**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

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

****

**LAB REPORT**

**on**

**OBJECT ORIENTED JAVA LAB**

**(22CS3PCOOJ)**

***Submitted by***

**RUSHIL BINDROO(1BM21CS172)**

***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**

**October-2022 to Feb-2023**

**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 (22CS3PCOOJ)” carried out by **RUSHIL BINDROO (1BM21CS172),** 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 during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a Object Oriented Java (22CS3PCOOJ) work prescribed for the said degree.

**Dr.Rajeshwari B.S**         **Dr. Jyothi S Nayak**

Assistant professor Professor and Head

Department of CSE Department of CSE

BMSCE, Bengaluru BMSCE, Bengaluru

`

**Table Of Contents**

|  |  |
| --- | --- |
| **S. No.** | **Experiment** |
| **1** | PROGRAM-01 |
| **2** | PROGRAM-02 |
| **3** | PROGRAM-03 |
| **4** | PROGRAM-04 |
| **5** | PROGRAM-05 |
| **6** | PROGRAM-06 |
| **7** | PROGRAM-07 |
| **8** | PROGRAM-08 |

**PROGRAM -01**

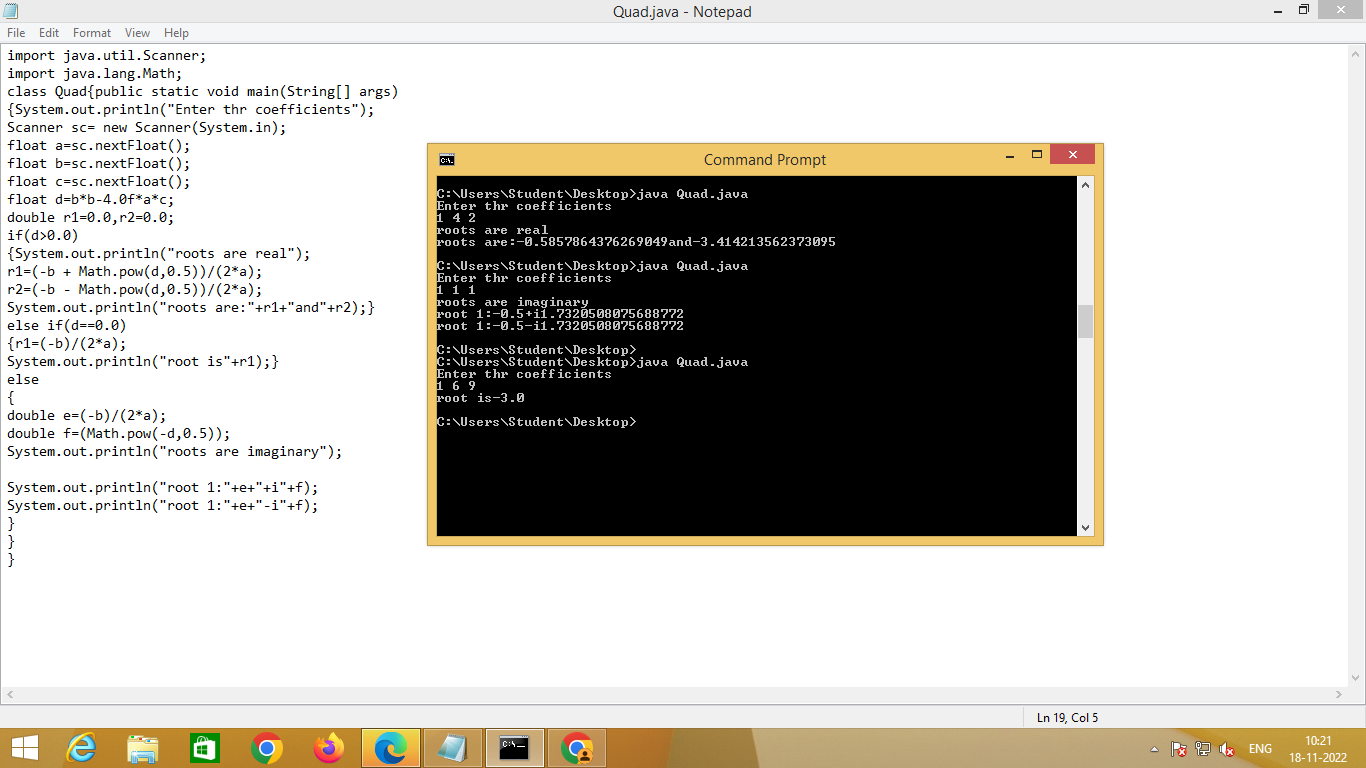
**QUESTION:**

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.

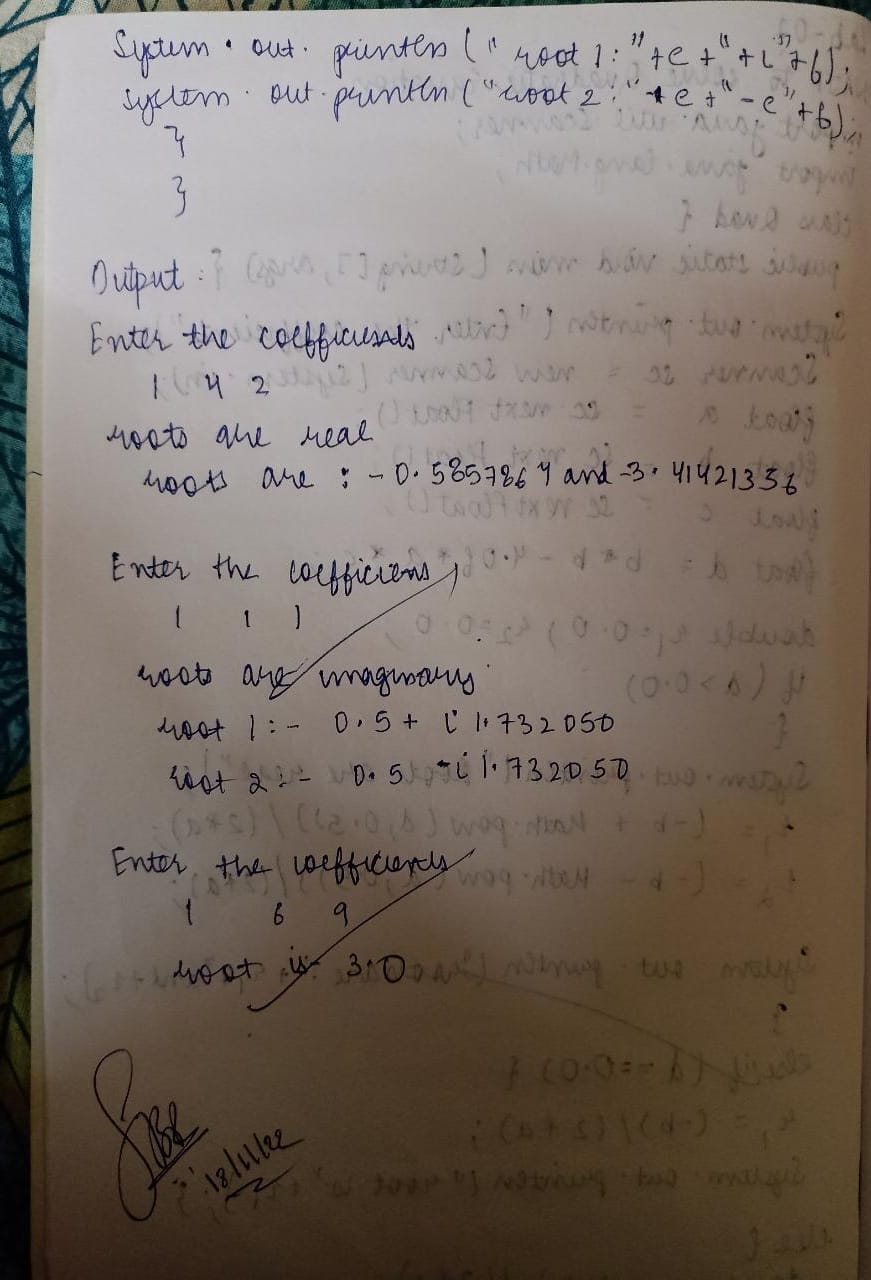
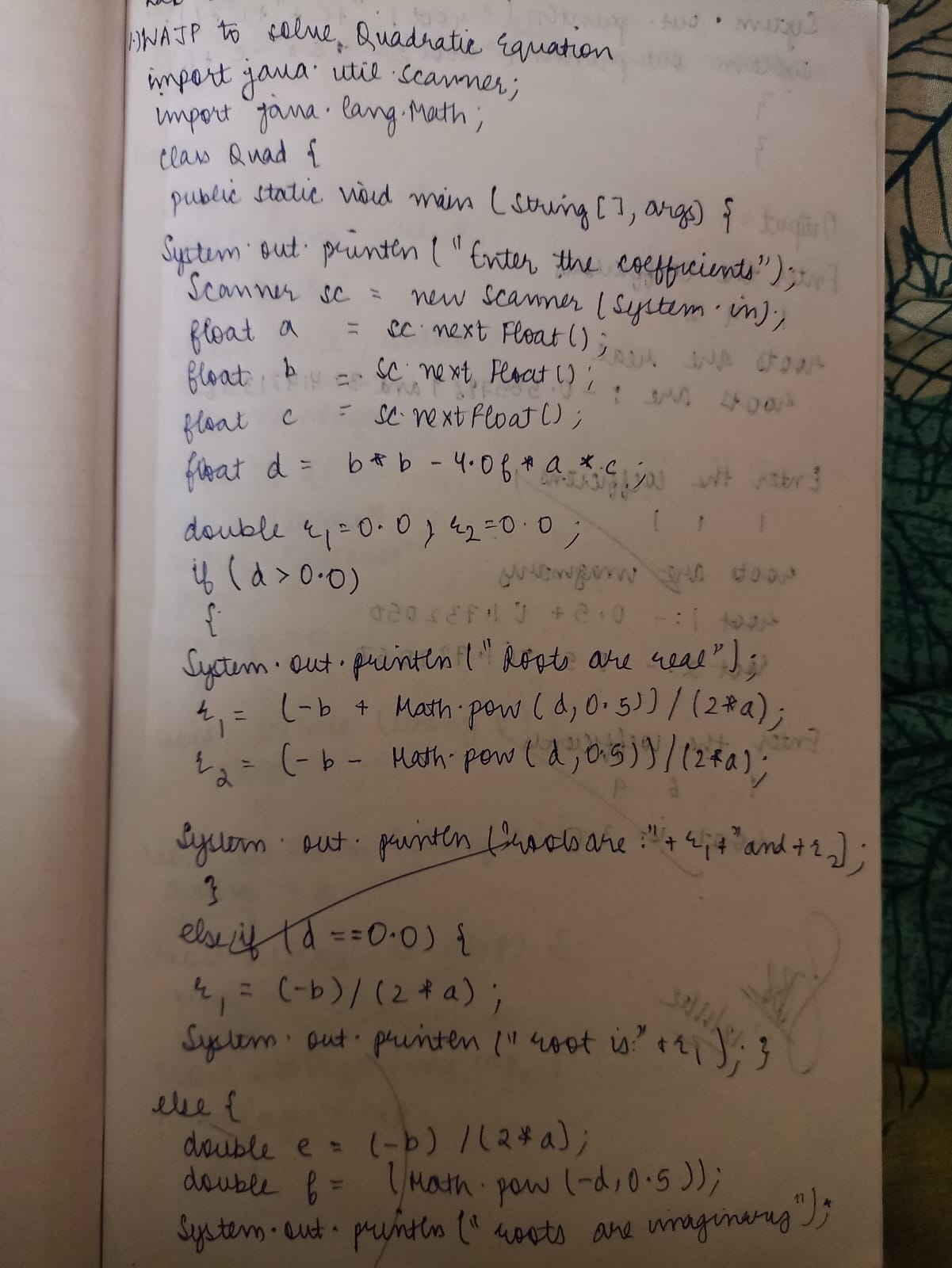
**CODE:**

|  |
| --- |
| importjava.util.\*; |
|  |  |
|  | class QuadraticEquation{ |
|  |  |
|  | public static void main(String args[]){ |
|  |  |
|  | Scanner sc = new Scanner(System.in); |
|  |  |
|  | System.out.println("Enter value of a: "); |
|  | double a = sc.nextDouble(); |
|  |  |
|  | System.out.println("Enter value of b: "); |
|  | double b = sc.nextDouble(); |
|  |  |
|  | System.out.println("Enter value of c: "); |
|  | double c = sc.nextDouble(); |
|  |  |
|  | double d = (b\*b)-(4\*a\*c); |
|  |  |
|  | if (d>0) |
|  | { |
|  | double r1 = (-b+Math.sqrt(d))/(2\*a); |
|  | double r2 = (-b-Math.sqrt(d))/(2\*a); |
|  |  |
|  | System.out.format("Root 1: %.2f", r1); |
|  | System.out.format("Root 2: %.2f", r2); |
|  | } |
|  |  |
|  | else if (d==0) |
|  | { |
|  | double r1,r2; |
|  | r1= r2 = -b/(2\*a); |
|  | System.out.format("Root 1 = Root 2 = %.2f", r1, r2); |
|  | } |
|  |  |
|  | else |
|  | { |
|  | double real = -b / (2 \* a); |
|  | double imaginary = Math.sqrt(-d) / (2 \* a); |
|  | System.out.format("Root1 = %.2f+%.2fi", real, imaginary); |
|  | System.out.format("Root2 = %.2f-%.2fi", real, imaginary); |
|  |  |
|  | } |
|  |  |
|  | } |

**OUTPUT:**



**Observation Book Pictures:**



**PROGRAM-02**

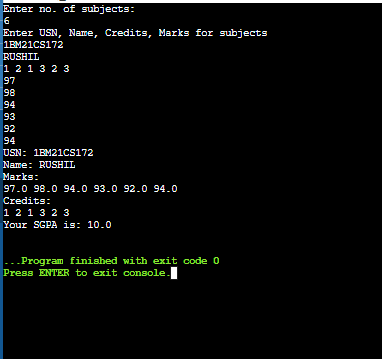
**QUESTION:**

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.

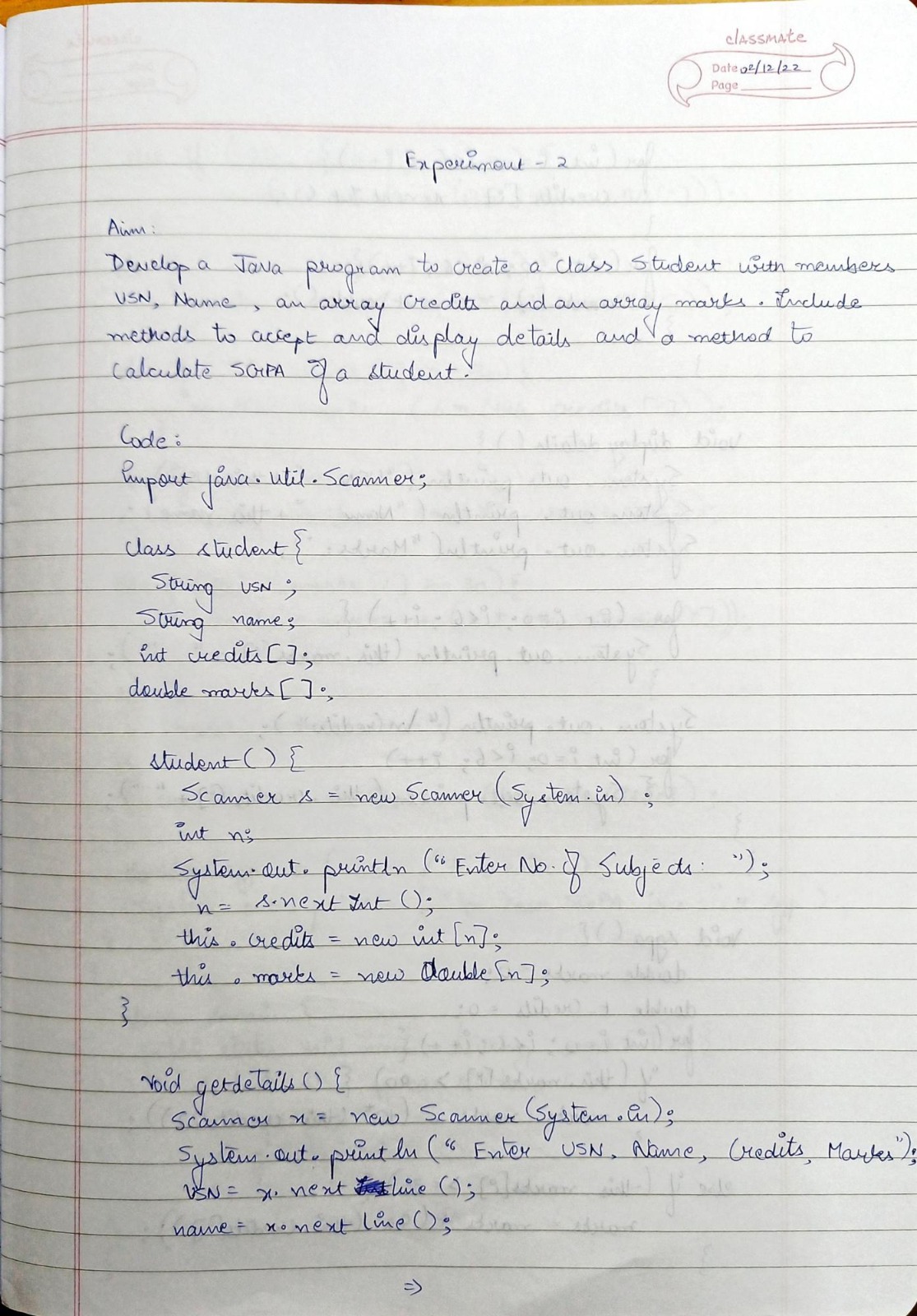
**CODE:**

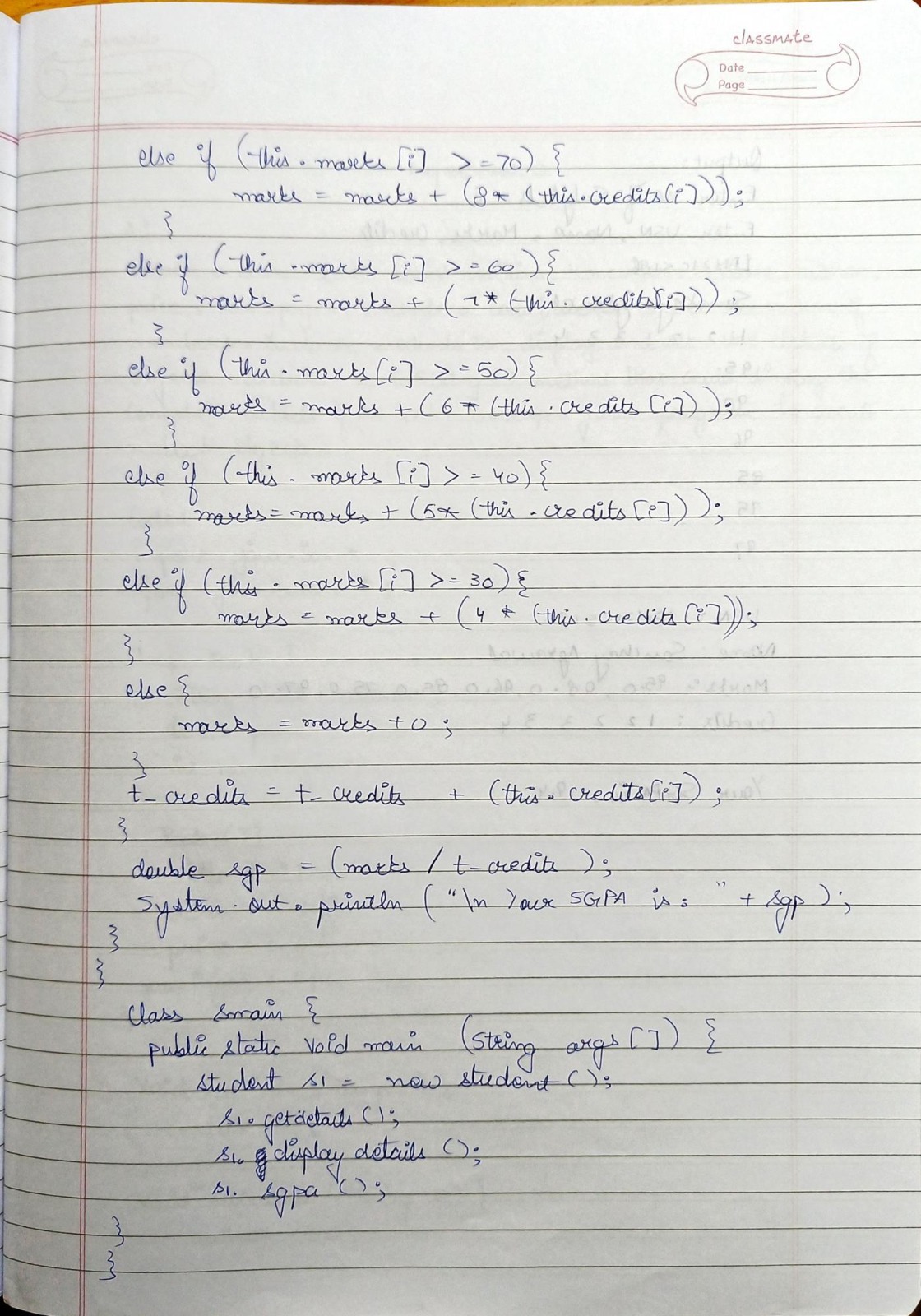
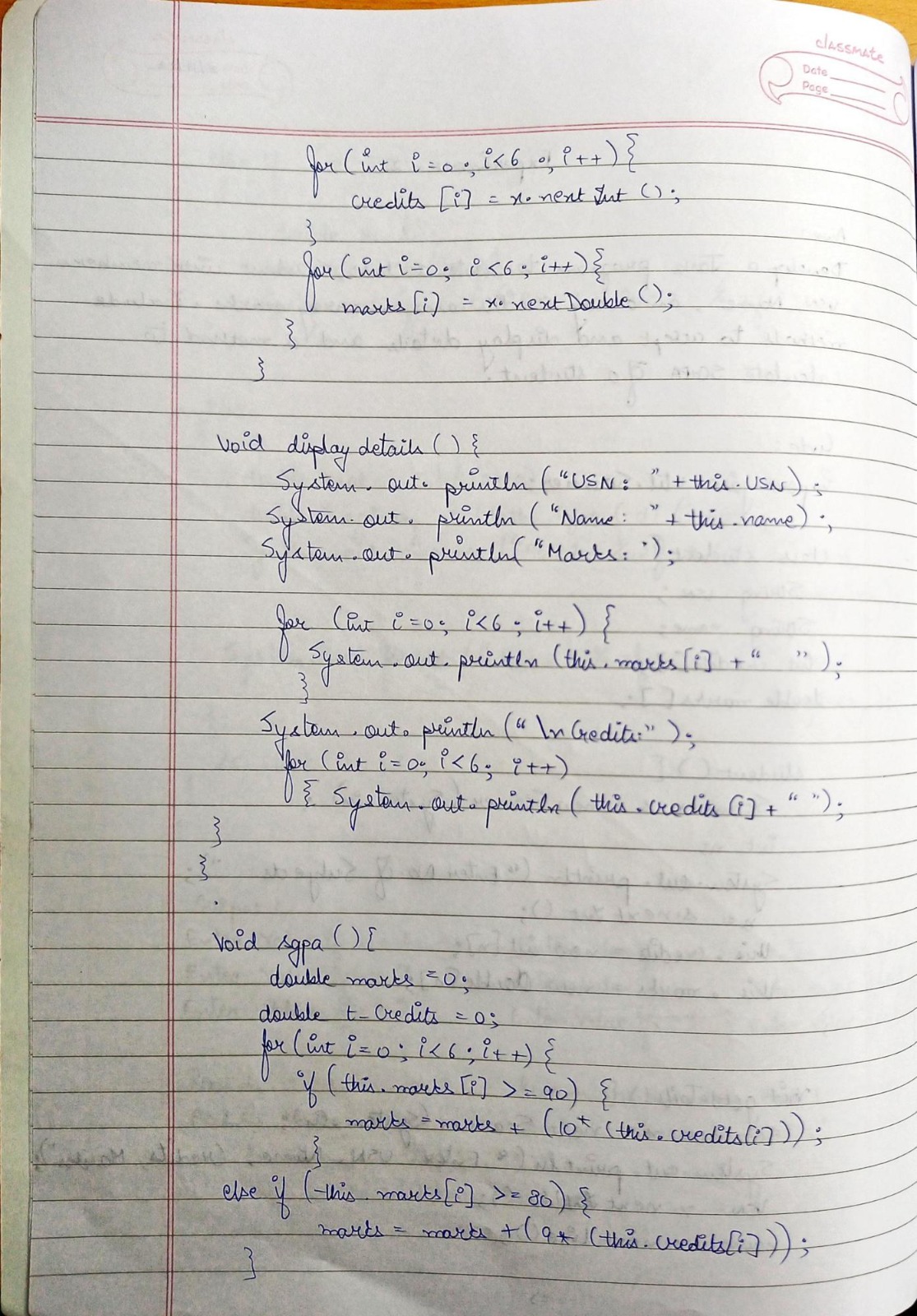
|  |
| --- |
| import java.util.Scanner; |
|  |  |
|  | class student { |
|  | String usn; |
|  | String name; |
|  | int credits[]; |
|  | double marks[]; |
|  |  |
|  | student() { |
|  | Scanner s=new Scanner(System.in); |
|  | int n; |
|  | System.out.println("Enter no. of subjects: "); |
|  | n=s.nextInt(); |
|  | this.credits=new int[n]; |
|  | this.marks=new double[n]; |
|  |  |
|  | } |
|  |  |
|  | void getsd() { |
|  | Scanner x=new Scanner(System.in); |
|  | System.out.println("Enter USN, Name, Credits, Marks for subjects"); |
|  | usn=x.nextLine(); |
|  | name=x.nextLine(); |
|  | for(int i=0;i<6;i++) { |
|  | credits[i]=x.nextInt(); |
|  | } |
|  | for(int i=0;i<6;i++) { |
|  | marks[i]=x.nextDouble(); |
|  | } |
|  |  |
|  | } |
|  |  |
|  | void putsd() { |
|  | System.out.println("USN: "+this.usn); |
|  | System.out.println("Name: "+this.name); |
|  | System.out.println("Marks: "); |
|  | for(int i=0;i<6;i++) { |
|  | System.out.print(this.marks[i]+" "); |
|  | } |
|  | System.out.println("\nCredits: "); |
|  | for(int i=0;i<6;i++) { |
|  | System.out.print(this.credits[i]+" "); |
|  | } |
|  | } |
|  |  |
|  | void sgpa() { |
|  | double marks=0; |
|  | double t\_credits=0; |
|  | for(int i=0;i<6;i++) { |
|  | if(this.marks[i]>=90) { |
|  | marks=marks+(10\*(this.credits[i])); |
|  | } |
|  | else if(this.marks[i]>=80) { |
|  | marks=marks+(9\*(this.credits[i])); |
|  | } |
|  | else if(this.marks[i]>=70) { |
|  | marks=marks+(8\*(this.credits[i])); |
|  | } |
|  | else if(this.marks[i]>=60) { |
|  | marks=marks+(7\*(this.credits[i])); |
|  | } |
|  | else if(this.marks[i]>=50) { |
|  | marks=marks+(6\*(this.credits[i])); |
|  | } |
|  | else if(this.marks[i]>=40) { |
|  | marks=marks+(5\*(this.credits[i])); |
|  | } |
|  | else if(this.marks[i]>=30) { |
|  | marks=marks+(4\*(this.credits[i])); |
|  | } |
|  | else { |
|  | marks=marks+0; |
|  | } |
|  | t\_credits=t\_credits+(this.credits[i]); |
|  | } |
|  | double sgp=(marks/t\_credits); |
|  | System.out.println("\nYour SGPA is: "+sgp); |
|  | } |
|  | } |
|  |  |
|  | class smain { |
|  | public static void main(String xx[]) { |
|  | student s1=new student(); |
|  | s1.getsd(); |
|  | s1.putsd(); |
|  | s1.sgpa(); |
|  | } |
|  | } |

**OUTPUT:**



**Observation Book Pictures:**

****



**PROGRAM-03**

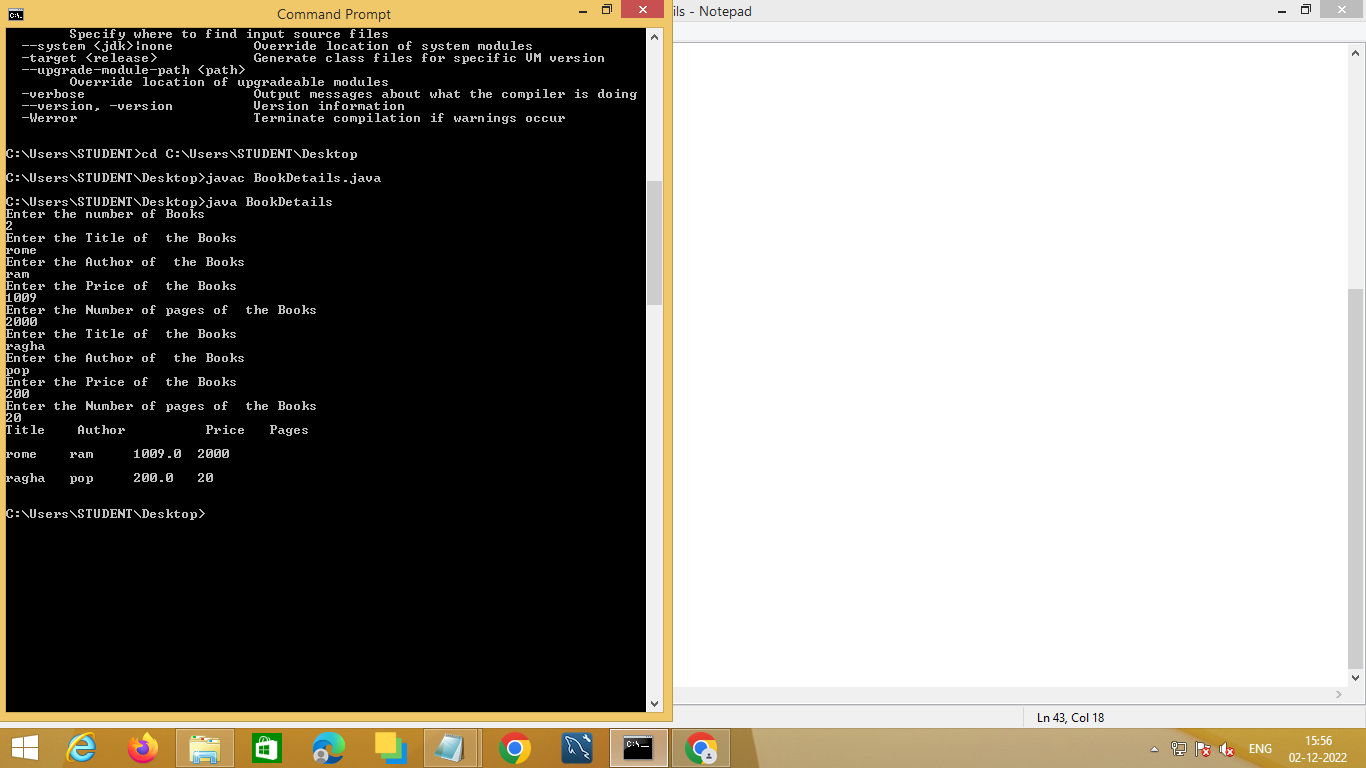
**QUESTION:**

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.

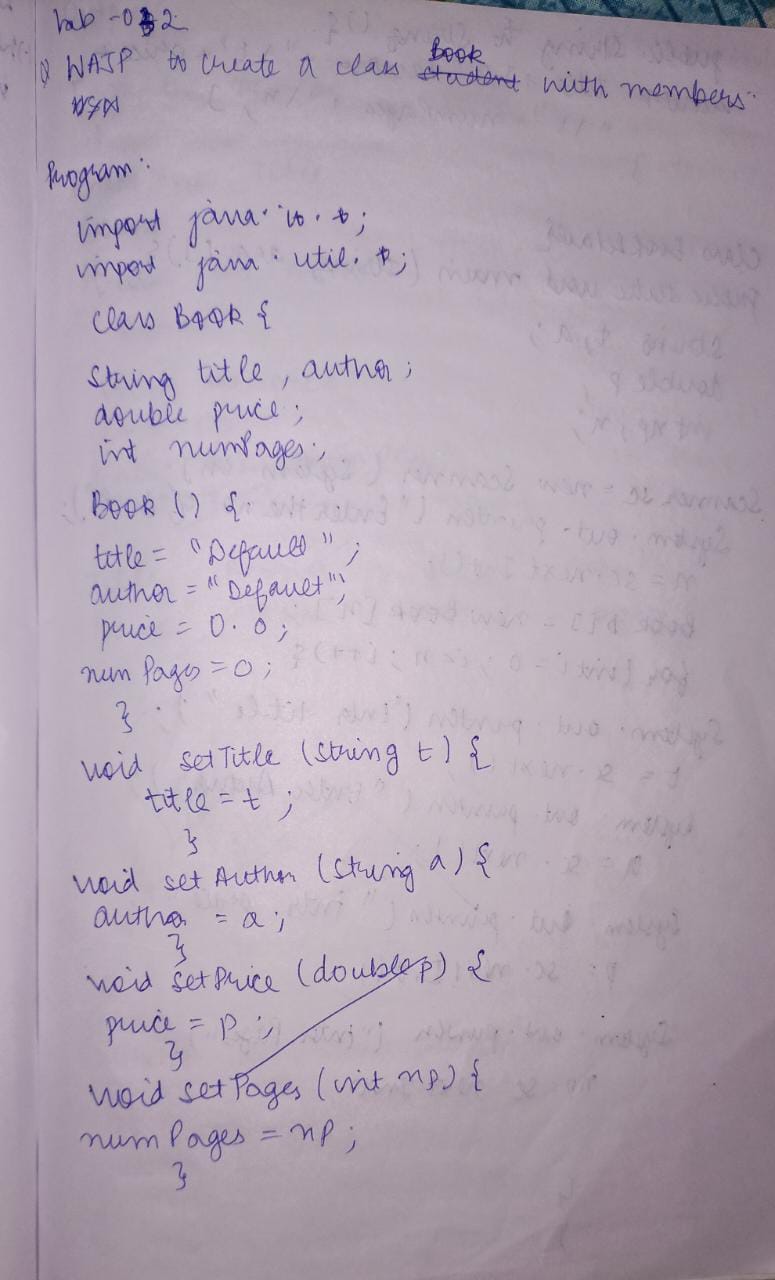
**CODE:**

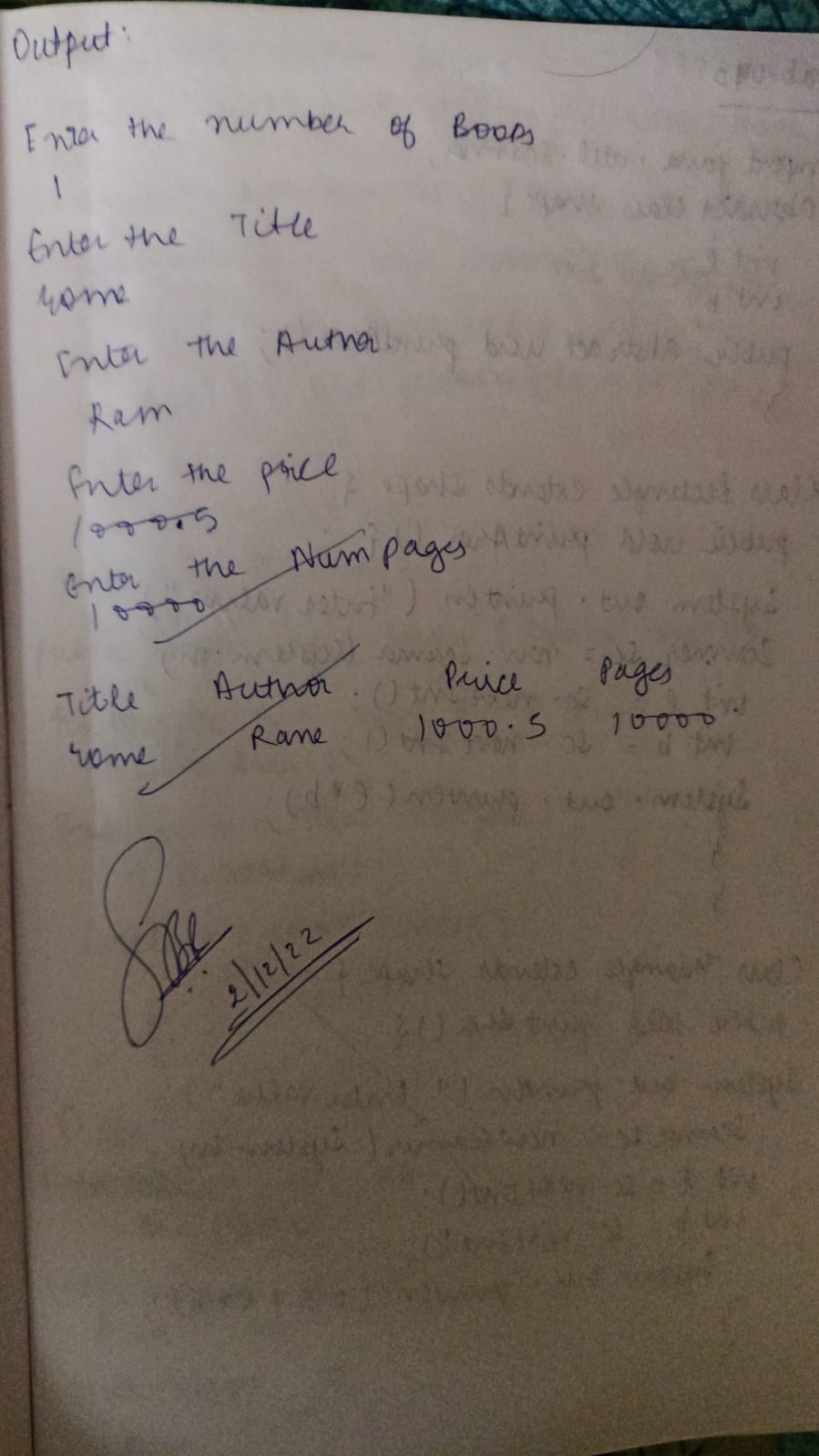
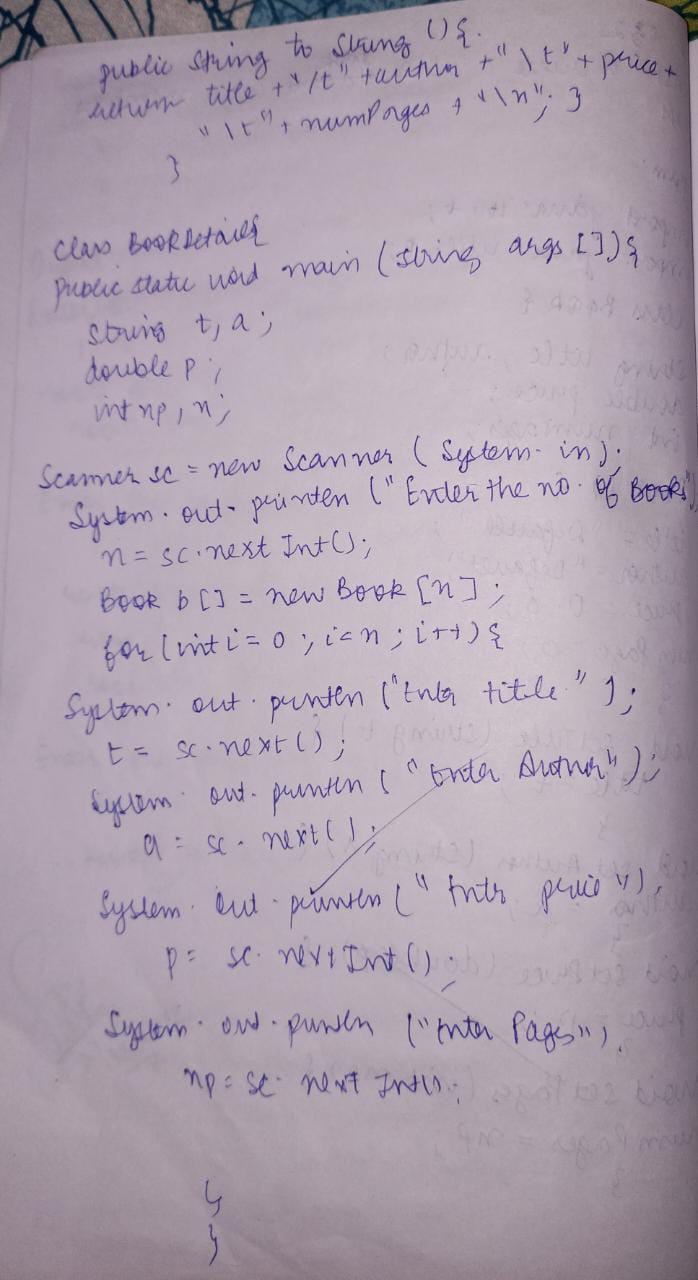
|  |
| --- |
| importjava.io.\*; |
|  | import java.util.\*; |
|  |  |
|  |  |
|  |  |
|  | class Book { |
|  |  |
|  | String title, author; |
|  | double price; |
|  | int numPages; |
|  |  |
|  | Book() { |
|  |  |
|  | title="Default"; |
|  | author="Default"; |
|  | price=0.0; |
|  | numPages=0; |
|  | } |
|  | void setTitle(String t) { |
|  |  |
|  | title=t; |
|  |  |
|  | } |
|  | void setAuthor(String a) { |
|  |  |
|  |  |
|  | author=a; |
|  |  |
|  | } |
|  | void setPrice(double p) { |
|  |  |
|  |  |
|  | price=p; |
|  |  |
|  | } |
|  | void setPages(int np) { |
|  |  |
|  |  |
|  | numPages=np; |
|  | } |
|  |  |
|  | public String toString() { |
|  |  |
|  | return title+"\t"+author+"\t"+price+"\t"+numPages+"\n"; |
|  | } |
|  | } |
|  |  |
|  | class BookDetails { |
|  |  |
|  | public static void main(String args[]) { |
|  | String t, a; |
|  | double p; |
|  | int np,n; |
|  | Scanner sc = new Scanner(System.in); |
|  | System.out.println("Enter the number of Books"); |
|  | n = sc.nextInt(); |
|  | Book b[]= new Book[n]; |
|  | for(int i=0; i<n;i++) { |
|  | System.out.println("Enter the Title of the Books"); |
|  | t= sc.next(); |
|  | System.out.println("Enter the Author of the Books"); |
|  | a= sc.next(); |
|  | System.out.println("Enter the Price of the Books"); |
|  | p= sc.nextDouble(); |
|  | System.out.println("Enter the Number of pages of the Books"); |
|  | np= sc.nextInt(); |
|  |  |
|  | b[i] = new Book(); |
|  | b[i].setTitle(t); |
|  | b[i].setAuthor(a); |
|  | b[i].setPrice(p); |
|  | b[i].setPages(np); |
|  | } |
|  |  |
|  | System.out.println("Title \t Author \t Price \t Pages\n"); |
|  | for(int i=0; i<n;i++) { |
|  | System.out.println(b[i]); |
|  | } |
|  |  |
|  | } |
|  | } |

**OUTPUT:**



**Observation Book Pictures:**

****



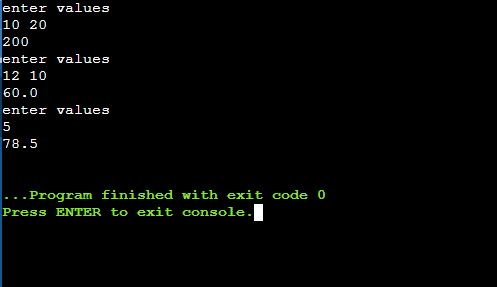
**PROGRAM-04**

**QUESTION:** 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.

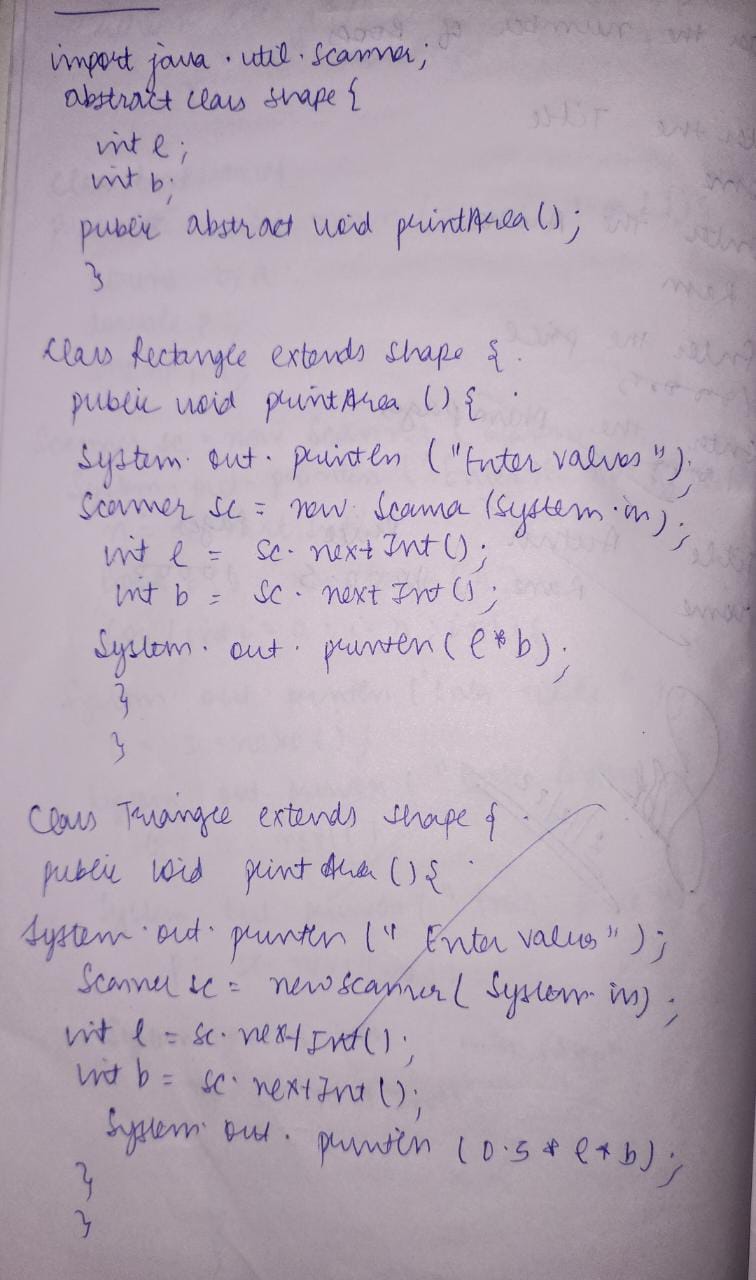
**CODE:**

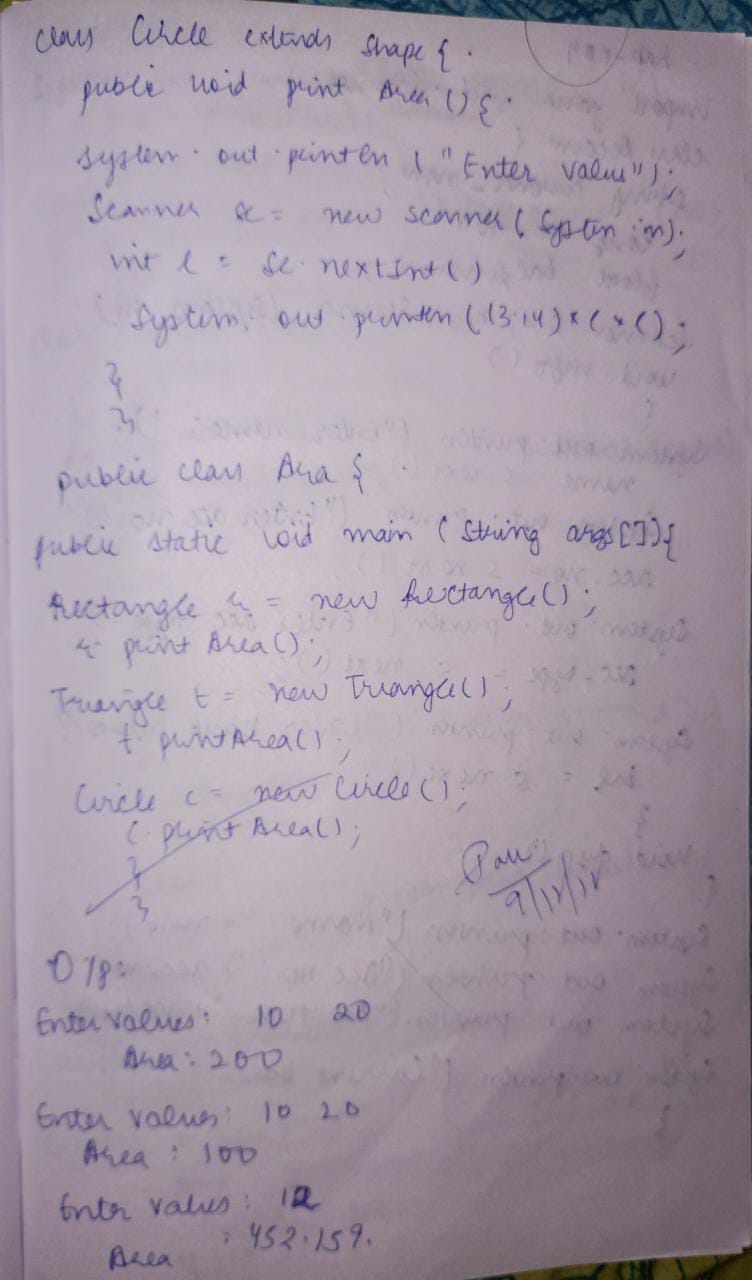
|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | abstract class Shape { |
|  |  |
|  | int a, b; |
|  |  |
|  | public Shape(int a, int b) { |
|  | this.a = a; |
|  | this.b = b; |
|  | } |
|  | abstract void Printarea(); |
|  | } |
|  |  |
|  | class Circle extends Shape { |
|  |  |
|  | Circle(int a, int b) { |
|  | super(a, b); |
|  | } |
|  |  |
|  | void Printarea() { |
|  | System.out.println("area of circle is " + (3.14 \* a \* a)); |
|  | } |
|  |  |
|  | } |
|  |  |
|  | class Rectangle extends Shape { |
|  | public Rectangle(int a, int b) { |
|  | super(a, b); |
|  | } |
|  |  |
|  | void Printarea() |
|  | { |
|  | System.out.println("area of rectangle is " + (a \* b)); |
|  | } |
|  |  |
|  | } |
|  |  |
|  | class Triangle extends Shape { |
|  | public Triangle(int a, int b) { |
|  | super(a, b); |
|  | } |
|  |  |
|  | void Printarea() |
|  |  |
|  | { |
|  |  |
|  | System.out.println("area of triangle is " + (0.5 \* a \* b)); |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | class Main { |
|  | public static void main(String args[]) { |
|  |  |
|  | Scanner in = new Scanner(System.in); |
|  |  |
|  | System.out.println("Enter dimension 1: "); |
|  | int x = in.nextInt(); |
|  | System.out.println("Enter dimension 2: "); |
|  | int y = in.nextInt(); |
|  |  |
|  |  |
|  | Shape b; |
|  | b = new Circle(x, y); |
|  | b.Printarea(); |
|  |  |
|  | b = new Rectangle(x, y); |
|  | b.Printarea(); |
|  |  |
|  | b = new Triangle(x, y); |
|  | b.Printarea(); |
|  | } |
|  | } |

**OUTPUT:**

****

**Observation Book Pictures:**

****



**PROGRAM-05**

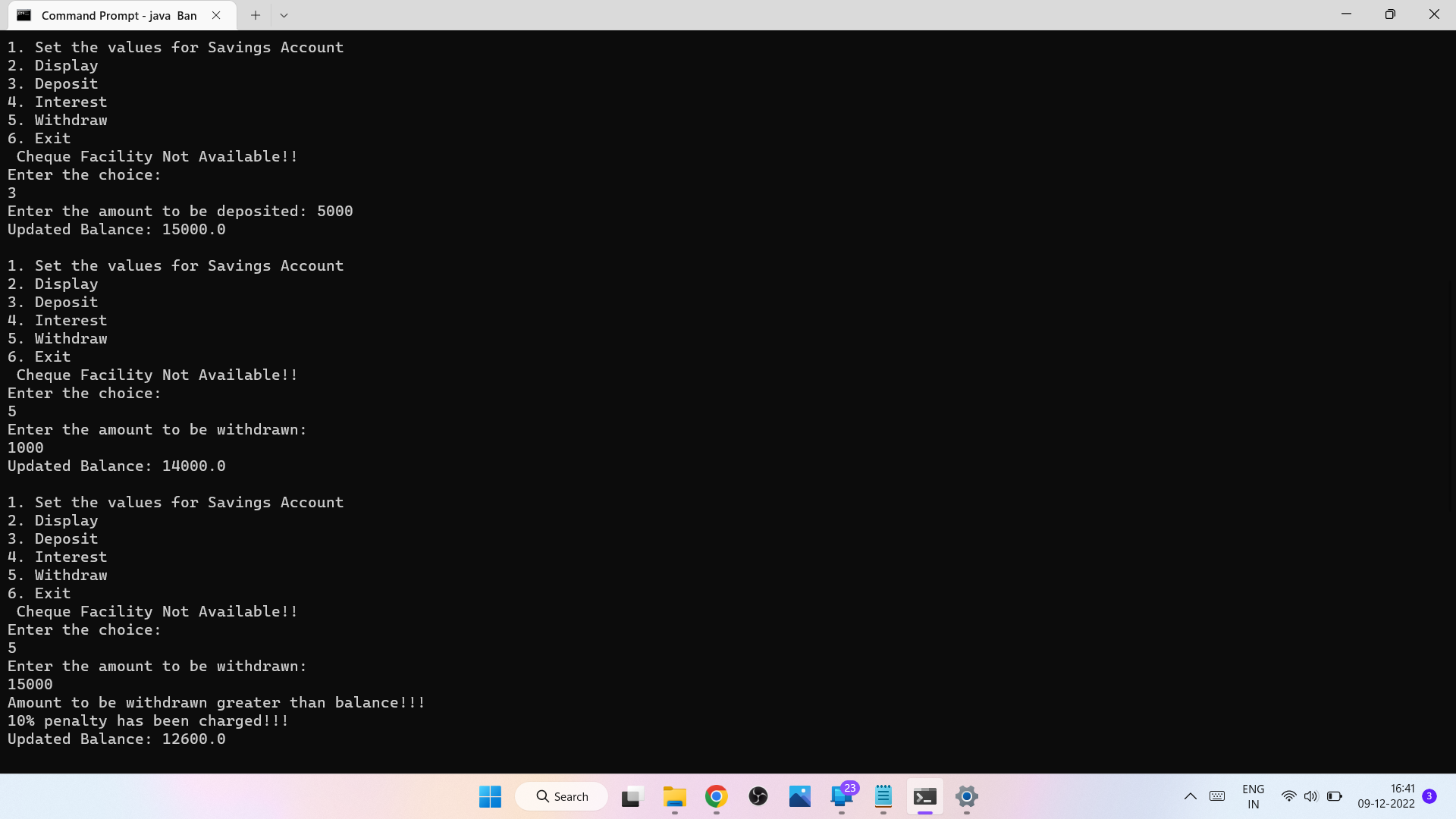
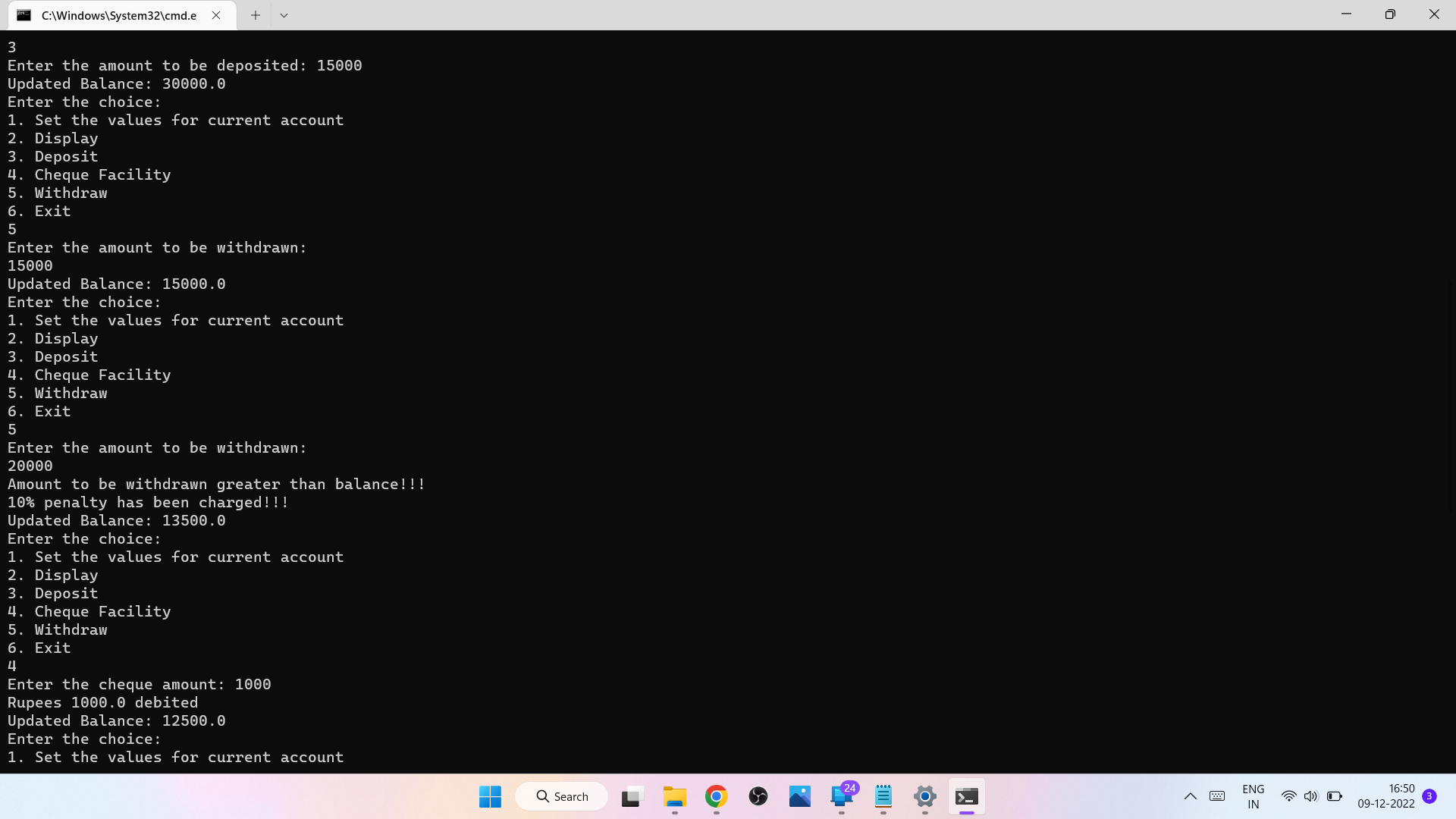
**QUESTION:**

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

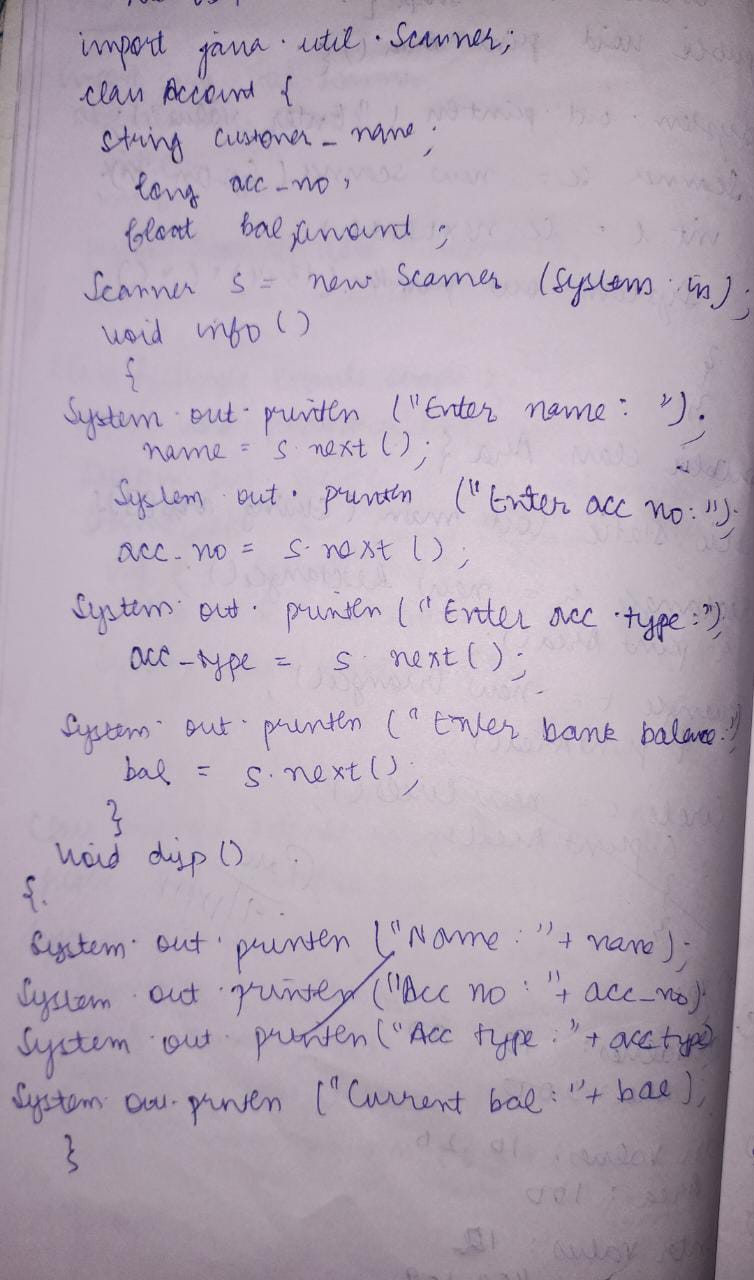
**CODE:**

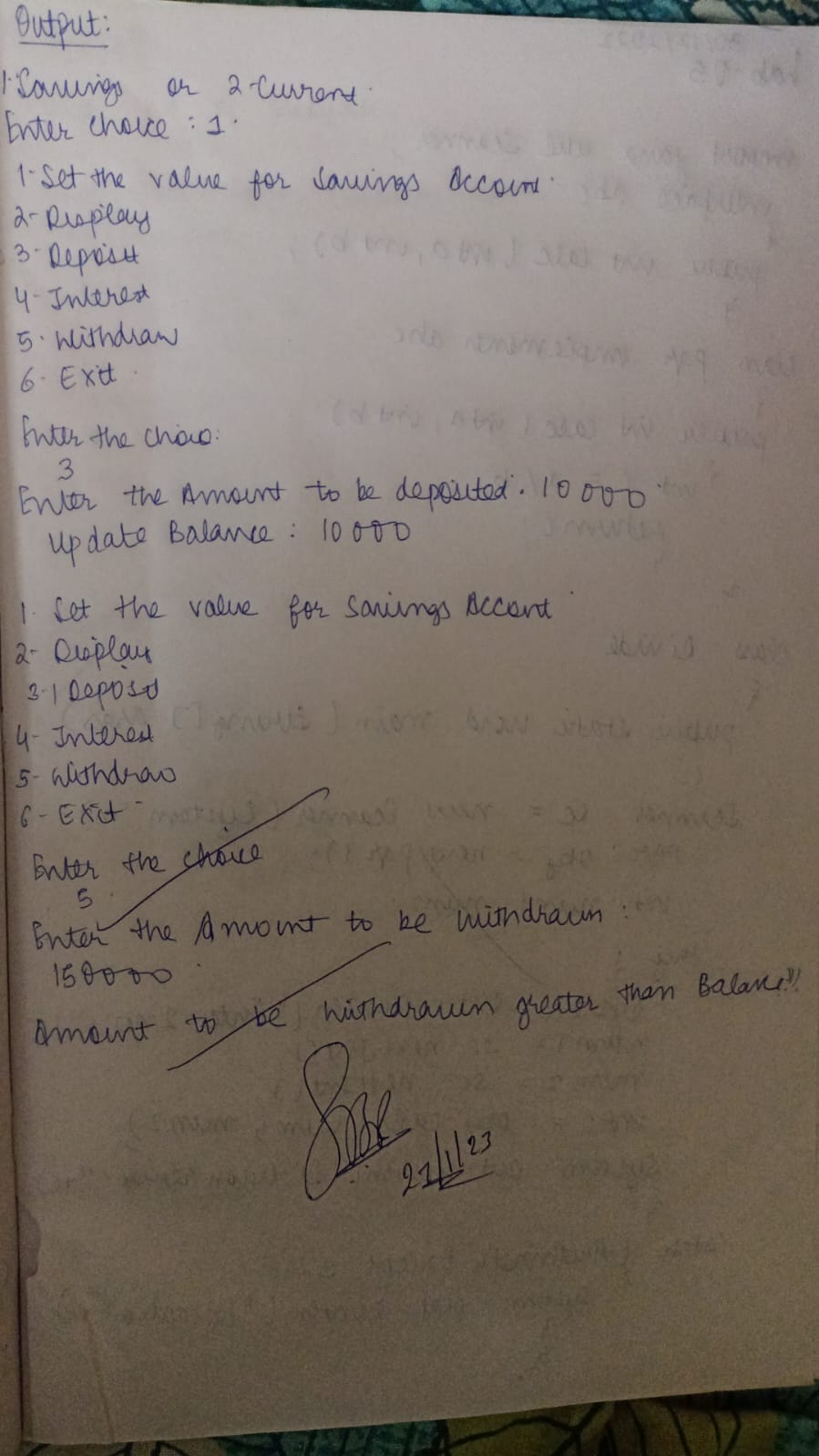
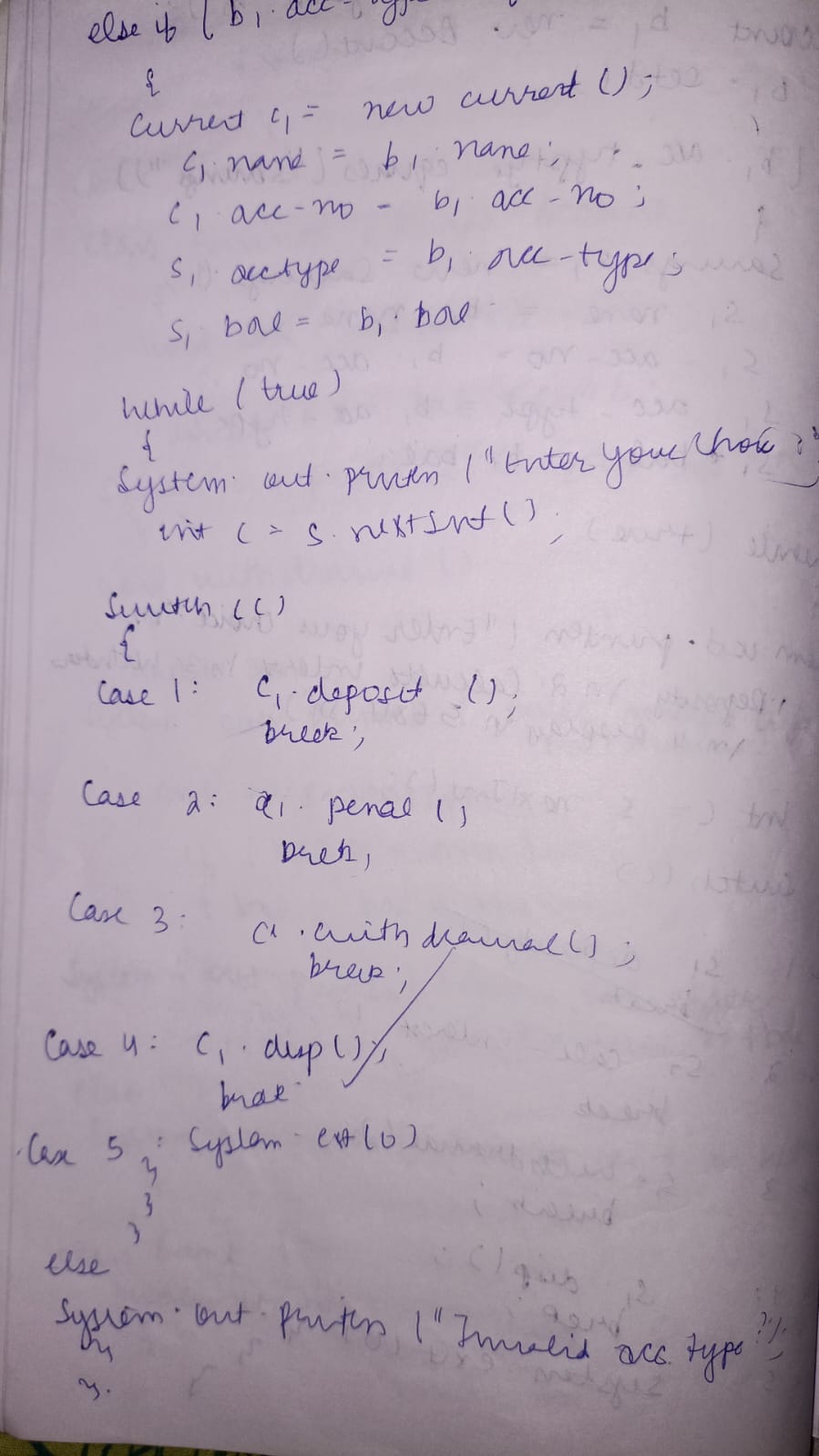
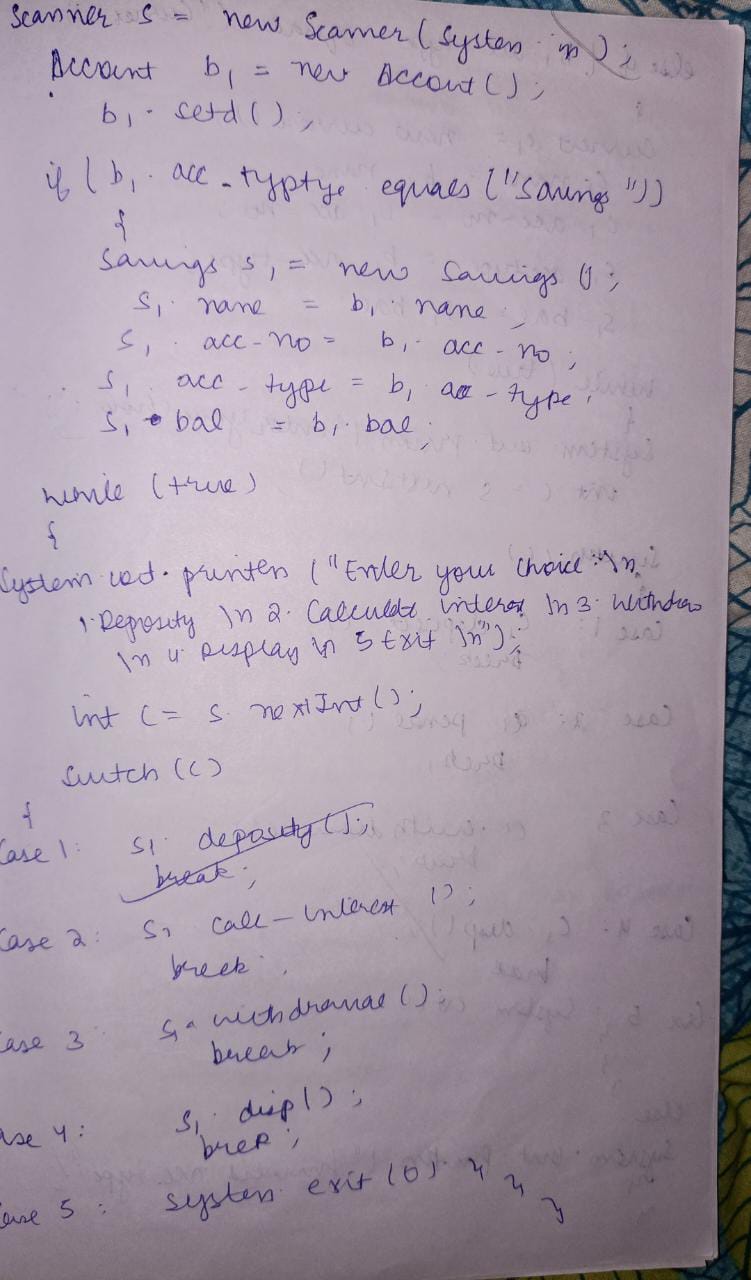
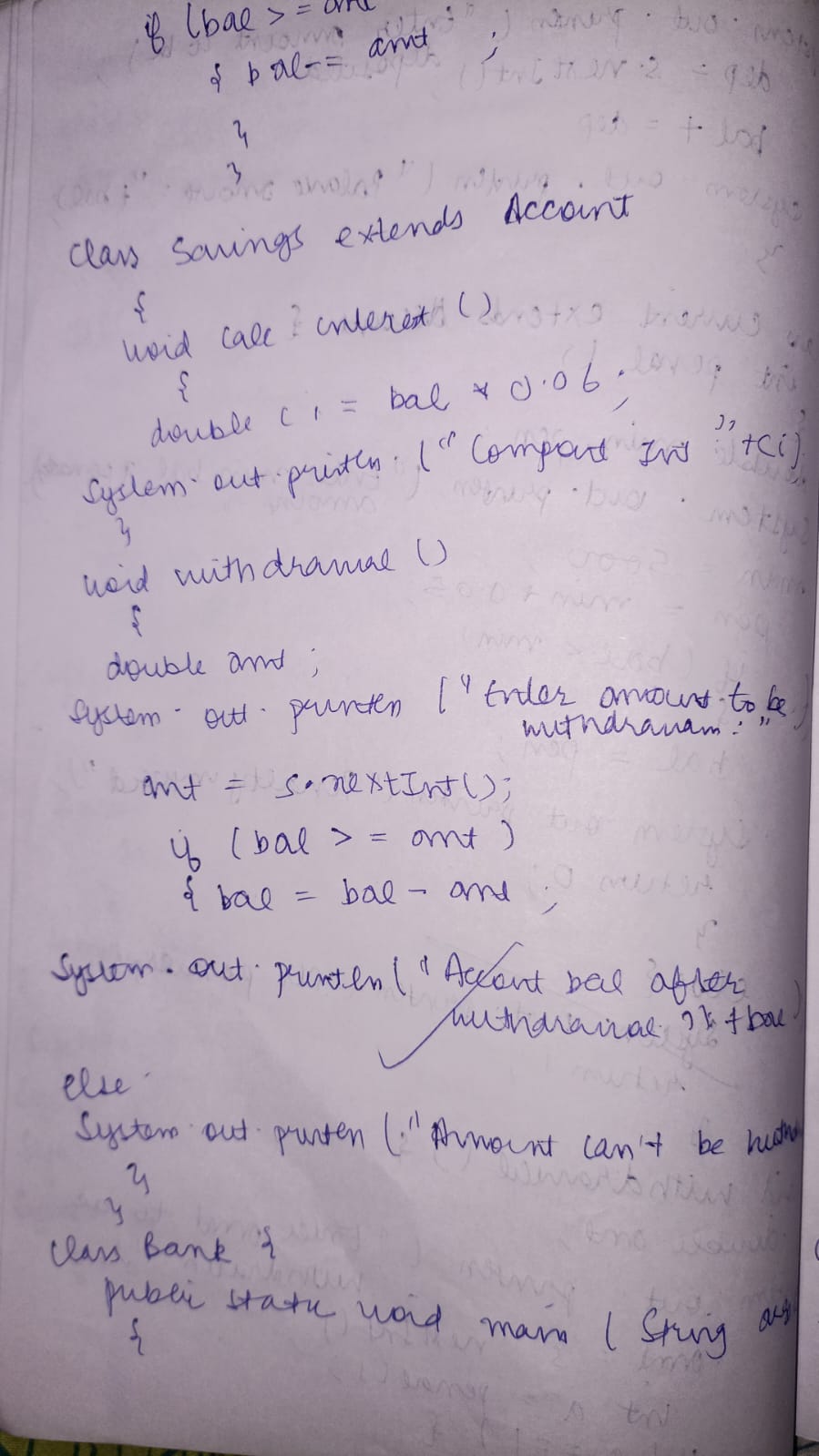
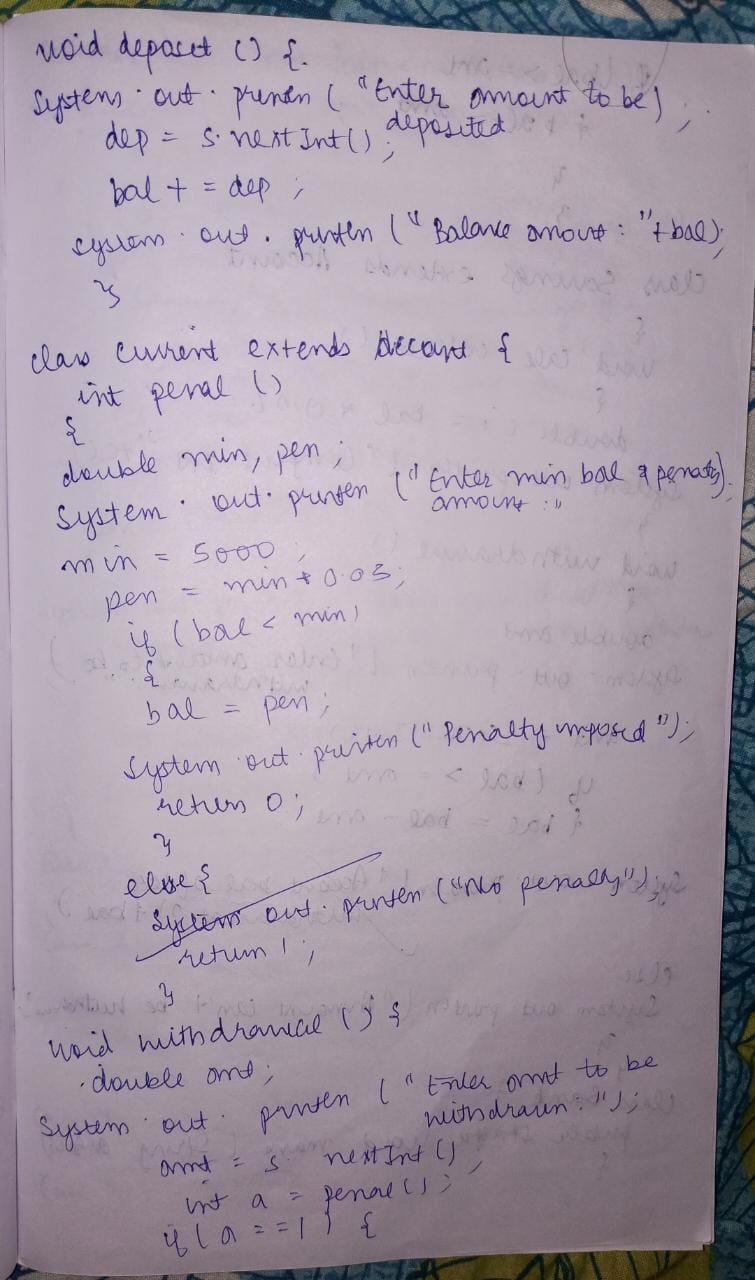
|  |
| --- |
| import java.util.Scanner; |
|  |  |
|  | class Account |
|  | { |
|  | String name; |
|  | int type; |
|  | long accno; |
|  | double balance; |
|  | void setA() |
|  | { |
|  | Scanner s=new Scanner(System.in); |
|  | System.out.print("Enter customer name: "); |
|  | name=s.nextLine(); |
|  |  |
|  | System.out.print("Enter account number: "); |
|  | accno=s.nextLong(); |
|  | System.out.println("Account Balance Should Not Be Less than 5000"); |
|  | System.out.print("Enter bank balance: "); |
|  | balance=s.nextDouble(); |
|  | } |
|  |  |
|  | void display() |
|  | { |
|  | System.out.println("Customer name is: "+name); |
|  | if(type==1) { |
|  | System.out.println("Customer account type is: Savings"); |
|  | } |
|  | else { |
|  | System.out.println("Customer account type is: Current"); |
|  | } |
|  | System.out.println("Customer account number is: "+accno); |
|  | System.out.println("Current balance is: "+balance); |
|  | } |
|  | void deposit() |
|  | { |
|  | System.out.print("Enter the amount to be deposited: "); |
|  | Scanner x=new Scanner(System.in); |
|  | double amt=x.nextDouble(); |
|  | balance+=amt; |
|  | System.out.println("Updated Balance: "+balance); |
|  | } |
|  | } |
|  |  |
|  |  |
|  | class Sav\_acct extends Account |
|  | { |
|  | double interest; |
|  | Scanner s=new Scanner(System.in); |
|  |  |
|  | Sav\_acct() { |
|  | type=1; |
|  | } |
|  | void cinterest() |
|  | { |
|  | int timey; |
|  | float irate; |
|  | int times; |
|  | System.out.println("Compound Interest details:"); |
|  |  |
|  | System.out.println("Enter time in years: "); |
|  | timey=s.nextInt(); |
|  | System.out.println("Enter rate of interest: "); |
|  | irate=s.nextFloat(); |
|  | System.out.println("Enter number of times: "); |
|  | times=s.nextInt(); |
|  | System.out.println("Interest will be compunded "+times+" times a year"); |
|  | interest=balance\*(Math.pow((1+irate/times),(times\*timey))); |
|  | balance+=interest; |
|  | System.out.println("Balance:"+balance); |
|  | } |
|  | void withdraw() |
|  | { |
|  | System.out.println("Enter the amount to be withdrawn: "); |
|  | double amt=s.nextDouble(); |
|  | if(balance>amt) |
|  | {balance-=amt; |
|  | System.out.println("Updated Balance: "+balance); |
|  | } |
|  | else |
|  | {System.out.println("Amount to be withdrawn greater than balance!!!"); |
|  | balance=balance-(balance/10); |
|  | System.out.println("10% penalty has been charged!!!"); |
|  | System.out.println("Updated Balance: "+balance); |
|  | } |
|  | } |
|  |  |
|  | } |
|  |  |
|  | class Curr\_acct extends Account |
|  | { |
|  | double check\_amt; |
|  |  |
|  | Curr\_acct() { |
|  | type=2; |
|  | } |
|  |  |
|  | void cheque() |
|  | { |
|  | System.out.print("Enter the cheque amount: "); |
|  | Scanner s=new Scanner(System.in); |
|  | check\_amt = s.nextDouble(); |
|  | if(check\_amt>balance) |
|  | { |
|  | System.out.println("Rs. 500 penalty imposed...Is it ok to proceed? Enter y for yes and n for no"); |
|  | String option=s.next(); |
|  | if(option.equals("y")) |
|  | {balance=balance-500;} |
|  | else {System.out.println("no Check debited");} |
|  | } |
|  | else |
|  | { |
|  | System.out.println("Rupees "+check\_amt+" debited"); |
|  | balance-=check\_amt; |
|  | System.out.println("Updated Balance: "+balance); |
|  | } |
|  | } |
|  | void withdraw() |
|  | { |
|  | System.out.println("Enter the amount to be withdrawn: "); |
|  | Scanner s=new Scanner(System.in); |
|  | double amt=s.nextDouble(); |
|  | if(balance>amt) |
|  | {balance-=amt; |
|  | System.out.println("Updated Balance: "+balance); |
|  | } |
|  |  |
|  | else |
|  | { System.out.println("Amount to be withdrawn greater than balance!!!"); |
|  | balance=balance-(balance/10); |
|  | System.out.println("10% penalty has been charged!!!"); |
|  | System.out.println("Updated Balance: "+balance);} |
|  | } |
|  | } |
|  |  |
|  | class Bank { |
|  | public static void main(String ss[]) { |
|  | String op1,op2; |
|  | Scanner s=new Scanner(System.in); |
|  | System.out.println("1. Savings or 2. Current"); |
|  | int q; |
|  | q=s.nextInt(); |
|  | if(q==1) { |
|  | Sav\_acct s1 = new Sav\_acct(); |
|  | while(true) { |
|  | System.out.print("\n1. Set the values for Savings Account\n2. Display\n3. Deposit\n4. Interest\n5. Withdraw\n6. Exit\n Cheque Facility Not Available!!\n"); |
|  | System.out.println("Enter the choice: "); |
|  | op1=s.next(); |
|  | switch(op1) |
|  | { |
|  | case "1":s1.setA(); |
|  | break; |
|  | case "2":s1.display(); |
|  | break; |
|  | case "3":s1.deposit(); |
|  | break; |
|  | case "4":s1.cinterest(); |
|  | break; |
|  | case "5":s1.withdraw(); |
|  | break; |
|  | case "6":System.exit(0); |
|  | } |
|  | } |
|  | } |
|  | else if(q==2) { |
|  | Curr\_acct c1 = new Curr\_acct(); |
|  | while(true) { |
|  | System.out.print("Enter the choice: \n1. Set the values for current account\n2. Display\n3. Deposit\n4. Cheque Facility\n5. Withdraw\n6. Exit\n"); |
|  | op2=s.next(); |
|  | switch(op2) |
|  | { |
|  | case "1":c1.setA(); |
|  | break; |
|  | case "2":c1.display(); |
|  | break; |
|  | case "3":c1.deposit(); |
|  | break; |
|  | case "4":c1.cheque(); |
|  | break; |
|  | case "5":c1.withdraw(); |
|  | break; |
|  | case "6":System.exit(0); |
|  | } |
|  | } |
|  | } |
|  | } |
|  | } |

**OUTPUT:**

**Observation Book Pictures:**

****



**PROGRAM-06**

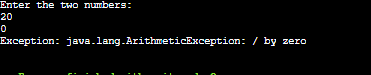
**QUESTION:**

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 NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

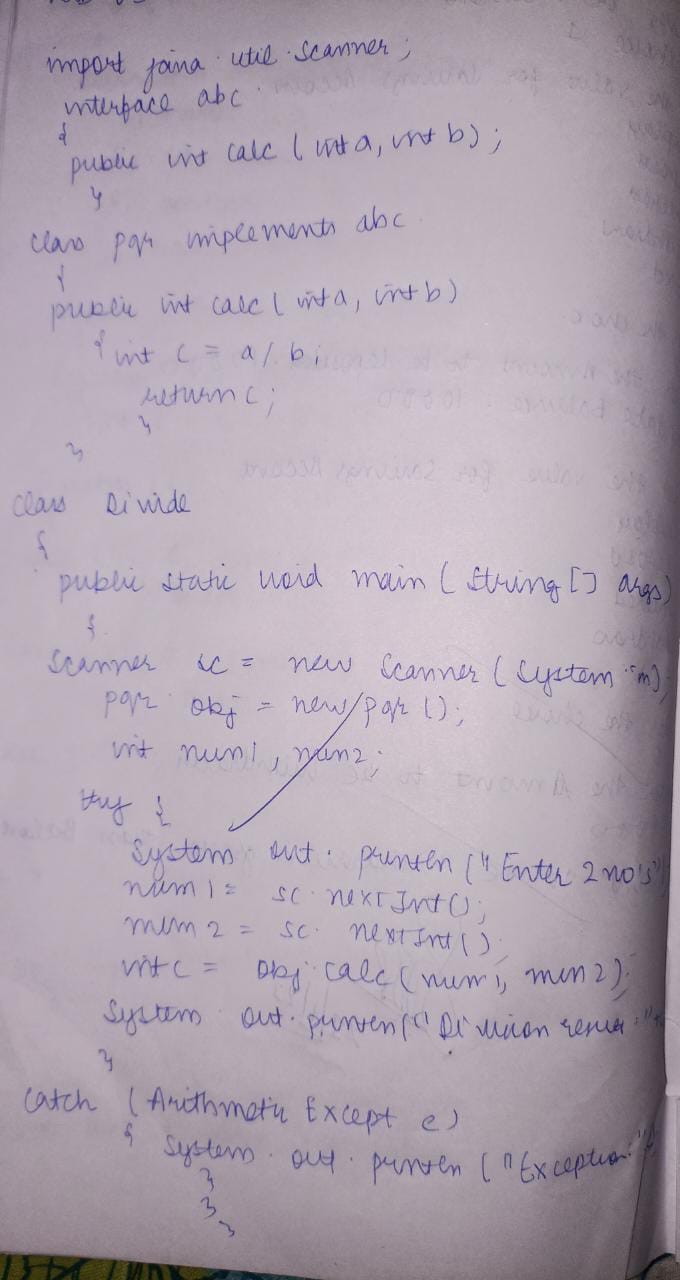
**CODE:**

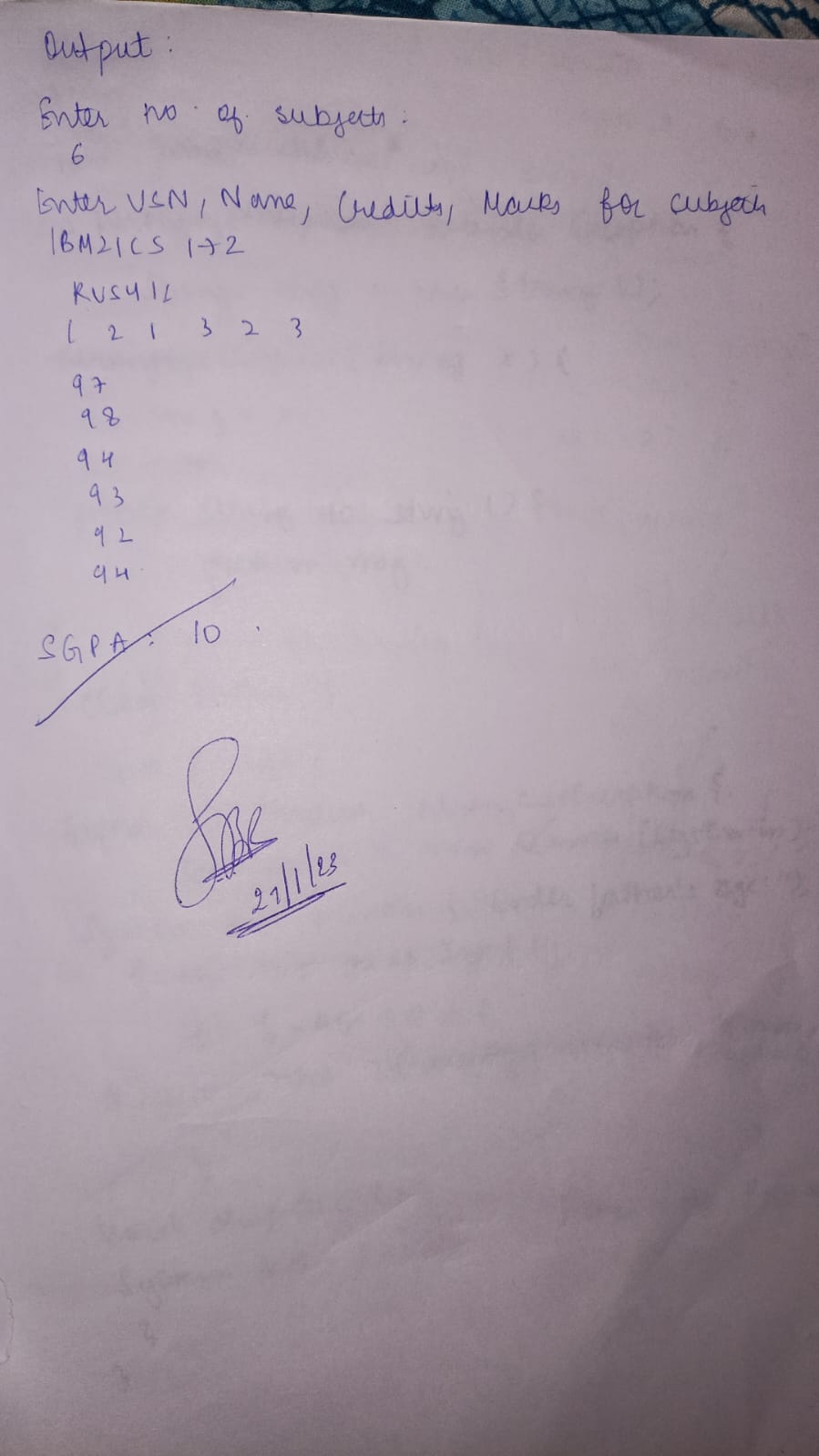
|  |
| --- |
|  |
| import java.util.Scanner; |
|  |  |
|  | interface abc |
|  | { |
|  | public int calc(int a,int b); |
|  | } |
|  |  |
|  | class pqr implements abc |
|  | { |
|  | public int calc(int a, int b) |
|  | { |
|  | int c = a/b; |
|  | return c; |
|  | } |
|  | } |
|  |  |
|  | class Divide |
|  | { |
|  | public static void main(String[] args) |
|  | { |
|  | Scanner sc = new Scanner(System.in); |
|  | pqr obj = new pqr(); |
|  | int num1,num2; |
|  |  |
|  | try |
|  | { |
|  | System.out.println("Enter the two numbers: "); |
|  | num1 = sc.nextInt(); |
|  | num2 = sc.nextInt(); |
|  | int c = obj.calc(num1,num2); |
|  | System.out.println("Division Result: "+c); |
|  | } |
|  |  |
|  | catch(ArithmeticException | NumberFormatException e) |
|  | { |
|  | System.out.println("Exception: "+e); |
|  | } |
|  | } |
|  | } |
|  |  |
|  |  |

**OUTPUT:**



**Observation Book Pictures:**

****



**PROGRAM-07**

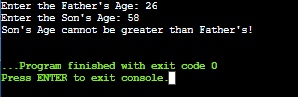
**QUESTION:**

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age=father’s age.

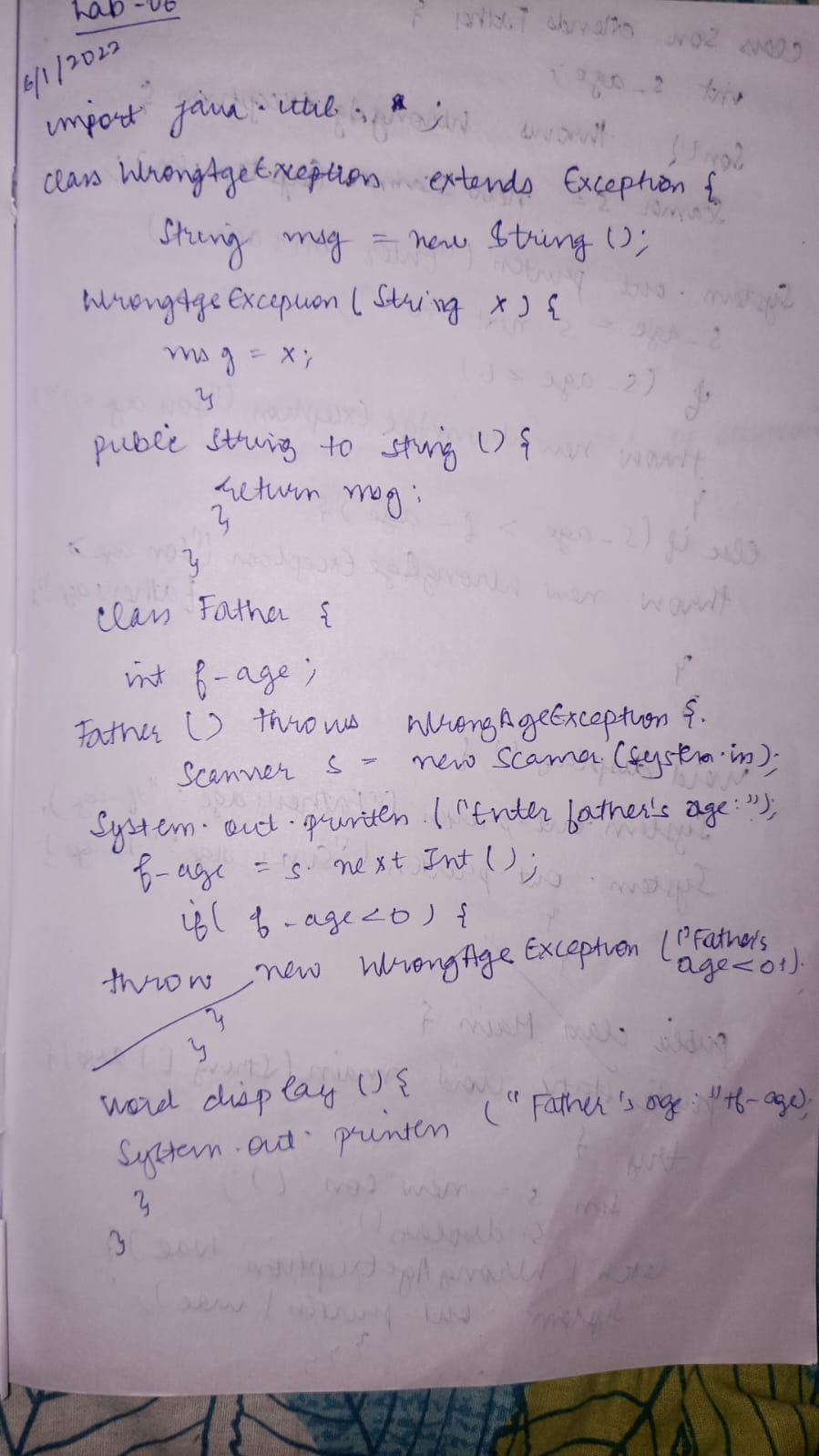
**CODE:**

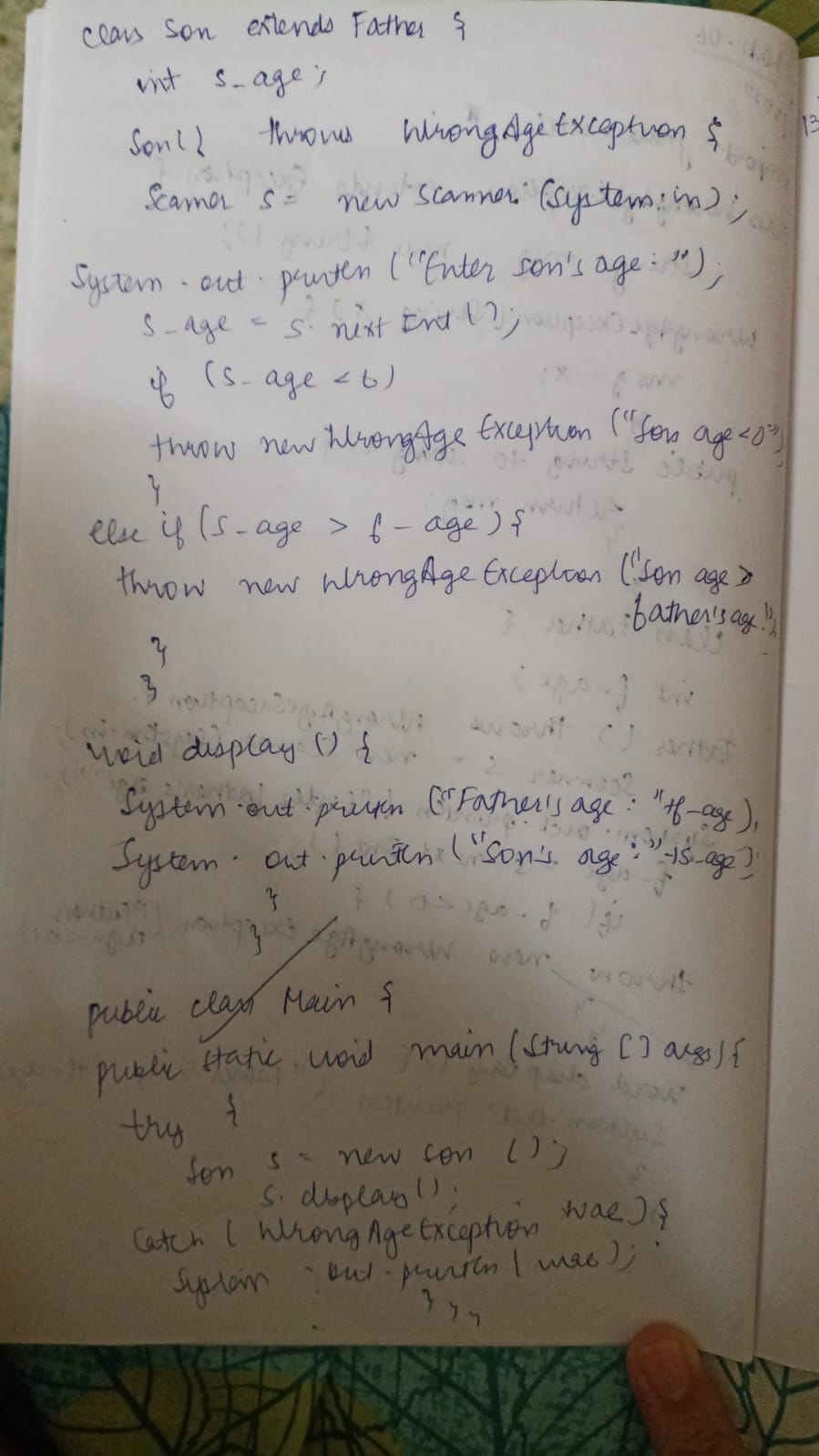
|  |
| --- |
|  |
| import java.util.\*; |
|  |  |
|  | class WrongAge extends Exception |
|  | { |
|  | public String getMessage() |
|  | { |
|  | return "Age Cannot Be Negative"; |
|  | } |
|  | } |
|  |  |
|  | class InvalidAge extends Exception |
|  | { |
|  | public String getMessage() |
|  | { |
|  | return "Son's Age cannot be greater than Father's!"; |
|  | } |
|  | } |
|  |  |
|  | class Father |
|  | { |
|  | Scanner s = new Scanner(System.in); |
|  | int fatherAge; |
|  | Father() throws WrongAge |
|  | { |
|  | System.out.print("Enter the Father's Age: "); |
|  | fatherAge = s.nextInt(); |
|  |  |
|  | try |
|  | { |
|  | if(fatherAge<0) |
|  | throw new WrongAge(); |
|  | } |
|  |  |
|  | catch(WrongAge e1) |
|  | { |
|  | System.out.println(e1.getMessage()); |
|  | System.exit(0); |
|  | } |
|  | } |
|  | } |
|  |  |
|  | class Son extends Father |
|  | { |
|  | int sonAge; |
|  | Son() throws WrongAge,InvalidAge |
|  | { |
|  | super(); |
|  | System.out.print("Enter the Son's Age: "); |
|  | sonAge = s.nextInt(); |
|  | try |
|  | { |
|  | if(sonAge<0) |
|  | throw new WrongAge(); |
|  | } |
|  |  |
|  | catch(WrongAge e2) |
|  | { |
|  | System.out.println(e2.getMessage()); |
|  | } |
|  |  |
|  | try |
|  | { |
|  | if(sonAge>fatherAge) |
|  | throw new InvalidAge(); |
|  | } |
|  |  |
|  | catch(InvalidAge e3) |
|  | { |
|  | System.out.println(e3.getMessage()); |
|  | } |
|  | } |
|  | } |
|  |  |
|  | class Agecheck |
|  | { |
|  | public static void main(String[] args) throws WrongAge,InvalidAge |
|  | { |
|  | new Son(); |
|  | }} |
|  |  |
|  |  |
|  |  |
|  |  |

**OUTPUT:**



**Observation Book Pictures:**

****



**PROGRAM-08**

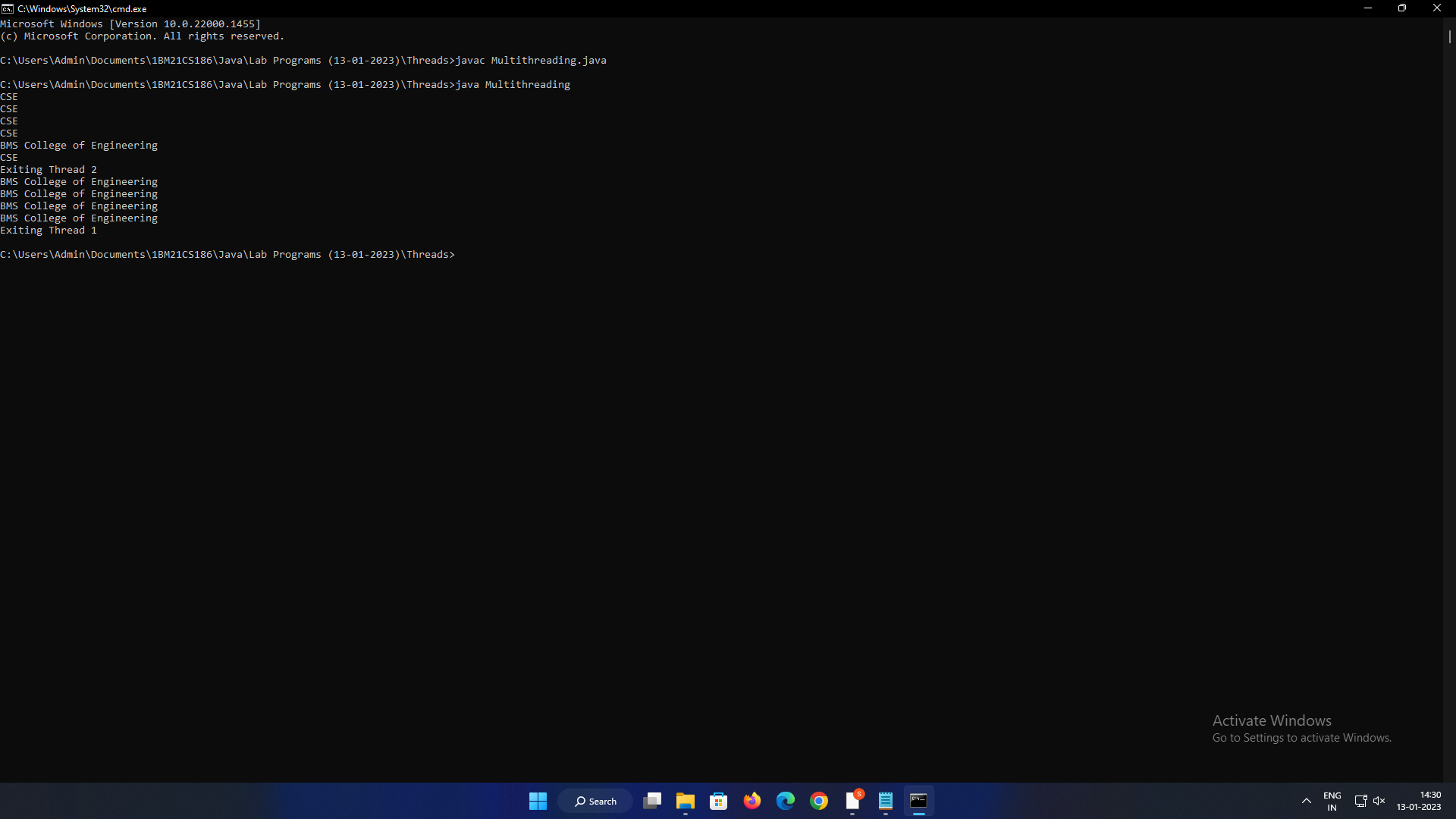
**QUESTION:**

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.

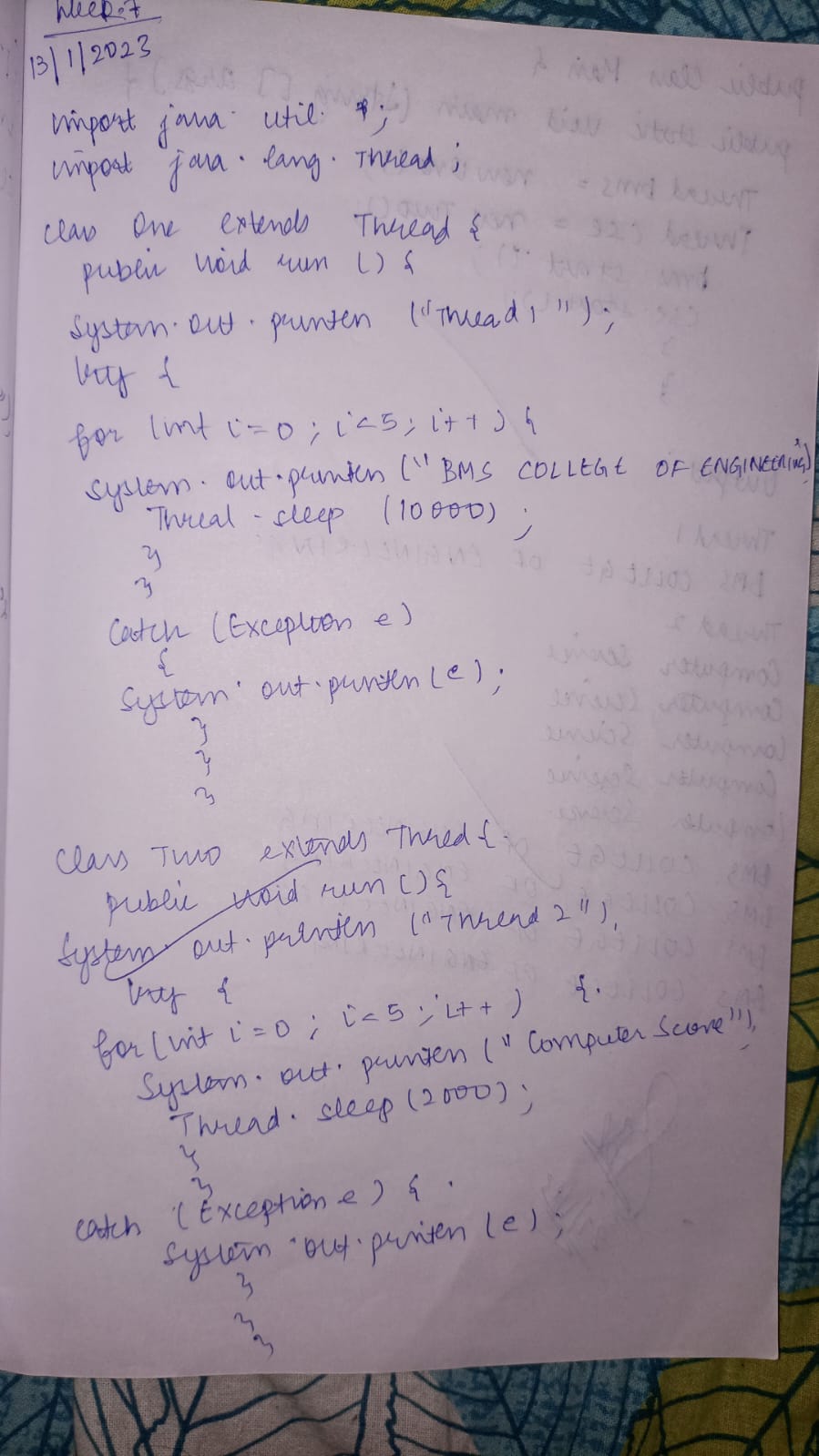
**CODE:**

|  |
| --- |
| import java.util.Scanner; |
|  |  |
|  | class BMSCE extends Thread { |
|  | synchronized public void run() |
|  | { |
|  | try |
|  | { |
|  | int i=0; |
|  | while (i<5) |
|  | { |
|  | sleep(10000); |
|  | System.out.println("BMS College of Engineering "); |
|  | i++; |
|  | } |
|  | } |
|  |  |
|  | catch (Exception e) { |
|  | } |
|  | System.out.println("Exiting Thread 1"); |
|  | } |
|  | } |
|  |  |
|  | class CSE extends Thread |
|  | { |
|  | synchronized public void run() |
|  | { |
|  | try |
|  | { |
|  | int i=0; |
|  | while (i<5) |
|  | { |
|  | sleep(2000); |
|  | System.out.println("CSE"); |
|  | i++; |
|  | } |
|  | } |
|  |  |
|  | catch (Exception e) { |
|  | } |
|  | System.out.println("Exiting Thread 2"); |
|  |  |
|  | } |
|  | } |
|  |  |
|  |  |
|  | class Multithreading |
|  | { |
|  | public static void main(String args[]) |
|  | { |
|  | BMSCE t1 = new BMSCE(); |
|  | CSE t2 = new CSE(); |
|  | t1.start(); |
|  | t2.start(); |
|  | } |
|  | } |

**OUTPUT:**



**Observation Book Pictures:**

****

