VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Kashvi Agarwal(1BM23CS142)

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 Kashvi Agarwal**(1BM23CS142),** who is a 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

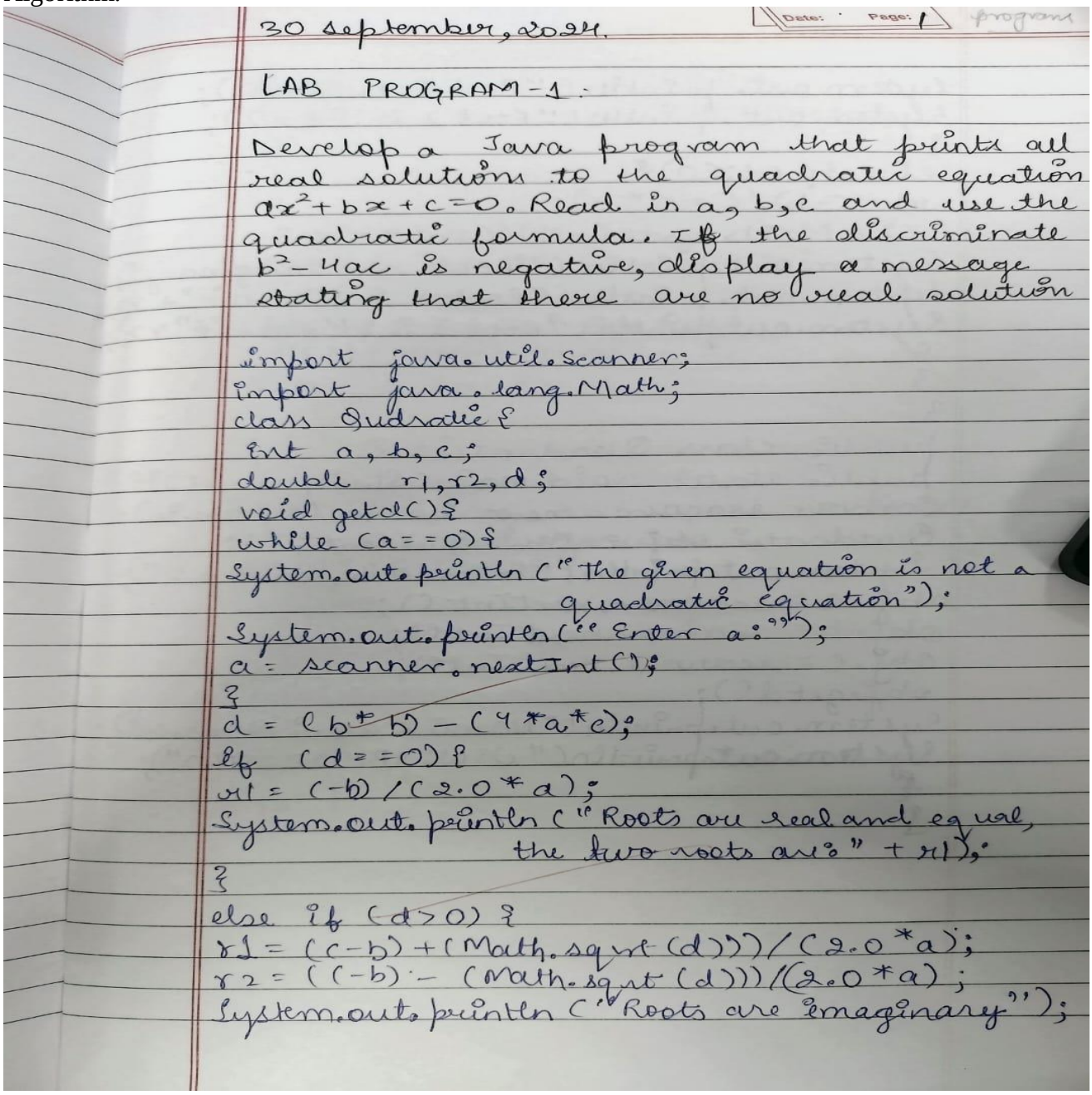
Index

Github Link:
https://github.com/KashviAgarwalcs23/Java-Lab-Programs.git

Program 1

Develop a Java 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:



30 September, 2024.

LAB PROGRAM-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 discriminate b^2-4ac is negative, display a message stating that there are no real solution

```
import java.util.Scanner;
import java.lang.Math;
class Quadratic {
 Int a, b, c;
 double r1, r2, d;
 void getd() {
 while (a==0) {
System.out.println("The given equation is not a
                    quadratic equation");
System.out.println("Enter a:");
a = scanner.nextInt();
}
d = (b*b) - (4*a*c);
If (d==0) {
r1 = (-b) / (2.0*a);
System.out.println("Roots are real and equal,
                    the two roots are:" + r1);
}
else if (d>0) {
r1 = (-b) + (Math.sqrt(d)) / (2.0*a);
r2 = ((-b) - (Math.sqrt(d))) / (2.0*a);
System.out.println("Roots are imaginary");
```

```
System.out.println("Root 1 is :"+r1);
System.out.println("Root 2 is :"+r2);
}
else if(d<0){
r1 = (-b)/(2.0*a);
r2 = Math.sqrt(-d)/(2.0*a);
System.out.println("Roots are imaginary");
System.out.println("Root 1 is :"+r1+" + "+r2);
System.out.println("Root 2 is :"+r1+" - "+r2);
}
}
}
public class Quadratic1 {
public static void main(String args[]){
Scanner scanner = new Scanner(System.in);
Quadratic obj = new Quadratic();
System.out.println("Enter a,b, and c :");
obj.a = scanner.nextInt();
obj.b = scanner.nextInt();
obj.c = scanner.nextInt();
obj.getd();
System.out.println("Name: K ashvi Agarwal");
System.out.println("USN: 1BM23CS142");
}
}
```

Output:

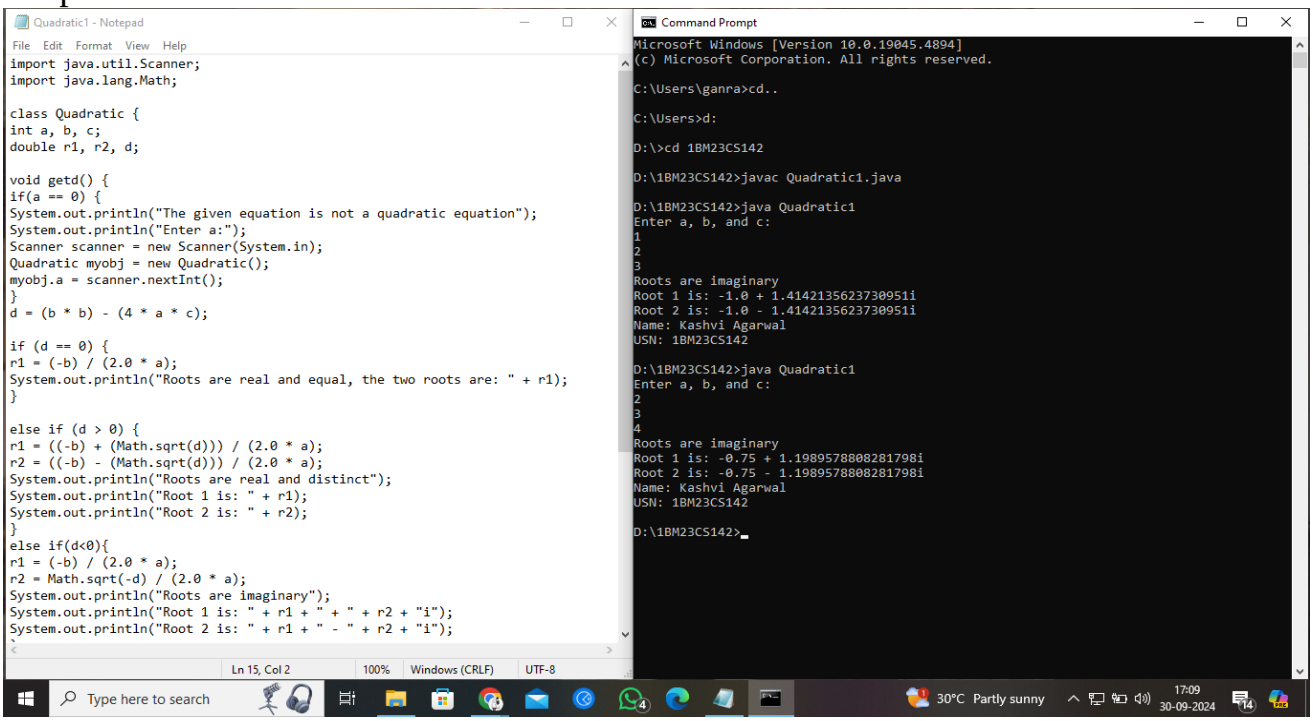
Enter a, b and c
2
1
2
Roots are imaginary
Root 1 is: -0.25 + 0.9682 i
Root 2 is: -0.25 - 0.9682 i
Name: Kashvi Agarwal
USN: 1BM23CS142
Enter a, b and c
1
3
2
Roots are real and equal. Roots are -1,
Name: Kashvi Agarwal
USN: 1BM23CS142

Code:

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

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

void getd() {
if(a == 0) {
System.out.println("The given equation is not a quadratic equation");
System.out.println("Enter a:");
Scanner scanner = new Scanner(System.in);
Quadratic myobj = new Quadratic();
myobj.a = scanner.nextInt();
}
d = (b * b) - (4 * a * c);

if (d == 0) {
r1 = (-b) / (2.0 * a);
System.out.println("Roots are real and equal, the two roots are: " + r1);
}

else if (d > 0) {
r1 = ((-b) + (Math.sqrt(d))) / (2.0 * a);
r2 = ((-b) - (Math.sqrt(d))) / (2.0 * a);
System.out.println("Roots are real and distinct");
System.out.println("Root 1 is: " + r1);
System.out.println("Root 2 is: " + r2);
}
else if(d<0){
r1 = (-b) / (2.0 * a);
r2 = Math.sqrt(-d) / (2.0 * a);
System.out.println("Roots are imaginary");
System.out.println("Root 1 is: " + r1 + " + " + r2 + "i");
System.out.println("Root 2 is: " + r1 + " - " + r2 + "i");
}
}
}
public class Quadratic1 {
public static void main(String args[]) {
Scanner scanner = new Scanner(System.in);
Quadratic myobj = new Quadratic();
System.out.println("Enter a, b, and c: ");
myobj.a = scanner.nextInt();
myobj.b = scanner.nextInt();
```

```
myobj.c = scanner.nextInt();
myobj.getd();
System.out.println("Name: Kashvi Agarwal");
System.out.println("USN: 1BM23CS142");
}
}
```

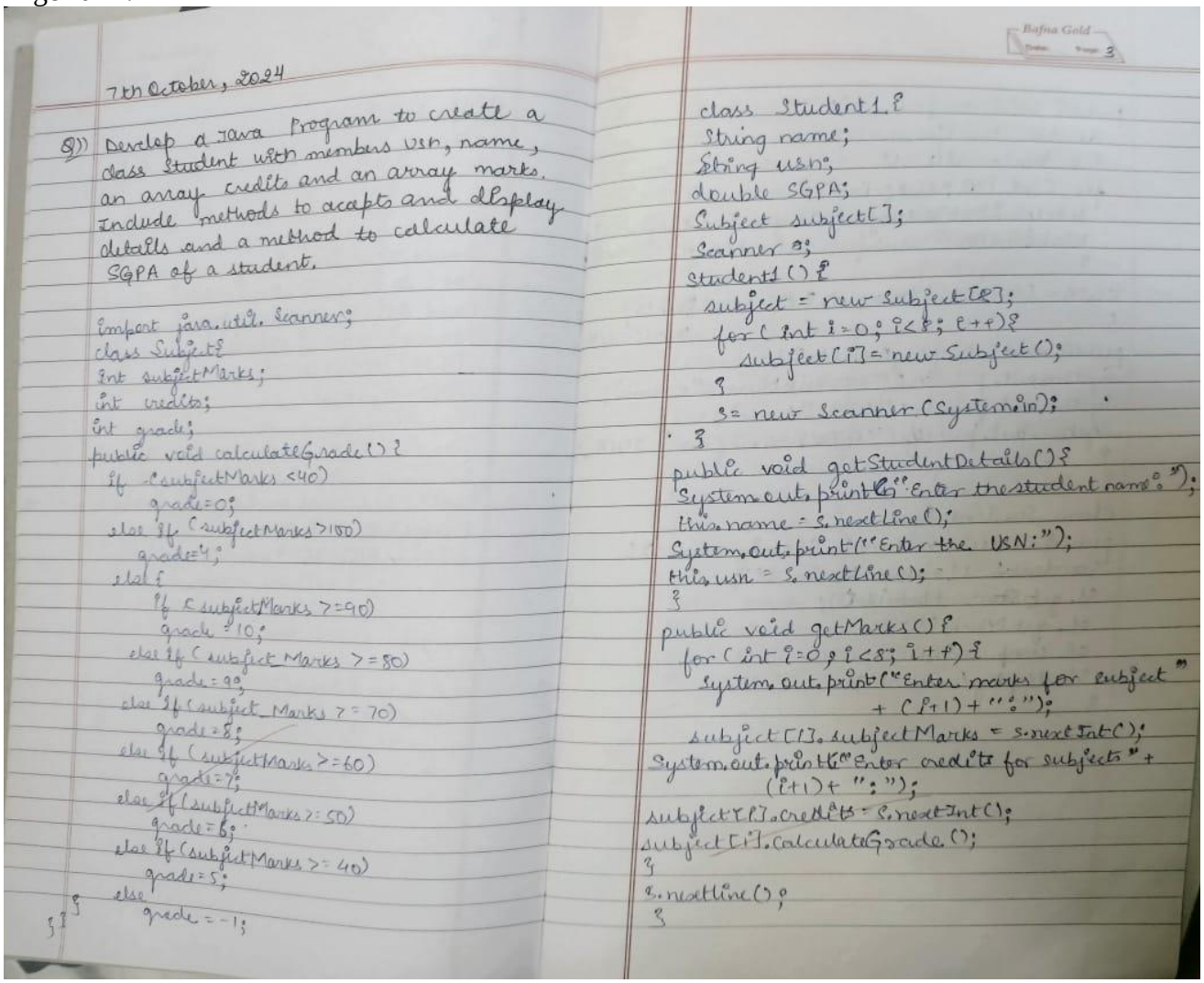
Output:



Program 2

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

Algorithm:

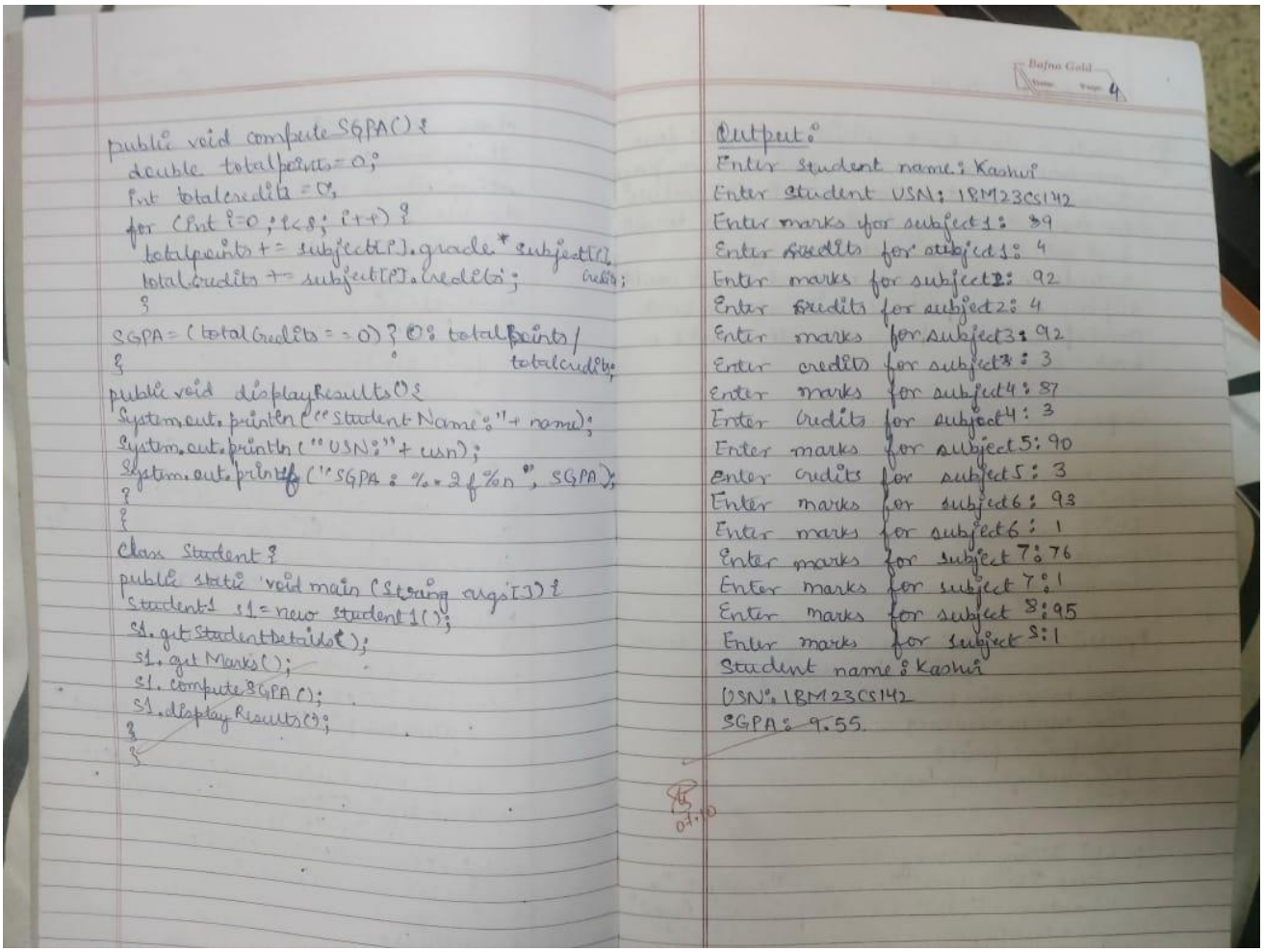


7th October, 2024

Q)) Develop a Java Program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accepts and display details and a method to calculate SGPA of a student.

```
import java.util.Scanner;
class Subject{
int subjectMarks;
int credits;
int grade;
public void calculateGrade(){
  if (subjectMarks <40)
      grade=0;
  else if (subjectMarks>100)
    grade=4;
  else{
      if (subjectMarks >=90)
        grade =10;
    else if (subject_Marks >=80)
        grade=9;
    else if (subject_Marks >=70)
        grade=8;
    else if (subjectMarks >=60)
        grade=7;
    else if (subjectMarks >=50)
        grade=6;
    else if (subjectMarks >=40)
        grade=5;
    else
        grade=-1;
}}
```

```
class Student1{
String name;
String usn;
double SGPA;
Subject subject[];
Scanner s;
Student1(){
  subject = new subject[8];
  for(int i=0; i<8; i++){
    subject[i]=new Subject();
  }
  s= new Scanner(System.in);
}
public void getStudentDetails(){
System.out.println("Enter the student name:");
this.name = s.nextLine();
System.out.println("Enter the USN:");
this.usn = s.nextLine();
}
public void getMarks(){
  for(int i=0;i<5;i++){
    System.out.print("Enter marks for subject"
                + (i+1)+":");
    subject[i].subjectMarks = s.nextInt();
System.out.println("Enter credits for subjects"+
          (i+1)+":");
subject[i].credits=s.nextInt();
subject[i].calculateGrade();
}
s.nextLine();
}
```



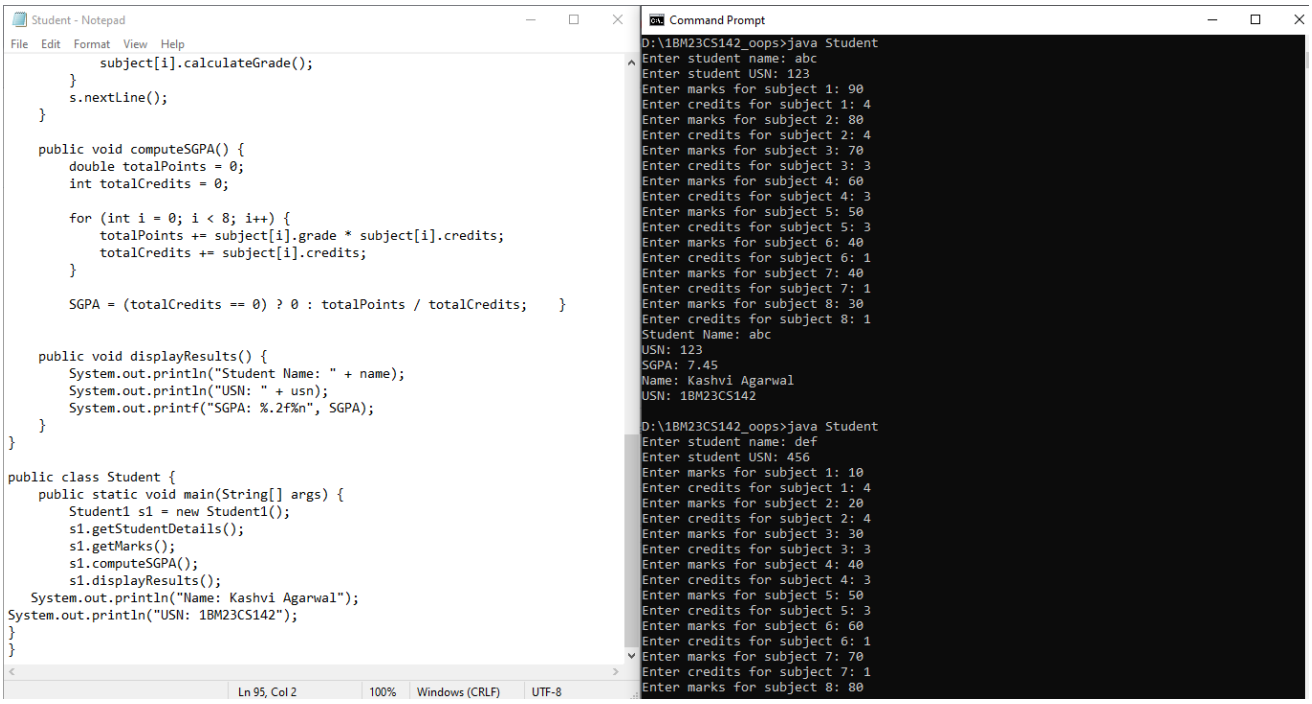
Code:

```
import java.util.Scanner;

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

    public void calculateGrade() {
        if (subjectMarks < 40)
            grade = 0;
        else if (subjectMarks > 100)
            grade = 4; // Invalid grade, assuming it’s an error
        else {
                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 = -1;
        }
    }
}

class Student1 {
    String name;
    String usn;
    double SGPA;
    Subject subject[];
    Scanner s;
    Student1() {
        subject = new Subject[8];
        for (int i = 0; i < 8; i++) {
            subject[i] = new Subject();
        }
        s = new Scanner(System.in);
    }


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


    public void getMarks() {
        for (int i = 0; i < 8; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            subject[i].subjectMarks = s.nextInt();
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            subject[i].credits = s.nextInt();
            subject[i].calculateGrade();
        }
        s.nextLine();
    }
```

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

        for (int i = 0; i < 8; i++) {
            totalPoints += subject[i].grade * subject[i].credits;
            totalCredits += subject[i].credits;
        }

        SGPA = (totalCredits == 0) ? 0 : totalPoints / totalCredits;    }


    public void displayResults() {
        System.out.println("Student Name: " + name);
        System.out.println("USN: " + usn);
        System.out.printf("SGPA: %.2f%n", SGPA);
    }
}

public class Student {
    public static void main(String[] args) {
        Student1 s1 = new Student1();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();
        s1.displayResults();
   System.out.println("Name: Kashvi Agarwal");
System.out.println("USN: 1BM23CS142");
}
}
```

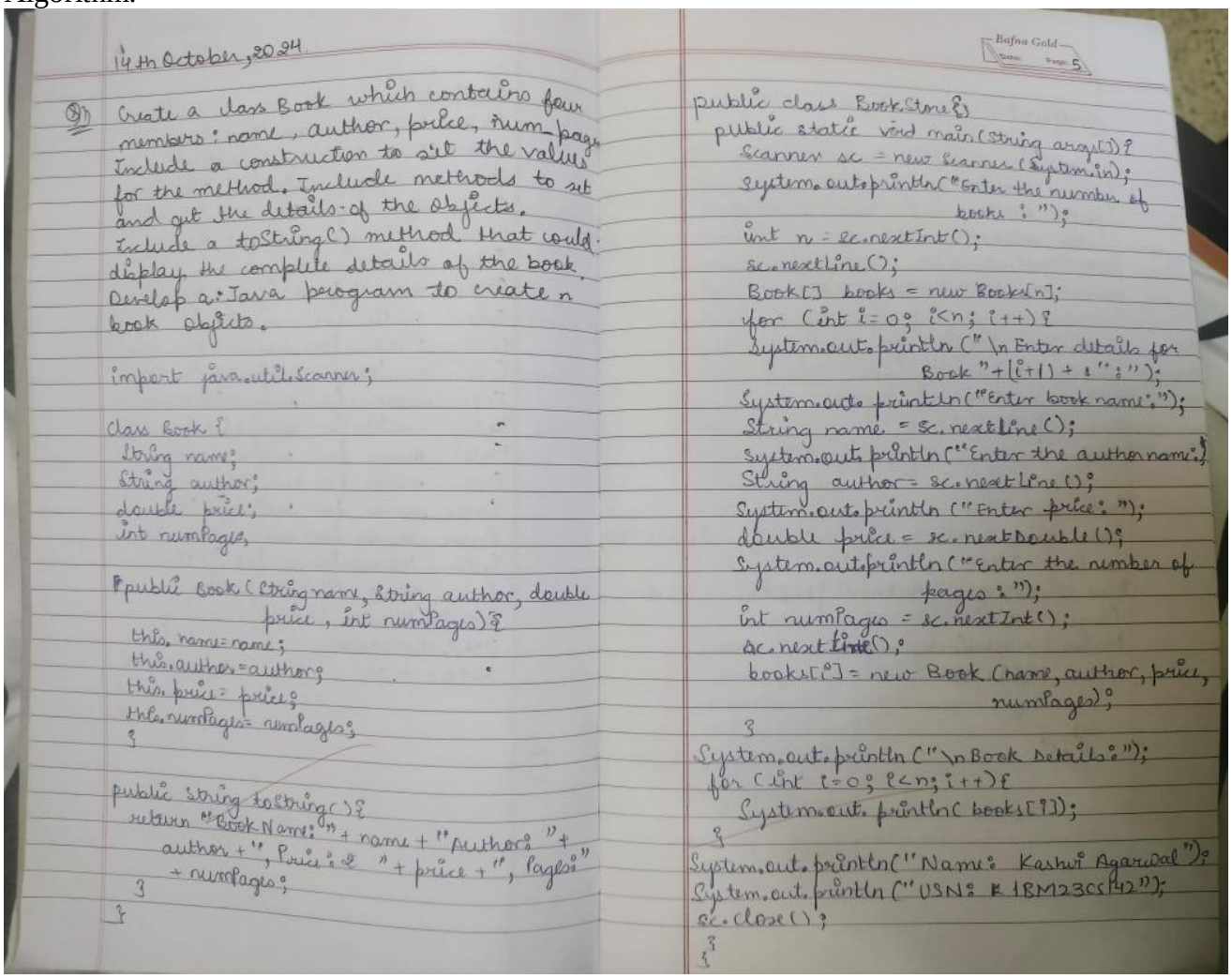
Output:


Student - Notepad
File Edit Format View Help
subject[i].calculateGrade();
}
s.nextLine();
}
public void computeSGPA() {
double totalPoints = 0;
int totalCredits = 0;
for (int i = 0; i < 8; i++) {
totalPoints += subject[i].grade * subject[i].credits;
totalCredits += subject[i].credits;
}
SGPA = (totalCredits == 0) ? 0 : totalPoints / totalCredits; }
public void displayResults() {
System.out.println("Student Name: " + name);
System.out.println("USN: " + usn);
System.out.printf("SGPA: %.2f%n", SGPA);
}
}
public class Student {
public static void main(String[] args) {
Student1 s1 = new Student1();
s1.getStudentDetails();
s1.getMarks();
s1.computeSGPA();
s1.displayResults();
System.out.println("Name: Kashvi Agarwal");
System.out.println("USN: 1BM23CS142");
}
}
Ln 95, Col 2
100%
Windows (CRLF)
UTF-8
Command Prompt
D:\1BM23CS142_oops>java Student
Enter student name: abc
Enter student USN: 123
Enter marks for subject 1: 90
Enter credits for subject 1: 4
Enter marks for subject 2: 80
Enter credits for subject 2: 4
Enter marks for subject 3: 70
Enter credits for subject 3: 3
Enter marks for subject 4: 60
Enter credits for subject 4: 3
Enter marks for subject 5: 50
Enter credits for subject 5: 3
Enter marks for subject 6: 40
Enter credits for subject 6: 1
Enter marks for subject 7: 40
Enter credits for subject 7: 1
Enter marks for subject 8: 30
Enter credits for subject 8: 1
Student Name: abc
USN: 123
SGPA: 7.45
Name: Kashvi Agarwal
USN: 1BM23CS142
D:\1BM23CS142_oops>java Student
Enter student name: def
Enter student USN: 456
Enter marks for subject 1: 10
Enter credits for subject 1: 4
Enter marks for subject 2: 20
Enter credits for subject 2: 4
Enter marks for subject 3: 30
Enter credits for subject 3: 3
Enter marks for subject 4: 40
Enter credits for subject 4: 3
Enter marks for subject 5: 50
Enter credits for subject 5: 3
Enter marks for subject 6: 60
Enter credits for subject 6: 1
Enter marks for subject 7: 70
Enter credits for subject 7: 1
Enter marks for subject 8: 80

Program 3

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

Algorithm:



14th October, 2024

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

```
import java.util.Scanner;

class Book {
    String name;
    String author;
    double price;
    int numPages;

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

    public String toString() {
        return "Book Name: " + name + "Author: " + author + ", Price: " + price + ", Pages: " + numPages;
    }
}

public class BookStore {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of books : ");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            System.out.println("\n Enter details for Book " + (i+1) + ":");
            System.out.println("Enter book name:");
            String name = sc.nextLine();
            System.out.println("Enter the author name:");
            String author = sc.nextLine();
            System.out.println("Enter price: ");
            double price = sc.nextDouble();
            System.out.println("Enter the number of pages : ");
            int numPages = sc.nextInt();
            sc.nextLine();
            books[i] = new Book(name, author, price, numPages);
        }
        System.out.println("\n Book Details:");
        for (int i = 0; i < n; i++) {
            System.out.println(books[i]);
        }
        System.out.println("Name: Kashvi Agarwal");
        System.out.println("USN: 1BM23CS142");
        sc.close();
    }
}
```

Output:

```
Enter the number of books: 2
Enter details for Book1:
Enter book name: abc
Enter author name: def
Enter price: 1234
Enter number of pages: 400

Enter the details for Book2:
Enter book name: ghi
Enter author name: jkl
Enter price: 5678
Enter number of pages: 500

Book details:
Book Name: abc, Author: def, Price: 1234.0,
    Pages: 400
Book Name: ghi, Author: jkl, Price: 5678.0,
Pages: 500
Name: Kashvi Agarwal
USN: 1BM23CS142
```

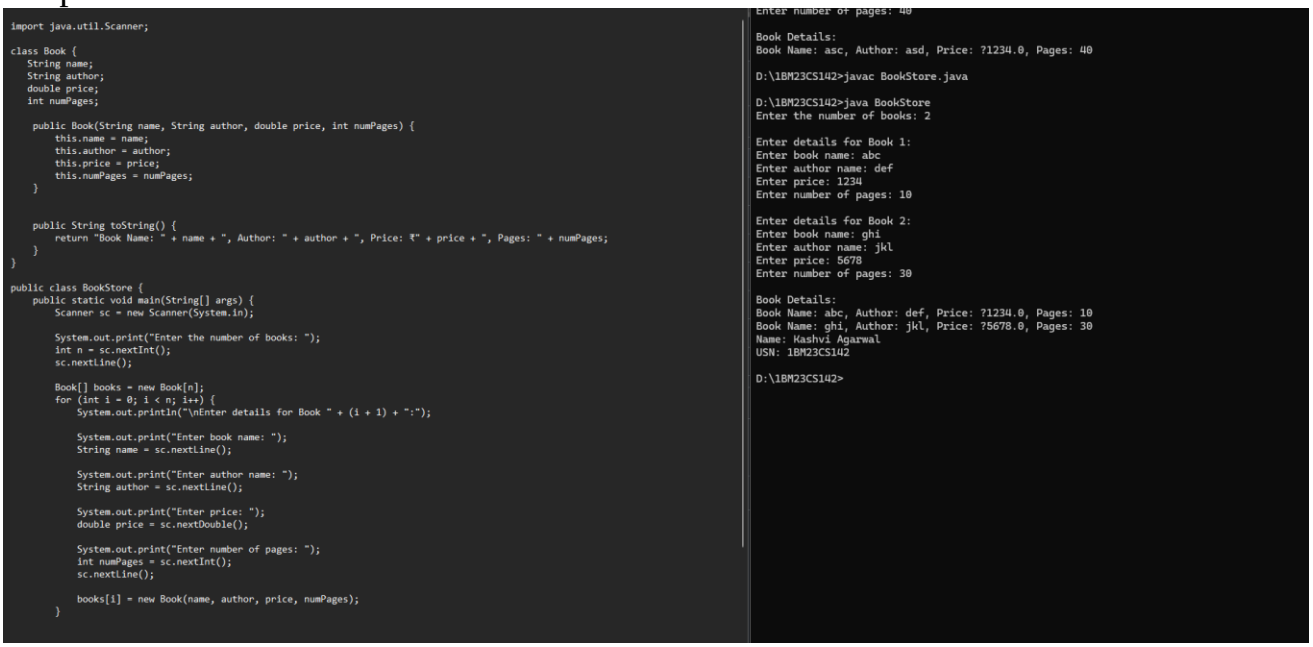
14.10

Code:

```
import java.util.Scanner;

class Book {
   String name;
   String author;
   double price;
   int numPages;

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


    public String toString() {
       return "Book Name: " + name + ", Author: " + author + ", Price: ₹" + price + ", Pages: "
+ numPages;
    }
}

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

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

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

          System.out.print("Enter book name: ");
          String name = sc.nextLine();

          System.out.print("Enter author name: ");
          String author = sc.nextLine();

          System.out.print("Enter price: ");
          double price = sc.nextDouble();

          System.out.print("Enter number of pages: ");
          int numPages = sc.nextInt();
```

```
            sc.nextLine();

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


        System.out.println("\nBook Details:");
        for (int i = 0; i < n; i++) {
            System.out.println(books[i]);
        }
System.out.println("Name: Kashvi Agarwal");
System.out.println("USN: 1BM23CS142");
        sc.close();
    }
}
```

Output:



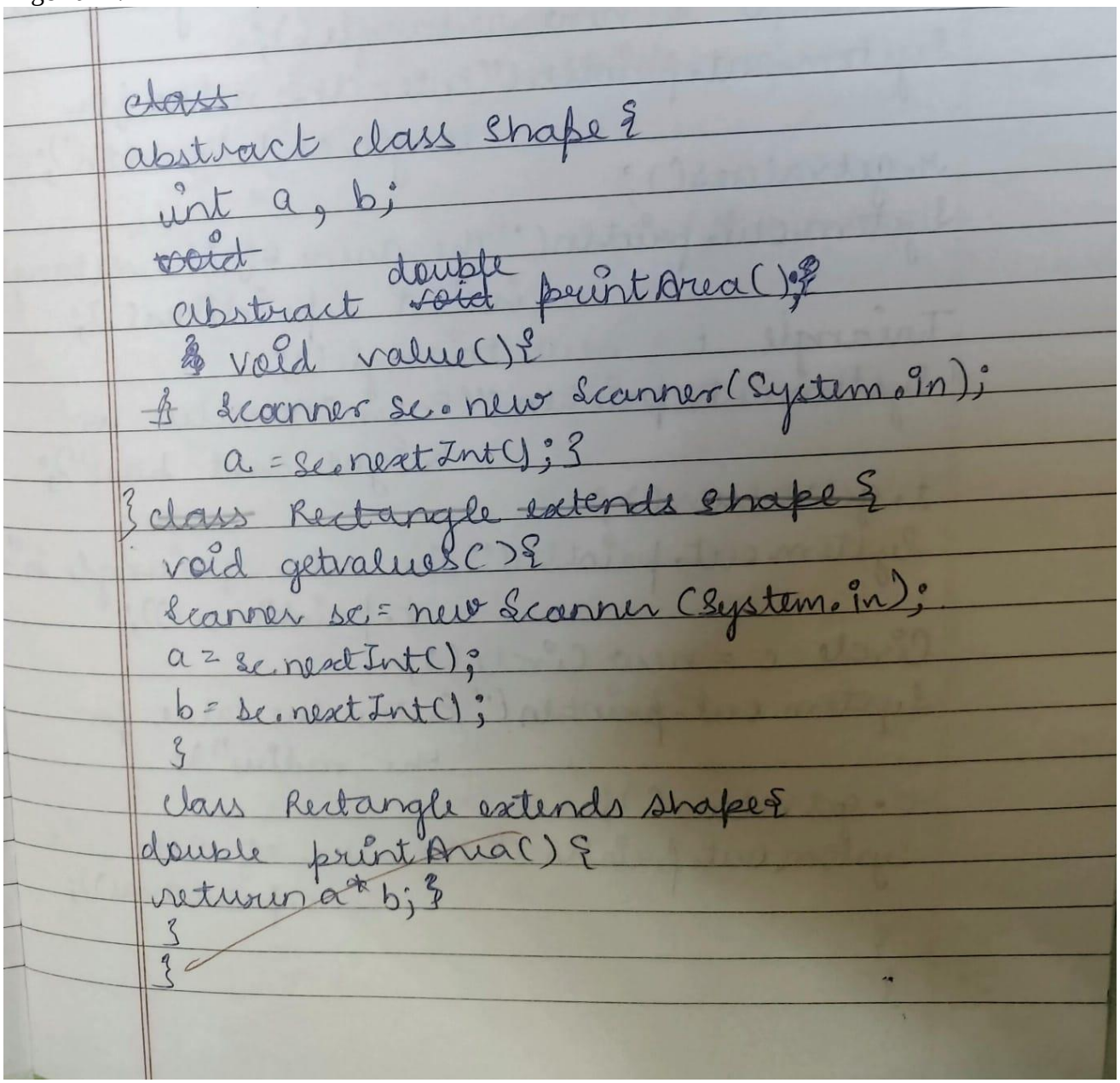
```
import java.util.Scanner;

class Book {
    String name;
    String author;
    double price;
    int numPages;

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


    public String toString() {
        return "Book Name: " + name + ", Author: " + author + ", Price: ₹" + price + ", Pages: " + numPages;
    }
}

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

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

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

            System.out.print("Enter book name: ");
            String name = sc.nextLine();

            System.out.print("Enter author name: ");
            String author = sc.nextLine();

            System.out.print("Enter price: ");
            double price = sc.nextDouble();

            System.out.print("Enter number of pages: ");
            int numPages = sc.nextInt();
            sc.nextLine();

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

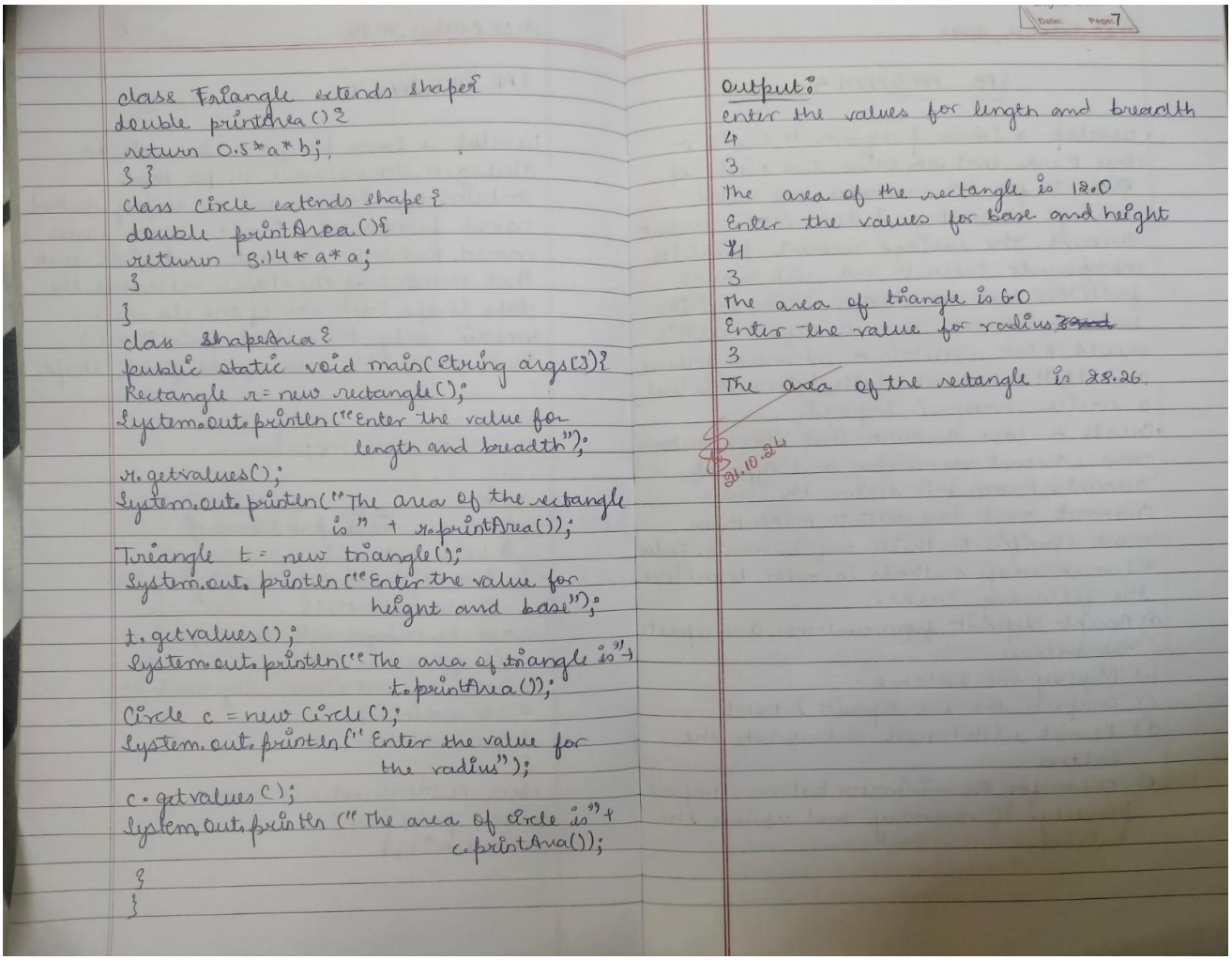
```
Enter number of pages: 40

Book Details:
Book Name: asc, Author: asd, Price: ?1234.0, Pages: 40

D:\1BM23CS142>javac BookStore.java

D:\1BM23CS142>java BookStore
Enter the number of books: 2

Enter details for Book 1:
Enter book name: abc
Enter author name: def
Enter price: 1234
Enter number of pages: 10

Enter details for Book 2:
Enter book name: ghi
Enter author name: jkl
Enter price: 5678
Enter number of pages: 30

Book Details:
Book Name: abc, Author: def, Price: ?1234.0, Pages: 10
Book Name: ghi, Author: jkl, Price: ?5678.0, Pages: 30
Name: Kashvi Agarwal
USN: 1BM23CS142

D:\1BM23CS142>
```

Program 4

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

Algorithm:





```
class Triangle extends shape{
double printArea(){
return 0.5*a*b;
}}
class Circle extends shape{
double printArea(){
return 3.14*a*a;
}
}
class ShapeArea{
public static void main(String args[]){
Rectangle r=new rectangle();
System.out.println("Enter the value for
                    length and breadth");
r.getvalues();
System.out.println("The area of the rectangle
                    is " + r.printArea());
Triangle t=new triangle();
System.out.println("Enter the value for
                    height and base");
t.getvalues();
System.out.println("The area of triangle is"+
                    t.printArea());
Circle c=new Circle();
System.out.println("Enter the value for
                    the radius");
c.getvalues();
System.out.println("The area of Circle is"+
                    c.printArea());
}
}
```

```
output:
Enter the values for length and breadth
4
3
The area of the rectangle is 12.0
Enter the values for base and height
4
3
The area of triangle is 6.0
Enter the value for radius
3
The area of the rectangle is 28.26
```

31.10.24

Code:

```
import java.util.Scanner;
abstract class shape{
int a,b;
abstract double printArea();
void getvalue(){
Scanner sc = new Scanner(System.in);
a = sc.nextInt();
b = sc.nextInt();
}
void value(){
Scanner sc = new Scanner(System.in);
a = sc.nextInt();
}
}
class Rectangle extends shape{
double printArea(){
return a*b;}
```

```
}
class Triangle extends shape{
double printArea(){
return a*b*0.5;}
}
class Circle extends shape{
double printArea(){
return a*a*3.14;}
}

class ShapeArea{
public static void main(String args[]){
Rectangle r = new Rectangle();
System.out.println("enter the values for length and breadth");
r.getvalue();
System.out.println("The area of the rectangle is " + r.printArea());
Triangle t = new Triangle();
System.out.println("enter the values for base and height");
t.getvalue();
System.out.println("The area of the triangle is " + t.printArea());
Circle c = new Circle();
System.out.println("enter the values for radius ");
c.value();
System.out.println("The area of the rectangle is " + c.printArea());
System.out.println("Name: Kashvi Agarwal");
System.out.println("USN: 1BM23CS142");
}
}
```

Output:

```java
import java.util.Scanner;
abstract class shape{
int a,b;
abstract double printArea();
void getvalue(){
Scanner sc = new Scanner(System.in);
a = sc.nextInt();
b = sc.nextInt();
}
void value(){
Scanner sc = new Scanner(System.in);
a = sc.nextInt();
}
}
class Rectangle extends shape{
double printArea(){
return a*b;}
}
class Triangle extends shape{
double printArea(){
return a*b*0.5;}
}
class Circle extends shape{
double printArea(){
return a*a*3.14;}
}

class ShapeArea{
public static void main(String args[]){
Rectangle r = new Rectangle();
System.out.println("enter the values for length and breadth");
r.getvalue();
System.out.println("The area of the rectangle is " + r.printArea());
Triangle t = new Triangle();
System.out.println("enter the values for base and height");
t.getvalue();
System.out.println("The area of the triangle is " + t.printArea());
Circle c = new Circle();
System.out.println("enter the values for radius ");
c.value();
System.out.println("The area of the rectangle is " + c.printArea());
System.out.println("Name: Kashvi Agarwal");
System.out.println("USN: 1BM23CS142");
}
}
```

```
The area of the rectangle is12.0
enter the values for base and height
4
3
The area of the triangle is6.0
enter the values for radius
3
The area of the rectangle is28.26

D:\1BM23CS142> javac ShapeArea.java

D:\1BM23CS142> java ShapeArea
enter the values for length and breadth
4
3
The area of the rectangle is 12.0
enter the values for base and height
4
3
The area of the triangle is 6.0
enter the values for radius
3
The area of the rectangle is 28.26
Name: Kashvi Agarwal
USN: 1BM23CS142

D:\1BM23CS142>
```

Program 5

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called a 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 a 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 deposits from customers and update the balance.

b)Display the balance.

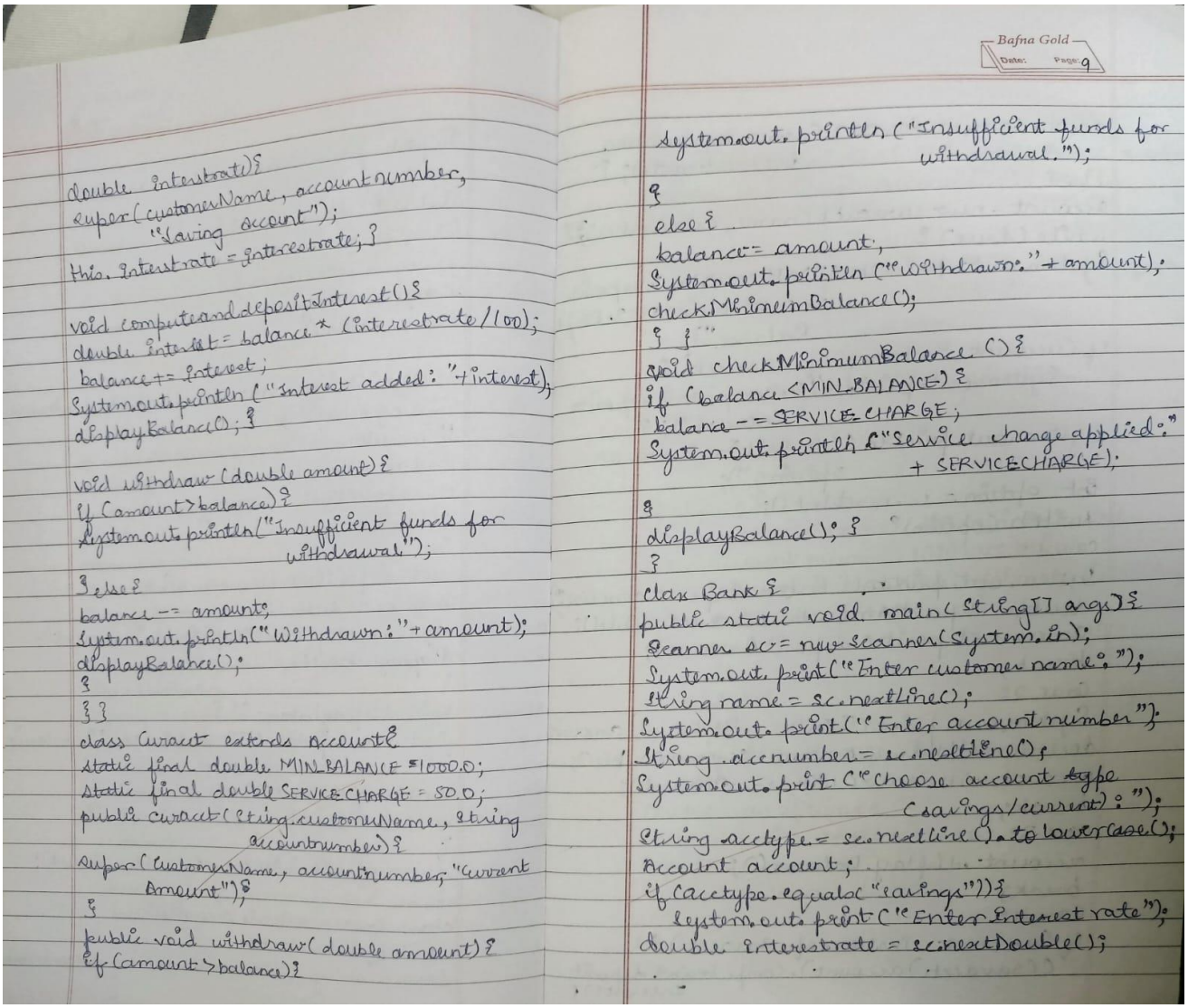
c)Compute and deposit interest

d)Permit withdrawal and update the balance

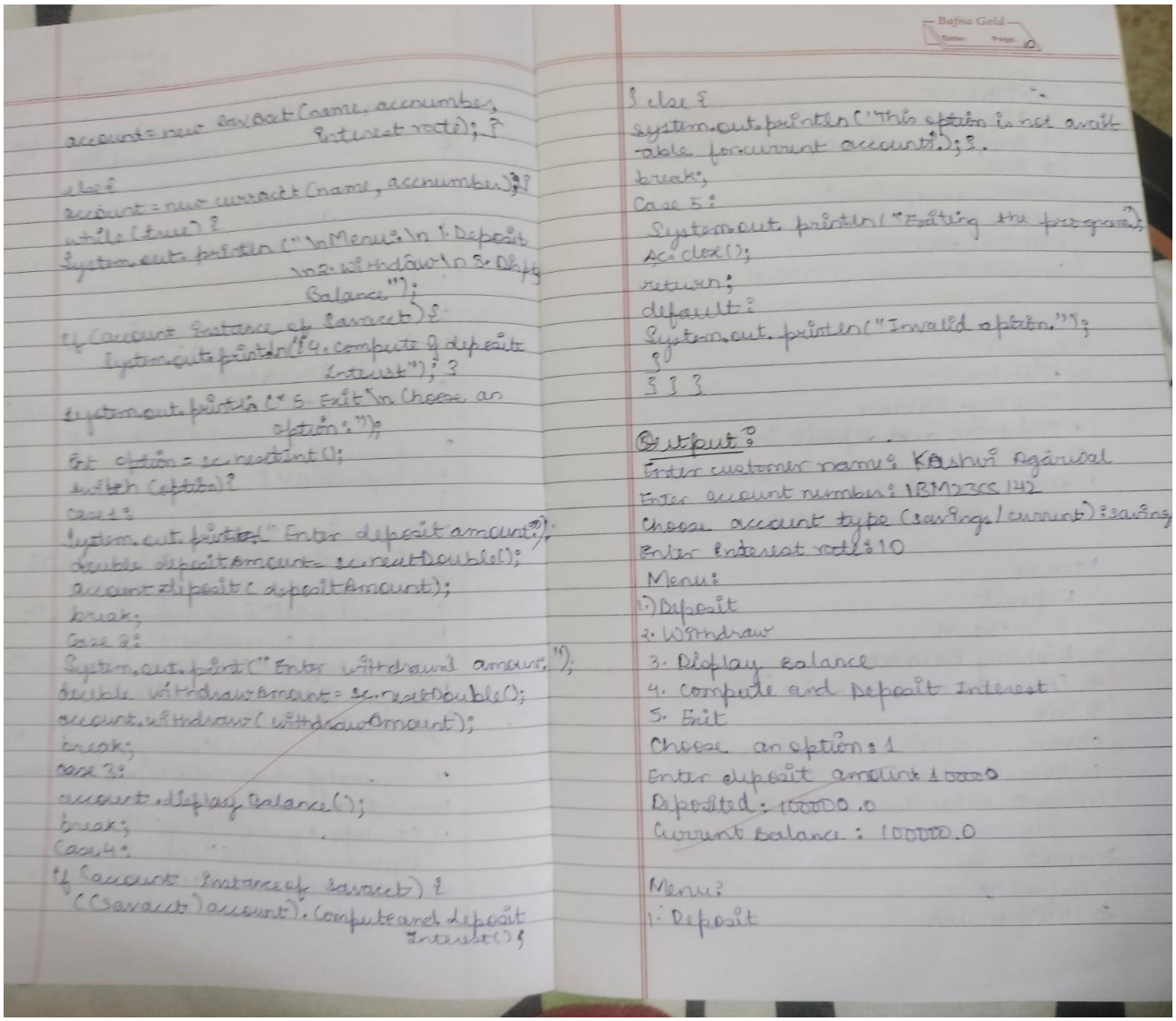
e) Check for the minimum balance, impose a penalty if necessary and update the balance.

Algorithm:

```
import java.util.Scanner;

abstract class account {
String customerName;
String accountnumber;
String accounttype;
double balance;

Account(String customername, String accountnumber,
  String accounttype) {
this.customername = customername;
this.accountnumber = accountnumber;
this.accounttype = accountype;
this.balance = 0.0;
}
void deposit(double amount) {
balance += amount;
System.out.println("Deposited: " + amount);
displayBalance();
}
void displayBalance() {
System.out.println("Current Balance:" + balance);
}
double getBalance() {
return balance;
}
abstract void withdraw(double amount);
}
class Savacc extends Account{
double interestrate;
Savacc(String customername, string accountnumb
```



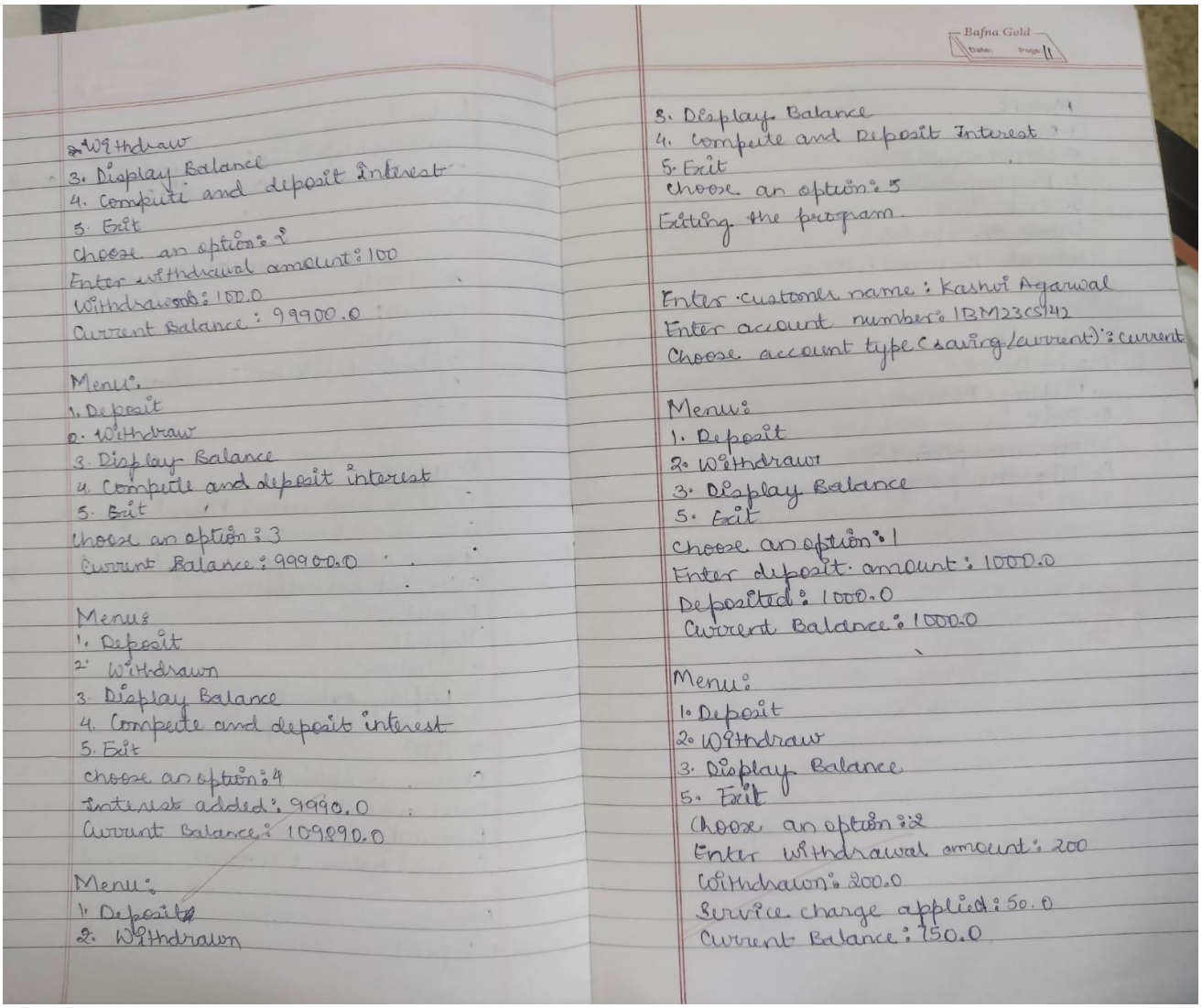
```
double interestrate){
super(customerName, accountnumber,
        "saving account");
this.interestrate = interestrate;}

void computeanddepositInterest(){
double interest = balance * (interestrate/100);
balance += interest;
System.out.println("Interest added: "+interest);
displayBalance();}

void withdraw(double amount){
if(amount>balance){
System.out.println("Insufficient funds for
                    withdrawal");
}else{
balance -= amount;
System.out.println("Withdrawn:"+amount);
displayBalance();
}}
class Curacct extends Account{
static final double MIN_BALANCE =1000.0;
static final double SERVICE_CHARGE = 50.0;
public Curacct(String customerName, String
                accountnumber){
super(CustomerName, accountnumber, "Current
        Amount");}
public void withdraw(double amount){
if(amount>balance){
System.out.println("Insufficient funds for
                    withdrawal.");
}
else{
balance -= amount;
System.out.println("Withdrawn:"+amount);
checkMinimumBalance();
}}
void checkMinimumBalance(){
if(balance<MIN_BALANCE){
balance -= SERVICE_CHARGE;
System.out.println("Service charge applied:"
                    +SERVICECHARGE);
}
displayBalance();}
}
class Bank{
public static void main(String[] args){
Scanner sc = new Scanner(System.in);
System.out.print("Enter customer name:");
String name = sc.nextLine();
System.out.print("Enter account number");
String accnumber = sc.nextLine();
System.out.print("choose account type
                (savings/current):");
String acctype = sc.nextline().tolowercase();
Account account;
if(acctype.equals("savings")){
    System.out.print("Enter Interest rate");
double interestrate = sc.nextDouble();
```



```
account = new SavAcct(name, accnumber,
                      interestrate);
else
account = new currAcct(name, accnumber);
while (true)
System.out.println("\nMenu:\n1. Deposit
\n2. Withdraw\n3. Disply
Balance");
if (account instance of SavAcct)
System.out.println("4. compute & deposit
Interest");
System.out.println("5. Exit\nChoose an
option:");
int option = sc.nextInt();
switch (option)
case 1:
System.out.println("Enter deposit amount");
double depositAmount = sc.nextDouble();
account.deposit(depositAmount);
break;
Case 2:
System.out.print("Enter withdrawal amount");
double withdrawAmount = sc.nextDouble();
account.withdraw(withdrawAmount);
break;
case 3:
account.displayBalance();
break;
Case 4:
if (account instance of SavAcct)
((SavAcct)account).computeandDepositInterest();
else
System.out.println("This option is not avail-
able for current account.");
break;
Case 5:
System.out.println("Exiting the program");
Sc.close();
return;
default:
System.out.println("Invalid option.");
}
}
}
```

Output:

```
Enter customer name: Kashvi Agarwal
Enter account number: IBM2365142
Choose account type (savings/current): savings
Enter interest rate: 10
Menu:
1.) Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Choose an option: 1
Enter deposit amount 100000
Deposited: 100000.0
Current balance: 100000.0

Menu:
1: Deposit
```



2. Withdraw
3. Display Balance
4. Compute and deposit Interest
5. Exit
Choose an option: 2
Enter withdrawal amount: 100
Withdrawn: 100.0
Current Balance: 99900.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute and deposit interest
5. Exit
Choose an option: 3
Current Balance: 99900.0

Menu:
1. Deposit
2. Withdrawn
3. Display Balance
4. Compute and deposit interest
5. Exit
Choose an option: 4
Interest added: 9990.0
Current Balance: 109890.0

Menu:
1. Deposit
2. Withdrawn
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Choose an option: 5
Exiting the program.

Enter customer name: Kashvi Agarwal
Enter account number: IBM23CS142
Choose account type (saving/current): current

Menu:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
Choose an option: 1
Enter deposit amount: 1000.0
Deposited: 1000.0
Current Balance: 1000.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
Choose an option: 2
Enter withdrawal amount: 200
Withdrawn: 200.0
Service charge applied: 50.0
Current Balance: 750.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
Choose an option: 3
Current Balance: 750.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Choose an option: 5
Exiting the program

28.10

Code:

```
import java.util.Scanner;

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

   public Account(String customerName, String accountNumber, String accountType) {
      this.customerName = customerName;
      this.accountNumber = accountNumber;
      this.accountType = accountType;
      this.balance = 0.0;
   }

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

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

   public double getBalance() {
      return balance;
   }

   public abstract void withdraw(double amount);
}

class SavAcct extends Account {
   private double interestRate;

   public SavAcct(String customerName, String accountNumber, double interestRate) {
      super(customerName, accountNumber, "Savings Account");
      this.interestRate = interestRate;
   }

   public void computeAndDepositInterest() {
      double interest = balance * (interestRate / 100);
      balance += interest;
      System.out.println("Interest added: " + interest);
      displayBalance();
```

```
    }

    @Override
    public void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient funds for withdrawal.");
        } else {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
            displayBalance();
        }
    }
}

class CurAcct extends Account {
   static final double MIN_BALANCE = 1000.0;
   static final double SERVICE_CHARGE = 50.0;

    public CurAcct(String customerName, String accountNumber) {
        super(customerName, accountNumber, "Current Account");
    }

    public void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient funds for withdrawal.");
        } else {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
            checkMinimumBalance();
        }
    }

     void checkMinimumBalance() {
        if (balance < MIN_BALANCE) {
            balance -= SERVICE_CHARGE;
            System.out.println("Service charge applied: " + SERVICE_CHARGE);
        }
        displayBalance();
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Welcome to the Bank!");

        System.out.print("Enter customer name: ");
```

```
String name = scanner.nextLine();

System.out.print("Enter account number: ");
String accNumber = scanner.nextLine();

System.out.print("Choose account type (savings/current): ");
String accType = scanner.nextLine().toLowerCase();

Account account;
if (accType.equals("savings")) {
    System.out.print("Enter interest rate: ");
    double interestRate = scanner.nextDouble();
    account = new SavAcct(name, accNumber, interestRate);
} else {
    account = new CurAcct(name, accNumber);
}

while (true) {
    System.out.println("\nMenu:");
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Display Balance");
    if (account instanceof SavAcct) {
        System.out.println("4. Compute and Deposit Interest");
    }
    System.out.println("5. Exit");
    System.out.print("Choose an option: ");
    int option = scanner.nextInt();

    switch (option) {
        case 1:
            System.out.print("Enter deposit amount: ");
            double depositAmount = scanner.nextDouble();
            account.deposit(depositAmount);
            break;
        case 2:
            System.out.print("Enter withdrawal amount: ");
            double withdrawAmount = scanner.nextDouble();
            account.withdraw(withdrawAmount);
            break;
        case 3:
            account.displayBalance();
            break;
        case 4:
            if (account instanceof SavAcct) {
                ((SavAcct) account).computeAndDepositInterest();
            } else {
```

```
                System.out.println("This option is not available for current accounts.");
            }
            break;
        case 5:
            System.out.println("Exiting the program.");
            scanner.close();
            return;
        default:
            System.out.println("Invalid option. Please try again.");
      }
    }
  }
}
```

Output:

```
import java.util.Scanner;

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

Account(String customerName, String accountNumber, String accountType) {
 this.customerName = customerName;
 this.accountNumber = accountNumber;
 this.accountType = accountType;
 this.balance = 0.0;
    }
void deposit(double amount) {
balance += amount;
System.out.println("Deposited: " + amount);
displayBalance();
    }
void displayBalance() {
System.out.println("Current Balance: " + balance);
    }
double getBalance() {
return balance;
}
abstract void withdraw(double amount);
}
class SavAcct extends Account {
 double interestRate;
SavAcct(String customerName, String accountNumber, double interestRate) {
super(customerName, accountNumber, "Savings Account");
this.interestRate = interestRate;
    }

void computeAndDepositInterest() {
double interest = balance * (interestRate / 100);
 balance += interest;
System.out.println("Interest added: " + interest);
 displayBalance();
    }
 void withdraw(double amount) {
 if (amount > balance) {
 System.out.println("Insufficient funds for withdrawal.");
} else {
balance -= amount;
System.out.println("Withdrawn: " + amount);
 displayBalance();
       }
    }
}
```

Ln 115, Col 8 | 3,583 characters | 100% | Windows (CRLF) | UTF-8

```
Enter customer name: Kashvi Agarwal
Enter account number: 1BM23CS142
Choose account type (savings/current): savings
Enter interest rate: 10

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Choose an option: 1
Enter deposit amount: 100000
Deposited: 100000.0
Current Balance: 100000.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Choose an option: 2
Enter withdrawal amount: 100
Withdrawn: 100.0
Current Balance: 99900.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Choose an option: 3
Current Balance: 99900.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Choose an option: 4
Interest added: 9990.0
Current Balance: 109890.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
```

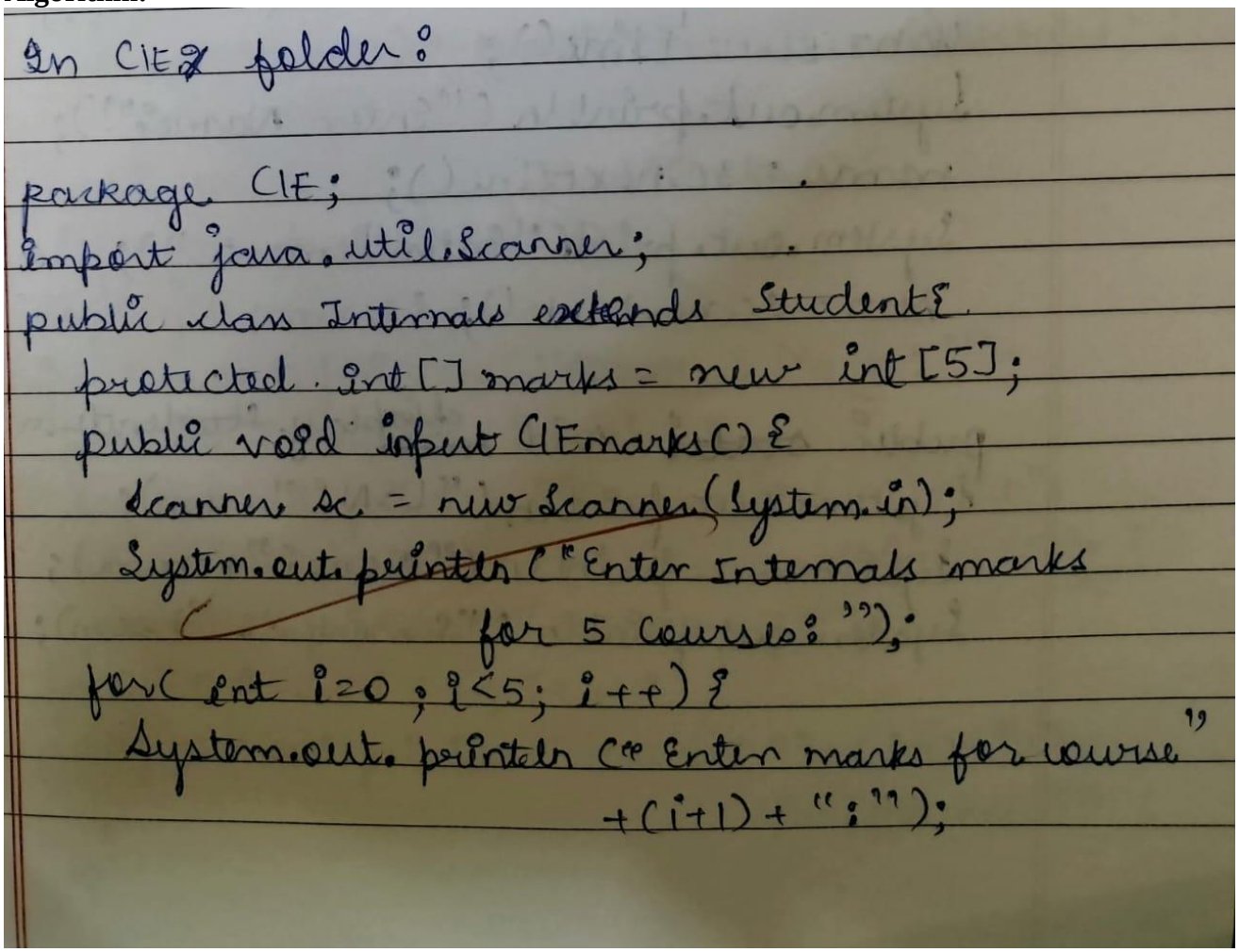
```
abstract class Account {
String customerName;
String accountNumber;
String accountType;
double balance;

Account(String customerName, String accountNumber, String accountType) {
 this.customerName = customerName;
 this.accountNumber = accountNumber;
 this.accountType = accountType;
 this.balance = 0.0;
    }
void deposit(double amount) {
balance += amount;
System.out.println("Deposited: " + amount);
displayBalance();
    }
void displayBalance() {
System.out.println("Current Balance: " + balance);
    }
double getBalance() {
return balance;
}
abstract void withdraw(double amount);
}
class SavAcct extends Account {
 double interestRate;
SavAcct(String customerName, String accountNumber, double interestRate) {
super(customerName, accountNumber, "Savings Account");
this.interestRate = interestRate;
    }

void computeAndDepositInterest() {
double interest = balance * (interestRate / 100);
 balance += interest;
System.out.println("Interest added: " + interest);
 displayBalance();
    }
 void withdraw(double amount) {
 if (amount > balance) {
 System.out.println("Insufficient funds for withdrawal.");
} else {
balance -= amount;
System.out.println("Withdrawn: " + amount);
 displayBalance();
        }
    }
}
```

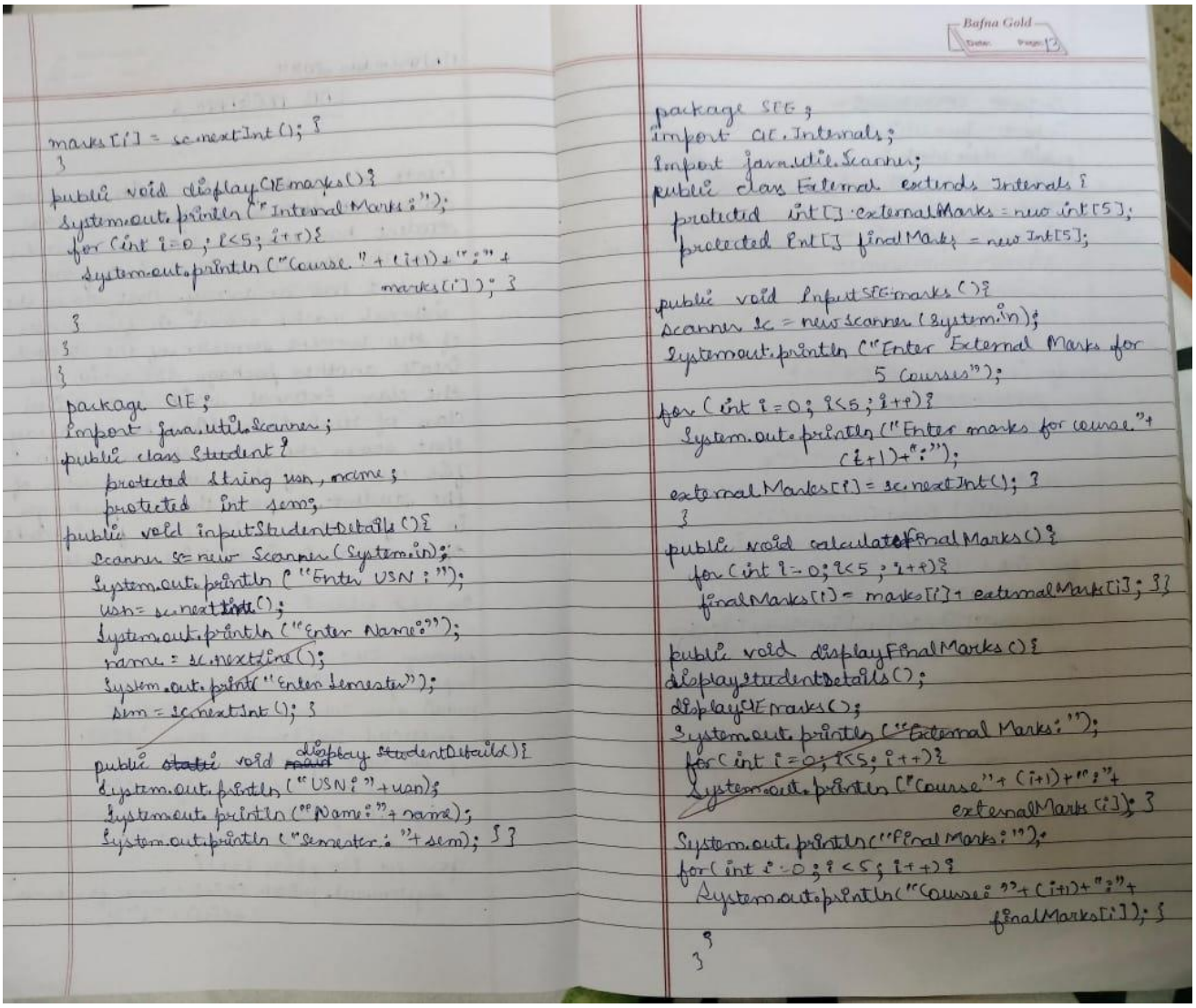
```
Exiting the program.

D:\1BM23CS142>javac Bank.java

D:\1BM23CS142>java Bank
Enter customer name: Kashvi Agarwal
Enter account number: 1BM23CS142
Choose account type (savings/current): current

Menu:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
Choose an option: 1
Enter deposit amount: 1000
Deposited: 1000.0
Current Balance: 1000.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
Choose an option: 2
Enter withdrawal amount: 200
Withdrawn: 200.0
Service charge applied: 50.0
Current Balance: 750.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
Choose an option: 3
Current Balance: 750.0

Menu:
1. Deposit
2. Withdraw
3. Display Balance
5. Exit
Choose an option: 5
Exiting the program.
```

Program 6

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

Algorithm:

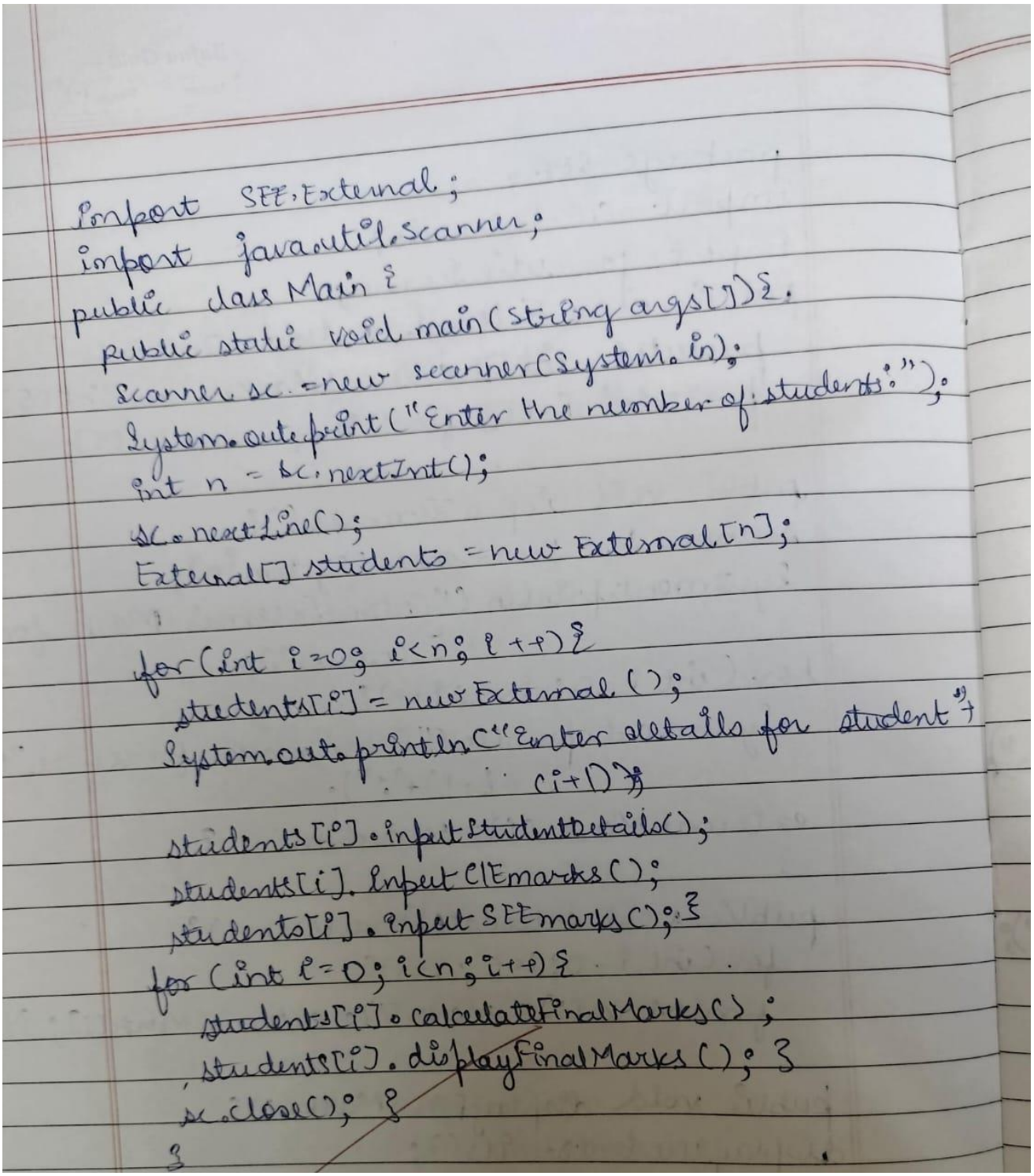




```
marks[i] = sc.nextInt(); }
}
public void displayCIEmarks(){
 System.out.println("Internal Marks:");
 for(int i=0; i<5; i++){
  System.out.println("Course."+(i+1)+":"+
                     marks[i]); }
}
}
package CIE;
import java.util.Scanner;
public class Student{
    protected String usn, name;
    protected int sem;
public void inputStudentDetails(){
 Scanner sc=new Scanner(System.in);
 System.out.println("Enter USN:");
 usn= sc.nextLine();
 System.out.println("Enter Name:");
 name = sc.nextLine();
 System.out.print("Enter Semester");
 sem = sc.nextInt(); }

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

Bafna Gold

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

public void inputSEEmarks(){
Scanner sc = new Scanner(System.in);
System.out.println("Enter External Marks for
                    5 Courses");
for(int i=0; i<5; i++){
 System.out.println("Enter marks for course."+
                    (i+1)+":");
externalMarks[i] = sc.nextInt(); }
public void calculateFinalMarks(){
 for(int i=0; i<5; i++){
  finalMarks[i] = marks[i]+externalMarks[i]; }}

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



Code:
Main class:

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

public class Main {
```

```
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = sc.nextInt();
        sc.nextLine(); // Consume the newline character
        External[] students = new External[n];


        for (int i = 0; i < n; i++) {
            students[i] = new External();
            System.out.println("Enter details for student " + (i + 1));
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
        }
        for (int i = 0; i < n; i++) {
            students[i].calculateFinalMarks();
            students[i].displayFinalMarks();
        }

        sc.close();
    }
}
```

In CIE package:

```
package CIE;

import java.util.Scanner;

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

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

    public void displayCIEmarks() {
        System.out.println("Internal Marks: ");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + marks[i]);
        }
```

```
    }
}

package CIE;

import java.util.Scanner;

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

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

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

In SEE Package:

```
package SEE;

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

public class External extends Internals {
    protected int[] externalMarks = new int[5];
    protected int[] finalMarks = new int[5];

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

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


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

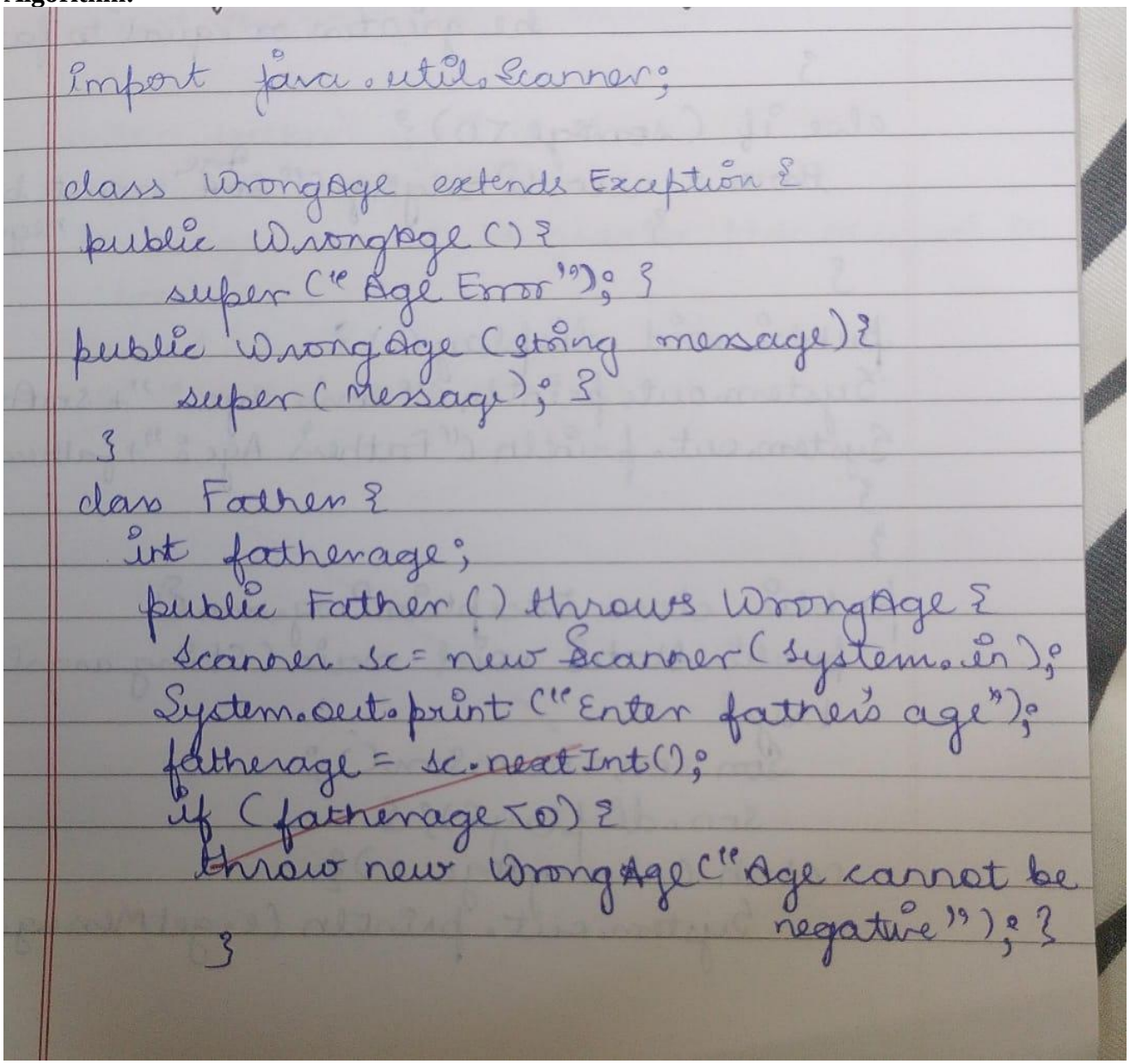
Output:

```
Enter marks for course 1:
94
Enter marks for course 2:
93
Enter marks for course 3:
92
Enter marks for course 4:
91
Enter marks for course 5:
90
USN: 1BM23CS142
Name: Kashvi Agarwal
Semester: 3
Internal Marks:
Course 1: 100
Course 2: 99
Course 3: 98
Course 4: 97
Course 5: 96
External Marks:
Course 1: 95
Course 2: 94
Course 3: 93
Course 4: 92
Course 5: 91
Final Marks:
Course 1: 195
Course 2: 193
Course 3: 191
Course 4: 189
Course 5: 187
USN: 1BM23CS142
Name: Kashvi Agarwal
Semester: 3
Internal Marks:
Course 1: 99
Course 2: 98
Course 3: 97
Course 4: 96
Course 5: 95
External Marks:
Course 1: 94
Course 2: 93
Course 3: 92
Course 4: 91
Course 5: 90
Final Marks:
Course 1: 193
Course 2: 191
Course 3: 189
Course 4: 187
```

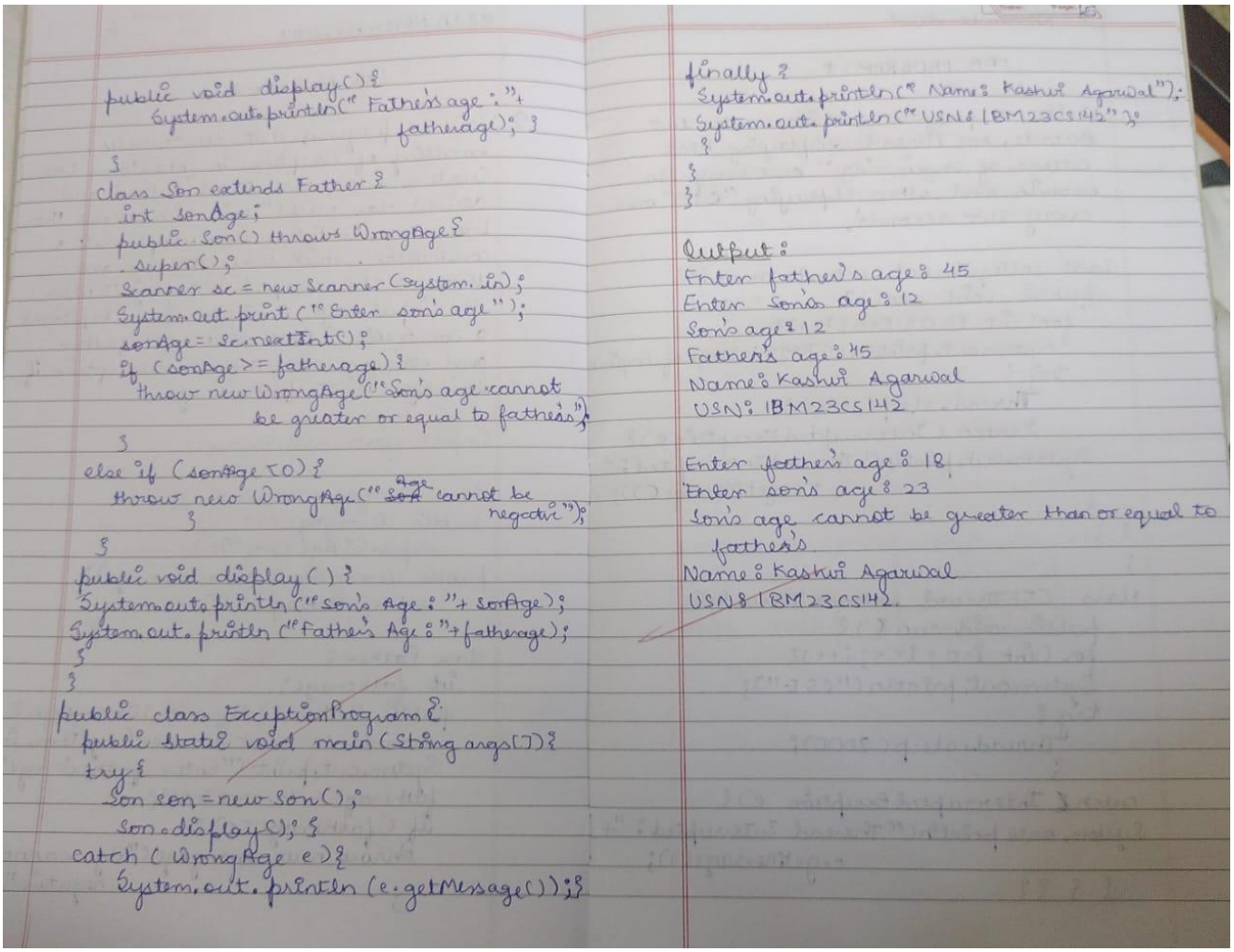
Program 7

Write a program that demonstrates handling of exceptions in inheritance trees. Create a base class called “Father” and a derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that causes both father and son’s age and throws an exception if son’s age is >=father’s age.

Algorithm:



```
import java.util.Scanner;

class WrongAge extends Exception {
 public WrongAge () {
    super ("Age Error"); }
public WrongAge (string message) {
    super (Message); }
 }
class Father {
  int fatherage;
  public Father () throws WrongAge {
   Scanner sc = new Scanner (System.in);
   System.out.print ("Enter father's age");
   fatherage = sc.nextInt();
   if (fatherage<0) {
    throw new WrongAge ("Age cannot be
                          negative"); } }
```



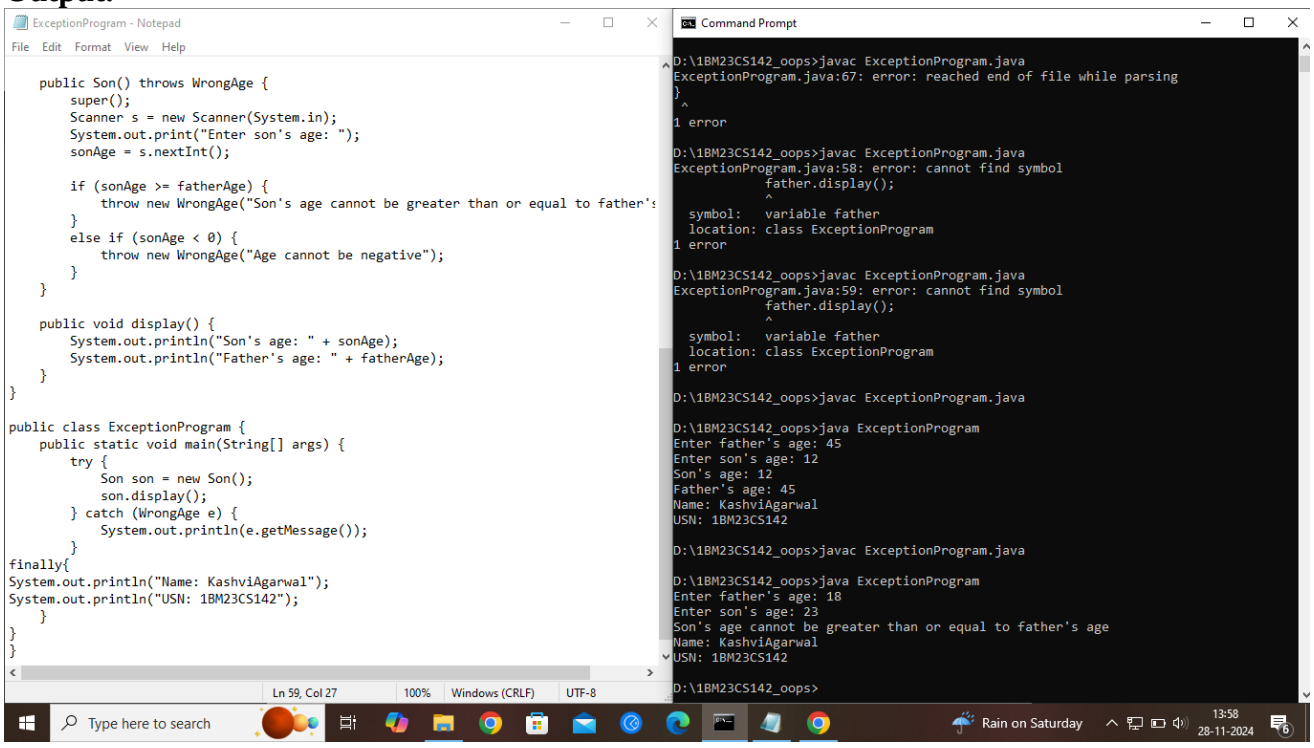
Code:

```
import java.util.Scanner;

class WrongAge extends Exception {

    public WrongAge() {
        super("Age Error");
    }

    public WrongAge(String message) {
        super(message);
    }
}

class Father {
    protected int fatherAge;

    public Father() throws WrongAge {
```

```
        Scanner s = new Scanner(System.in);
        System.out.print("Enter father's age: ");
        fatherAge = s.nextInt();

        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }

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

class Son extends Father {
    private int sonAge;

    public Son() throws WrongAge {
        super();
        Scanner s = new Scanner(System.in);
        System.out.print("Enter son's age: ");
        sonAge = s.nextInt();

        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to father's age");
        }
        else if (sonAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }

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

public class ExceptionProgram {
    public static void main(String[] args) {
        try {
            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println(e.getMessage());
        }
finally{
System.out.println("Name: KashviAgarwal");
```

```
System.out.println("USN: 1BM23CS142");
    }
}
}
```

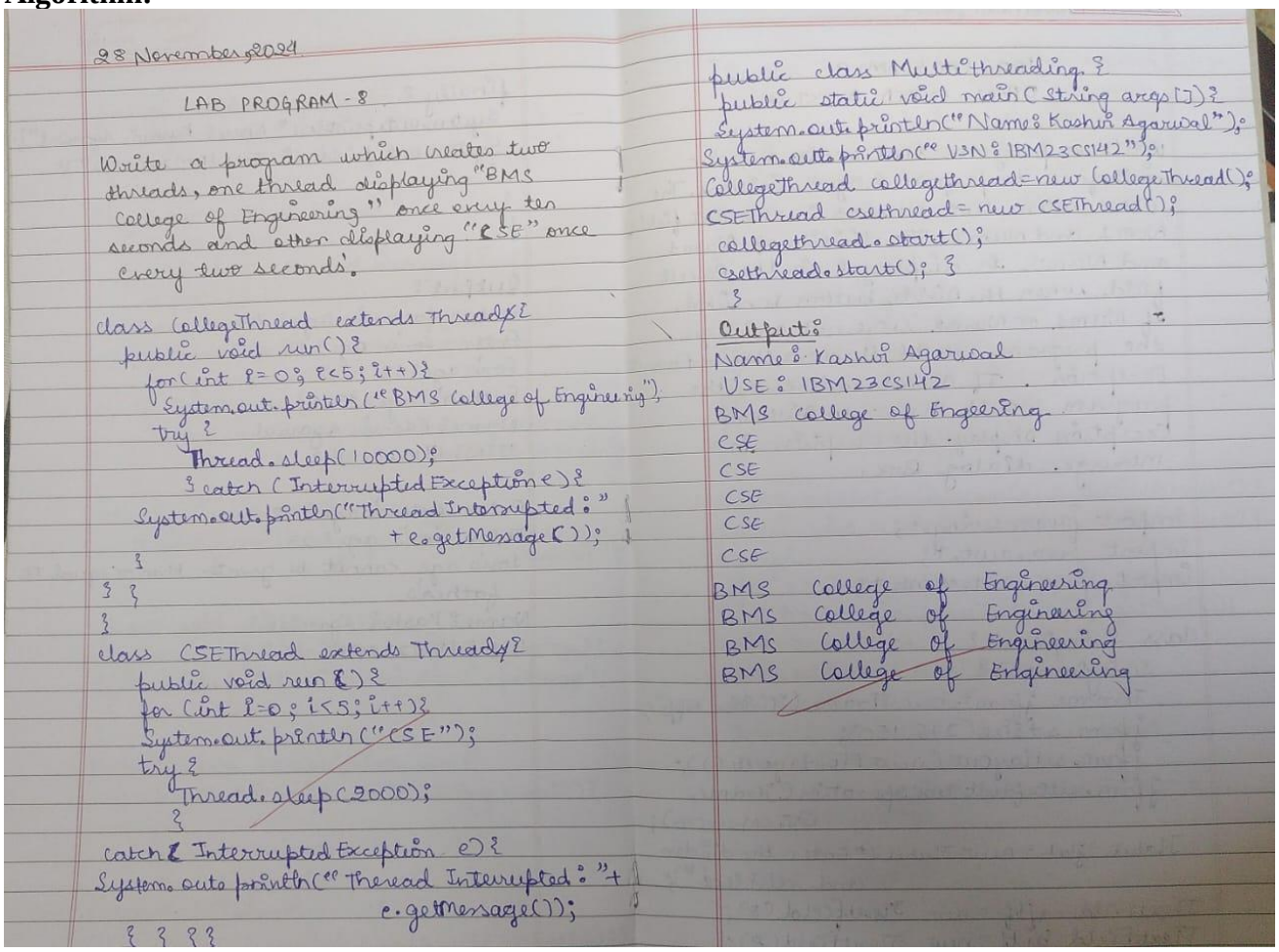
Output:



Program 8

Write a program which creates two threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds.

Algorithm:



28 November 2024

LAB PROGRAM - 8

Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and other displaying "CSE" once every two seconds.

```
class CollegeThread extends Thread{
  public void run(){
    for(int i=0; i<5; i++){
      System.out.println("BMS College of Engineering");
      try{
        Thread.sleep(10000);
      }catch (InterruptedException e){
        System.out.println("Thread Interrupted:"
                           +e.getMessage());
      }
  }}
}
class CSEThread extends Thread{
  public void run(){
    for(int i=0; i<5; i++){
      System.out.println("CSE");
      try{
        Thread.sleep(2000);
      }
    catch(InterruptedException e){
    System.out.println("Thread Interrupted:"+
                       e.getMessage());
    }}}
```

```
public class Multithreading{
  public static void main(String args[]){
  System.out.println("Name: Kashvi Agarwal");
  System.out.println("USN: 1BM23CS142");
  CollegeThread collegethread=new CollegeThread();
  CSEThread csethread=new CSEThread();
  collegethread.start();
  csethread.start();}
  }
```

Output:

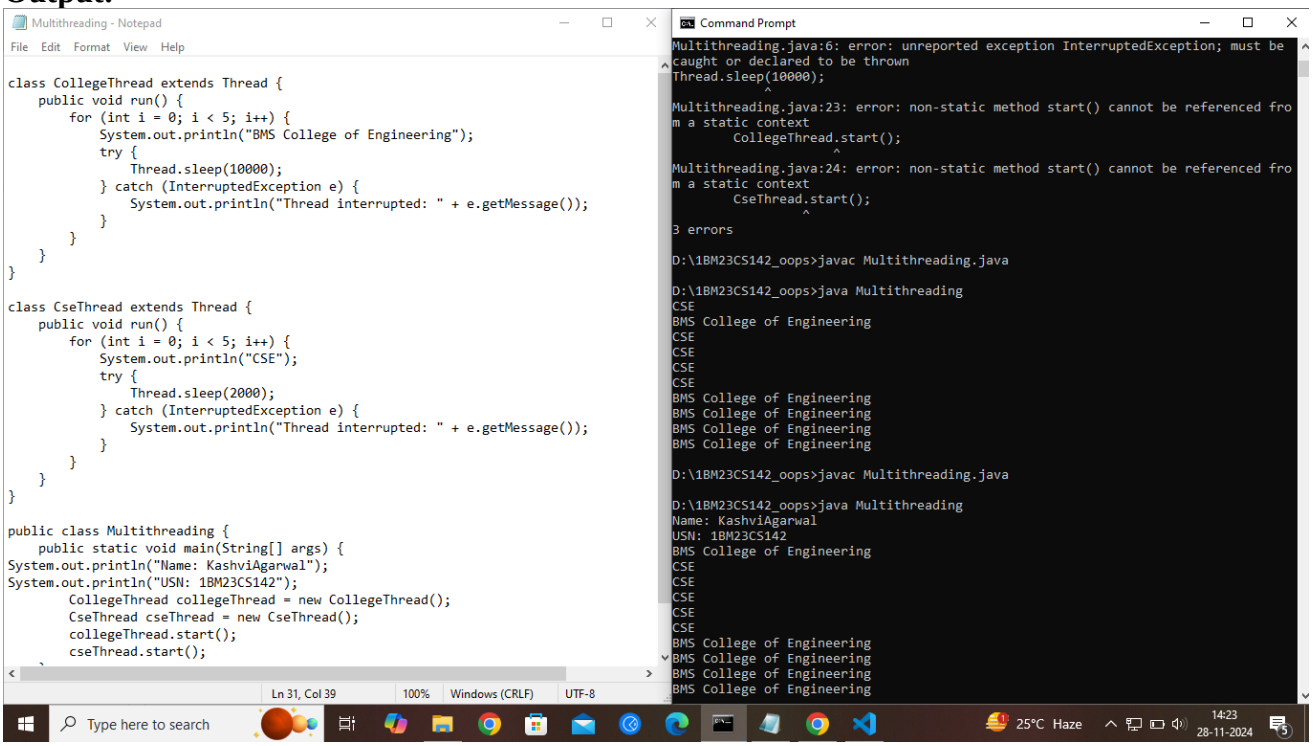
```
Name: Kashvi Agarwal
USE: 1BM23CS142
BMS College of Engeering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
```

Code:

```
class CollegeThread extends Thread {
    public void run() {
        for (int i = 0; i < 5; i++) {
            System.out.println("BMS College of Engineering");
            try {
                Thread.sleep(10000);
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted: " + e.getMessage());
            }
        }
    }
}
```

```
class CseThread extends Thread {
    public void run() {
        for (int i = 0; i < 5; i++) {
            System.out.println("CSE");
            try {
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted: " + e.getMessage());
            }
        }
    }
}

public class Multithreading {
    public static void main(String[] args) {
System.out.println("Name: KashviAgarwal");
System.out.println("USN: 1BM23CS142");
        CollegeThread collegeThread = new CollegeThread();
        CseThread cseThread = new CseThread();
        collegeThread.start();
        cseThread.start();
    }
}
```

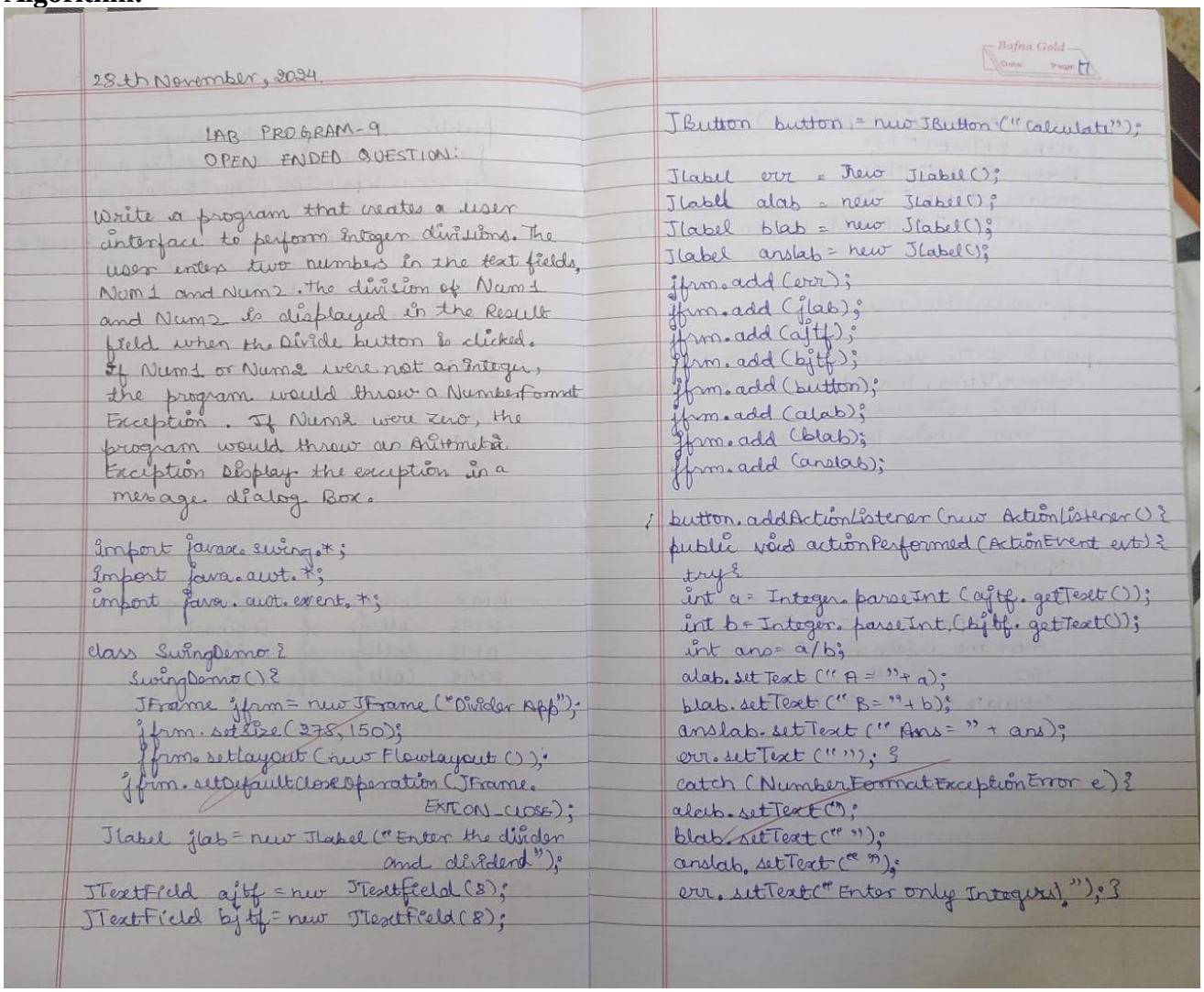
Output:



Program 9

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

Algorithm:



28th November, 2024.

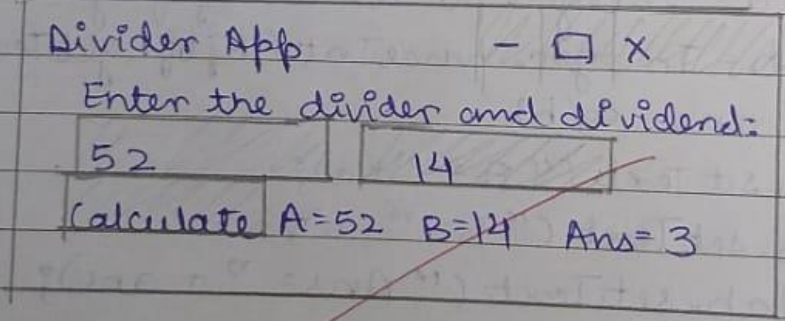
LAB PROGRAM-9
OPEN ENDED QUESTION:

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

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

class SwingDemo {
    SwingDemo() {
    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 dividon
                              and divident");
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 (NumberFormatExceptionError 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[]){
SwingUtilities.invokeLater(new Runnable(){
   public void run(){
      new SwingDemo();}
});
}
}
```

Output:

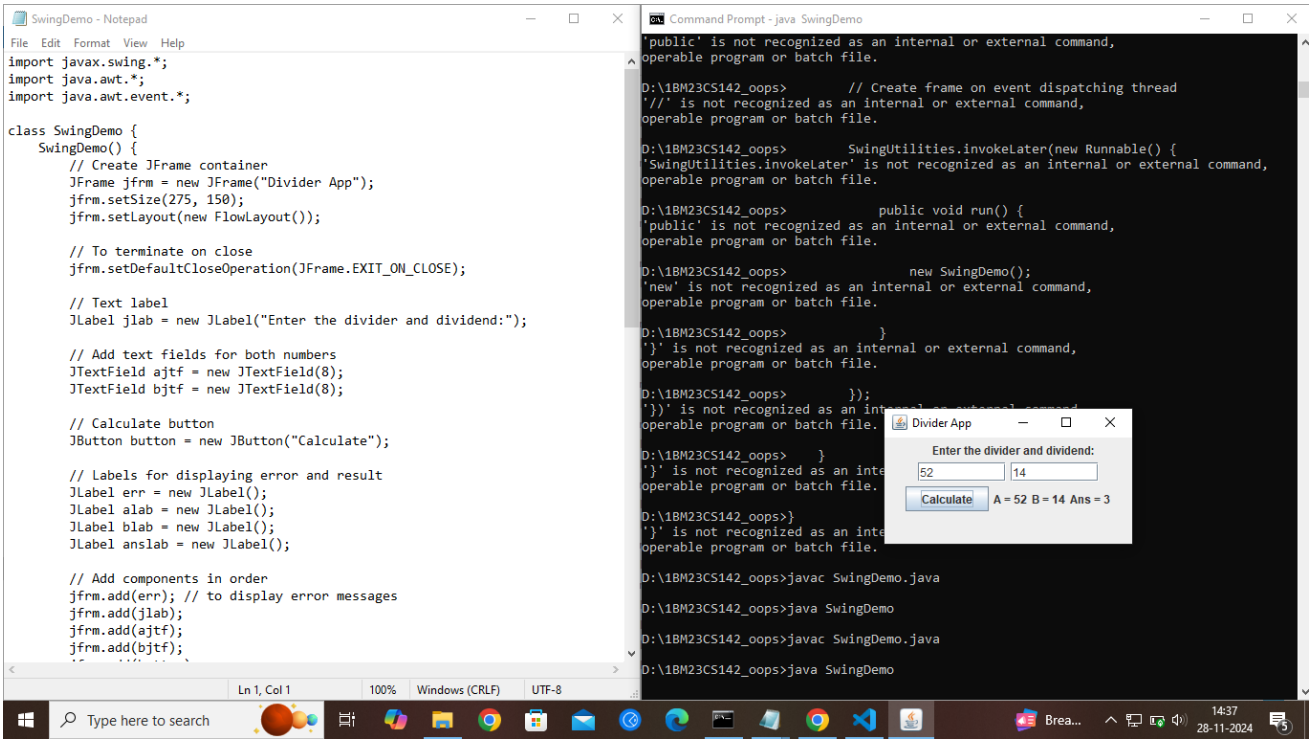


Code:

```
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 dividend:");

      // Add text fields for both numbers
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);

      // Calculate button
      JButton button = new JButton("Calculate");

      // Labels for displaying error and result
      JLabel err = new JLabel();
      JLabel alab = new JLabel();
      JLabel blab = new JLabel();
      JLabel anslab = new JLabel();

      // Add components in order
      jfrm.add(err); // to display error messages
      jfrm.add(jlab);
      jfrm.add(ajtf);
      jfrm.add(bjtf);
      jfrm.add(button);
      jfrm.add(alab);
      jfrm.add(blab);
      jfrm.add(anslab);

      // Button action listener
      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("");  // Clear any previous errors
            } 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[]) {

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

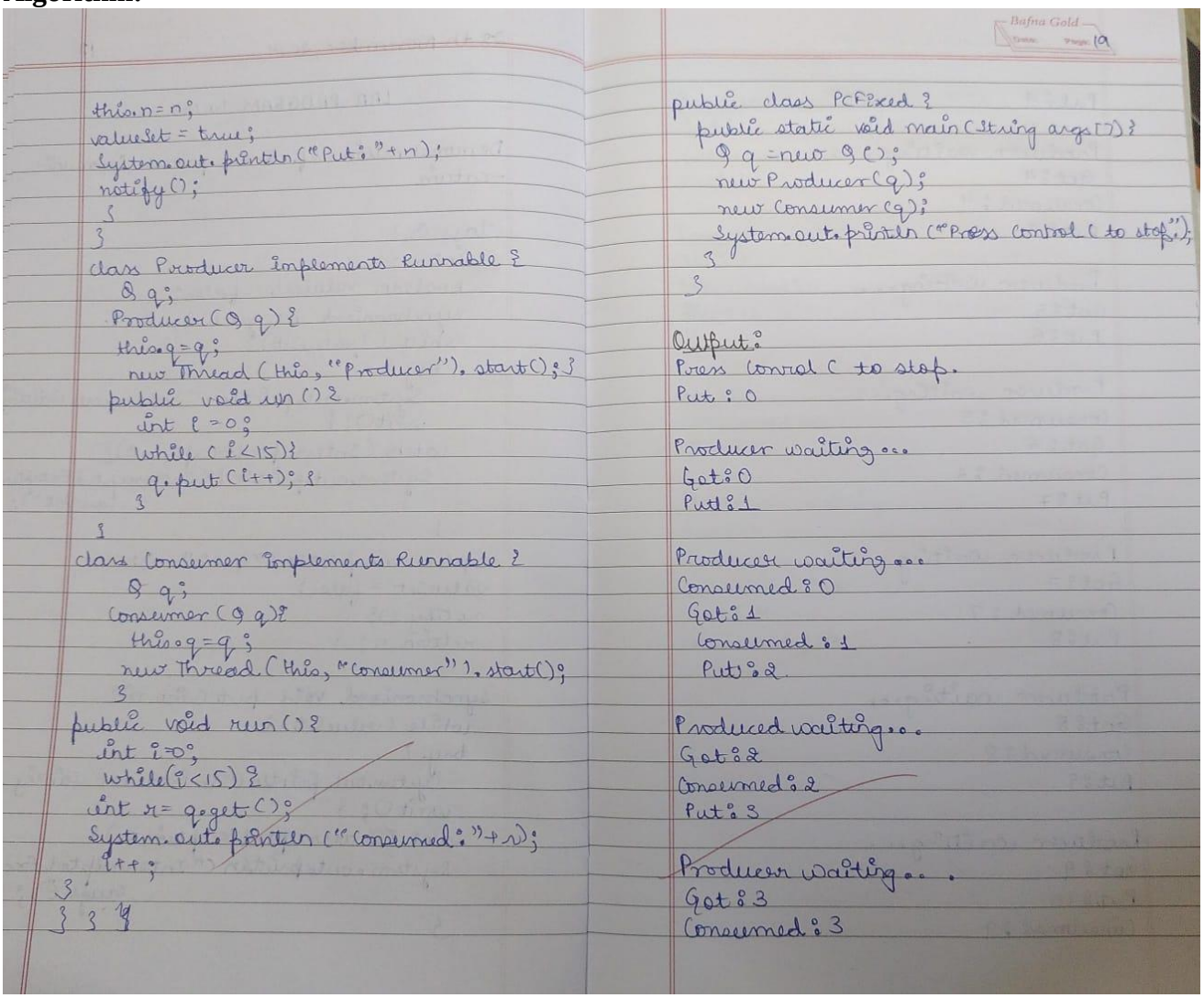
Output:


SwingDemo - Notepad
File Edit Format View Help
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 dividend:");
// Add text fields for both numbers
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
// Calculate button
JButton button = new JButton("Calculate");
// Labels for displaying error and result
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();
// Add components in order
jfrm.add(err); // to display error messages
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8
Command Prompt - java SwingDemo
'public' is not recognized as an internal or external command,
operable program or batch file.
D:\1BM23CS142_oops> // Create frame on event dispatching thread
'//' is not recognized as an internal or external command,
operable program or batch file.
D:\1BM23CS142_oops> SwingUtilities.invokeLater(new Runnable() {
'SwingUtilities.invokeLater' is not recognized as an internal or external command,
operable program or batch file.
D:\1BM23CS142_oops> public void run() {
'public' is not recognized as an internal or external command,
operable program or batch file.
D:\1BM23CS142_oops> new SwingDemo();
'new' is not recognized as an internal or external command,
operable program or batch file.
D:\1BM23CS142_oops> }
'}' is not recognized as an internal or external command,
operable program or batch file.
D:\1BM23CS142_oops> });
operable program or batch file.
D:\1BM23CS142_oops> }
operable program or batch file.
D:\1BM23CS142_oops>}
operable program or batch file.
D:\1BM23CS142_oops>javac SwingDemo.java
D:\1BM23CS142_oops>java SwingDemo
D:\1BM23CS142_oops>javac SwingDemo.java
D:\1BM23CS142_oops>java SwingDemo
Divider App
Enter the divider and dividend:
52
14
Calculate
A = 52 B = 14 Ans = 3
Type here to search
14:37
28-11-2024

Program 10(a)

Demonstrate Inter process Communication

Algorithm:

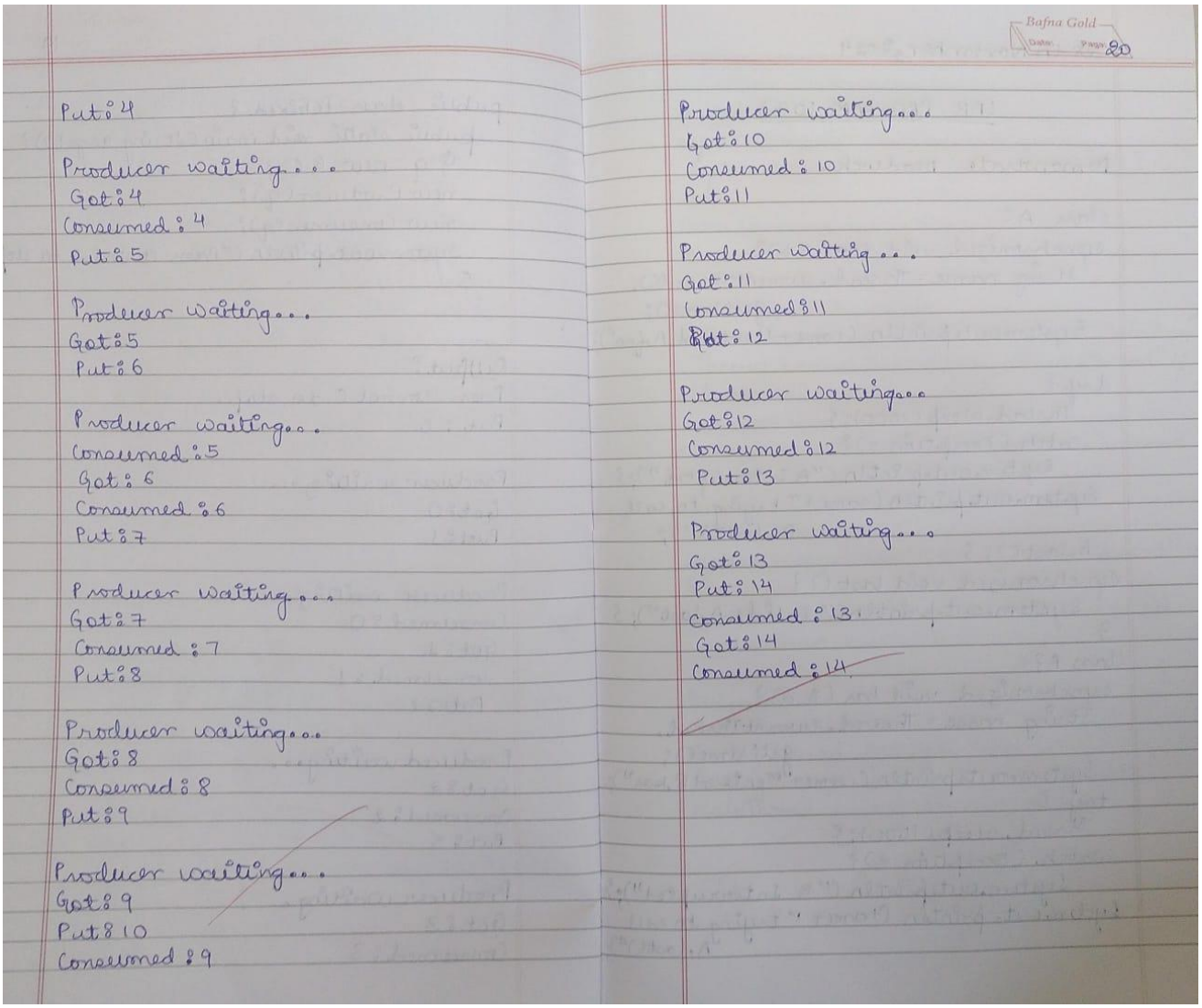


```
this.n=n;
valueSet = true;
System.out.println("Put: "+n);
notify();
}
}
class Producer implements Runnable {
Q q;
Producer(Q q){
this.q=q;
new Thread(this,"Producer").start();}
public void run(){
int i=0;
while(i<15){
q.put(i++);}
}
}
class Consumer implements Runnable{
Q q;
Consumer(Q q){
this.q=q;
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[]){
Q q=new Q();
new Producer(q);
new Consumer(q);
System.out.println("Press Control C to stop.");
}
}
```

Output:

```
Press Control C to stop.
Put: 0

Producer waiting...
Got: 0
Put: 1

Producer waiting...
Consumed: 0
Got: 1
Consumed: 1
Put: 2

Produced waiting...
Got: 2
Consumed: 2
Put: 3

Producer waiting...
Got: 3
Consumed: 3
```



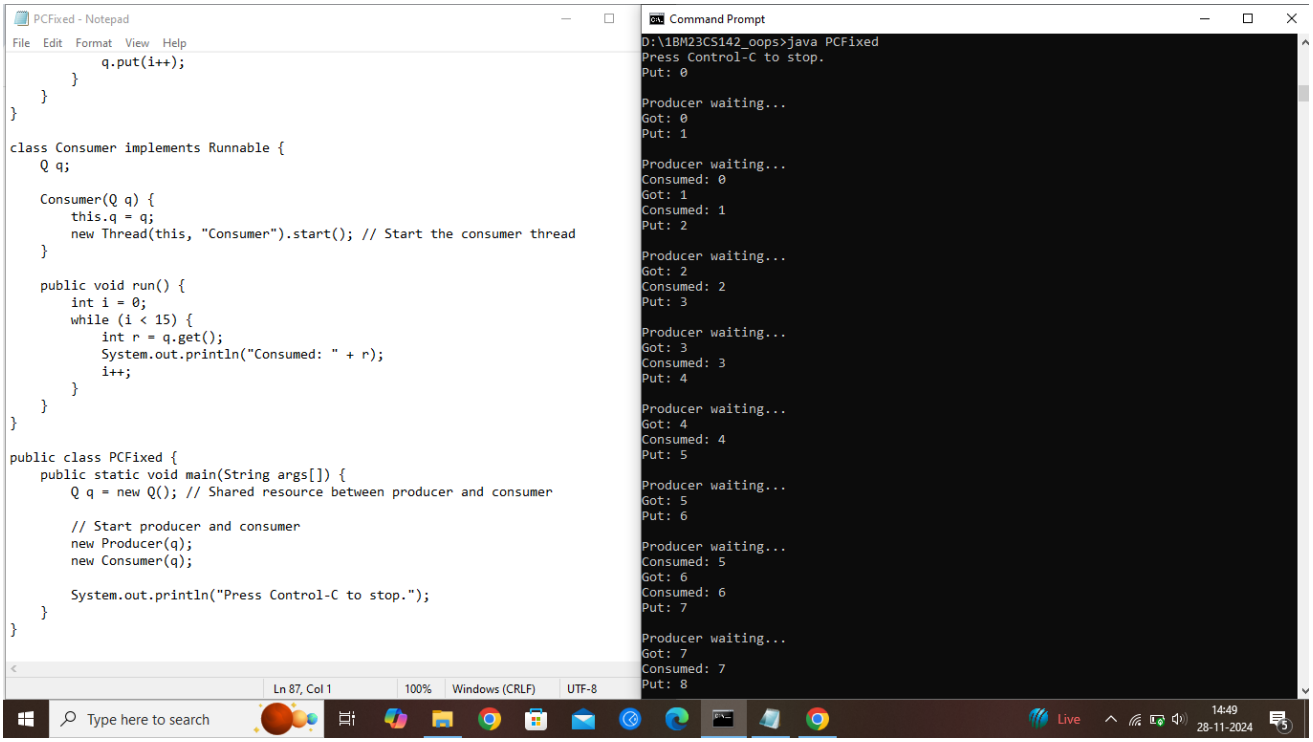
Code:

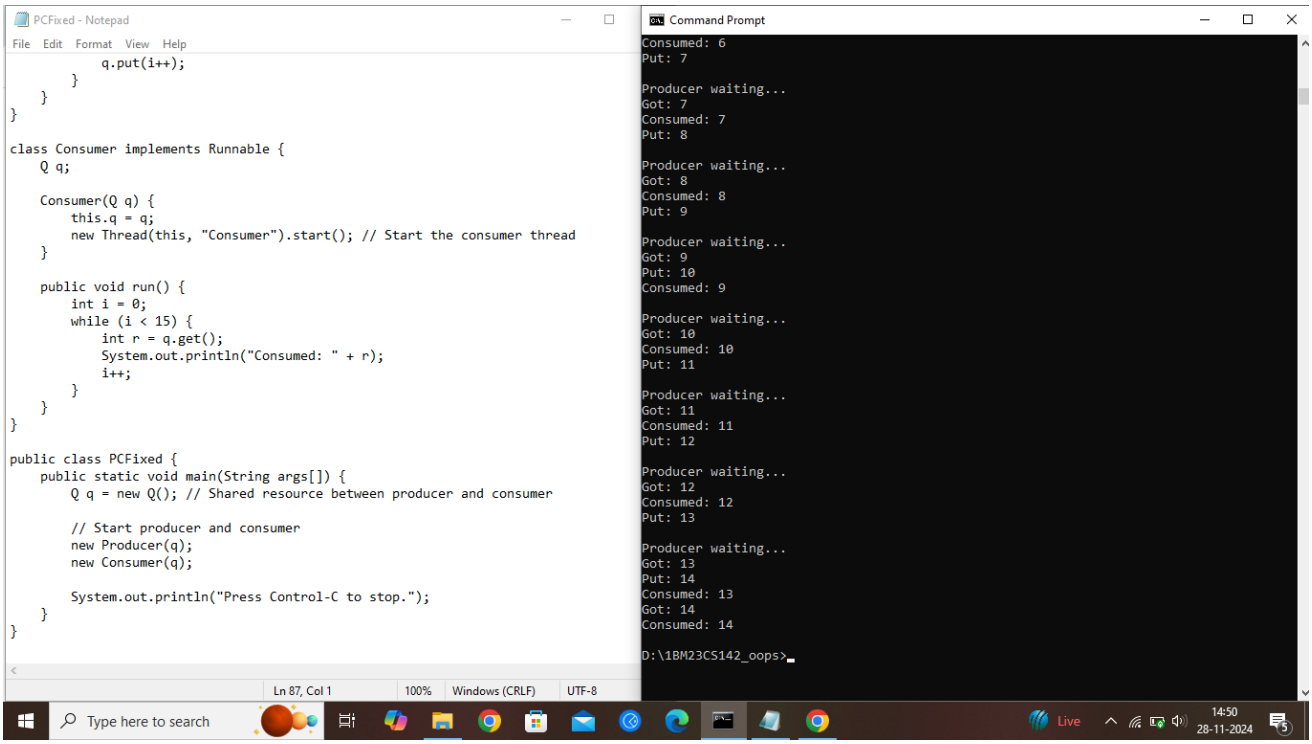
```
class Q {
    int n;
    boolean valueSet = false;

    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("\nConsumer waiting...");
                wait();  // Wait until the producer has produced an item.
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        }
        // Consume the item
        System.out.println("Got: " + n);
```

```
        valueSet = false;

        // Notify the producer that the consumer is done consuming
        notify();
        return n;
    }

    synchronized void put(int n) {
        while (valueSet) {
            try {
                System.out.println("\nProducer waiting...");
                wait();  // Wait until the consumer has consumed the item.
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        }
        // Produce the item
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);

        // Notify the consumer that the item is produced
        notify();
    }
}

class Producer implements Runnable {
    Q q;

    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start(); // Start the producer thread
    }

    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(); // Start the consumer thread
    }

    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[]) {
        Q q = new Q(); // Shared resource between producer and consumer

        // Start producer and consumer
        new Producer(q);
        new Consumer(q);

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

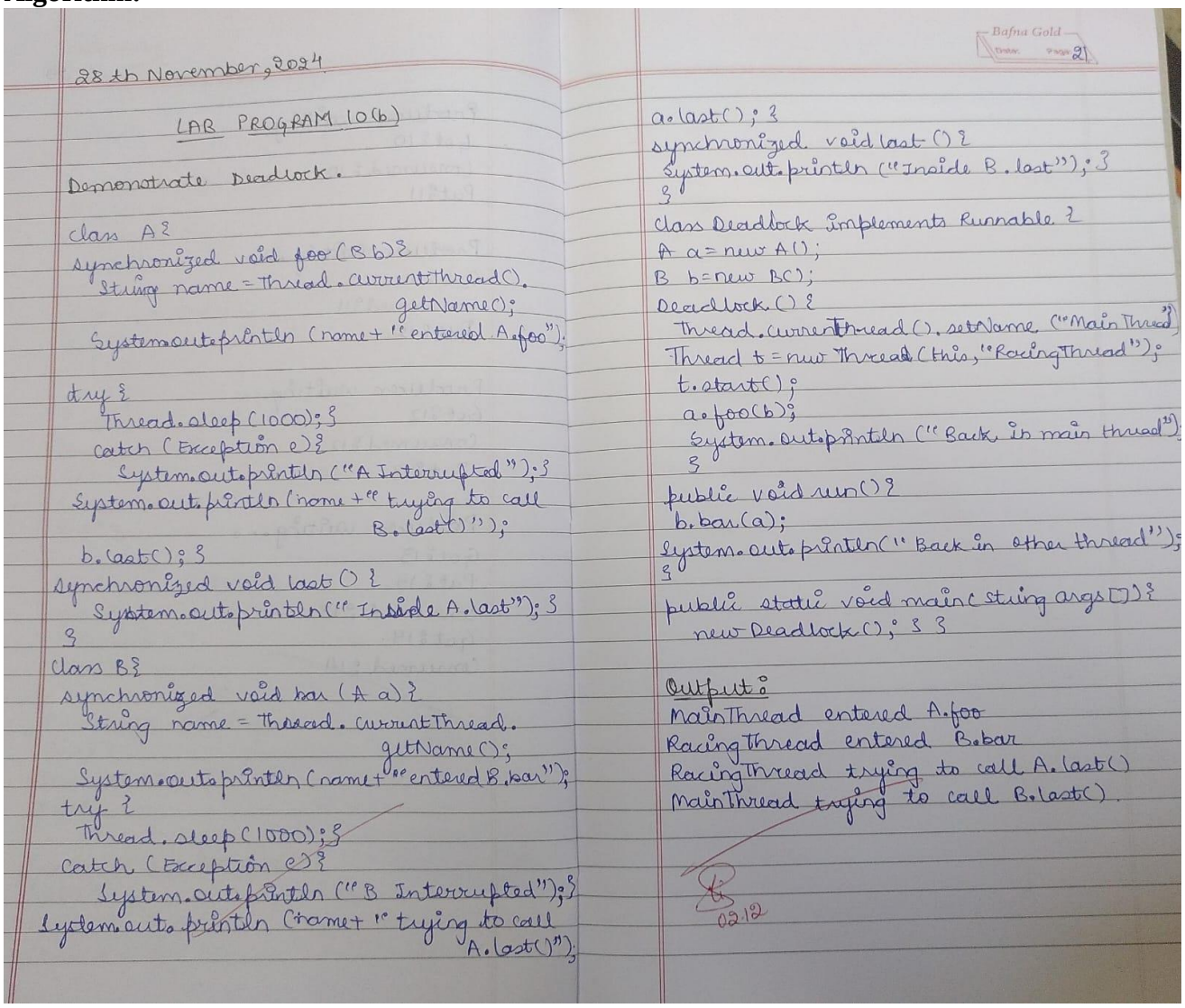
Output:




PCFixed - Notepad
File Edit Format View Help
q.put(i++);
}
}
}
class Consumer implements Runnable {
Q q;
Consumer(Q q) {
this.q = q;
new Thread(this, "Consumer").start(); // Start the consumer thread
}
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[]) {
Q q = new Q(); // Shared resource between producer and consumer
// Start producer and consumer
new Producer(q);
new Consumer(q);
System.out.println("Press Control-C to stop.");
}
}
Ln 87, Col 1
100%
Windows (CRLF)
UTF-8
Command Prompt
Consumed: 6
Put: 7
Producer waiting...
Got: 7
Consumed: 7
Put: 8
Producer waiting...
Got: 8
Consumed: 8
Put: 9
Producer waiting...
Got: 9
Put: 10
Consumed: 9
Producer waiting...
Got: 10
Consumed: 10
Put: 11
Producer waiting...
Got: 11
Consumed: 11
Put: 12
Producer waiting...
Got: 12
Consumed: 12
Put: 13
Producer waiting...
Got: 13
Put: 14
Consumed: 13
Got: 14
Consumed: 14
D:\1BM23CS142_oops>
Type here to search
Live
14:50
28-11-2024

Program 10(b)
Demonstrate Deadlock

Algorithm:



28th November, 2024

LAB PROGRAM 10(b)

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

Output:

```
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()
```

02.12

Code:

```
class A {

   synchronized void foo(B b) {
      String name = Thread.currentThread().getName();
      System.out.println(name + " entered A.foo");

      try {
         Thread.sleep(1000);
      } catch (Exception e) {
         System.out.println("A Interrupted");
      }
```

```
        System.out.println(name + " trying to call B.last()");
        b.last();
    }

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

class B {

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

        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println("B Interrupted");
        }

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

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

class Deadlock implements Runnable {

    A a = new A();
    B b = new B();

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

        a.foo(b); // MainThread gets lock on A
        System.out.println("Back in main thread");
    }

    public void run() {
        b.bar(a); // RacingThread gets lock on B
```

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

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

Output: