VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object-Oriented Java Programming

(23CS3PCOOJ)

Submitted by

KARAN SURESH NAIR(1BM23CS139)

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 **KARAN SURESH NAIR(1BM23CS139),** 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.

Seema Patil Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

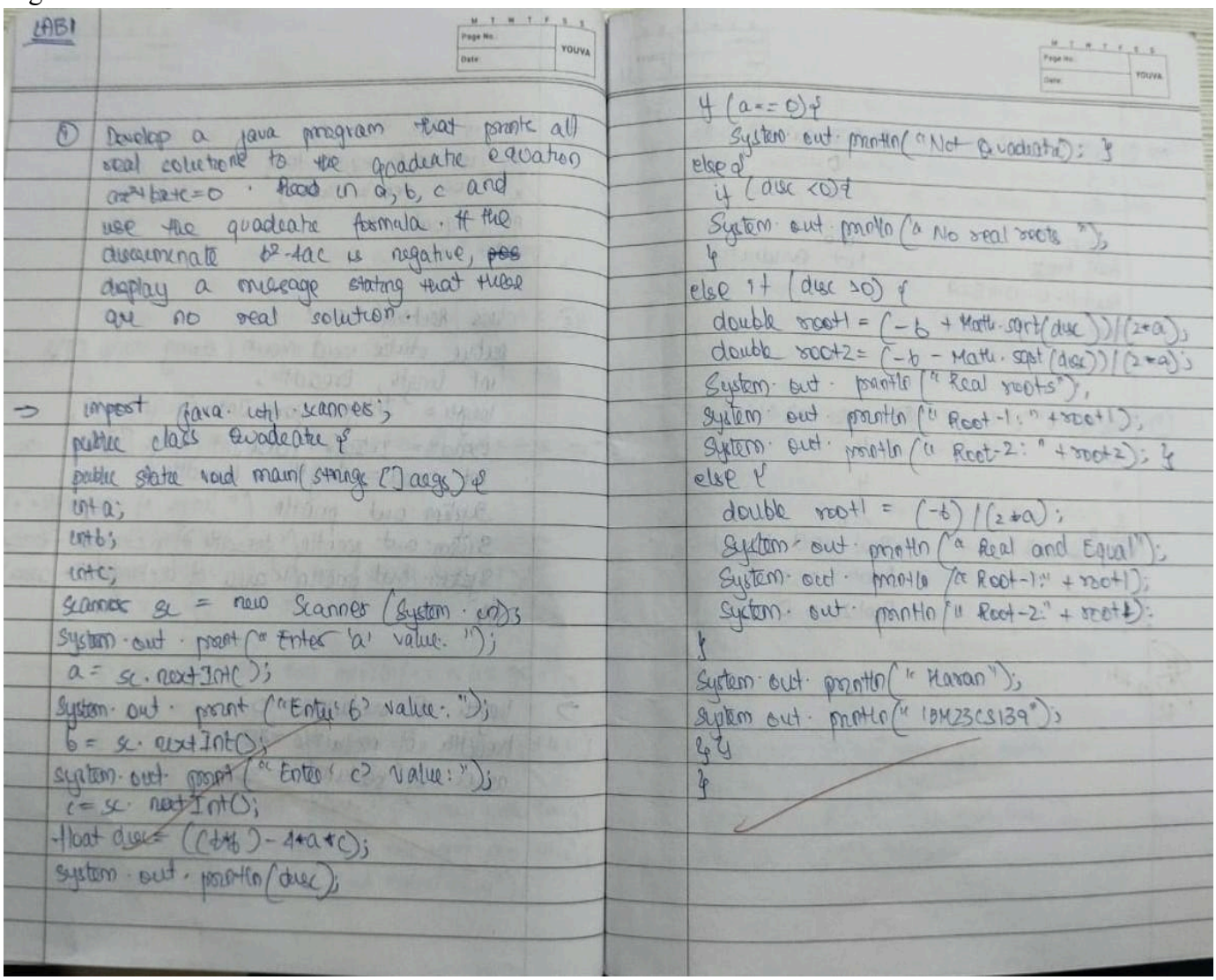
Index

Sl. No.	Date	Experiment Title	Page No.
1	30-9-24	Quadratic Equation	4
2	07-10-24	SGPA Calculator	8
3	14-10-24	Book Class	14
4	21-10-24	Abstraction	18
5	28-10-24	Inheritance	22
6	11-11-24	Package	31
7	02-12-24	Exception Handling	38
8	02-12-24	GUI	43
9	02-12-24	MultiThreading	47
10	02-12-24	Inter-Process Communication & Deadlock	51

GitHub Link:
https://github.com/KaranSureshNair/OOJ-Lab-139/tree/main

Program 1
Quadratic Equation

Algorithm:



Page No.
Date: YOUVA

⇒ ① Enter 'a', 'b', 'c':
3
8
1
52.0
Real Roots
Root-1:- 0.1314829
Root-2:- 2.5351837

② Enter 'a', 'b', 'c':
0
2
4
4.0
Not quadratic

③ Enter 'a', 'b', 'c':
10
4
2
-64.0
No real roots
1

④ Enter 'a', 'b', 'c':
4
4
1
0.0
Real and Equal
Root1: 0.0
Root-2: 0.0

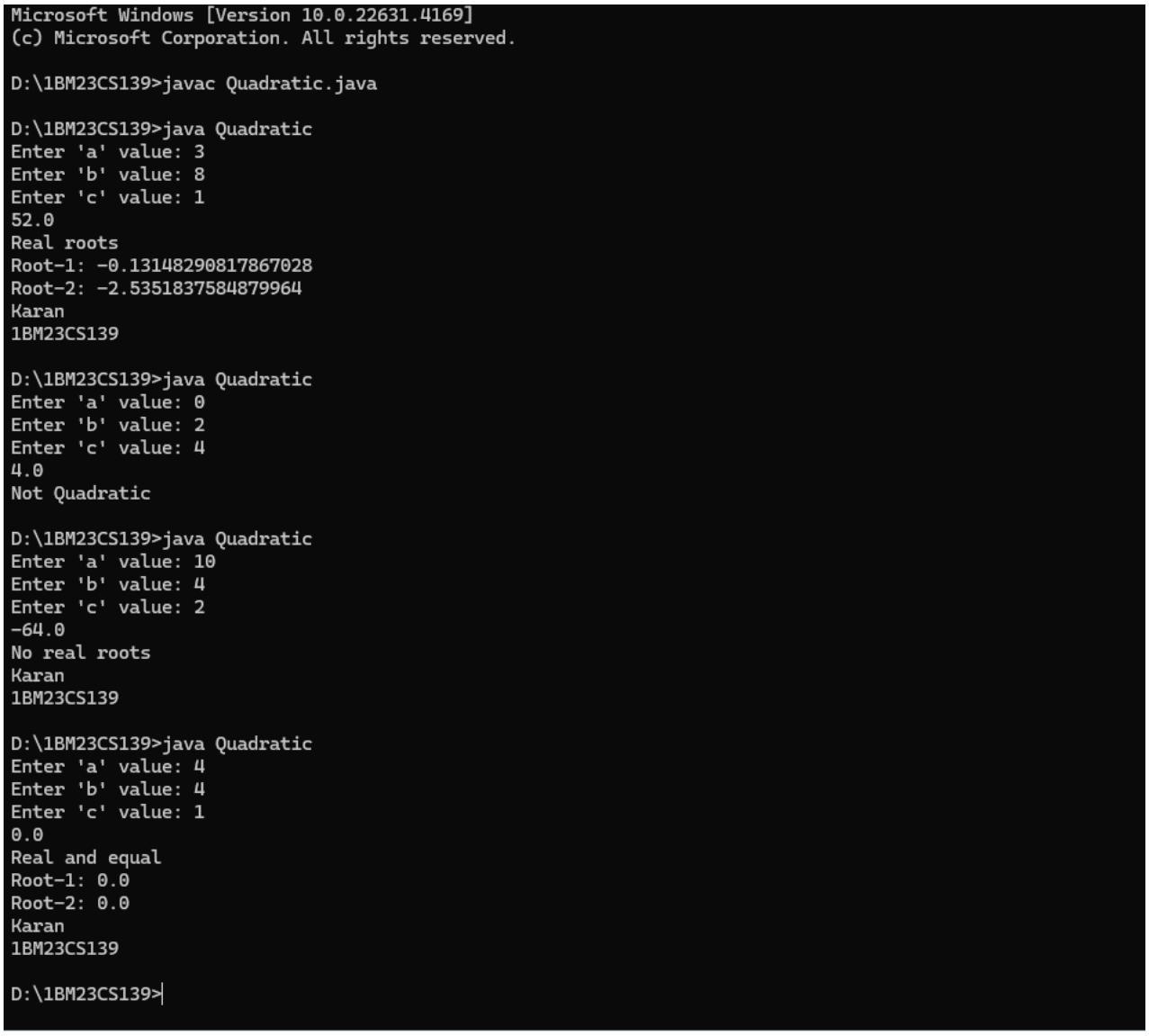
01.10.24

Code:

```
import java.util.Scanner;

public class Quadratic {

    public static void main(String[] args) {
        int a;
        int b;
        int c;

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter 'a' value: ");
        a = sc.nextInt();

        System.out.print("Enter 'b' value: ");
        b = sc.nextInt();

        System.out.print("Enter 'c' value: ");
        c = sc.nextInt();

        float disc = ((b * b) - 4 * a * c);
        System.out.println(disc);

        if (a == 0) {
            System.out.println("Not Quadratic");
        } else {
            if (disc < 0) {
                System.out.println("No real roots");
            } else if (disc > 0) {
                double root1 = (-b + Math.sqrt(disc)) / (2 * a);
                double root2 = (-b - Math.sqrt(disc)) / (2 * a);

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

                System.out.println("Real and equal");
                System.out.println("Root-1: " + root1);
                System.out.println("Root-2: " + root1);
            }

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

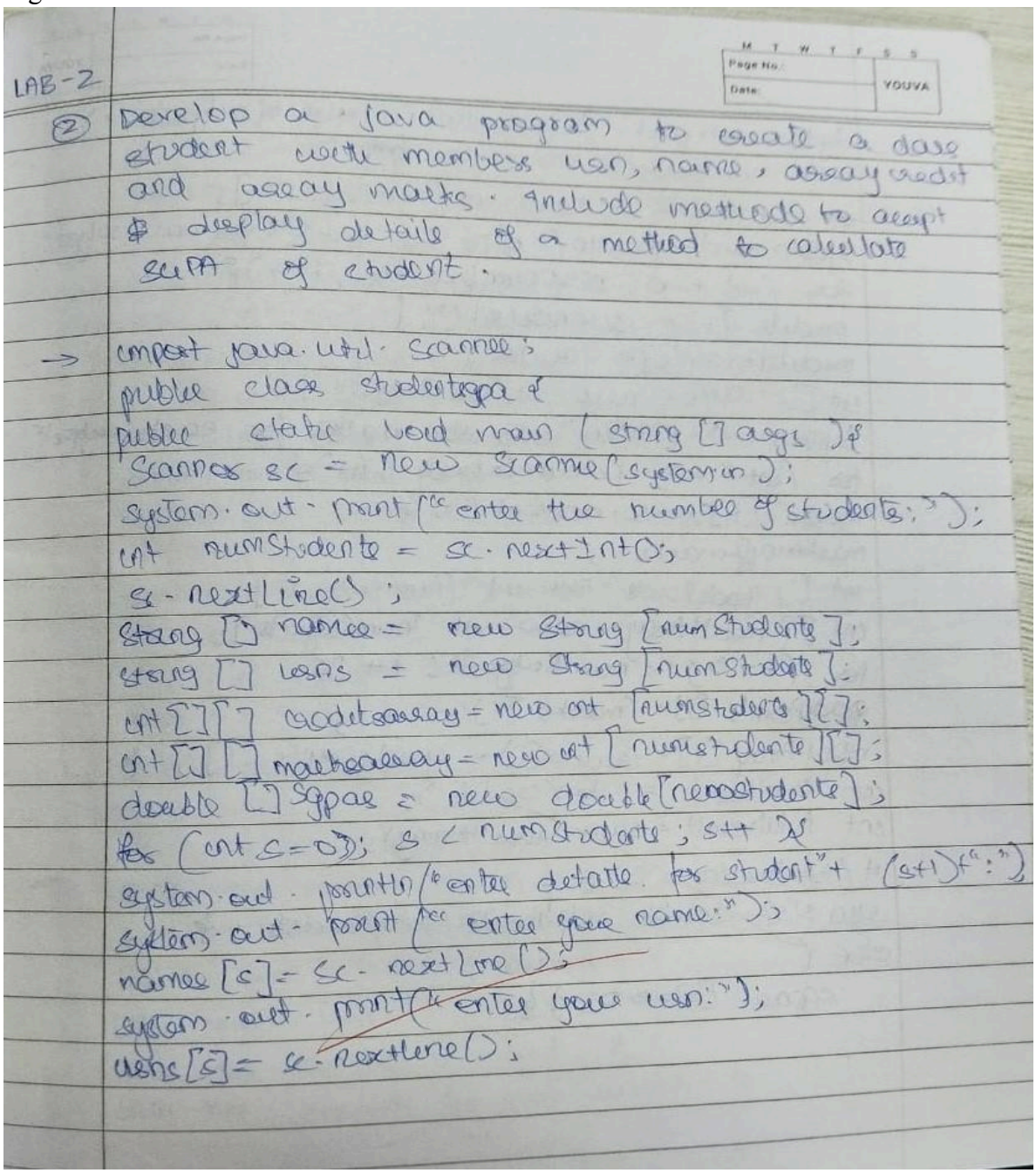
```
            System.out.println("1BM23CS139");
        }

        sc.close(); // Closing the scanner
    }
}
```

Output:



Program 2
SGPA Calculator

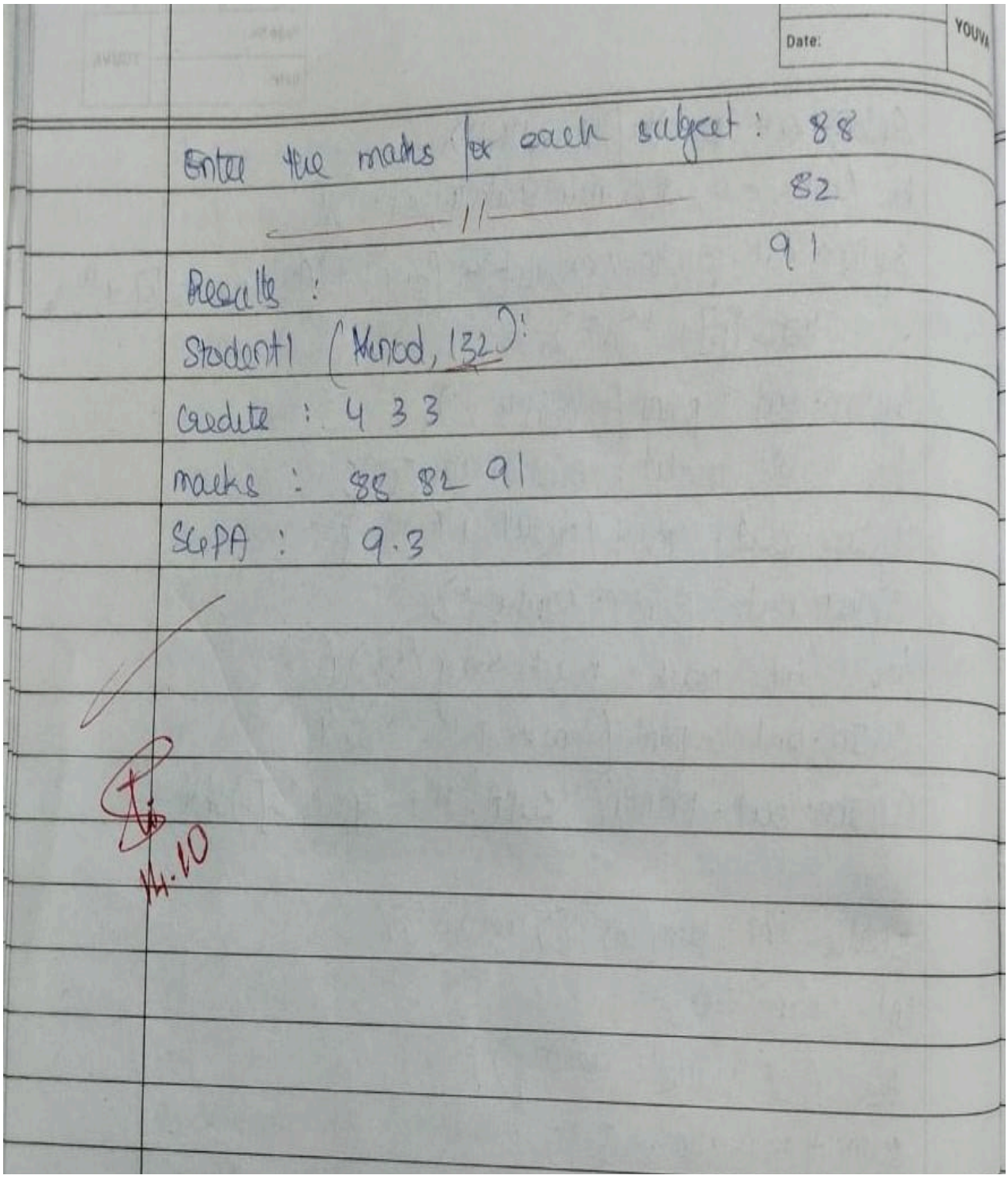
Algorithm:



```
System.out.print("enter the number of subjects:");
int numSubjects = sc.nextInt();
int[] credits = new int[numSubjects];
System.out.println("enter the credits for each subject");
for(int i=0; i<numSubjects; i++){
credits[i] = sc.nextInt();}
creditsArray[s] = credits;
int[] marks = new int[numSubjects];
System.out.println("enter the marks for each subject");
for(int i=0; i<numSubjects; i++){
marks[i] = sc.nextInt();}
marksArray[s] = marks;
int[] gradePoints = new int[numSubjects];
int[] resultArray = new int[numSubjects];
for(i=0; i<numSubjects; i++){
gradePoints[i] = (marks[i]/10)+1;
resultArray[i] = credits[i] * gradePoints[i];}
int totalCredits = sum(credits);
int totalResult = sum(resultArray);
if(totalCredits != 0){
sgpa[s] = double TotalResult / totalCredits;}
else{
sgpas[s] = 0.0;}}
```

```
System.out.println("Results");
for(int s=0; s<numStudents; s++){
System.out.println("Student"+(s+1)+"("+names[s]+"):");
System.out.print("Credits:");
for(int credit: creditsArray[s]){
System.out.print(credit+" ");}
System.out.print("Marks:");
for(int mark: marksArray[s]){
System.out.print(mark+" ");}
System.out.println("SGPA:"+sgpas[s]);}}

static int sum(int[] array){
int sum=0;
for(int value: array){
sum += value;}
return sum;}}
```

```
Enter the number of student: 1
Enter details of student 1:
Enter your name: vinod
Enter your usn: 132
Enter the number of subject: 3
Enter the credits for each subject: 4
3
3
```


Date:
YOUVA
Enter the marks for each subject: 88
82
91
Results :
Student (Murad, 132):
Credits : 4 3 3
marks : 88 82 91
SGPA : 9.3
14.10

Code:

```
import java.util.Scanner;

public class StudentsGPA {

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

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

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

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

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

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

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

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

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

            int[] gradePoints = new int[numSubjects];
            int[] resultArray = new int[numSubjects];
```

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

            int totalCredits = sum(credits);
            int totalResult = sum(resultArray);

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

        System.out.println("\n--- Results ---");
        for (int s = 0; s < numStudents; s++) {
            System.out.println("Student " + (s + 1) + " (" + names[s] + ", " + usns[s] + "):");
            System.out.print("Credits: ");
            for (int credit : creditsArray[s]) {
                System.out.print(credit + " ");
            }
            System.out.println();

            System.out.print("Marks: ");
            for (int mark : marksArray[s]) {
                System.out.print(mark + " ");
            }
            System.out.println();

            System.out.println("SGPA: " + sgpas[s]);
            System.out.println();
        }

        sc.close(); // Close the scanner
    }

    static int sum(int[] array) {
        int sum = 0;
        for (int value : array) {
            sum += value;
        }
        return sum;
    }
}
```

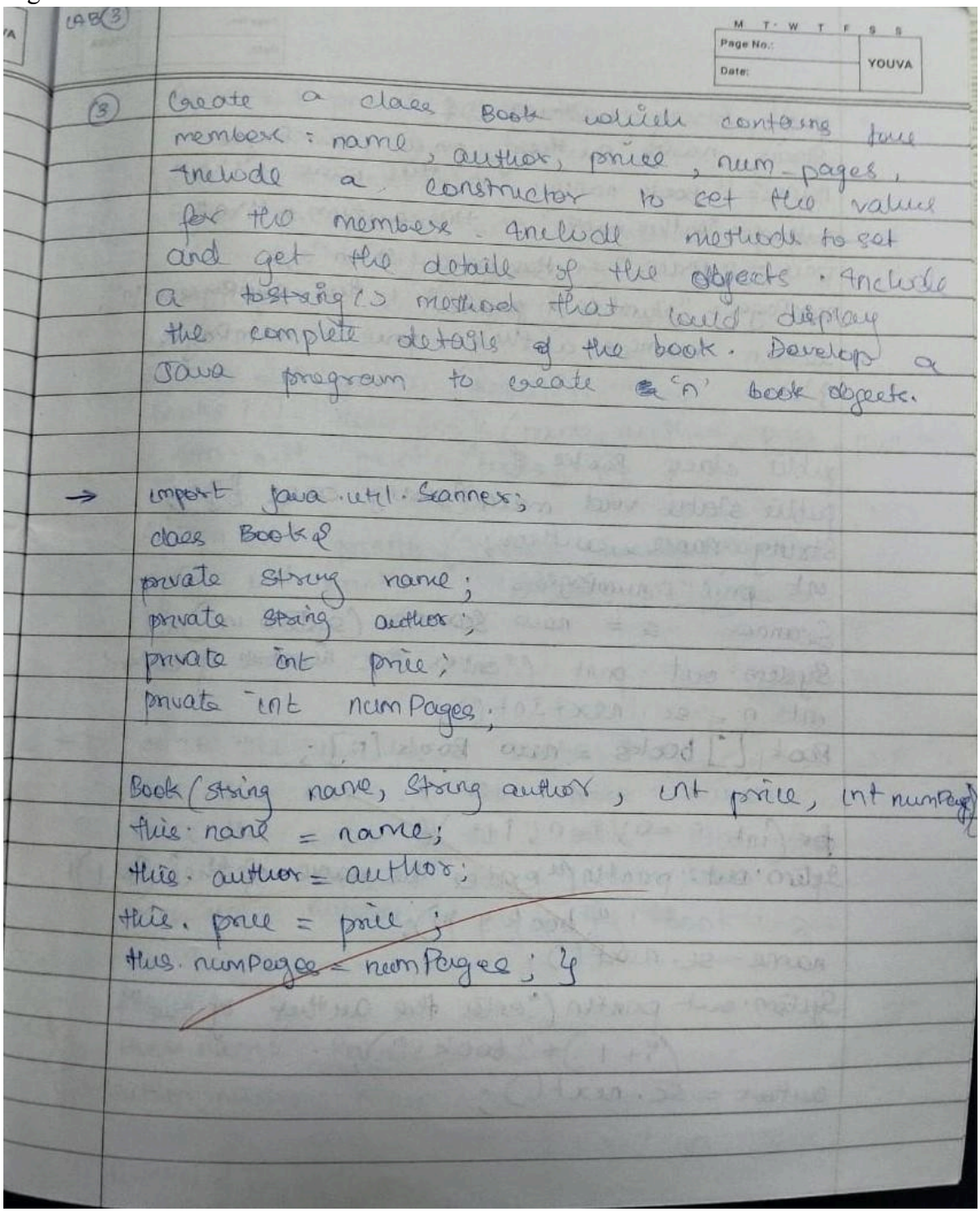
Output:

```
C:\Users\Karan\OneDrive\Desktop\JAVA COLLEGE> c: && cd "c:\Users\Karan\OneDrive\Desktop\JAVA COLLEGE" && cmd /C "C:\Users\Karan\AppData\Roami
Code\User\globalStorage\pleiades.java-extension-pack-jdk\java\latest\bin\java.exe --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -c
C:\Users\Karan\AppData\Roaming\Code\User\workspaceStorage\593b80fa01ff5fca0a9381dff8d5acd8\redhat.java\jdt_ws\JAVA COLLEGE_b6c8edf2\bin" stud
sgpa "
Enter the number of students: 2
Enter details for student 1:
Enter your name: karan
Enter your USN: 139
Enter the number of subjects: 3
Enter the credits for each subject:
4
3
3
Enter the marks for each subject out of 100:
88
82
91
Enter details for student 2:
Enter your name: Enter your USN: vinod 132
Enter the number of subjects: 3
```

```
Enter details for student 2:
Enter your name: Enter your USN: vinod 132
Enter the number of subjects: 3
Enter the credits for each subject:
4
3
3
Enter the marks for each subject out of 100:
78
98
73

--- Results ---
Student 1 (karan, 139):
Credits: 4 3 3
Marks: 88 82 91
SGPA: 9.3

Student 2 (, vinod 132):
Credits: 4 3 3
Marks: 78 98 73
SGPA: 8.6
```

Program 3
Book Class

Algorithm:


LAB 3

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.

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

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

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

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

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

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

System.out.println("Kasan Suresh Nair");
System.out.println("1BM23CS130");
}}
```

```
-> enter the number of books: 1
enter the name of the 1 book: Kasan
enter the author of the 1 book: Nair
enter the price of the 1 book: 1000
enter the number of pages of 1 book: 2
Result:
Book name: Kasan
Author name: Nair
Price: 1000
Number of pages: 2
```

Code:

```
import java.util.Scanner;

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

    // Constructor to initialize Book attributes
    Book(String name, String author, int price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    // Overriding toString method to return book details
    public String toString() {
        String name, author, price, numPages;
        name = "Book name: " + this.name + "\n";
        author = "Author name: " + this.author + "\n";
        price = "Price: " + this.price + "\n";
        numPages = "Number of pages: " + this.numPages + "\n";
        return name + author + price + numPages;
    }
}

public class Books {
    public static void main(String args[]) {
        String name, author;
        int price, numPages;
        Scanner sc = new Scanner(System.in);

        // Input number of books
        System.out.print("Enter the number of books: ");
        int n = sc.nextInt();
        Book[] books = new Book[n];

        // Input details for each book
        for (int i = 0; i < n; i++) {
            System.out.println("Enter the details for Book " + (i + 1) + ":");

            System.out.print("Enter the name of the book: ");
            name = sc.next(); // Using next() for single-word names; use nextLine() if names may have
spaces
```

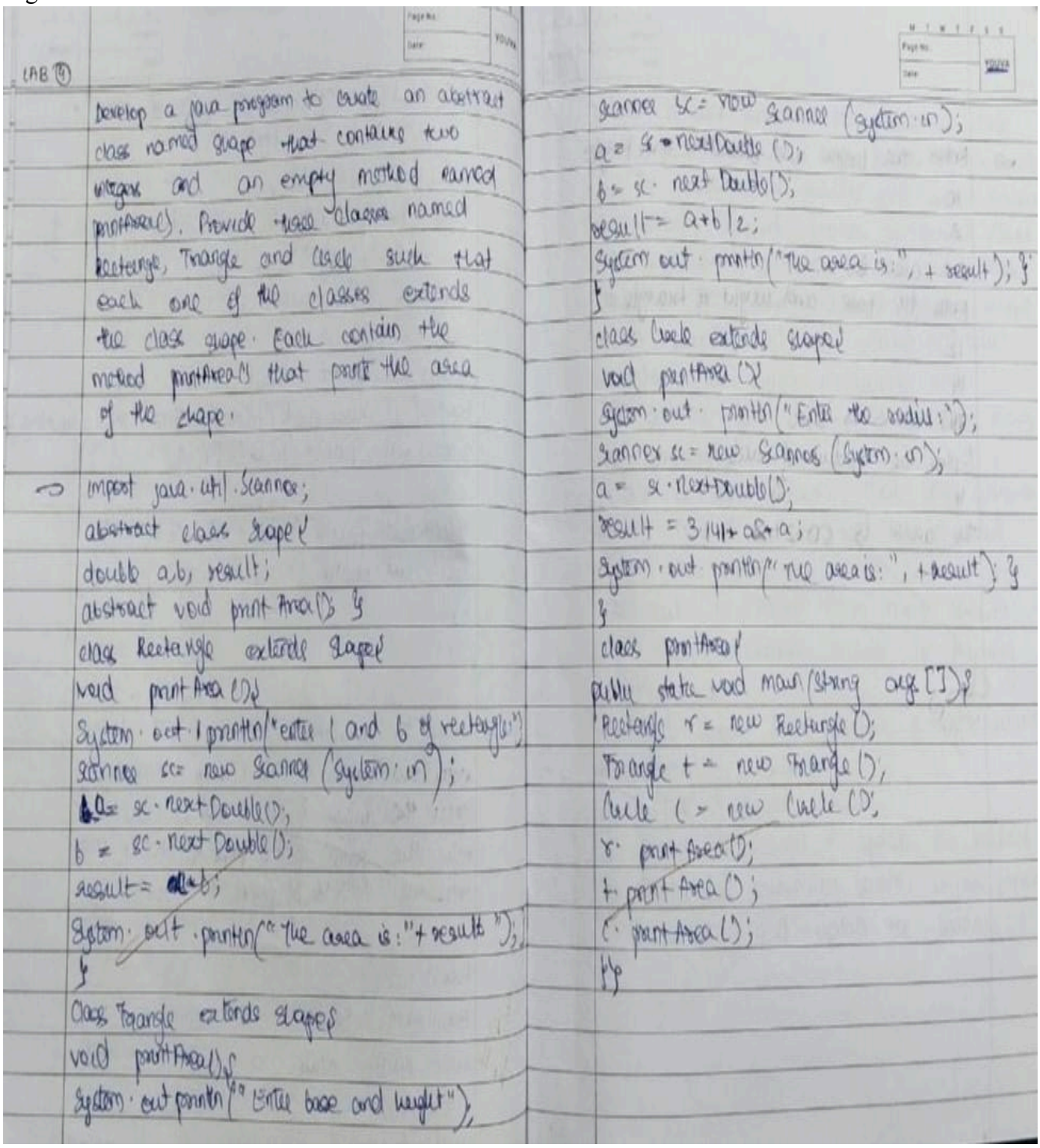
```
            System.out.print("Enter the author of the book: ");
            author = sc.next(); // Similar adjustment if multi-word input is expected

            System.out.print("Enter the price of the book: ");
            price = sc.nextInt();

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

            books[i] = new Book(name, author, price, numPages); // Create and store Book object

            System.out.println("\nResult:");
            System.out.println(books[i]); // Print book details
        }

        sc.close(); // Close the scanner
    }
}
```

Output:

```
D:\1BM23CS139>java Books
enter the number of books:2
enter the name of the 1 Book:
Karan
enter the author of the 1 book:
Nair
enter the price of the :1 book:
1000
enter the number of pages of the 1 Book:
2
Result
Book name: Karan
Author name:Nair
Price: 1000
Number of pages: 2

enter the name of the 2 Book:
Adiga
enter the author of the 2 book:
Raghavendra
enter the price of the :2 book:
10000
enter the number of pages of the 2 Book:
1
Result
Book name: Adiga
Author name:Raghavendra
Price: 10000
Number of pages: 1
```

Program 4
Abstraction

Algorithm:



Enter the length and breadth of rectangle:
10
2
The area is 20.0
Enter the base and height of triangle :
12
14
The area is 84.0
Enter the radius of circle:
4
The area is 50.272

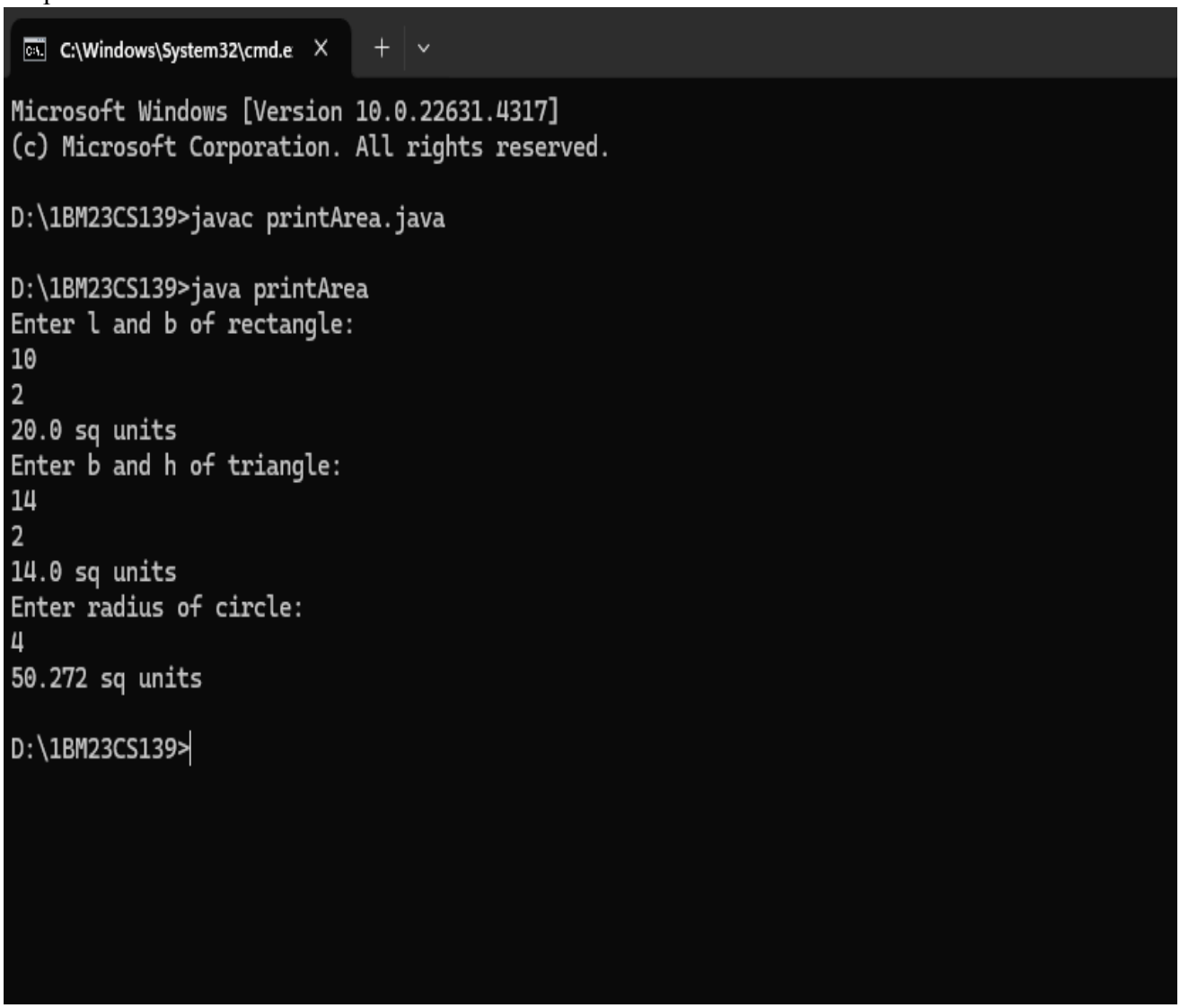
21.10

Code:

```
import java.util.Scanner;

abstract class Shape {
   double a, b, result;

   abstract void printArea();
}

class Rectangle extends Shape {

   void printArea() {
      System.out.println("Enter length and breadth of the rectangle:");
      Scanner s = new Scanner(System.in);
      a = s.nextDouble();
      b = s.nextDouble();
      result = a * b;
      System.out.println(result + " sq units");
   }
}

class Triangle extends Shape {

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

class Circle extends Shape {

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

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

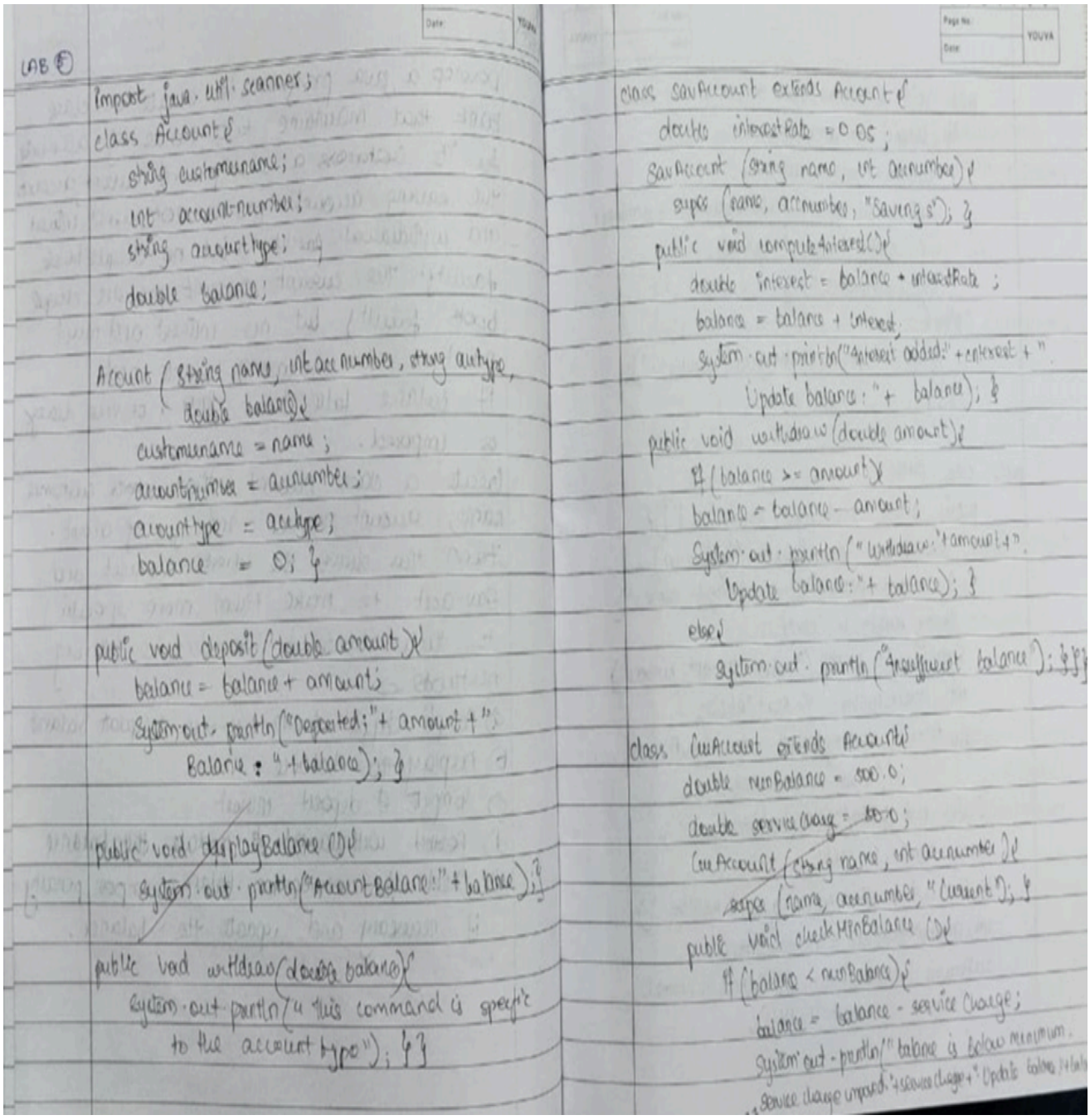
```
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();

        r.printArea();
        t.printArea();
        c.printArea();
    }
}
```

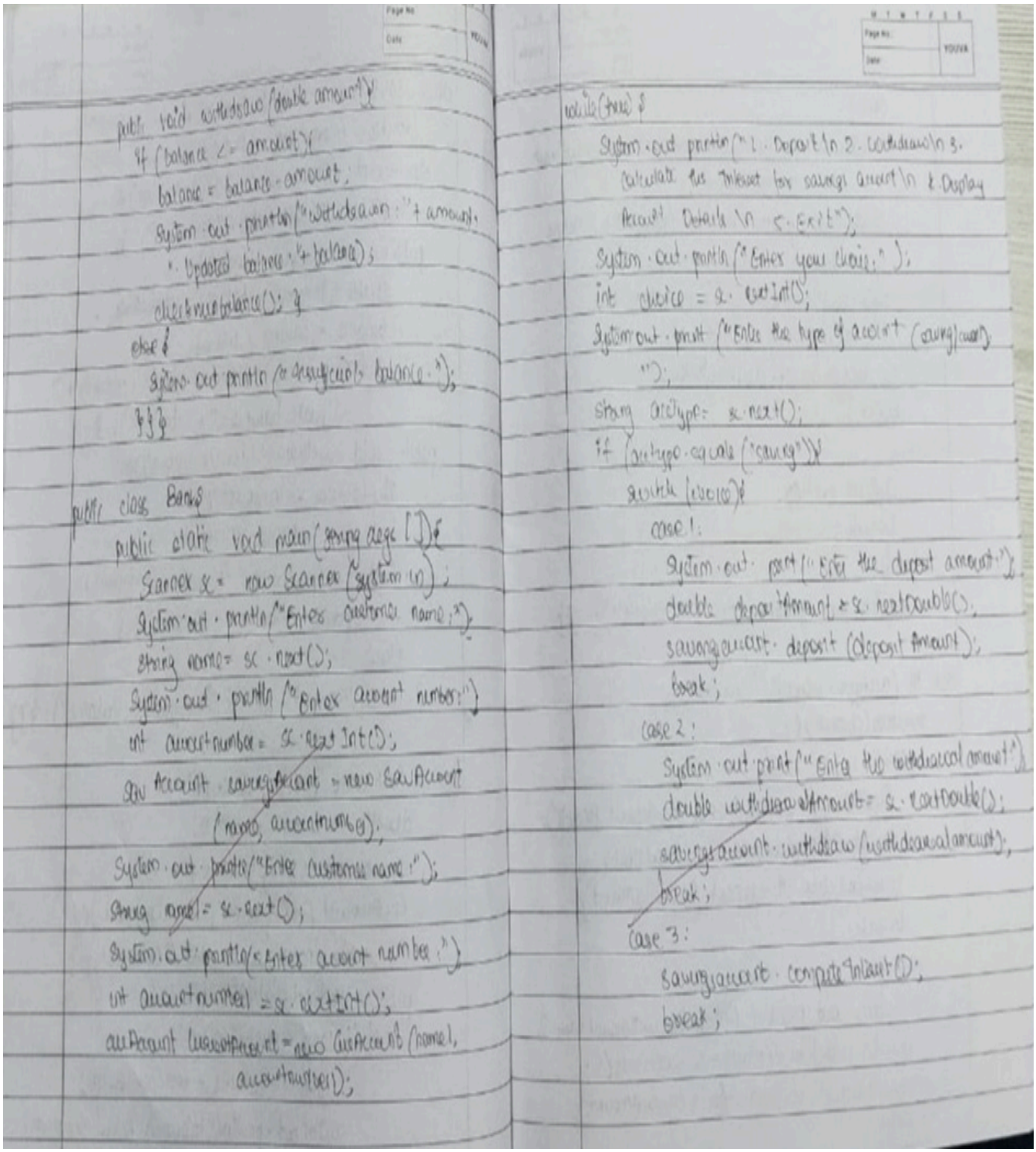
Output:



Program 5
Inheritance

Algorithm:





```
public void withdraw(double amount){
if(balance <= amount){
balance = balance - amount;
System.out.println("Withdrawn: "+amount+
". Updated balance: "+balance);
checkMinBalance(); }
else{
System.out.println("Insufficient balance.");
}}}

public class Bank
public static void main(String args[]){
Scanner sc = new Scanner(System.in);
System.out.println("Enter customer name:");
String name = sc.next();
System.out.println("Enter account number:")
int accountNumber = sc.nextInt();
SavAccount savingsAcct = new SavAccount
(name, accountNumber);
System.out.println("Enter customer name:");
String name1 = sc.next();
System.out.println("Enter account number:")
int accountNumber1 = sc.nextInt();
CurAccount currentAcct = new CurAccount(name1,
accountNumber1);

while(true){
System.out.println("1. Deposit\n2. Withdrawn\n3.
Calculate the interest for savings account\n4.Display
Account Details\n5. Exit");
System.out.println("Enter your choice:");
int choice = sc.nextInt();
System.out.println("Enter the type of account (savings
current)
");
String acctype = sc.next();
if(acctype.equals("savings")){
switch(choice){
case 1:
System.out.println("Enter the deposit amount:");
double depositAmount = sc.nextDouble();
savingsAcct.deposit(depositAmount);
break;
case 2:
System.out.println("Enter the withdrawal amount:");
double withdrawalAmount = sc.nextDouble();
savingsAcct.withdraw(withdrawalAmount);
break;
case 3:
savingsAcct.computeInterest();
break;
```

```
case 1:
    System.out.println("Customer name:" + savingsAccount
        .customerName);
    System.out.println("Account Number:" + savings
        Account.accountNumber);
    System.out.println("Type of Account:" +
        savingsAccount.accountType);
    savingsAccount.displayBalance();
    break;
case 5:
    System.exit(0);
    break;
default:
    System.out.println("Invalid Choice");
}

else if (acctype.equals("current")){
    switch(choice){
    case 1:
        System.out.println("Enter the deposit Amount:");
        double depositamount = sc.nextDouble();
        currentAccount.deposit(depositamount);
        break;
    case 2:
        System.out.println("Enter the withdrawal amount");
        double withdrawalAmount = sc.nextDouble();
        currentAccount.withdraw(withdrawalAmount);
        break;
    case 3:
        System.out.println("Current account don't earn
            interest");
        break;
    case 4:
        System.out.println("Customer name:" + currentAccount.
            customerName);
        System.out.println("Account number:" +currentAccount
            .accountNumber);
        System.out.println("Type of Account:" + currentAccount.
            accountType);
        currentAccount.displayBalance();
        break;
    case 5:
        System.exit(0);
    default:
        System.out.println("Invalid choice.");
    }
}
else{
    System.out.println("Invalid account type.");
}
}}
```

1. Deposit
2. Withdraw
3. Compute Interest for saving Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account: saving
Interest added: 2000.0. Update balance: 12000.0

1. Deposit
2. Withdraw
3. Compute Interest for saving Account
4. Display Account Details
5. Exit
Enter your choice: 4
Enter the type of account: saving
Customer name: Kavya
Account number: 123
Type of Account: Savings
Account Balance: 12000.0

1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account: saving
Enter the withdraw amount: 5000.0
Insufficient balance.

28.10

Code:

```
import java.util.Scanner;

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


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


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

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


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


class SavAccount extends Account {
    double interestRate = 0.05;

    SavAccount(String name, int accNumber) {
        super(name, accNumber, "Savings");
    }


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

```
    }


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

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

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


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

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

```
        SavAccount savingsAccount = new SavAccount(name, accountnumber);
System.out.println("Enter customer name:");
        String name1=sc.next();
        System.out.println("Enter account number:");
        int accountnumber1=sc.nextInt();
        CurAccount currentAccount = new CurAccount(name1, accountnumber1);

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

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

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

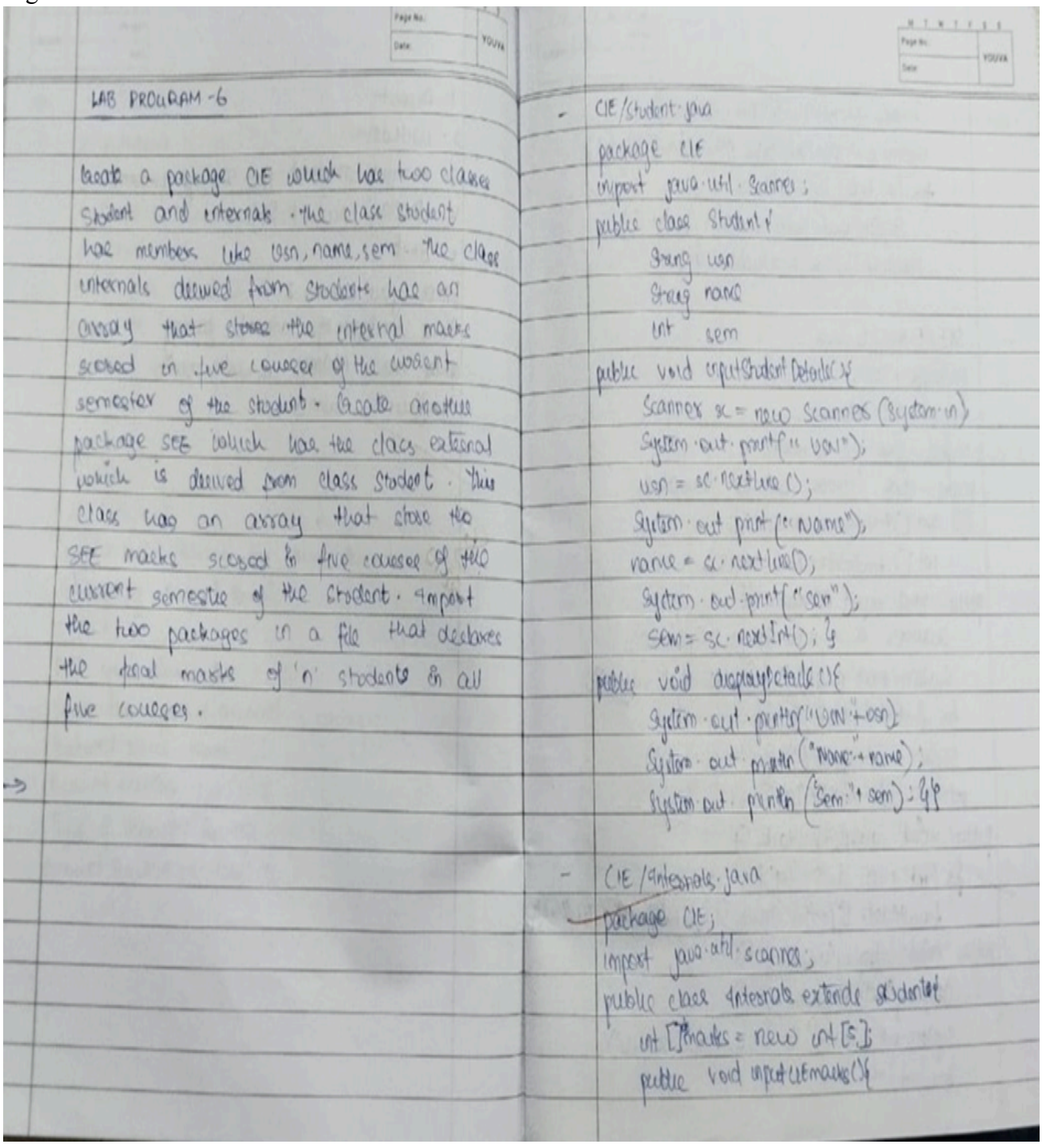
```
                    System.out.print("Enter the deposit amount: ");
                    double depositAmount = sc.nextDouble();
                    currentAccount.deposit(depositAmount);
                    break;
                case 2:
                    System.out.print("Enter the withdrawal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

Output:

```
-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 3
Enter the type of account (saving/current): saving
Interest added: 2000.0. Updated balance: 42000.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 4
Enter the type of account (saving/current): saving
Customer name: Karan
Account number: 123
Type of Account: Savings
Account Balance: 42000.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account (saving/current): 50000
Invalid account type.

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account (saving/current): saving
Enter the withdrawal amount: 50000
Insufficient balance.
```

Program 6
Package

Algorithm:



LAB 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 Students 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 derived from class Student. This class has an array that store 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.

- CIE/Student.java

```
package CIE
import java.util.Scanner;
public class Student {
    String usn
    String name
    int sem
public void inputStudentDetails() {
    Scanner sc = new Scanner(System.in)
    System.out.print("usn");
    usn = sc.nextLine();
    System.out.print("name");
    name = sc.nextLine();
    System.out.print("sem");
    sem = sc.nextInt(); }
public void displayDetails() {
    System.out.println("USN:"+usn);
    System.out.println("Name:"+name);
    System.out.println("Sem:"+sem); }}
```

- CIE/Internals.java

```
package CIE;
import java.util.Scanner;
public class Internals extends Student {
    int[] marks = new int[5];
    public void inputCIEMarks() {
```

Scanner sc = new Scanner(System.in);
System.out.println("Enter marks for 5 subjects");
for(int i=0; i<5; i++){
System.out.print("Subject"+(i+1)+":");
internals[i] = sc.nextInt(); }}

- SEE/Externals.java

package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals{
int[] marks = new int[5];
int[] finalmarks = new int[5];
public void inputSEEmarks(){
Scanner sc = new Scanner(System.in);
System.out.println("Enter SEE of 5 subjects");
for(int i=0; i<5; i++){
System.out.print("Subject"+(i+1)+":");
marks[i] = sc.nextInt(); }}
public void calculateFinalMarks(){
for(int i=0; i<5; i++){
finalMarks[i] = this.marks[i] + marks[i]; }}
public void displayFinalMarks(){
displayStudentDetails();
System.out.println("Final marks of 5 subjects");
for(int i=0; i<5; i++){
System.out.println("Subject"+(i+1)+":"+finalmarks[i]);
}}}

- Main.java

import SEE.Externals;
import java.util.Scanner;
public class Main{
public static void main(String args[]){
Scanner sc = new Scanner(System.in);
System.out.print("enter number of students");
int n = sc.nextInt();
Externals[] students = new Externals[n];
for(int i=0; i<n; i++){
System.out.println("\n Enter details for Student:"
+(i+1)+":");
students[i] = new Externals();
students[i].inputStudentDetails();
students[i].inputCIEmarks();
students[i].inputSEEmarks();
students[i].calculateFinalMarks(); }
System.out.println("\n FinalMarks of students:");
for(int i=0; i<n; i++){
System.out.println("\n Student"+(i+1)+":");
students[i].displayFinalMarks(); }
scanner.close(); }}

Page No.
Date
YOUVA

Enter the number of students: 1
Enter details for students here 1:
Enter your usn here: 139
Enter your name here: Kaeran
Enter your semester here: 3
Enter your CIE marks for the 5 subjects here:
Subject1: 98
Subject2: 99
subject3: 100
Subject4: 100
Subject5: 97
Enter the SEE marks for the 5 subjects here:
Subject 1: 100
Subject 2: 100
Subject3: 98
Subject4: 99
subject5: 95
Final marks of Students:
Student1:
Name: Kaeran
USN: 139
semester: 3
the final marks of the 5 subject are,
Subject 1: 99 Subject 5: 96
Subject 2: 99
Subject 3: 99
Subject 4: 99

Code:

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

public class Internal extends Student{

protected int[]ciemarks=new int[5];
public void InputCIEMarks(){
   Scanner sc = new Scanner(System.in);
   System.out.println("Enter your CIE marks for the 5 subjects here:");
   for(int i=0;i<5;i++){
      System.out.print("Subject"+(i+1)+":");
      ciemarks[i]=sc.nextInt();
   }


   System.out.println("CIE marks are as follows:");
   for(int i=0;i<5;i++){
      System.out.print("\nSubject"+(i+1)+":"+ciemarks[i]);

   }
}
}



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

   public void inputSEEMarks(){
      Scanner sc= new Scanner(System.in);
      System.out.println("\nEnter the 5 SEE Marks here:");
      for(int i=0;i<5;i++){
         System.out.print("\nSubject"+(i+1)+":");
         seemarks[i]=sc.nextInt();
      }

   }
   public void Calculatefinalmarks(){
```

```
        for(int i=0;i<5;i++){
            finalmarks[i]=(this.ciemarks[i]+this.seemarks[i])/2;
        }
    }
    public void Displayfinalmarks(){
        DisplayStudent();
        System.out.println("\nThe final marks of the 5 subjects are:");
        for(int i=0;i<5;i++){
            System.out.print("\nSubject"+(i+1)+":"+finalmarks[i]);
        }
    }
}



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

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

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

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

    }

}


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

```
public class main {
    public static void main(String args[]){
    Scanner sc = new Scanner(System.in);
    System.out.println("\nEnter the number of students:");
    int n = sc.nextInt();
    External[]student = new External[n];
    for(int i=0; i<n;i++){
        System.out.println("\nEnter details for Students here"+(i+1)+":");
        student[i]= new External();
        student[i].InputStudent();
        student[i].InputCIEMarks();
        student[i].inputSEEMarks();
        student[i].Calculatefinalmarks();}
    System.out.println("\nFinalmarks of students:");
    for(int i=0;i<n;i++){
            System.out.println("\nStudent"+(i+1)+":");
            student[i].Displayfinalmarks();
        }
        sc.close();
    }



}
```

Output:

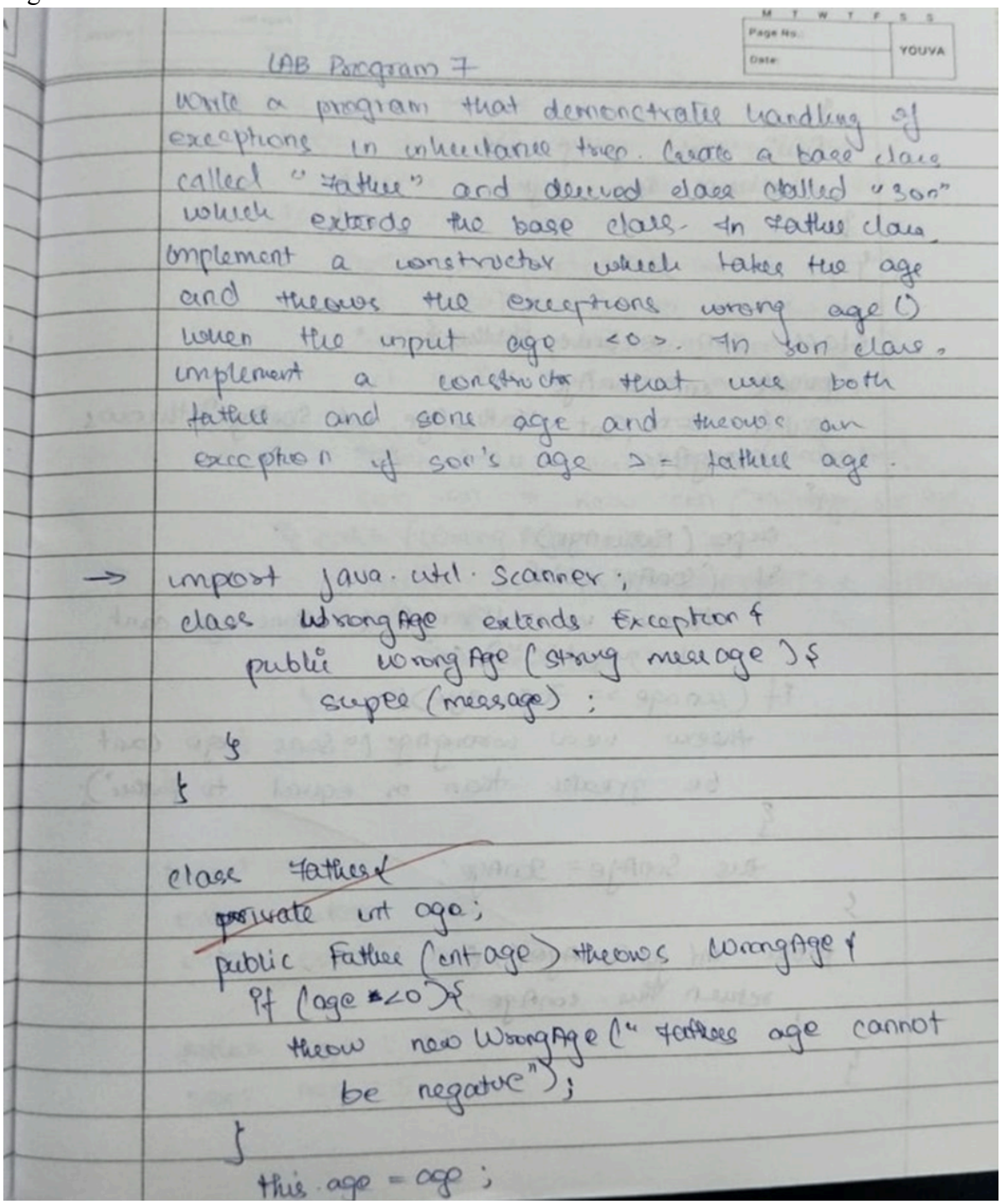
```
PS D:\1BM23CS139> cd "d:\1BM23CS139\" ; if ($?) { javac main.java } ; if ($?) { java main }

Enter the number of students:
2

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

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

Subject1:100

Subject2:100

Subject3:99

Subject4:98

Subject5:95
```

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

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

Subject1:100

Subject2:100

Subject3:92

Subject4:91

Subject5:98

Finalmarks of students:

Student1:
Name:Karna
USN:139
Semester:3

The final marks of the 5 subjects are:

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

Program 7

Exception Handling

Algorithm:



LAB Program 7

Write a program that demonstrates handling of exceptions in inheritance trees. 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 exceptions wrongAge() when the input age <0>. In Son class, implement a constructor that uses both father and son age and throws an exception if son's age >= father age.

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

class Father{
    private int age;
    public Father(int age) throws WrongAge{
        if(age<0){
            throw new WrongAge("Fathers age cannot
                be negative");
        }
        this.age = age;
```

```
}
Public int getAge(){
  return this.age
}
}

class Son extends Father{
private int SonAge;
public Son(int FatherAge, int SonAge) throws
  WrongAge
{
  super(FatherAge)
  if (SonAge<0){
    throw new WrongAge("Sons age cant
      be negative"); }
  if (SonAge >= FatherAge){
    throw new WrongAge("Sons age cant
      be greater than or equal to father")
  }
  this.SonAge = SonAge;
}
public int getSonAge(){
  return this.SonAge;
  }
}
```

```
public class Main{
  public static void main (String []args){
    Scanner Scanner = new Scanner(System.in);
  try{
    System.out.print("Enter Father age");
    int FatherAge = Scanner.nextInt()
    System.out.print("Enter Sons age");
    int SonAge = Scanner.nextInt();

    Father father = new Father(FatherAge);
    Son son = new Son(FatherAge, SonAge);
  } catch (WrongAge e){
    System.out.println("Error"+e.getMessage())
  } finally {
      Scanner.close();
    }
  }
}
```

Output ⇒

enter father age 23
enter son age : 5

father age : 23
sons age : 5

Date: YOUVA

enter fathers age: 57
enter sons age: 64
error: sons age cant be greater than fathers age.

enter fathers age: -1
enter sons age: 7
error: fathers age cant be negative.

enter fathers age: 53
enter sons age: -6
error: sons age cant be negative.

Code:

```
import java.util.Scanner;
class WrongAge extends Exception {
    public WrongAge() {
        super("Age Error");
    }
    public WrongAge(String message) {
        super(message);
    }
}

class InputScanner {
    Scanner s = new Scanner(System.in);
    public int getInput() {
        return s.nextInt();
    }
}


class Father extends InputScanner {
    protected int fatherAge;
    public Father() throws WrongAge {
        System.out.print("Enter father's age: ");
        fatherAge = getInput(); // Read father's age
        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }

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

class Son extends Father {
    private int sonAge;
    public Son() throws WrongAge {
        super();
        System.out.print("Enter son's age: ");
        sonAge = getInput();

        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");
        }
```

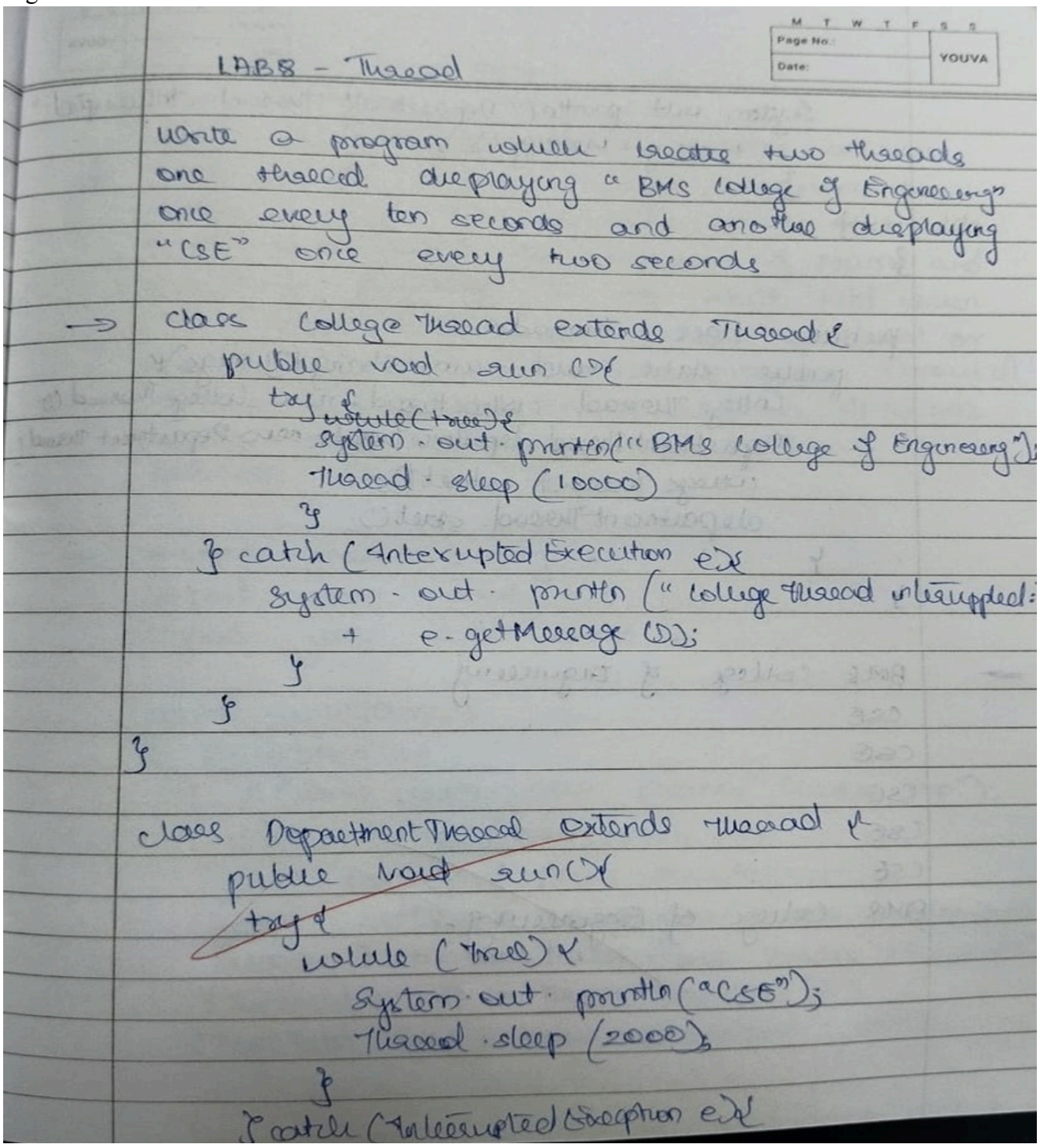
```
    }
    public void displaySonAge() {
        System.out.println("Son's age: " + sonAge);
    }
}

public class Age {
    public static void main(String[] args) {
        try {
            Son son = new Son();
            son.displayFatherAge();
            son.displaySonAge();
        } catch (WrongAge e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:

```
C:\Users\Karan\OneDrive\Desktop\javaclass>java Age
Enter father's age: 55
Enter son's age: 10
Father's age: 55
Son's age: 10

C:\Users\Karan\OneDrive\Desktop\javaclass>java Age
Enter father's age: -5
Error: Age cannot be negative

C:\Users\Karan\OneDrive\Desktop\javaclass>java Age
Enter father's age: 10
Enter son's age: 50
Error: Son's age cannot be greater than or equal to father's age
```

Program 8
GUI

Algorithm:



LAB8 - Thread

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

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

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

```
            System.out.println("Department Thread Interrupted
                + e.getMessage());
        }
    }
}

public class Thread{
    public static void main(String[] args){
        CollegeThread collegeThread = new CollegeThread();
        DepartmentThread departmentThread = new DepartmentThread();
        collegeThread.start();
        departmentThread.start();
    }
}
```

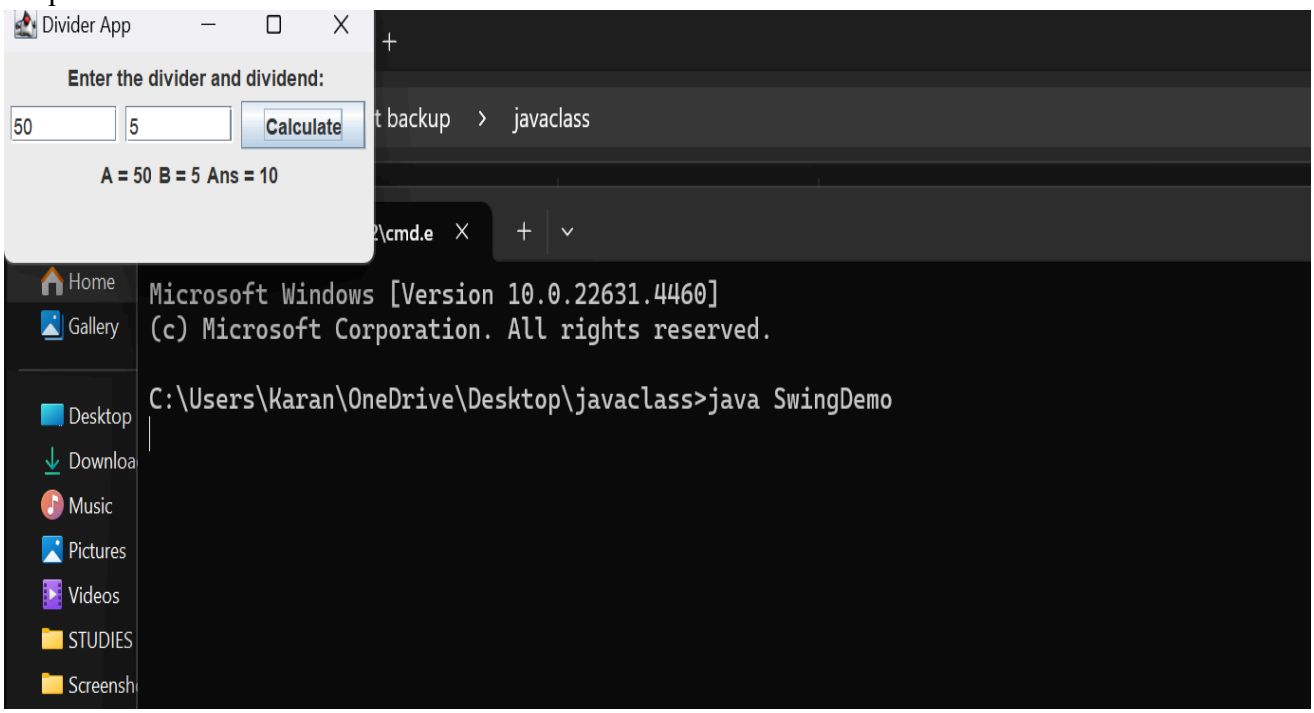
→ BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering.

Code:

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

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

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

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

      JButton button = new JButton("Calculate");

      // Labels for displaying error and results
      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;
               err.setText("");
               alab.setText("A = " + a);
               blab.setText("B = " + b);
               anslab.setText("Ans = " + ans);
            } catch (NumberFormatException e) {
```

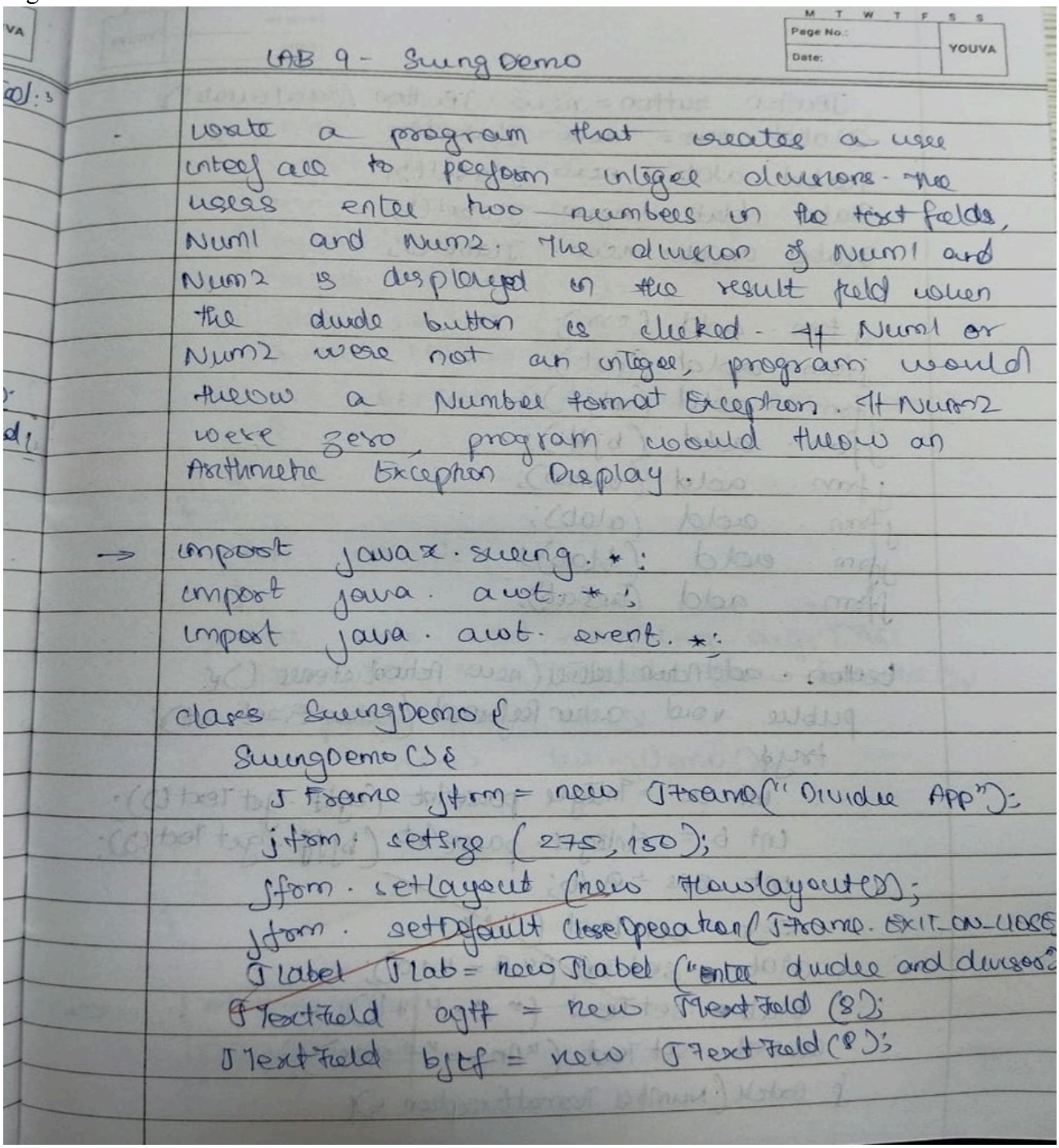
```
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("Enter Only Integers!");
            } catch (ArithmeticException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");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

    jfrm.setVisible(true);
  }

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

Output:



Program 9
MultiThreading

Algorithm:

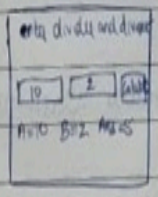


LAB 9 - SwingDemo

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, program would throw a NumberFormat Exception. If Num2 were zero, program would throw an Arithmetic Exception Display.

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

class SwingDemo {
    SwingDemo() {
    JFrame jfrm = new JFrame("Divide App");
      jfrm.setSize(275, 150);
      jfrm.setLayout(new FlowLayout());
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
      JLabel jlab = new JLabel("enter divide and divisor
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);
```

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

jfrm.add(err);
jfrm.add(alab);
jfrm.add(atft);
jfrm.add(bjft);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

button.addActionListener(new ActionListener(){
  public void actionPerformed(ActionEvent ae){
    try{
      int a = Integer.parseInt(atft.getText());
      int b = Integer.parseInt(bjft.getText());
      int ans = a/b;
      err.setText("");
      alab.setText("A="+a);
      blab.setText("B="+b);
      anslab.setText("Ans="+ans);
    } catch(NumberFormatException ex){
      alab.setText("");
      blab.setText("");
      anslab.setText("");
      err.setText("Enter only Integers");
    } catch(ArithmeticException ex){
      alab.setText("");
      blab.setText("");
      anslab.setText("");
      err.setText("B should be NON ZERO");
    }
  }
});
jfrm.setVisible(true);
}

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



Code:

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

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

public class thread {
    public static void main(String[] args) {
        // Create and start threads
        CollegeThread collegeThread = new CollegeThread();
        DepartmentThread departmentThread = new DepartmentThread();

        collegeThread.start();
        departmentThread.start();
    }
}
```

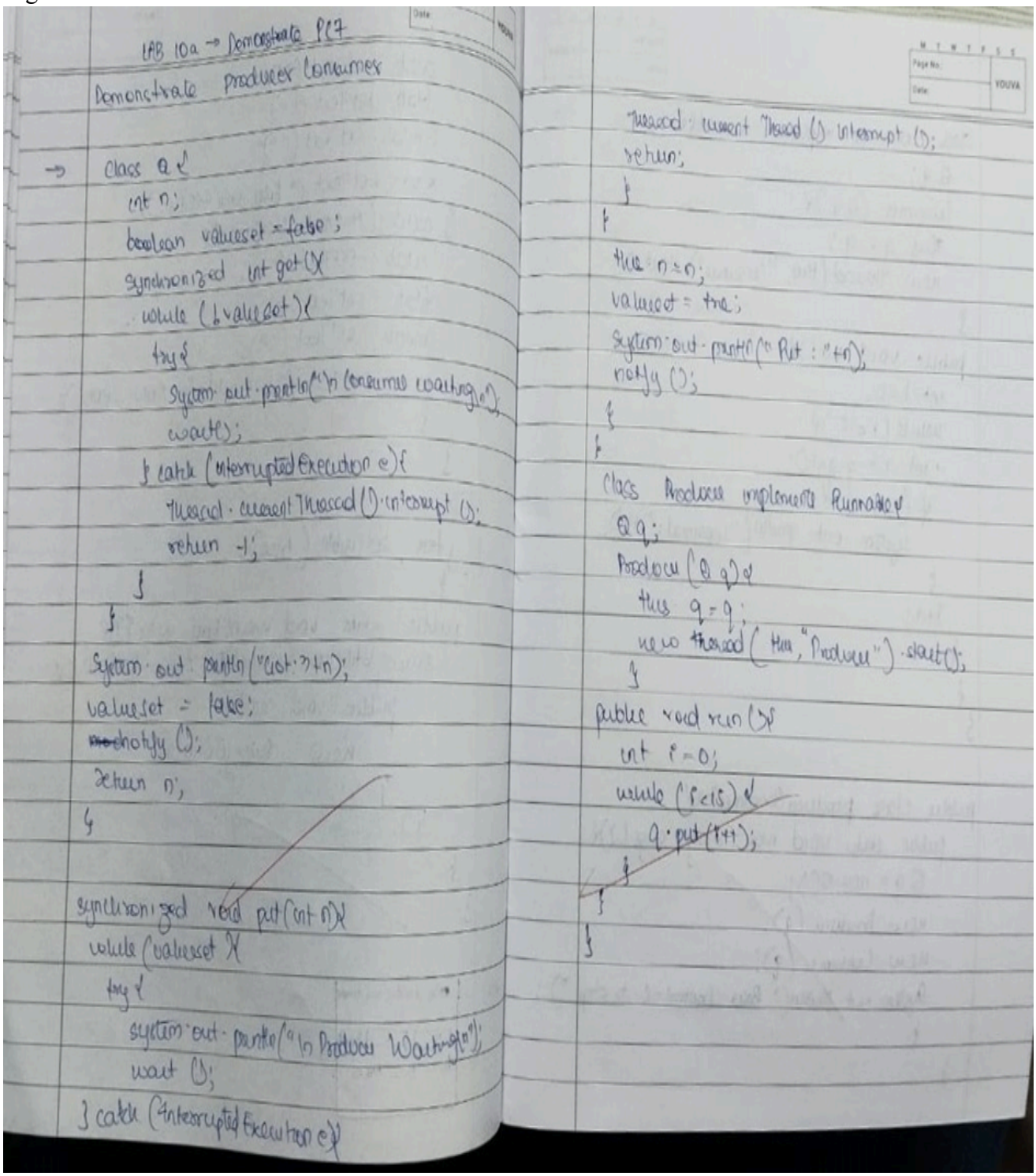
Output:

```
C:\Users\Karan\OneDrive\Desktop\javaclass>java thread
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
```

Program 10

Inter-Process Communication & Deadlock

Algorithm:



```
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();
  if(r!=-1){
    System.out.println("Consumed: "+r);
  }
  i++;
  }
}
}

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

```
Press Control-C to stop
Put: 0
Producer waiting
Got: 0
Put: 1
Producer waiting
Consumed 0
Got: 1
Consumed :1
Put : 2
```

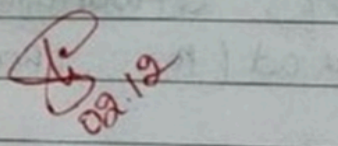
LAB 10b → Demonstrate Deadlock

Demonstrate Deadlock

```
package lab;
class A{
  synchronized void foo(B b){
    String name = Thread.currentThread().getName();
    System.out.println(name + " entered A.foo");
    try{
      Thread.sleep(1000);
    } catch(InterruptedException 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(InterruptedException e){
      System.out.println("B interrupted");
    }
    System.out.println(name + " trying to call A.last()");
    a.last();
  }
  void last(){
    System.out.println("Inside B.last");
  }
}

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

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



Code:

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

    synchronized int get() {
        // Wait until the producer has put a value
        while (!valueSet) {
            try {
                System.out.println("\nConsumer waiting\n");
                wait();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();  // Set interrupt flag
                return -1; // Handle interruption gracefully
            }
        }

        System.out.println("Got: " + n);
        valueSet = false;
        notify();  // Notify the producer that the consumer has consumed the item
        return n;
    }

    synchronized void put(int n) {
        // Wait until the consumer consumes the previous value
        while (valueSet) {
            try {
                System.out.println("\nProducer waiting\n");
                wait();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();  // Set interrupt flag
                return; // Handle interruption gracefully
            }
        }

        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        notify();  // Notify the consumer that a new item is available
    }
}

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();
            if (r != -1) { // Handle interrupted exception gracefully
                System.out.println("Consumed: " + r);
            }
            i++;
        }
    }
}

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


//Deadlock
package Lab;

class A {
    synchronized void foo(B b) {
```

```
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered A.foo");

        try {
            Thread.sleep(1000);  // This may throw InterruptedException
        } catch (InterruptedException 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);  // This may throw InterruptedException
        } catch (InterruptedException e) {
            System.out.println("B Interrupted");
        }

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

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

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

    Deadlock() {
        // Start the thread and set its name
        Thread.currentThread().setName("MainThread");
        Thread t = new Thread(this, "RacingThread");
        t.start();
```

```
        // Main thread acquires lock on a and calls foo
        a.foo(b);

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

    public void run() {
        // This method runs in the new thread
        b.bar(a);
    }

    public static void main(String[] args) {
        // Create the Deadlock instance and trigger the deadlock scenario
        new Deadlock();
    }
}
```

Output:

```
PS C:\Users\STUDENT> cd "d:\1BM23CS139\LAB fina;l\LAB10\" ; if ($?) { javac ProducerConsumerFix
Press Control-C to stop.
Put: 0

Producer waiting

Got: 0
Put: 1

Producer waiting

Consumed: 0
Got: 1
Consumed: 1
Put: 2

Producer waiting

Got: 2
Consumed: 2
Put: 3

Producer waiting

Got: 3
Consumed: 3
Put: 4

Producer waiting

Got: 4
Consumed: 4
Put: 5

Producer waiting

Got: 5
Consumed: 5
Put: 6

Producer waiting

Got: 6
Consumed: 6
Put: 7

Producer waiting

Got: 7
Consumed: 7
Put: 8

Producer waiting

Got: 8
```

```
"C:\Users\Karan\AppData\Local\Programs\Eclipse Adoptium\jdk-17.0.8.101-hotspot\bin\java.exe" "-javaagent:C:\Users\Karan\OneDrive\Desktop\CODING\intellijidea\Inte
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
RacingThread trying to call A.last()
Inside B.last
Inside A.last
Back in main thread

Process finished with exit code 0
```