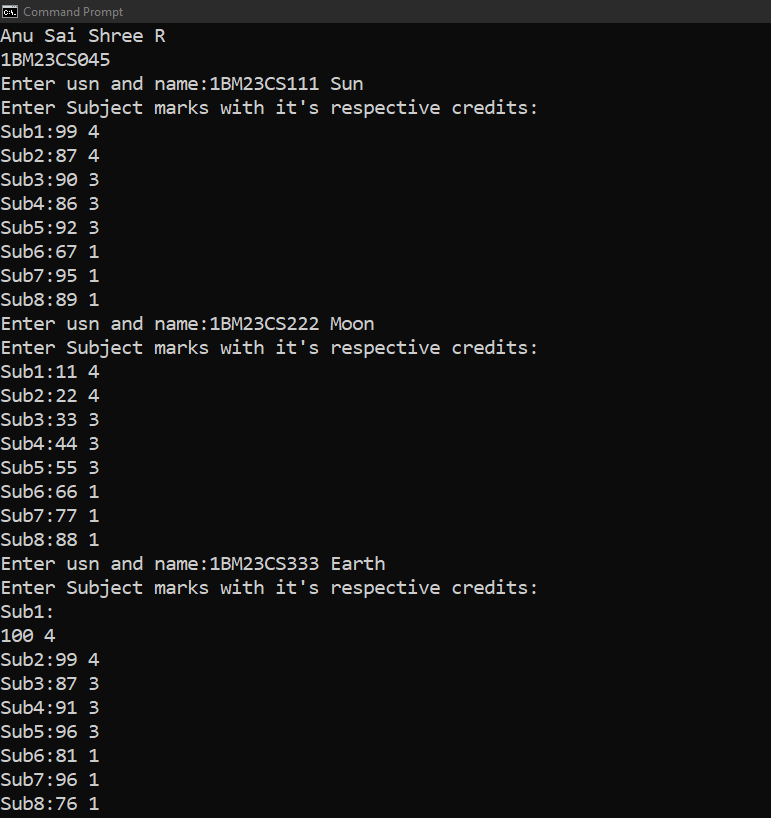
stud[i].display();

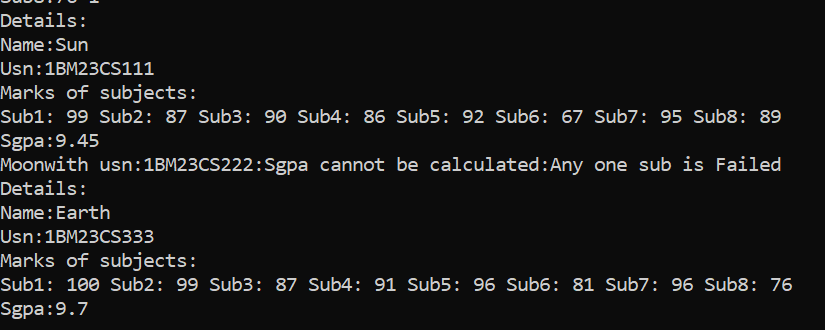
}

}

}

Output:



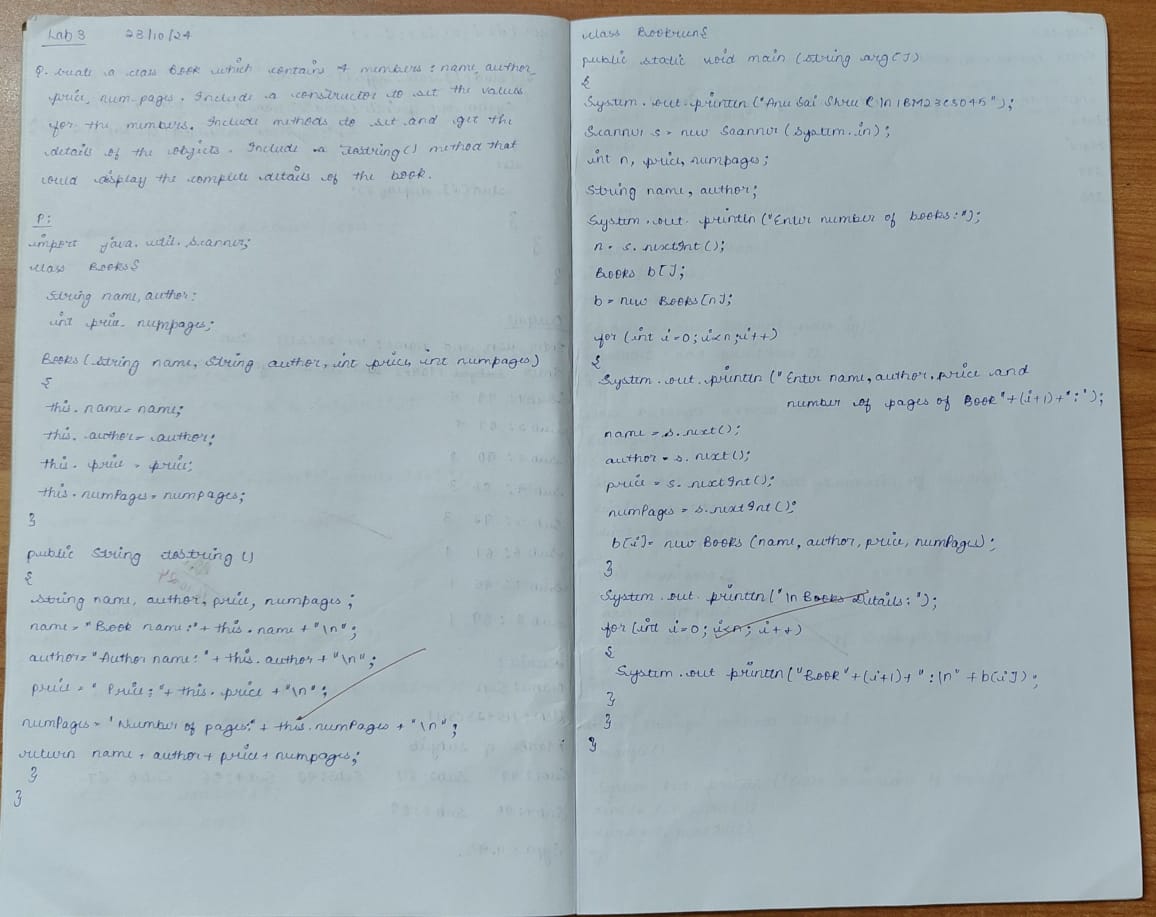


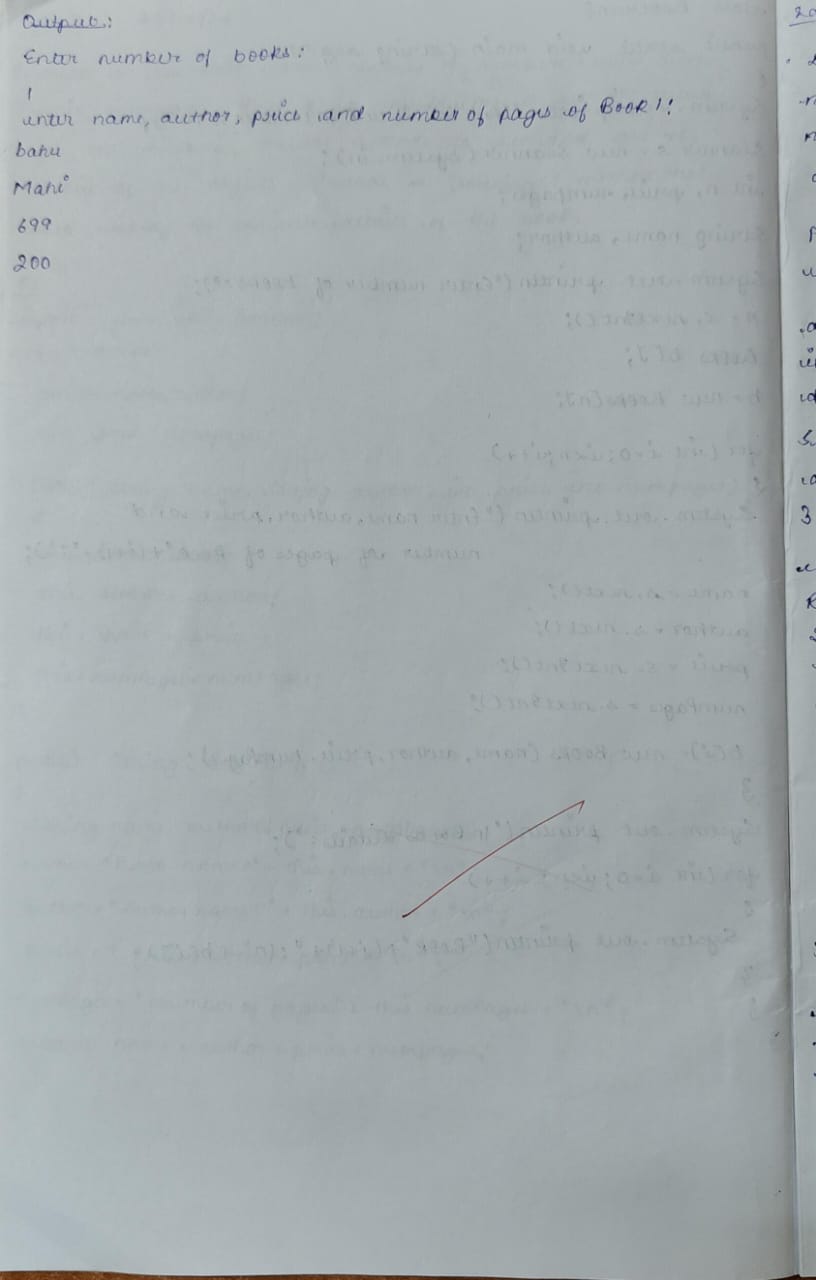
# Program 3

## Demonstration of Array of objects of each book type

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:





Code:

import java.util.Scanner;

class Books {

String name, 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;

}

}

class Bookrun {

public static void main(String arg[]) {

System.out.println("Anu Sai Shree R\n1BM23CS045");

Scanner s = new Scanner(System.in);

int n, price, numPages;

String name, author;

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

n = s.nextInt();

Books b[];

b = new Books[n];

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

System.out.println("enter name,author,price and number of pages of Book" + (i + 1) + ":");

name = s.next();

author = s.next();

price = s.nextInt();

numPages = s.nextInt();

b[i] = new Books(name, author, price, numPages);

}

System.out.println("\nBooks Details:");

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

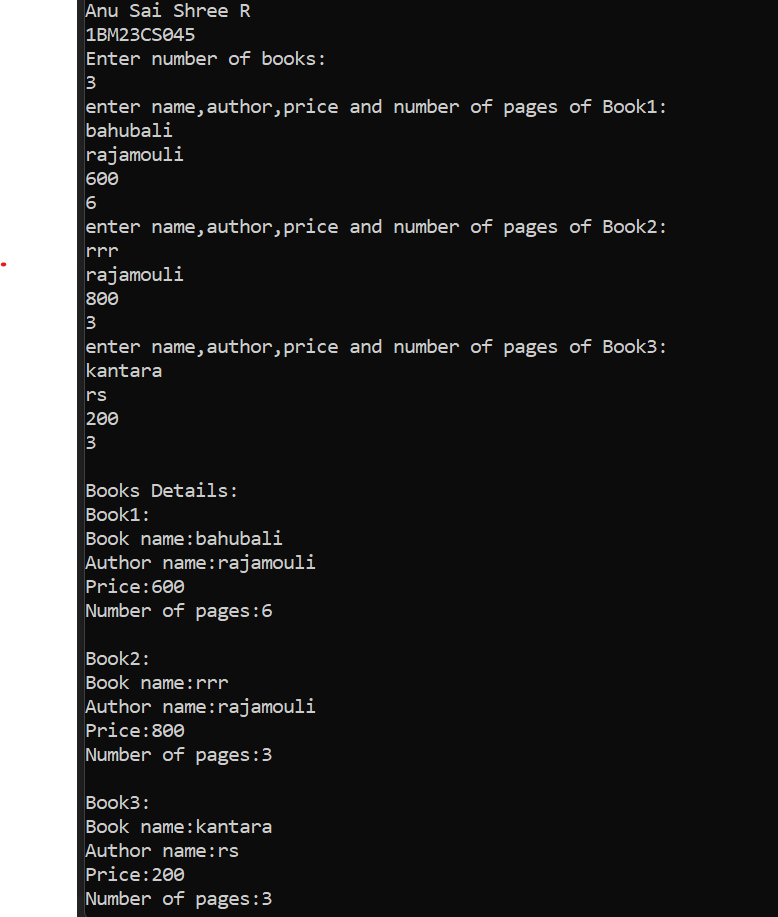
System.out.println("Book" + (i + 1) + ":\n" + b[i]);

}

}

}

Output:

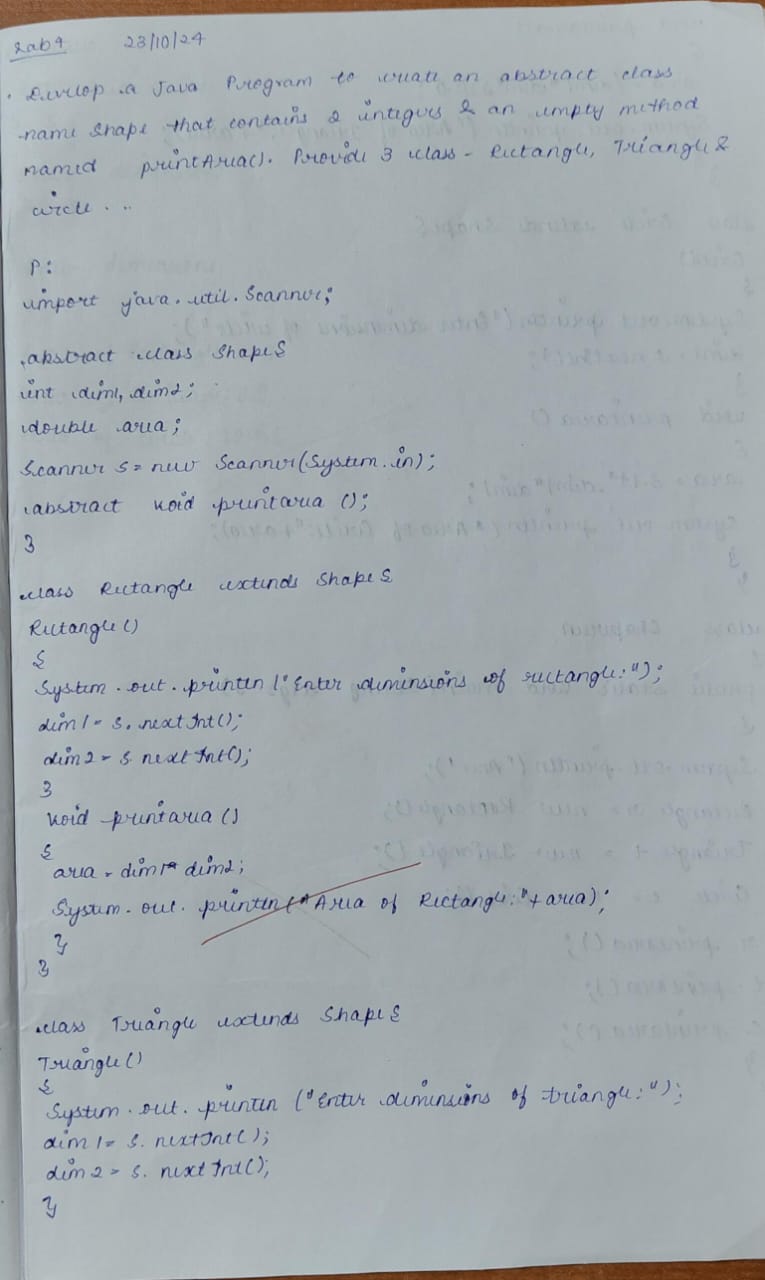


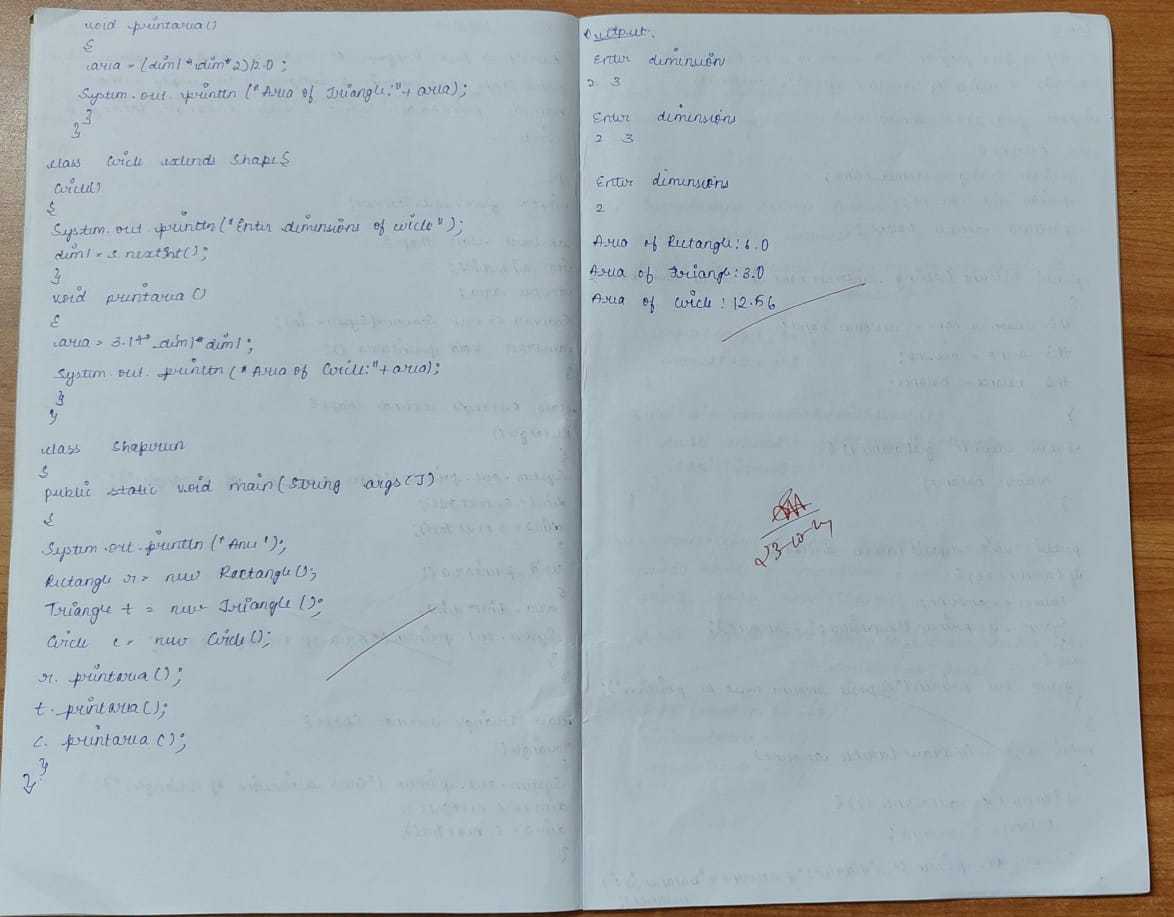
# Program 4

## Demonstration Of Abstract Class

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

Algorithm:



Code:

import java.util.Scanner;

abstract class Shape {

int dim1, dim2;

double area;

Scanner s = new Scanner(System.in);

abstract void printarea();

}

class Rectangle extends Shape {

Rectangle() {

System.out.println("Enter dimensions of rectangle:");

dim1 = s.nextInt();

dim2 = s.nextInt();

}

void printarea() {

area = dim1 \* dim2;

System.out.println("Area of Rectangle:" + area);

}

}

class Triangle extends Shape {

Triangle() {

System.out.println("Enter dimensions of triangle:");

dim1 = s.nextInt();

dim2 = s.nextInt();

}

void printarea() {

area = (dim1 \* dim2) / 2.0;

System.out.println("Area of Triangle:" + area);

}

}

class Circle extends Shape {

Circle() {

System.out.println("Enter dimensions of circle:");

dim1 = s.nextInt();

}

void printarea() {

area = 3.14 \* dim1 \* dim1;

System.out.println("Area of Circle:" + area);

}

}

class Shaperun {

public static void main(String args[]) {

System.out.println("Anu Sai Shree R\n1BM23CS045");

Rectangle r = new Rectangle();

Triangle t = new Triangle();

Circle c = new Circle();

r.printarea();

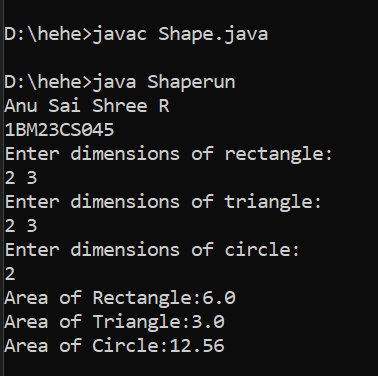
t.printarea();

c.printarea();

}

}

Output:

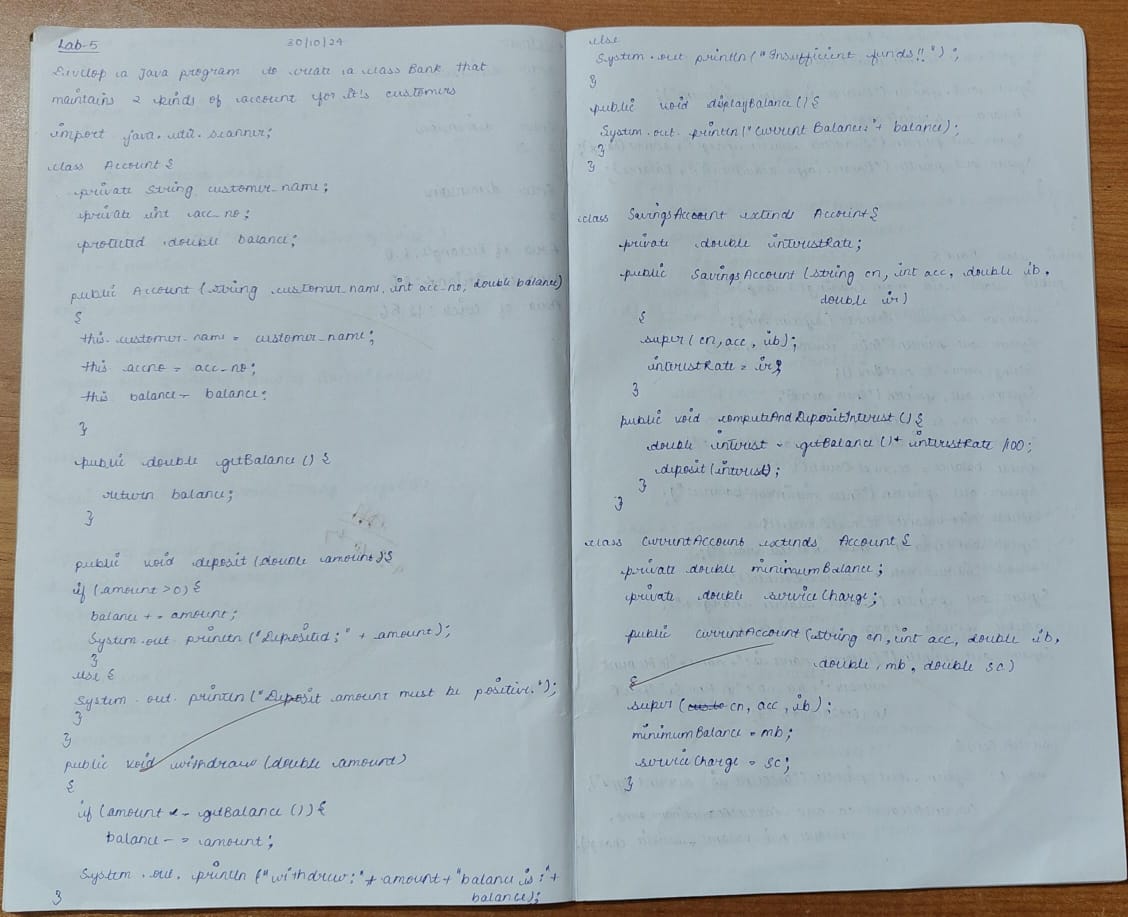


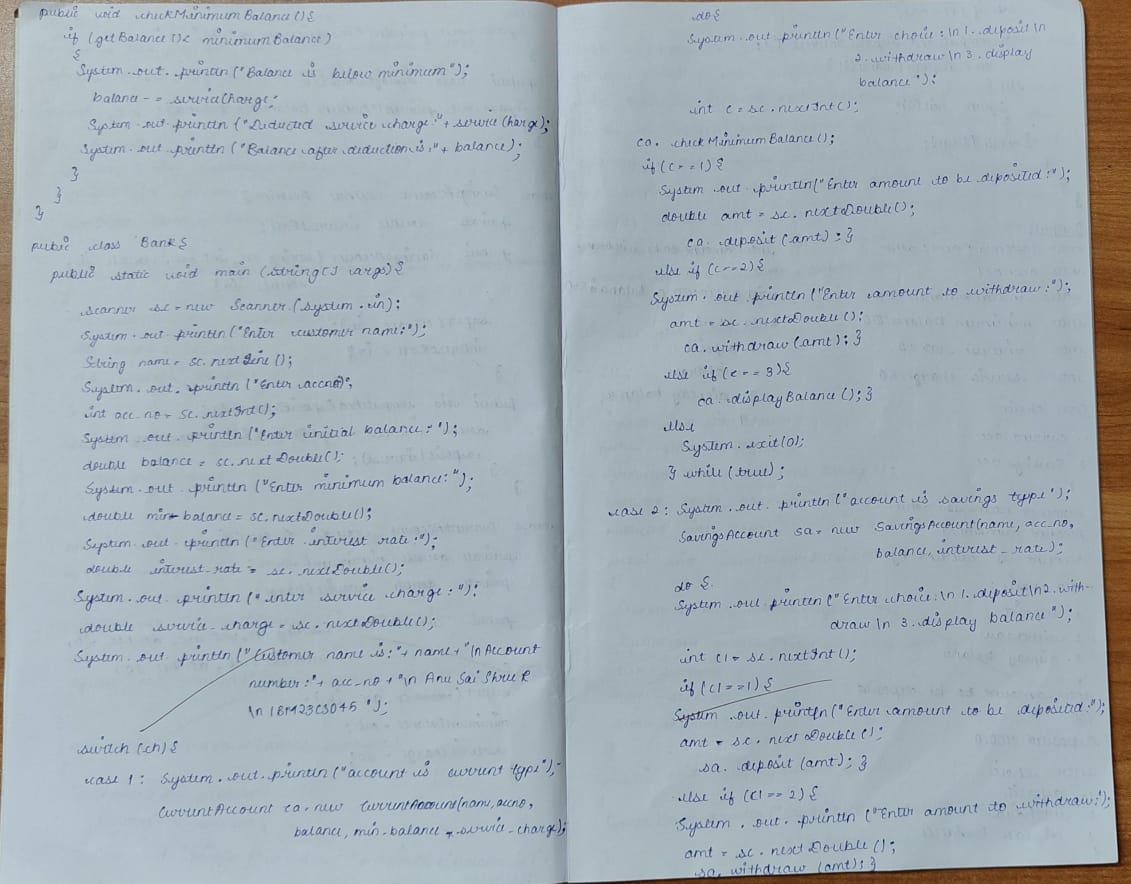
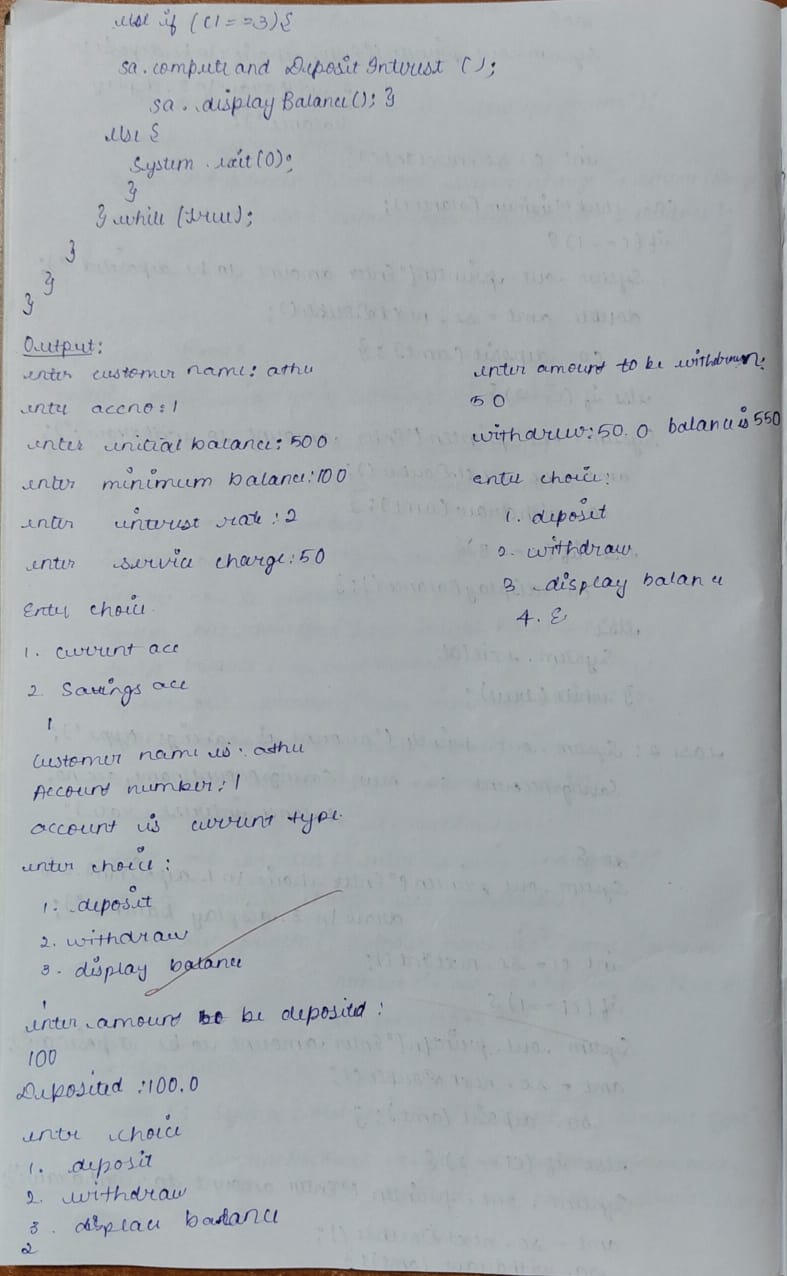
# Program 5

## Implementation of Bank

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:



Code:

import java.util.Scanner;

class Account {

private String customer\_name;

private int acc\_no;

protected double balance;

public Account(String customer\_name, int acc\_no, double balance) {

this.customer\_name = customer\_name;

this.acc\_no = acc\_no;

this.balance = balance;

}

public double getBalance() {

return balance;

}

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

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

} else {

System.out.println("Deposit amount must be positive.");

}

}

public void withdraw(double amount)

{

if(amount<=getBalance()){

balance-=amount;

System.out.println("withdrew:"+amount + " balance is:"+ balance);

}

else

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

}

public void displayBalance(){

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

}

}

class SavingsAccount extends Account {

private double interestRate;

public SavingsAccount(String customerName, int accountNumber, double initialBalance, double interestRate) {

super(customerName, accountNumber, initialBalance);

this.interestRate = interestRate;

}

public void computeAndDepositInterest() {

double interest = getBalance() \* interestRate / 100;

deposit(interest);

}

}

class CurrentAccount extends Account {

private double minimumBalance;

private double serviceCharge;

public CurrentAccount(String customerName, int accountNumber, double initialBalance, double minimumBalance, double serviceCharge) {

super(customerName, accountNumber, initialBalance);

this.minimumBalance = minimumBalance;

this.serviceCharge = serviceCharge;

}

public void checkMinimumBalance() {

if (getBalance() < minimumBalance) {

System.out.println("Balance is below minimum");

balance-=serviceCharge;

System.out.println("Deducted service charge:" +serviceCharge);

System.out.println("Balance after deduction is:"+balance);

}

}

}

public class Bank {

public static void main(String[] args) {

System.out.println("Anu Sai Shree R\n1BM23CS045");

Scanner sc = new Scanner(System.in);

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

String name=sc.nextLine();

System.out.println("enter accno:");

int acc\_no=sc.nextInt();

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

double balance=sc.nextDouble();

System.out.println("enter minimum balance:");

double minimum\_balance=sc.nextDouble();

System.out.println("enter interest rate:");

double interest\_rate=sc.nextDouble();

System.out.println("enter service charge:");

double service\_charge=sc.nextDouble();

System.out.println("Enter choice:\n 1.Current acc\n 2.Savings acc");

int ch=sc.nextInt();

System.out.println("Customer name is:"+ name+"\nAccount number:"+acc\_no);

switch(ch){

case(1):

System.out.println("account is current type");

CurrentAccount ca = new CurrentAccount(name,acc\_no,balance,minimum\_balance,service\_charge);

do{ System.out.println("enter choice:\n 1.deposit\n 2.withdraw\n 3.display balance");

int c=sc.nextInt();

ca.checkMinimumBalance();

if(c==1){

System.out.println("enter amount to be deposited:");

double amt=sc.nextDouble();

ca.deposit(amt);}

else if(c==2){

System.out.println("enter amount to withdraw:");

double amt=sc.nextDouble();

ca.withdraw(amt);}

else if(c==3){

ca.displayBalance();}

else

System.exit(0);

}while(true);

case(2):

System.out.println("account is savings type");

SavingsAccount sa=new SavingsAccount(name,acc\_no,balance,interest\_rate);

do{ System.out.println("enter choice:\n 1.deposit\n 2.withdraw\n 3.display balance");

int c1=sc.nextInt();

if(c1==1){

System.out.println("enter amount to be deposited:");

double amt=sc.nextDouble();

sa.deposit(amt);}

else if(c1==2){

System.out.println("enter amount to withdraw:");

double amt=sc.nextDouble();

sa.withdraw(amt);}

else if(c1==3){

sa.computeAndDepositInterest();

sa.displayBalance();}

else{

System.exit(0);

}

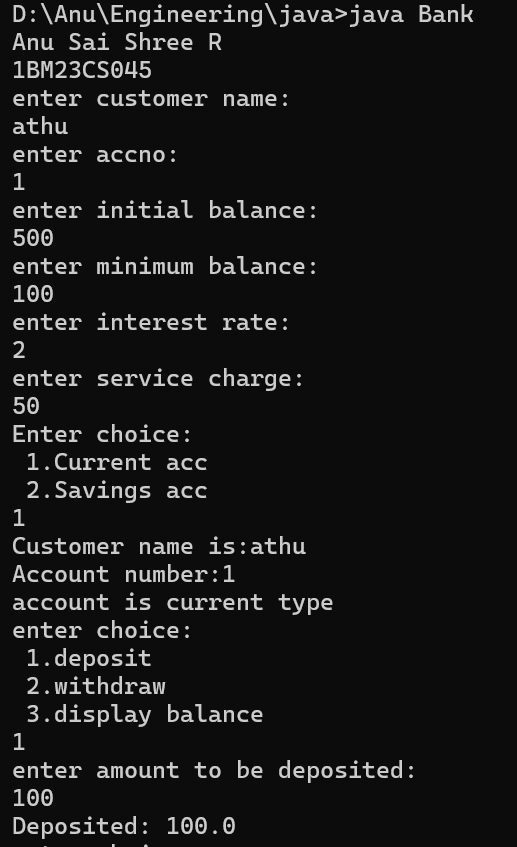
}while(true);

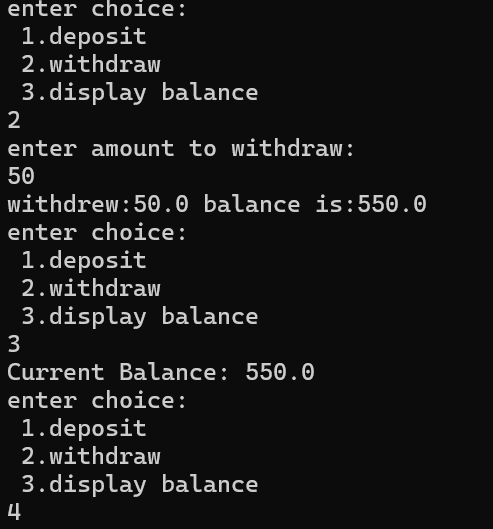
}

}

}

Output:



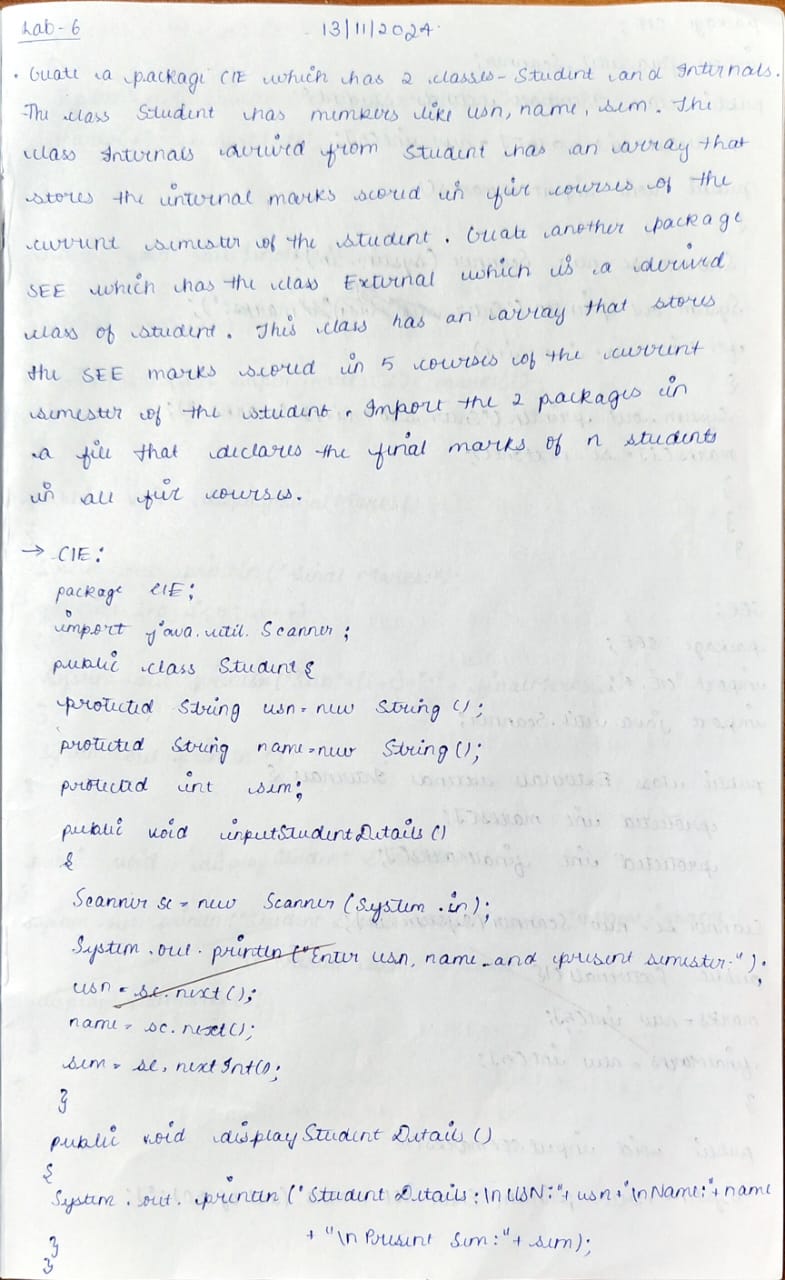


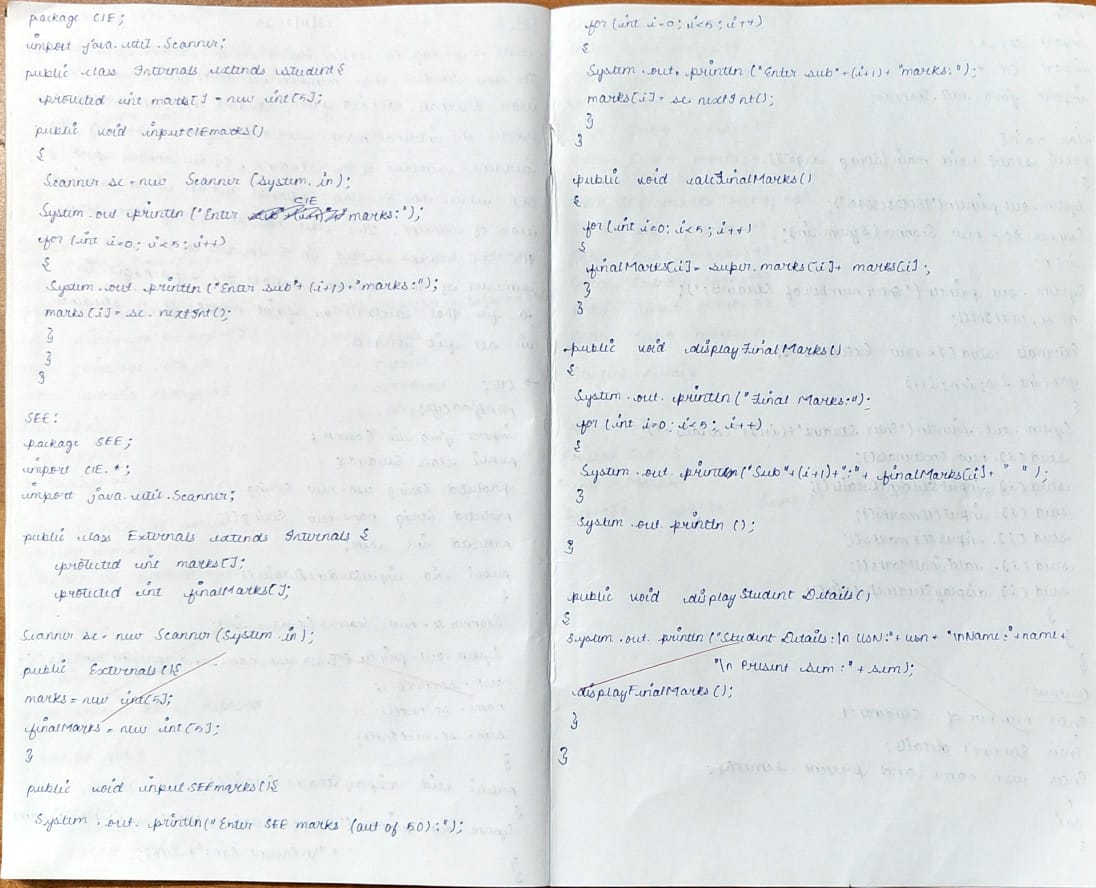
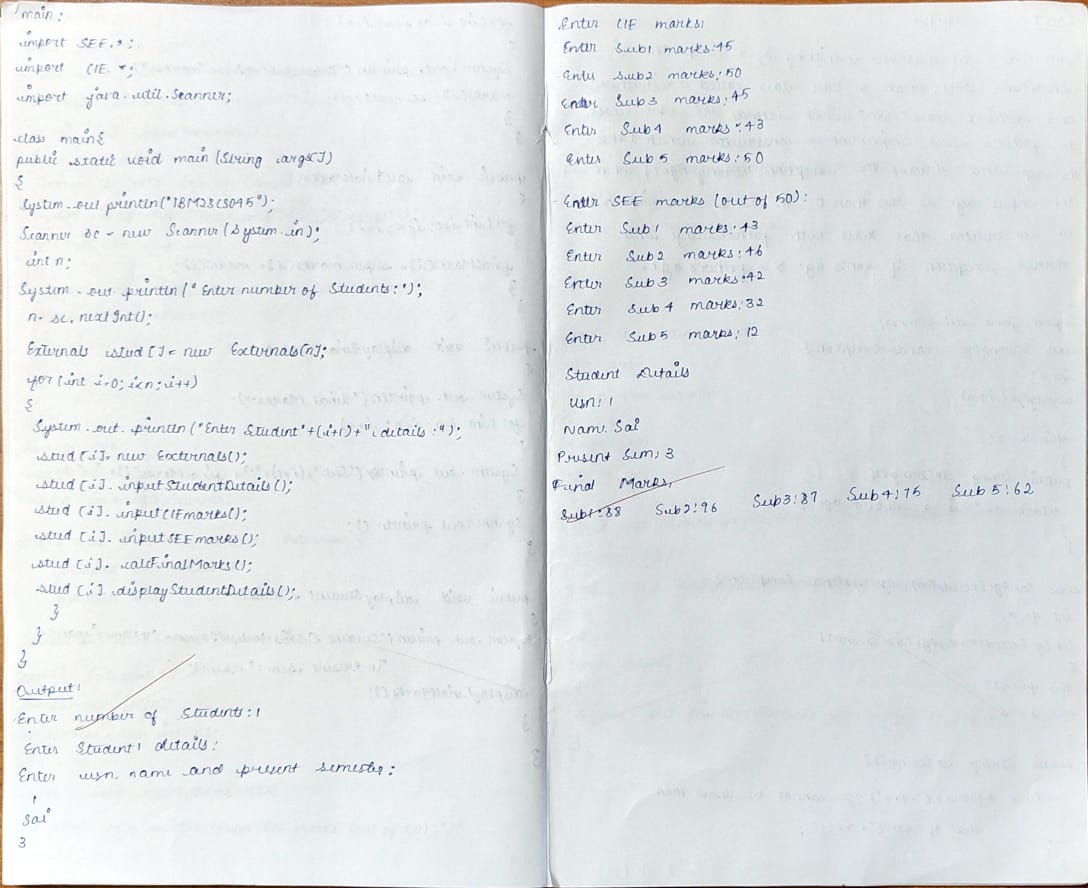
# Program 6

## Demonstration Of Package

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:



Code:

* CIE
* Student.java

package CIE;

import java.util.Scanner;

public class Student {

protected String usn = new String();

protected String name = new String();

protected int sem;

public void inputStudentDetails() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter usn,name and present semester :");

usn = sc.next();

name = sc.next();

sem = sc.nextInt();

}

public void displayStudentDetails() {

System.out.println("Student details:\nUsn:" + usn + "\nName:" + name + "\nPresent Sem:" + 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 CIE marks:");

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

System.out.println("Enter sub" + (i + 1) + "marks:");

marks[i] = sc.nextInt();

}

}

}

* SEE
* Externals.java

package SEE;

import CIE.\*;

import java.util.Scanner;

public class Externals extends Internals {

protected int marks[];

protected int finalMarks[];

Scanner sc = new Scanner(System.in);

public Externals() {

marks = new int[5];

finalMarks = new int[5];

}

public void inputSEEmarks() {

System.out.println("Enter SEE marks(out of 50):");

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

System.out.println("Enter sub" + (i + 1) + "marks:");

marks[i] = sc.nextInt();

}

}

public void calcFinalMarks() {

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

finalMarks[i] = super.marks[i] + marks[i];

}

}

public void displayFinalMarks() {

System.out.println("Final Marks:");

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

System.out.print("Sub" + (i + 1) + ":" + finalMarks[i] + " ");

}

System.out.println();

}

public void displayStudentDetails() {

System.out.println("Student details:\nUsn:" + usn + "\nName:" + name + "\nPresent Sem:" + sem);

displayFinalMarks();

}

}

* main.java

import SEE.\*;

import CIE.\*;

import java.util.Scanner;

class Main {

public static void main(String args[]) {

System.out.println("1BM23CS045\nAnu Sai Shree R");

Scanner sc = new Scanner(System.in);

int n;

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

n = sc.nextInt();

Externals stud[] = new Externals[n];

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

System.out.println("Enter Student" + (i + 1) + " details:");

stud[i] = new Externals();

stud[i].inputStudentDetails();

stud[i].inputCIEmarks();

stud[i].inputSEEmarks();

stud[i].calcFinalMarks();

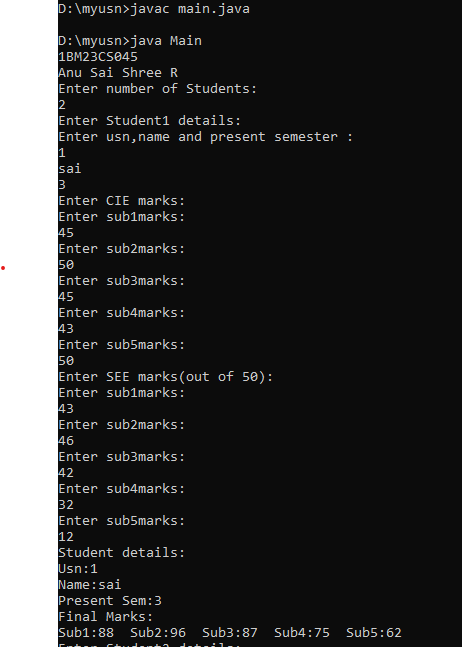
stud[i].displayStudentDetails();

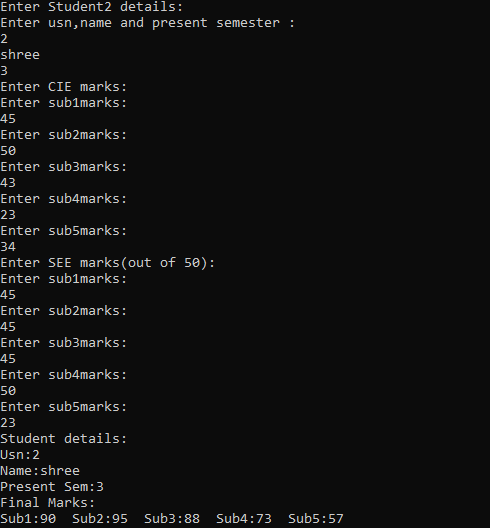
}

}

}

Output:



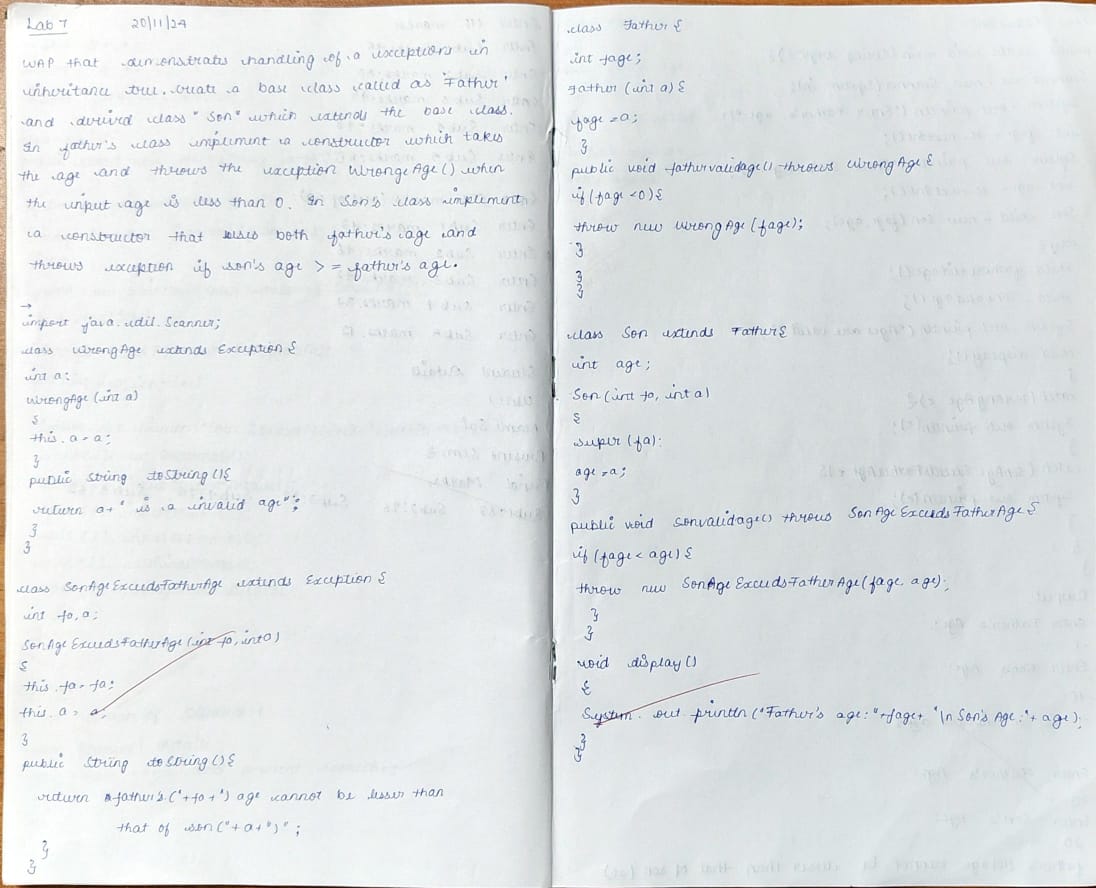


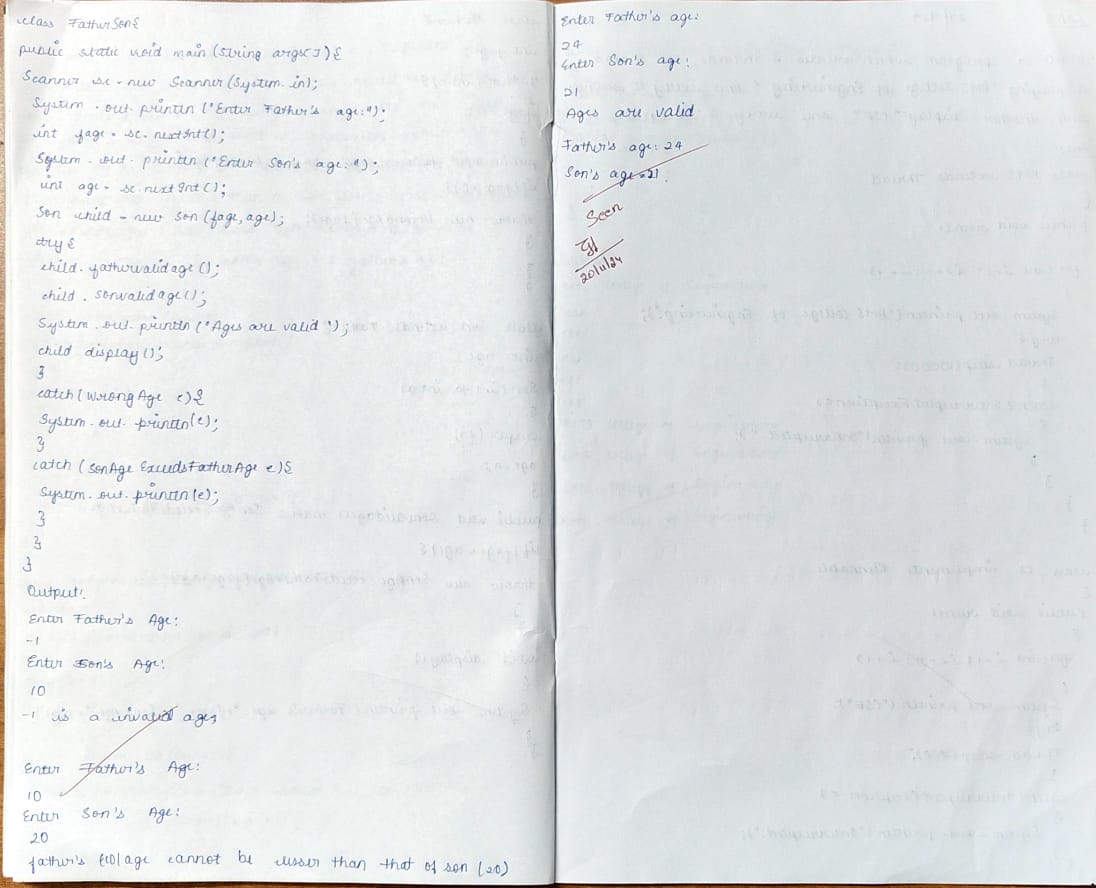
# Program 7

## Demonstration Of Exception Handling

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=father’s age.

Algorithm:



Code:

import java.util.Scanner;

class WrongAge extends Exception {

int a;

WrongAge(int a) {

this.a = a;

}

public String toString() {

return a + " is a invalid Age";

}

}

class SonAgeExceedsFatherAge extends Exception {

int fa, a;

SonAgeExceedsFatherAge(int fa, int a) {

this.fa = fa;

this.a = a;

}

public String toString() {

return "father's(" + fa + ") age cannot be lesser than that of son(" + a + ")";

}

}

class Father {

int fage;

Father(int a) {

fage = a;

}

public void fathervalidage() throws WrongAge {

if (fage < 0) {

throw new WrongAge(fage);

}

}

}

class Son extends Father {

int age;

Son(int fa, int a) {

super(fa);

age = a;

}

public void sonvalidage() throws SonAgeExceedsFatherAge {

if (fage < age) {

throw new SonAgeExceedsFatherAge(fage, age);

}

}

void display() {

System.out.println("Father's age:" + fage + "\nSon's age:" + age);

}

}

class FatherSon {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

System.out.println("Anu Sai Shree R\n1BM23CS045");

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

int fage = sc.nextInt();

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

int age = sc.nextInt();

Son child = new Son(fage, age);

try {

child.fathervalidage();

child.sonvalidage();

System.out.println("Ages are valid");

child.display();

} catch (WrongAge e) {

System.out.println(e);

} catch (SonAgeExceedsFatherAge e) {

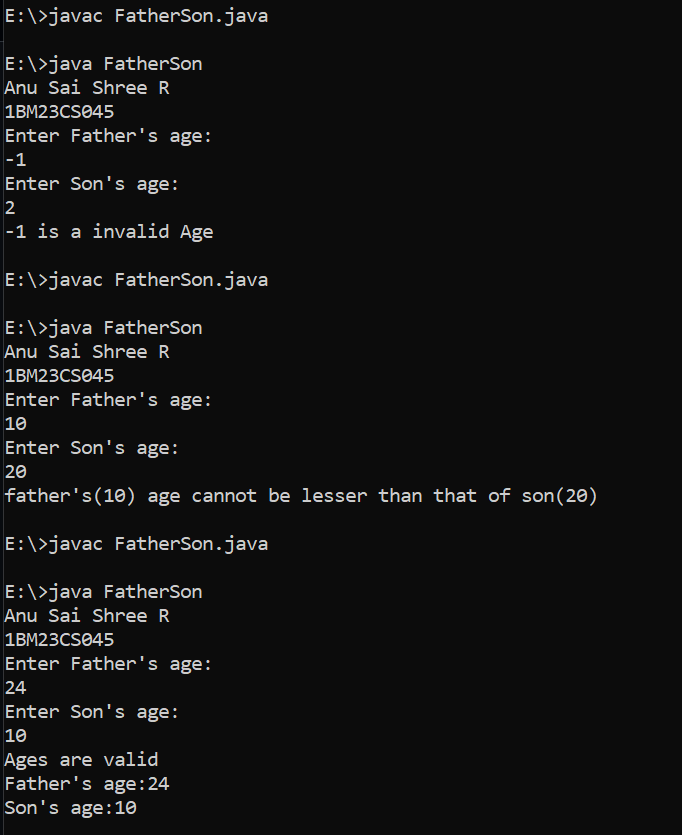
System.out.println(e);

}

}

}

Output:

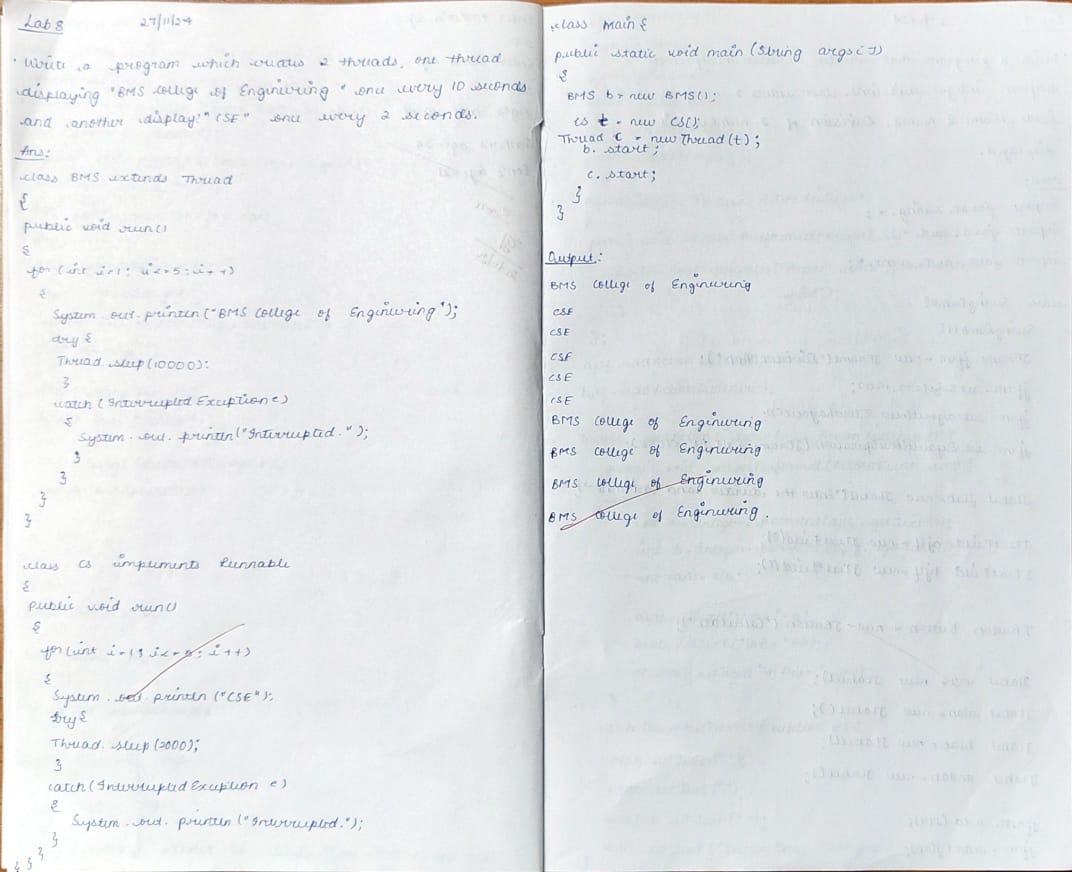


# Program 8

## Demonstration Of Threads

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:

Code:

class BMS extends Thread {

public void run() {

for (int i = 1; i <= 5; i++) {

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

try {

Thread.sleep(10000);

} catch (InterruptedException e) {

System.out.println("interrupted.");

}

}

}

}

class CS extends Thread {

public void run() {

for (int i = 1; i <= 5; i++) {

System.out.println("CSE");

try {

Thread.sleep(2000);

} catch (InterruptedException e) {

System.out.println("interrupted.");

}

}

}

}

class threadExtends {

public static void main(String args[]) {

System.out.println("Anu Sai Shree R\n1BM23CS045");

BMS b = new BMS();

CS c = new CS();

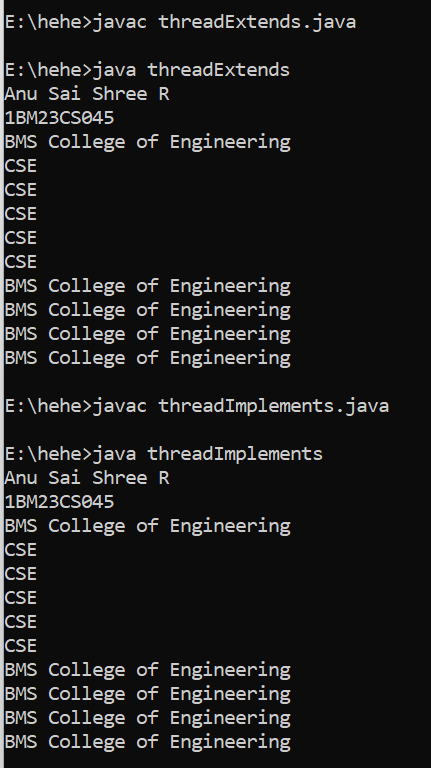
b.start();

c.start();

}

}

Output:

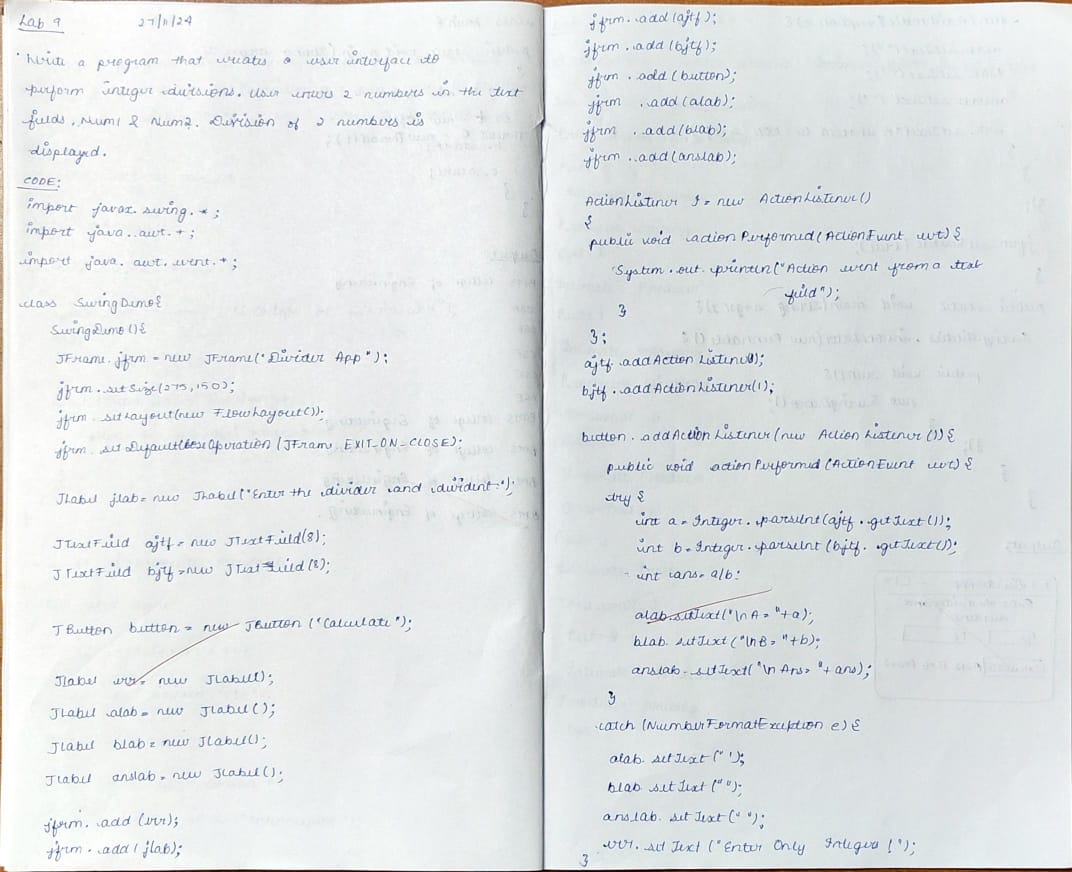
****

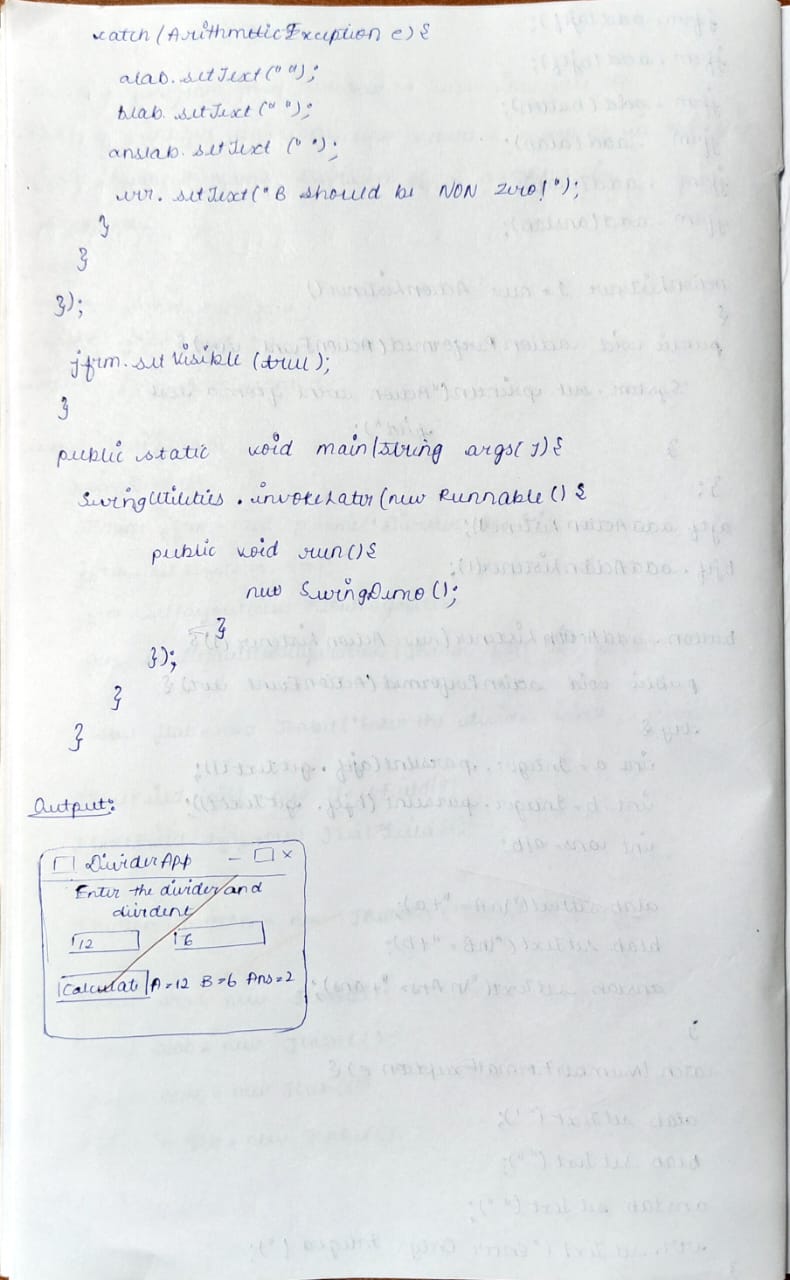
# Program 9

## User Interface for division of 2 Numbers(Open Ended Exercise)

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:





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();

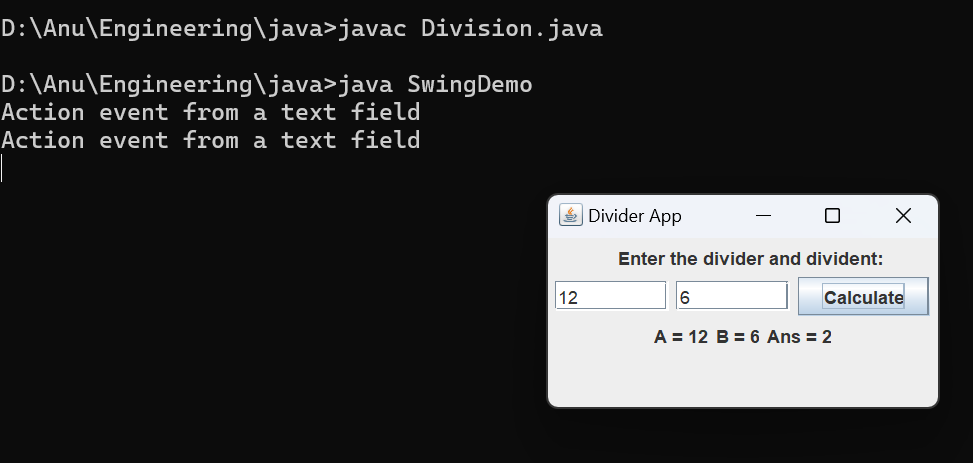
}

});

}

}

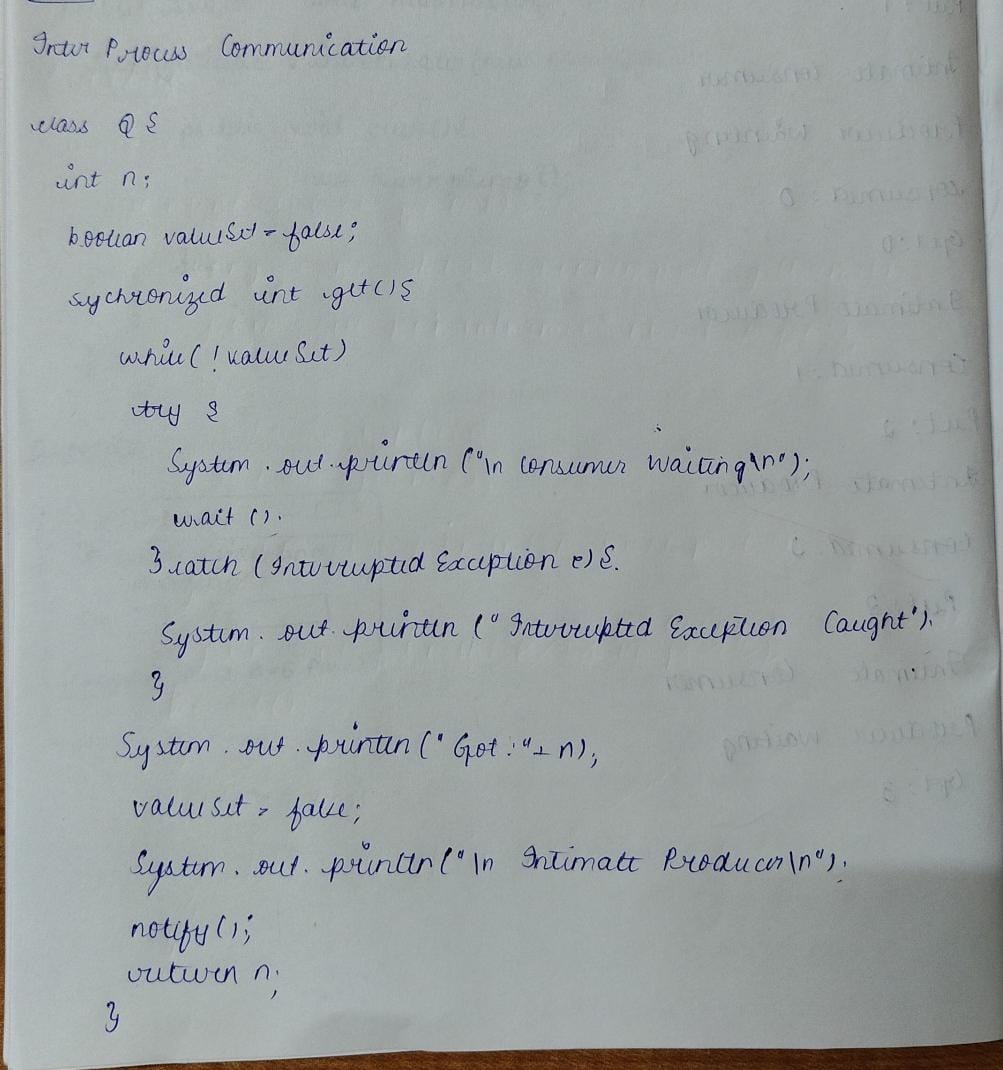
Output:

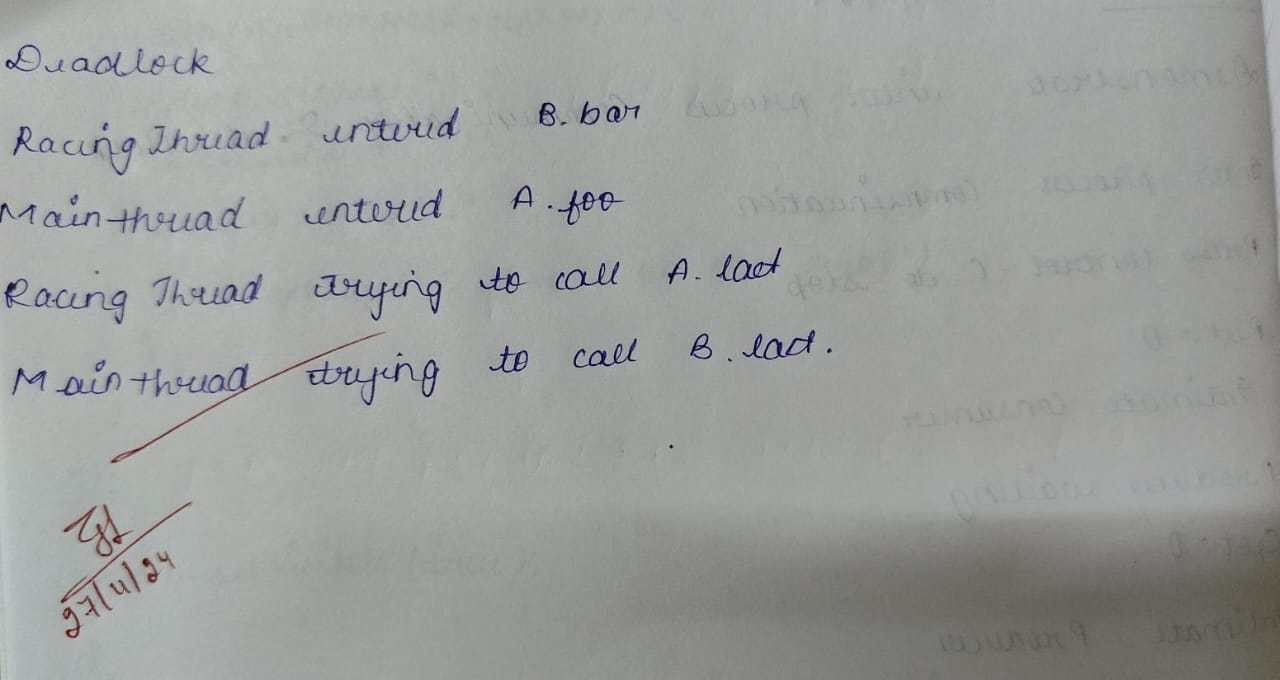
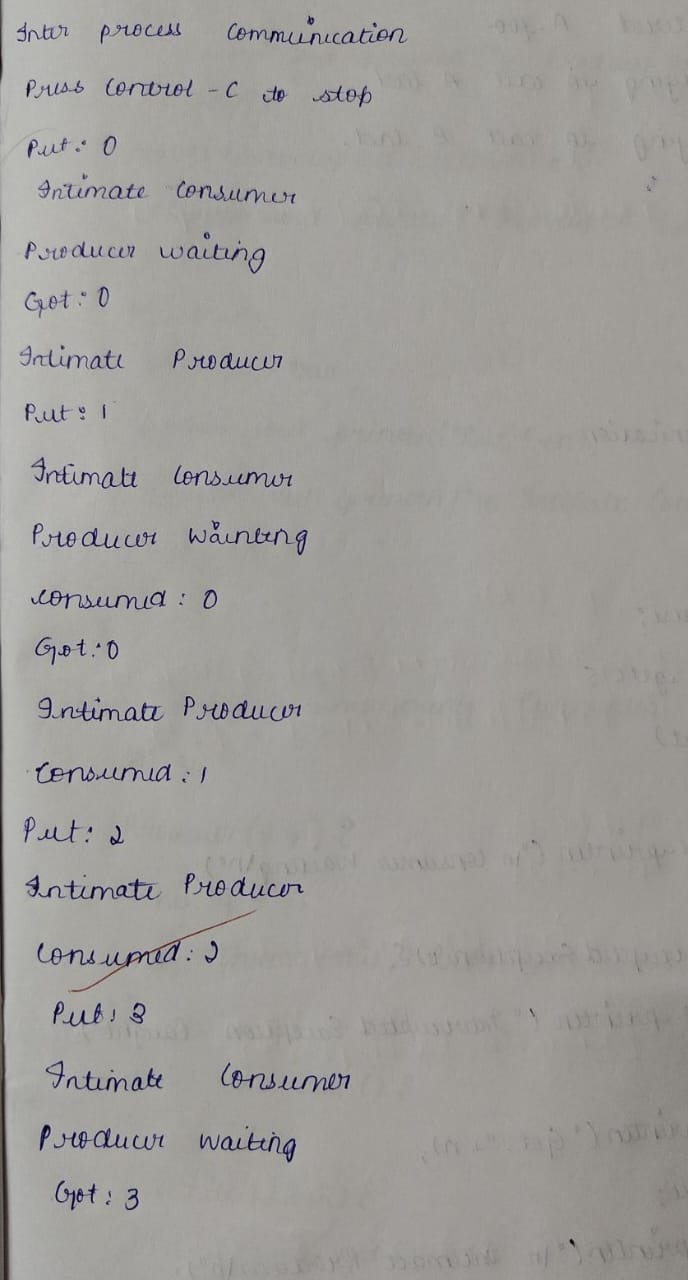
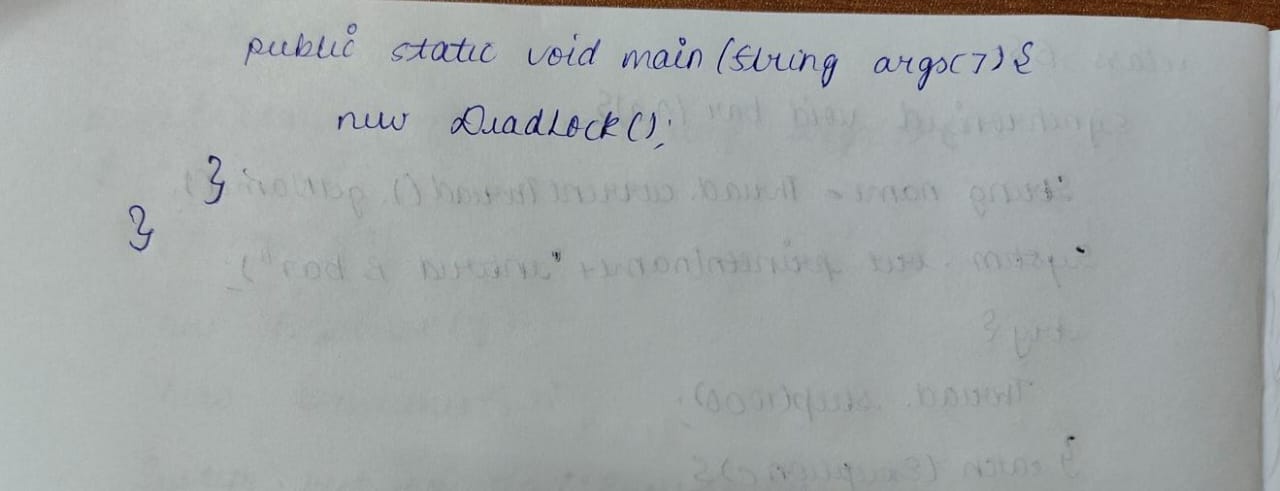
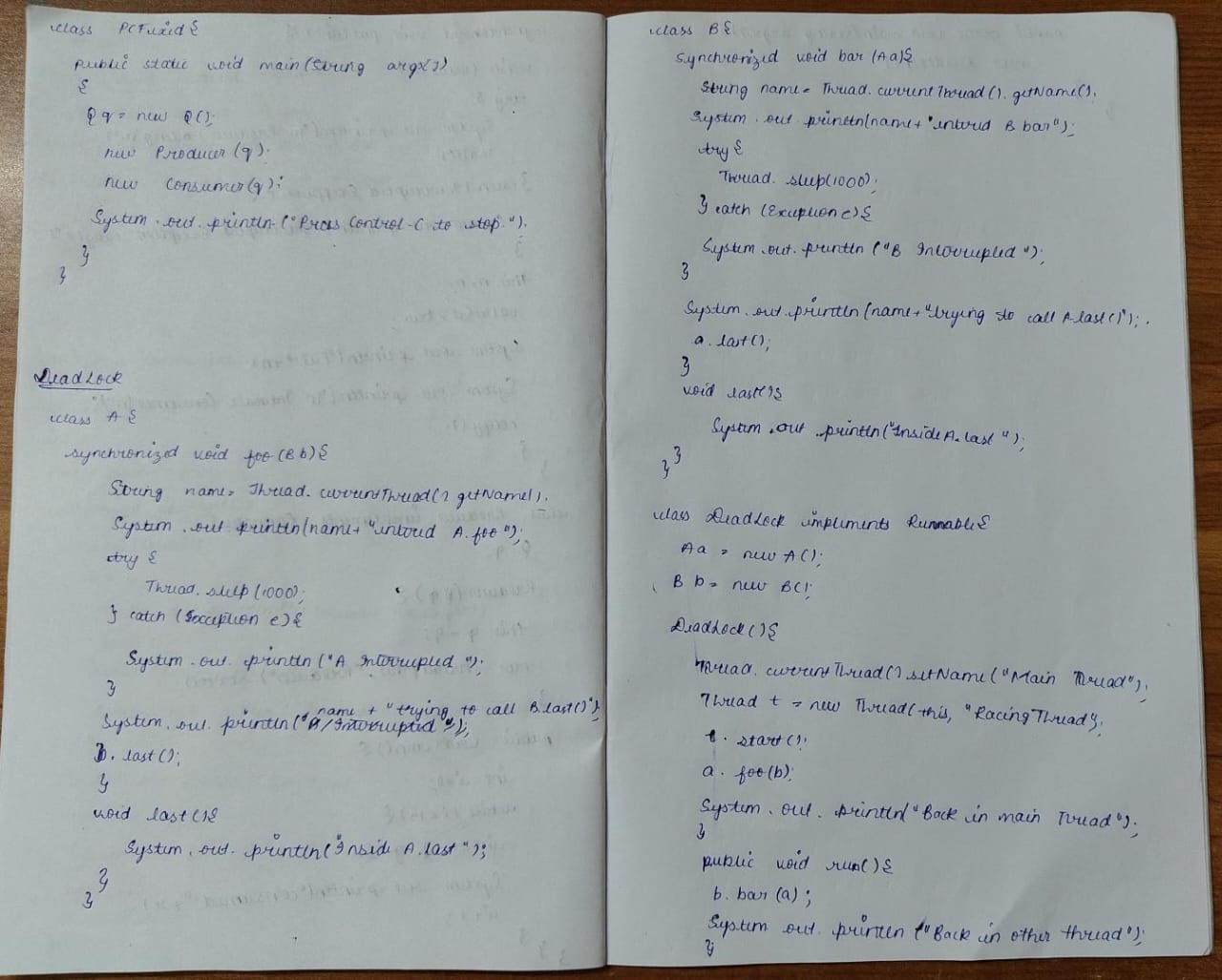
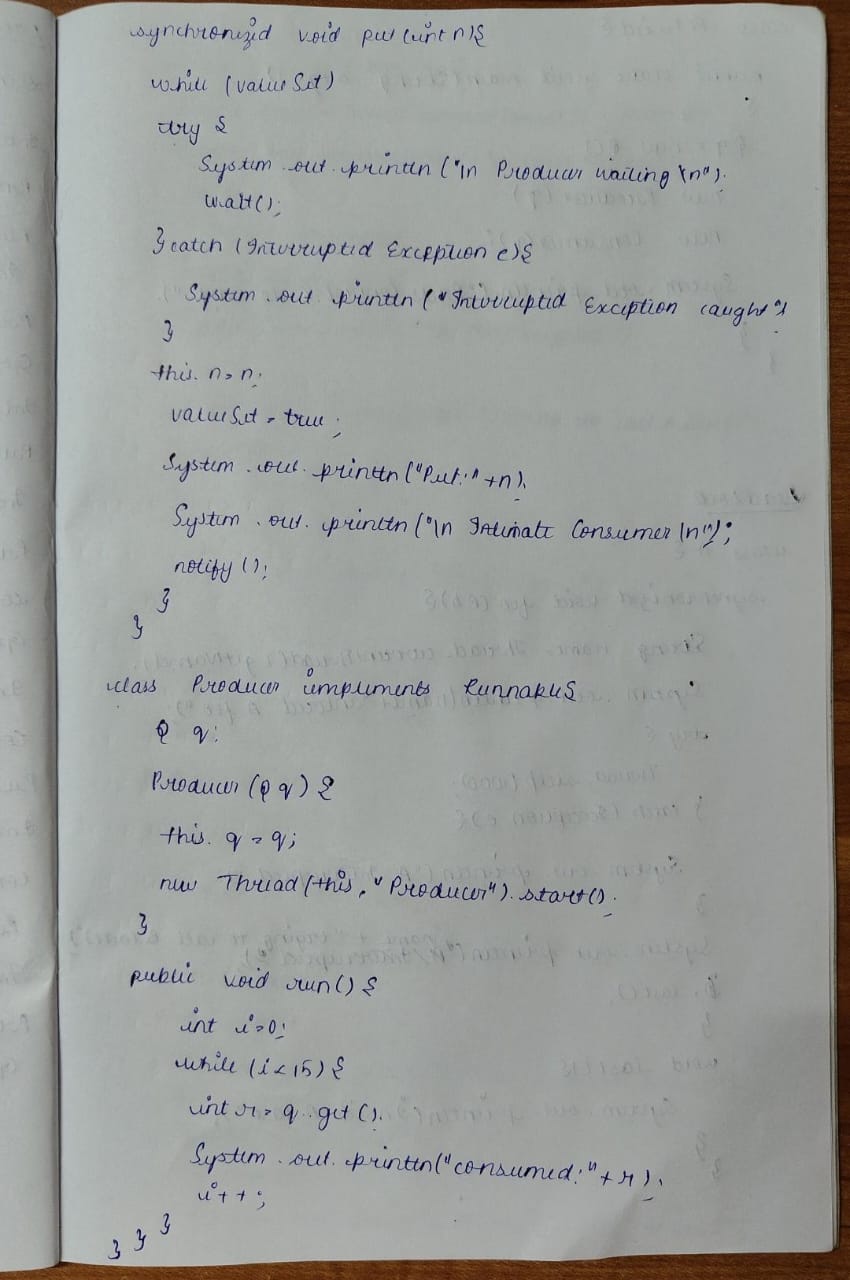


# Program 10

## Demonstrate Inter process Communication and deadlock(Open Ended Exercise)

Algorithm:



Code:

Inter process Communication

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);

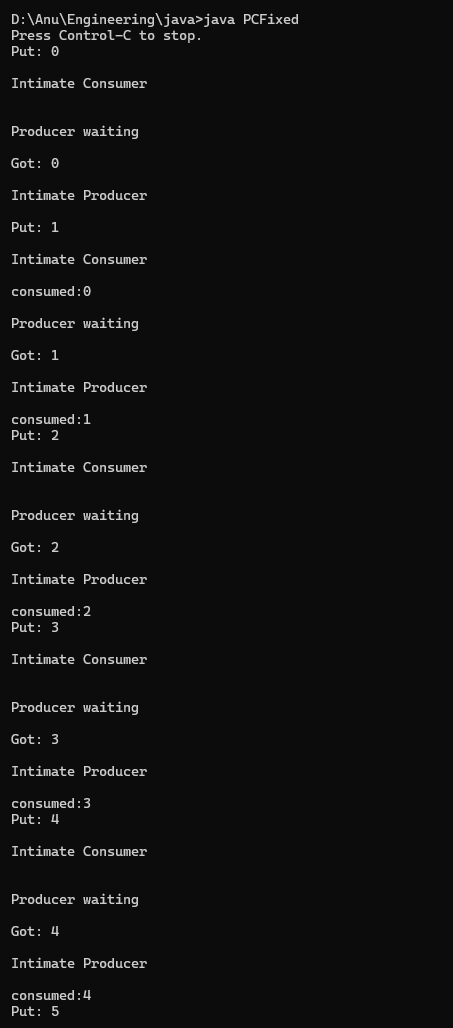
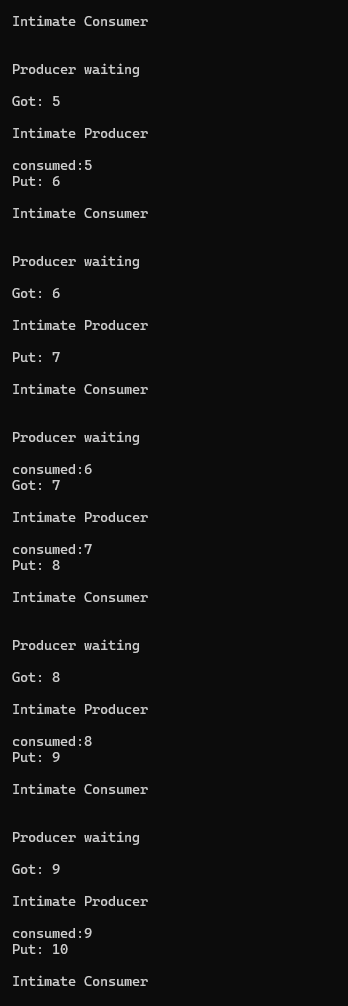
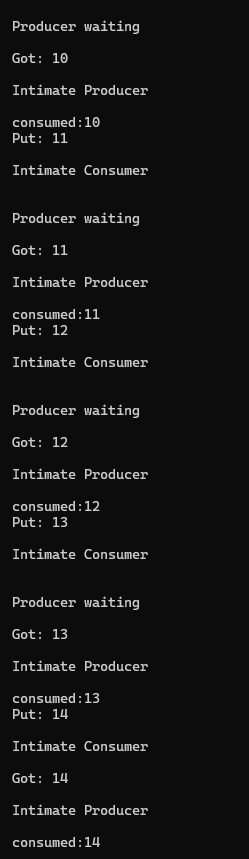
new Consumer(q);

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

}

}

Output:

-Deadlock

class A {

synchronized void foo(B b) {

String name = Thread.currentThread().getName();

System.out.println(name + " entered

A.foo ");

try {

Thread.sleep(1000);

} catch (Exception e) {

System.out.println("A Interrupted");

}

System.out.println(name + " trying to

call B.last()

");

b.last();

}

void last() {

System.out.println("Inside A.last");

}

}

class B {

synchronized void bar(A a) {

String name =

Thread.currentThread().getName();

System.out.println(name + " entered

B.bar ");

try {

Thread.sleep(1000);

} catch (Exception e) {

System.out.println("B Interrupted");

}

System.out.println(name + " trying to call A.last()"); a.last();

}

void last() {

System.out.println("Inside A.last");

}

}

class Deadlock implements Runnable {

A a = new A();

B b = new B();

Deadlock() {

Thread.currentThread().setName("MainThread");

Thread t = new Thread(this, "RacingThread");

t.start();

a.foo(b); // get lock on a in this

thread.

System.out.println("Back in main

thread ");

}

public void run() {

b.bar(a); // get lock on b in other

thread.

System.out.println("Back in other

thread ");

}

public static void main(String args[]) {

new Deadlock();

}

}

Output:

