VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

T TILAK REDDY (1WA23CS005)

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 **T TILAK REDDY (1WA23CS005),** 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.

Syed Akram Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link:
https://github.com/nivas-bit/lab-programs

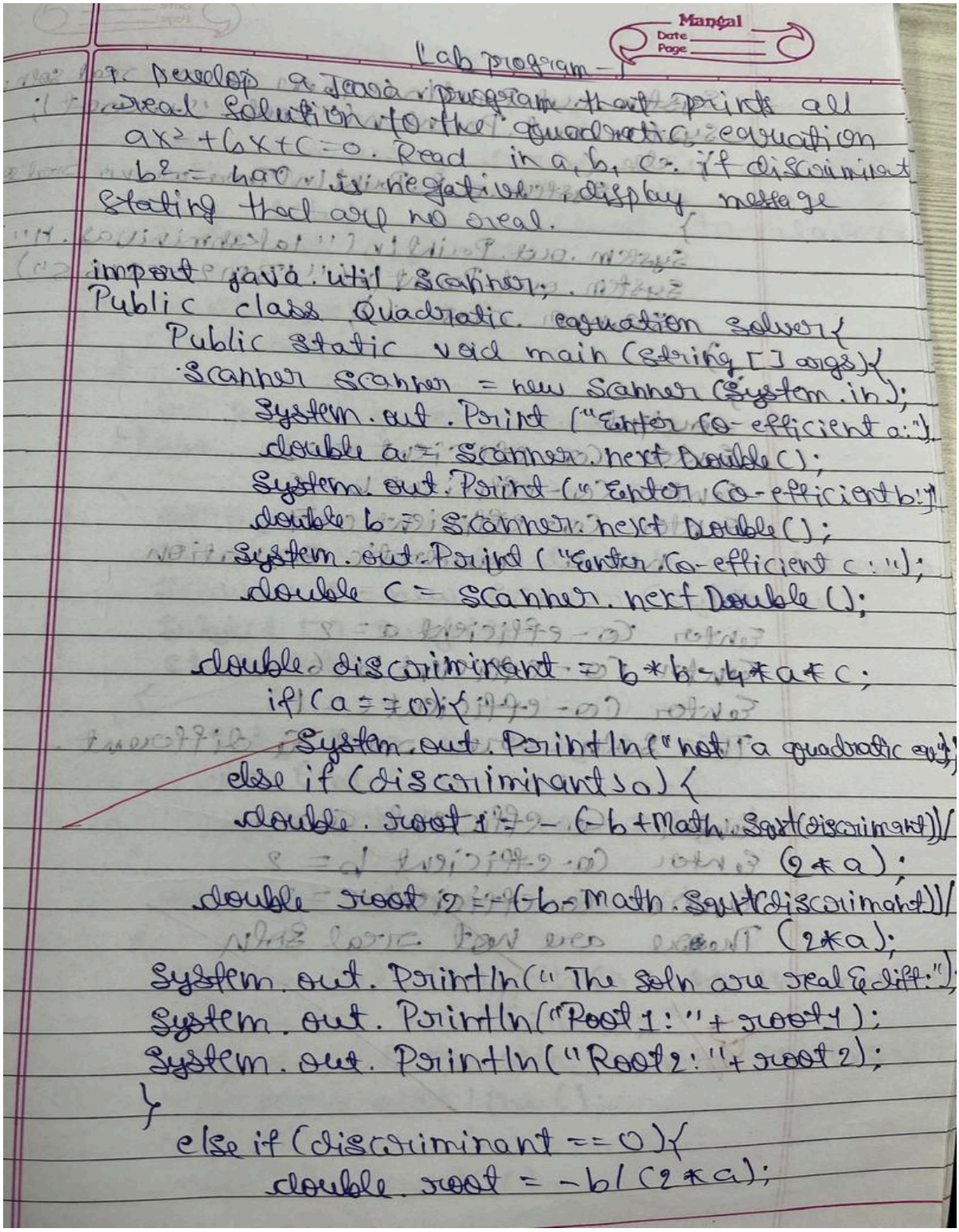
Program 1
Implement Quadratic Equation

```
import java.util.Scanner;

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


      System.out.print("Enter coefficient a: ");
      double a = scanner.nextDouble();

      System.out.print("Enter coefficient b: ");
      double b = scanner.nextDouble();

      System.out.print("Enter coefficient c: ");
      double c = scanner.nextDouble();


      double discriminant = b * b - 4 * a * c;
      if(a==0){
                  System.out.println("not a quadratic equation");}



      else if (discriminant > 0) {

         double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
         double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
         System.out.println("The solutions are real and different:");
         System.out.println("Root 1: " + root1);
         System.out.println("Root 2: " + root2);
      } else if (discriminant == 0) {

         double root = -b / (2 * a);
         System.out.println("There is one real solution:");
```

```
            System.out.println("Root: " + root);
        } else {

            System.out.println("There are no real solutions.");

        }
                    System.out.println("lakshminivas h");
                    System.out.println("1BM23CS165");



        scanner.close();
    }
}
```

```
PS D:\1BM23CS165> javac  QuadraticEquationSolver.java
PS D:\1BM23CS165> java  QuadraticEquationSolver.java
Enter coefficient a: 0
Enter coefficient b: 6
Enter coefficient c: 7
not a quadratic equation
lakshminivas h
1BM23CS165
PS D:\1BM23CS165> javac  QuadraticEquationSolver.java
PS D:\1BM23CS165> java  QuadraticEquationSolver.java
Enter coefficient a: 2
Enter coefficient b: 6
Enter coefficient c: 1
The solutions are real and different:
Root 1: -0.177124344467704 64
Root 2: -2.8228756555322954
lakshminivas h
1BM23CS165
PS D:\1BM23CS165> javac  QuadraticEquationSolver.java
PS D:\1BM23CS165> java  QuadraticEquationSolver.java
Enter coefficient a: 1
Enter coefficient b: 2
Enter coefficient c: 3
There are no real solutions.
lakshminivas h
1BM23CS165
PS D:\1BM23CS165>
```

source code:



Lab program -

1. Develop a Java program that prints all real solution to the quadratic equation $ax^2+bx+c=0$. Read in a, b, c. if discriminant b^2-4ac is negative, display message stating that are no real.

```
import java.util.Scanner;
Public class Quadratic equation solver{
    Public static void main (String [] args){
    Scanner scanner = new Scanner (System.in);
        System.out.Print ("Enter Co-efficient a:");
        double a = scanner.next Double();
        System.out.Print ("Enter Co-efficient b:");
        double b = Scanner.next Double();
    System.out.Print ("Enter Co-efficient c:");
        double c = scanner.next Double();

    double discriminant = b*b-4*a*c;
        if(a==0){
            System.out.Println("not a quadratic eq");
        else if (discriminant>0){
        double root1 = (-b+Math.Sqrt(discriminant))/
                        (2*a);
        double root2 = (-b-Math.Sqrt(discriminant))/
                        (2*a);
    System.out.Println("The Soln are real & diff:");
    System.out.Println("Root1:"+root1);
    System.out.Println("Root2:"+root2);
    }
      else if (discriminant==0){
          double root = -b/(2*a);
```

```
System.out.Println("There is one real soln:");
System.out.Println("Root:"+sroot);
} else {
    System.out.Println("There are no real soln:");
}
System.out.Println("lakshminivas.H");
System.out.Println("1BM23CS146");
Scanner.close();
}
}
```

output

Enter Co-efficient a=0
Enter Co-efficient b=6
Enter Co-efficient c=7
not a quadratic Equation

Enter Co-efficient a=2
Enter Co-efficient b=6
Enter Co-efficient c=1
The Soln are real & different.

Enter Co-efficient a=1
Enter Co-efficient b=2
Enter Co-efficient c=3
There are not real Soln

program 2:
sgpa of a student

```
import java.util.Scanner;

class Student {
    String usn;
    String name;
    int credits[];
    int marks[];
    int numSubjects;


    Student(int numSubjects) {
        this.numSubjects = numSubjects;
        credits = new int[numSubjects];
        marks = new int[numSubjects];
    }


    void acceptDetails() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = scanner.nextLine();
        System.out.print("Enter Name: ");
        name = scanner.nextLine();
        for (int i = 0; i < numSubjects; i++) {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = scanner.nextInt();
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
```

```
            marks[i] = scanner.nextInt();
        }
    }


    void displayDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        for (int i = 0; i < numSubjects; i++) {
            System.out.println("Subject " + (i + 1) + " - Credits: " +
credits[i] + ", Marks: " + marks[i]);
        }
    }


    double calculateSGPA() {
        int totalCredits = 0;
        double totalPoints = 0.0;

        for (int i = 0; i < numSubjects; i++) {
            int gradePoint = getGradePoint(marks[i]);
            totalPoints += gradePoint * credits[i];
            totalCredits += credits[i];
        }

        return totalPoints / totalCredits;
    }


    int getGradePoint(int marks) {
        if (marks >= 90) return 10;
```

```
    else if (marks >= 80) return 9;
    else if (marks >= 70) return 8;
    else if (marks >= 60) return 7;
    else if (marks >= 50) return 6;
    else if (marks >= 40) return 5;
    else return 0;
}

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


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


    System.out.print("Enter number of subjects per student: ");
    int numSubjects = scanner.nextInt();


    Student[] students = new Student[numStudents];


    for (int i = 0; i < numStudents; i++) {
     System.out.println("\nEntering details for Student " + (i + 1));
        students[i] = new Student(numSubjects);
        students[i].acceptDetails();
    }


    for (int i = 0; i < numStudents; i++) {
```

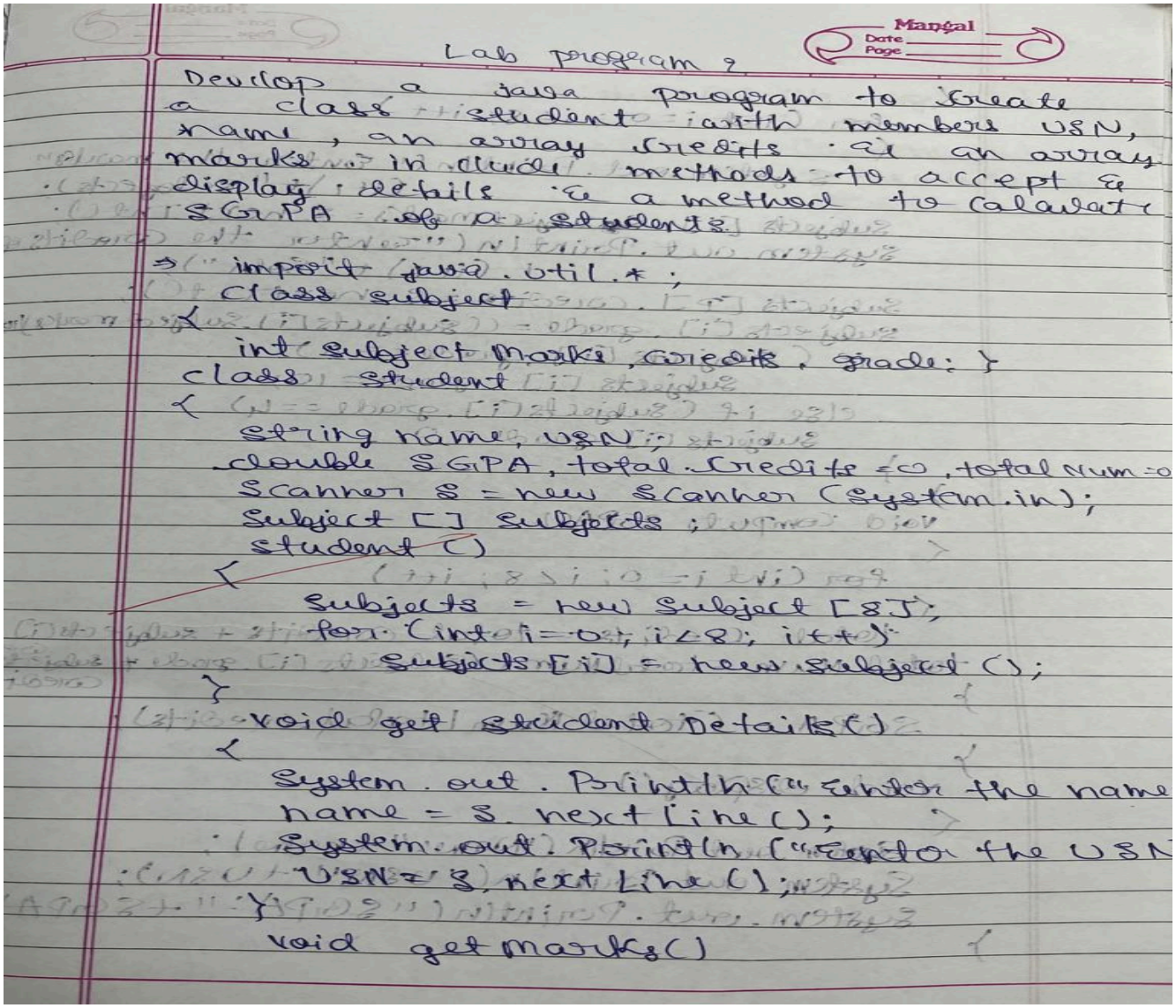
```
            System.out.println("\nDetails of Student " + (i + 1));
            students[i].displayDetails();
            double sgpa = students[i].calculateSGPA();
            System.out.printf("SGPA: %.2f\n", sgpa);
        }
    System.out.println("Name : LAKSHMINIVAS H ");
    System.out.println("USN : 1BM23CS165");
     }
}
```

```
Enter number of students: 2
Enter number of subjects per student: 2

Entering details for Student 1
Enter USN: 1BM23CS165
Enter Name: NIVAS H
Enter credits for subject 1: 4
Enter marks for subject 1: 90
Enter credits for subject 2: 4
Enter marks for subject 2: 95

Entering details for Student 2
Enter USN: 1BM23CS167
Enter Name: VINAY
```

```
Enter credits for subject 1: 4
Enter marks for subject 1: 88
Enter credits for subject 2: 3
Enter marks for subject 2: 99

Details of Student 1
USN: 1BM23CS165
Name: NIVAS H
Subject 1 - Credits: 4, Marks: 90
Subject 2 - Credits: 4, Marks: 95
SGPA: 10.00

Details of Student 2
USN: 1BM23CS167
```

```
Details of Student 2
USN: 1BM23CS167
Name: VINAY
Subject 1 - Credits: 4, Marks: 88
Subject 2 - Credits: 3, Marks: 99
SGPA: 9.43
Name : LAKSHMINIVAS H
USN : 1BM23CS165
```

SOURCE CODE:



Lab program 8

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.

```
import java.util.*;
class Subject
{
    int subjectMarks, credits, grade; }
class Student
{
    String name, usn;
    double SGPA, totalCredits = 0, totalNum = 0
    Scanner s = new Scanner(System.in);
    Subject[] subjects;
    Student()
    {
        subjects = new Subject[8];
        for(int i=0; i<8; i++)
            subjects[i] = new Subject();
    }
    void getStudentDetails()
    {
        System.out.println("Enter the name
        name = s.nextLine();
        System.out.println("Enter the USN
        usn = s.nextLine();
    }
    void getMarks()
```

```
{
for (int i=0; i<8; i++)
{
System.out.Println("Enter the marks" + (i+1) + "subjects");
Subjects[i].SubjectMarks = S.nextInt();
System.out.Println("Enter the Credits of" + (i+1) + "Sub");
Subjects[i].credits = S.nextInt();
Subjects[i].grade = ((Subjects[i].SubjectMarks/10) + 1);
if (Subjects[i].grade == 11)
    Subjects[i].grade = 10;
else if (Subjects[i].grade == 4)
    Subjects[i].grade = 0;
}
}
void computeSGPA()
{
for (int i=0; i<8; i++)
{
totalCredits = totalCredits + subjects[i].credits;
totalNum = totalNum + (Subjects[i].grade * subjects[i].credits);
}
SGPA = (totalNum / totalCredits);
}
void display()
{
System.out.Println("Name:" + name);
System.out.Println("USN:" + USN);
System.out.Println("SGPA:" + SGPA);
}
```

```
Public static void main(String args[])
{
  Scanner sc = new Scanner(System.in);
  System.out.Println("Enter the number of students");
  int n;
  n = sc.nextInt();
  Student[] s = new Student[n];
  for(int i=0; i<n; i++)
  {
    s[i] = new student();
  }
  for(int i=0; i<2; i++)
  {
    s[i].getstudentDetails();
    s[i].getMarks();
    s[i].computeSGPA();
    s[i].display();
  }
}
}
```

output

Enter the number of students

2

Enter the name;

Nivas

Enter the USN

1BM23CS165

Enter the marks 1 subject

Enter the .credits 1 subject

Enter the marks 2 subject

Enter the credits 2 subject

Enter the marks 3 Subject
Enter the credits 3 Subject
Enter the marks 4 Subject
Enter the credits 4 Subject
Enter the marks 5 Subject
Enter the credits 5 Subject
Enter the marks 6 Subject
Enter the credits 6 Subject
Enter the marks 7 Subject
Enter the credits 7 Subject
Enter the marks 8 Subject
Enter the credits 8 Subject.

PROGRAM 3
to create n book objects

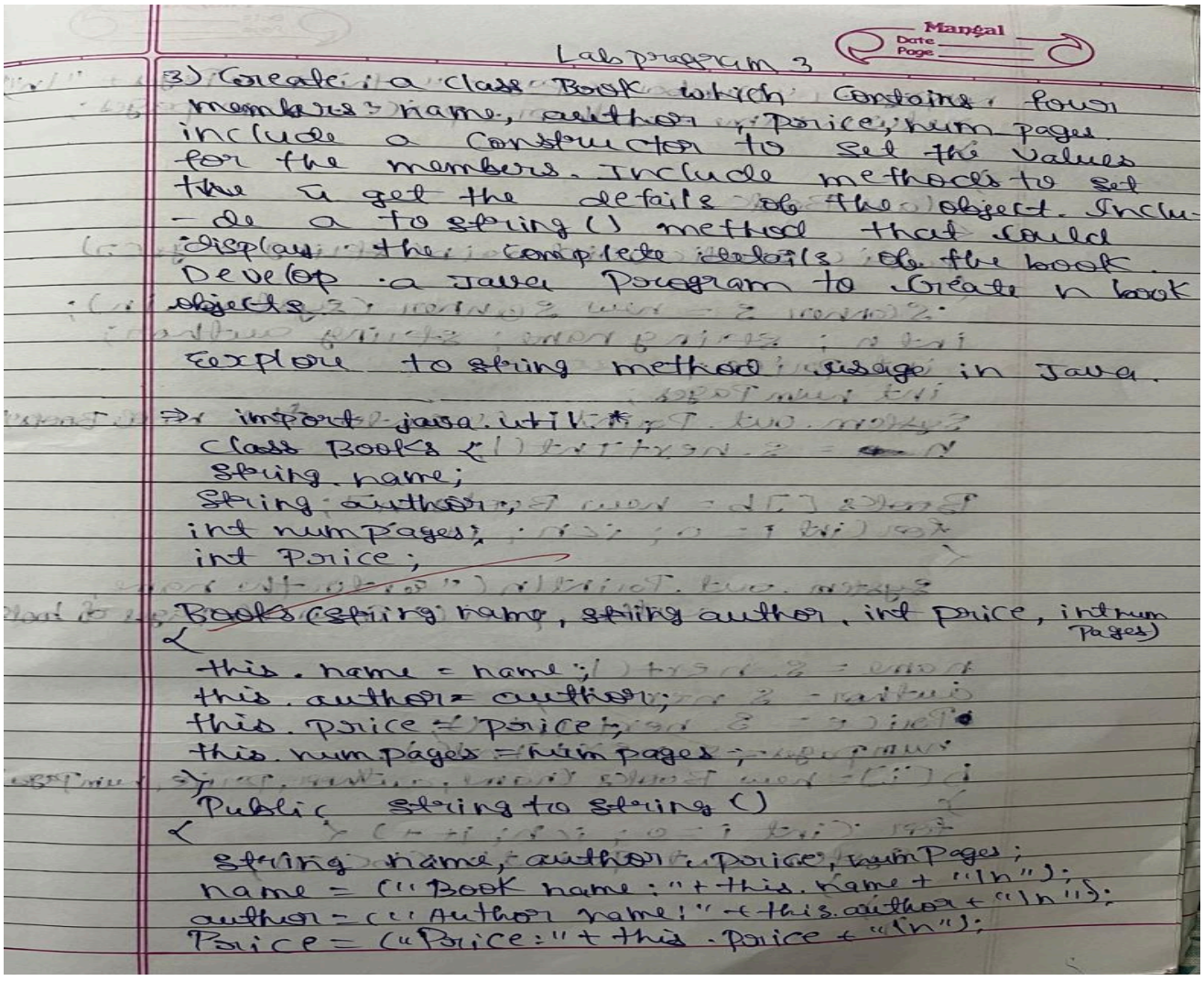
```
import java.util.*;
class Books{
String name,author;
int numPages,price;


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

 class Book
{

public static void main(String args[])

{

Scanner s = new Scanner(System.in);

int n; String name; String author; int price; int numPages;

System.out.println(" enter the number of books");
n = s.nextInt();
```

```
Books[] b = new Books[n];
for ( int i=0;i<n;i++)
{
System.out.println(" enter the name,author, price and number of pages of books");

name=s.next();
author=s.next();
price=s.nextInt();
numPages=s.nextInt();
b[i]=new Books(name,author,price,numPages);

}
for(int i=0;i<n;i++){
System.out.println(b[i]);

      }

    }

  }
```

```
 enter the number of books
2
 enter the name,author, price and number of pages of books
science
nivas
100
120
 enter the name,author, price and number of pages of books
maths
vinay
234
134
Book name: science
Author name: nivas
```

```
Price: 100
Number of pages: 120

Book name: maths
Author name: vinay
Price: 234
Number of pages: 134

name : LAKSHMINIVAS H
usn  : 1BM23CS165
```

source code:



```
numpages=("Number of Pages: "+this.numPages+"\n");
return name+author+price+numpages;
}
}
class Book{
public static void main(String args[])
{
Scanner s = new Scanner(System.in);
int n; String name; String author;
int price;
int numPages;
System.out.println("Enter the no of Books");
n = s.nextInt();

Books []b = new Books[n];
for(int i=0; i<n; i++)
{
System.out.println("Enter the name, author, price & no of pages of book");

name = s.next();
author = s.next();
Price = s.nextInt();
numpages = s.nextInt();
b[i] = new Books(name, author, price, numpages);
}
for(int i=0; i<n; i++){
System.out.println(b[i]);
}
}
```

output

Enter the number of books
3
Enter the name, author, price & number of Pages of books.
Science
Nivas
100
120

Enter the name, author, price. & number of Pages of books
maths
Vinay
234
199

Book name: Science
Author name: Nivas
Price: 100
Number of Pages: 120

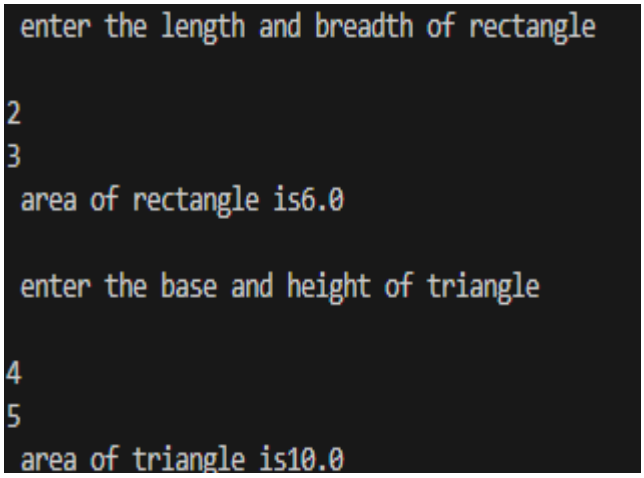
Book name: maths
Author name: Vinay
Price: 234
Number of Pages: 120

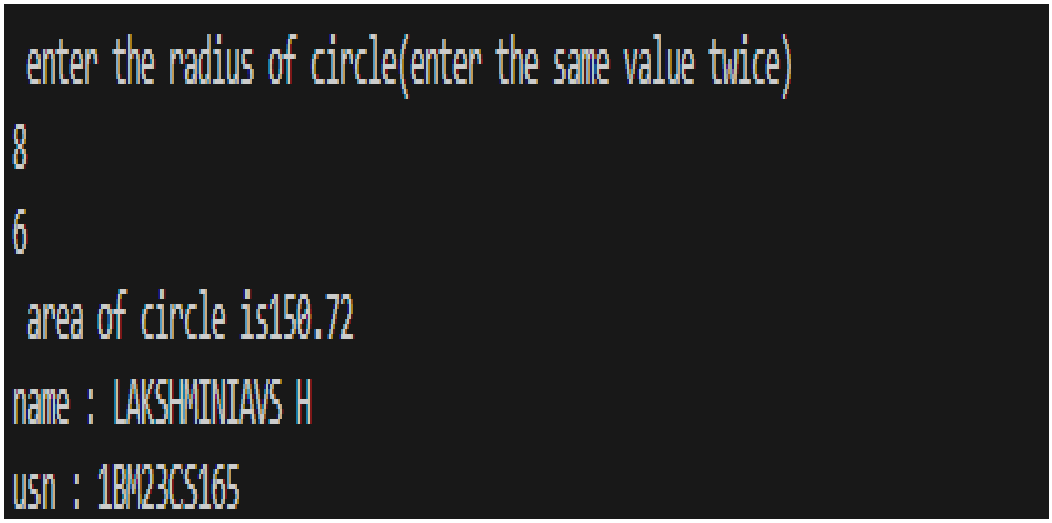
14/10/24

PROGRAM 4
print the area of the given shape

```
import java.util.*;
abstract class shape{
double x,y,area;

void accept(){
Scanner s=new Scanner(System.in);
x=s.nextDouble();
y=s.nextDouble();
}
abstract void printArea();
}

class rectangle extends shape{
void printArea(){
area = x*y;
System.out.println(" area of rectangle is"+ area);

}}
class triangle extends shape{
void printArea(){
area=0.5*x*y;
System.out.println(" area of triangle is"+ area);

}}
class circle extends shape{
void printArea(){
area = 3.14*x*y;
System.out.println(" area of circle is"+ area);

}}
class Area{
public static void main(String args[]){
 rectangle r= new rectangle();
System.out.println(" \n enter the length and breadth of rectangle\n ");
r.accept();
r.printArea();
```

```
 triangle t= new triangle();
System.out.println("\n enter the base and height of triangle \n");
t.accept();
t.printArea();
 circle c= new circle();
System.out.println("\n enter the radius of circle(enter the same value twice)");
c.accept();
c.printArea();

System.out.println("name : LAKSHMINIAVS H");
System.out.println("usn : 1BM23CS165");

}}
```





source code:

Mangal
Date
Page

Lab 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 & 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.

```
import java.util.*;
abstract class shape{
  double x, y, area;

  void accept(){
  Scanner s = new Scanner (System.in);
    x = s.nextDouble();
    y = s.nextDouble();
  }
  abstract void printArea();
  }
    class rectangle extends shape{
      void printArea(){
        area = x*y;
        System.out.println("area of rectangle is"+area);
      }
  }
  class triangle extends shape{
    void printArea(){
      area = 3.14*x*y;
    System.out.println("area of circle is"+area);
  }}
```

```
class Area
Public static void main (String args []) {
    rectangle r = new .rectangle ();
System. out. Print ln ("\n enter the length & breadt
                        of rectangle \n");
    r. accept ();
    r. Print Area ();
      triangle t = new triangle ();
System. out. Print ln ("\n enter the base & heigh
                        of height of triangle \n")
    t. accept ();
    t. Print Area ();
      Circle c = new .circle ();
System. out. Print ln ("\n enter the radius of Circle"
    c. accept ();
    c. print Area ();
}}
```

output

enter the length & breadth of rectangle

2

3

area of rectangle is 6.

enter the base & heigh of triangle

4

5

area of triangle is 10

enter the radius of circle

8

6

area of circle is 150.72.

25/2/10/24

PROGRAM 5:
create a bank class

```
import java.util.Scanner;

class Account {
    private String customerName;
    private String accountNumber;
    private double balance;

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

    public String getCustomerName() {
        return customerName;
    }

    public String getAccountNumber() {
        return accountNumber;
    }

    public double getBalance() {
        return balance;
    }

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

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

    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
```

```
            System.out.println("Withdrawn: " + amount);
        } else {
            System.out.println("Insufficient funds.");
        }
    }
}

class SavingsAccount extends Account {
    private double interestRate;

    public SavingsAccount(String customerName, String accountNumber, double
interestRate) {
        super(customerName, accountNumber);
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest() {
        double interest = getBalance() * interestRate / 100;
        deposit(interest);
        System.out.println("Interest of " + interest + " added.");
    }
}

class CurrentAccount extends Account {
    private double minimumBalance;
    private double serviceCharge;

    public CurrentAccount(String customerName, String accountNumber, double
minimumBalance, double serviceCharge) {
        super(customerName, accountNumber);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }

    @Override
    public void withdraw(double amount) {
        super.withdraw(amount);
        checkMinimumBalance();
    }
```

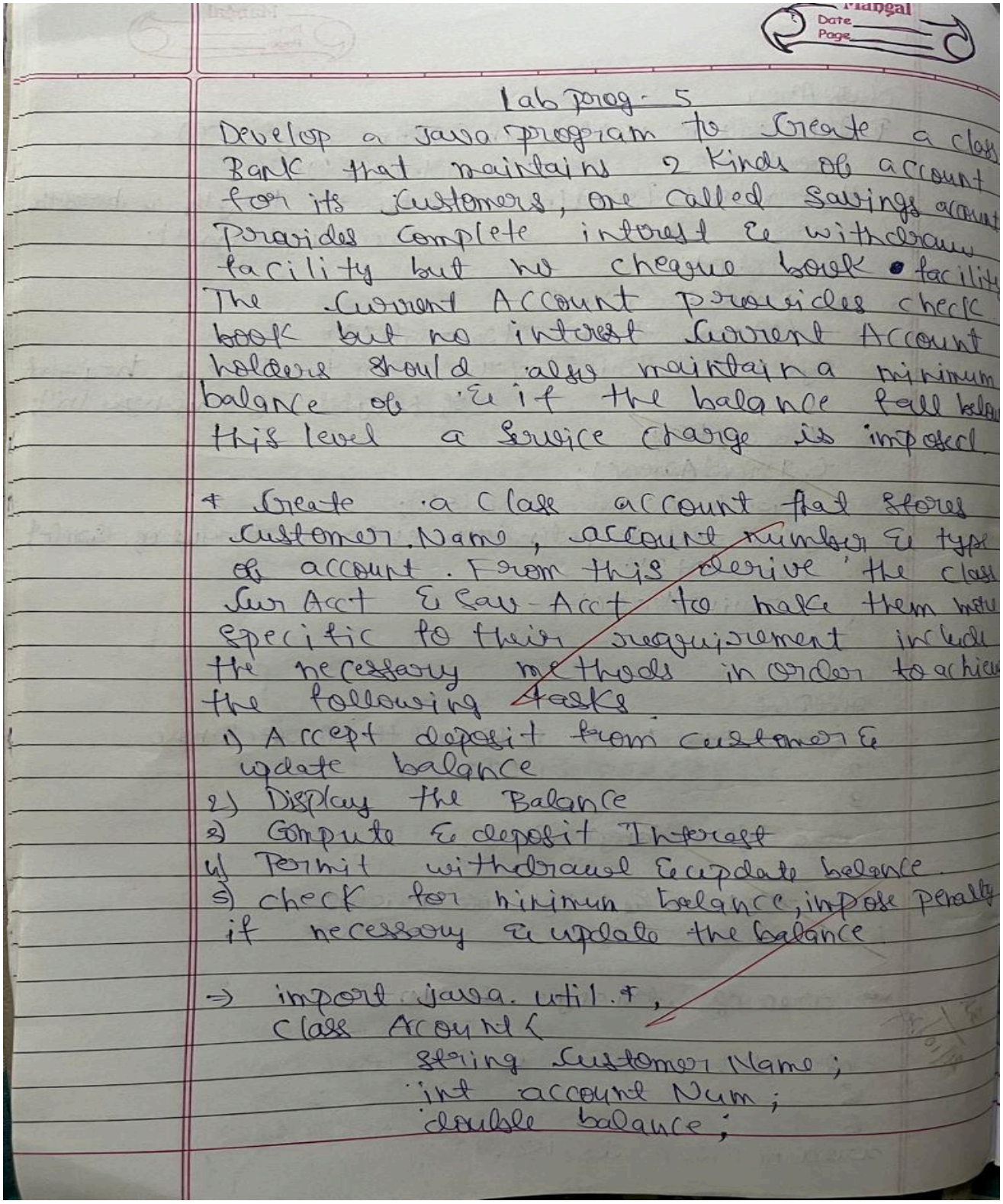
```
    private void checkMinimumBalance() {
        if (getBalance() < minimumBalance) {
            super.withdraw(serviceCharge);
            System.out.println("Service charge of " + serviceCharge + " applied for low
balance.");
        }
    }
}

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 account type (savings/current): ");
        String accountType = scanner.nextLine();

        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();
        System.out.print("Enter account number: ");
        String accountNumber = scanner.nextLine();

        Account account;

        if (accountType.equalsIgnoreCase("savings")) {
            System.out.print("Enter interest rate: ");
            double interestRate = scanner.nextDouble();
            account = new SavingsAccount(customerName, accountNumber, interestRate);
        } else {
            System.out.print("Enter minimum balance: ");
            double minimumBalance = scanner.nextDouble();
            System.out.print("Enter service charge: ");
            double serviceCharge = scanner.nextDouble();
            account = new CurrentAccount(customerName, accountNumber,
minimumBalance, serviceCharge);
        }

        while (true) {
            System.out.println("\n1. Deposit");
            System.out.println("2. Withdraw");
```

```
            System.out.println("3. Display Balance");
            if (account instanceof SavingsAccount) {
                System.out.println("4. Compute and Deposit Interest");
            }
            System.out.println("5. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter amount to deposit: ");
                    double depositAmount = scanner.nextDouble();
                    account.deposit(depositAmount);
                    break;
                case 2:
                    System.out.print("Enter amount to withdraw: ");
                    double withdrawAmount = scanner.nextDouble();
                    account.withdraw(withdrawAmount);
                    break;
                case 3:
                    account.displayBalance();
                    break;
                case 4:
                    if (account instanceof SavingsAccount) {
                        ((SavingsAccount) account).computeAndDepositInterest();
                    } else {
                        System.out.println("This option is not available for current accounts.");
                    }
                    break;
                case 5:
                    System.out.println("Thank you for using the bank. Goodbye!");
                    scanner.close();
                    return;
                default:
                    System.out.println("Invalid option. Please try again.");
            }
        }
    }
}
```

```
C:\BMSCE\OOJ>javac Bank1.java

C:\BMSCE\OOJ>java Bank1
Enter the Name and Account Number And Balance, Interest Rate
Likhith D
21
1000
Which type of account do you have?
 1.Saving Account
 2.Current Account
1
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
1
Enter the amount :
500
Balance after Deposit = 1500.0
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
2
Enter the amount :
300
Balance after Withdraw = 1200.0
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
3
Interest is 60.0
YOUR NEW BALANCE IS 1260.0
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
4
Customer Name Likhith D
Account Number 21
Account Type 1
Balance = 1200.0
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
5
Invalid choice
```

```
C:\BMSCE\OOJ>javac Bank1.java

C:\BMSCE\OOJ>java Bank1
Enter the Name and Account Number And Balance, Interest Rate
Likhith D
22
700
Which type of account do you have?
 1.Saving Account
 2.Current Account
2
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
1
Enter the amount :
200
Balance after Deposit = 900.0
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
2
Enter the amount :
500
Service charge will be imposed
Balance after Service charge imposed = 350.0
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
3
not possible for current account
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
4
Customer Name Likhith D
Account Number 22
Account Type 2
Balance = 350.0
             MENU
1.Deposit
 2.withdraw
3.Interest Calculation
4.Account Details
5.Exit
5
Invalid choice
```

source code:



Lab prog - 5

Develop a java program to create a class Bank that maintains 2 kinds of account for its customers, one called savings account provides complete interest & withdraw facility but no cheque book facility. The Current Account provides check book but no interest Current Account holders should also maintain a minimum balance of & if the balance fall below this level a service charge is imposed.

* Create a class account that stores customer Name, account number & type of account. From this derive the class Cur Acct & Sav-Acct to make them more specific to their requirement include the necessary methods in order to achieve the following tasks.
  1) Accept deposit from customer & update balance
  2) Display the Balance
  3) Compute & deposit Interest
  4) Permit withdrawal & update balance
  5) check for minimum balance, impose penalty if necessary & update the balance

⇒ import java.util.*;
Class Account {
    String customerName;
    int accountNum;
    double balance;

```
{
    Account (String customerName, int accountNum,
                    double balance)
    this.customerName = customerName;
    this.accountNum = accountNum;
    this.balance = balance;
}
    void deposit (double amount){
        balance = balance+ amount;
        System.out.println("Balance After
                                Deposit =" + Balan
    }
    void withdraw (double amount){
        if (amount > balance)
        System.out.println("Insufficient funds");
        else
        {
        balance = balance - amount;
        System.out.println("Balance after withdraw =
                                + Balance);
        }
    }
    void getBalance(){
        System.out.println("Balance =". + Balance);
    }
}

class SavingAccount extends Account{
    double interestRate;
SavingAccount(String customerName, int accountNumber
        double balance, double interestRate){
    super(customerName, accountNumber, balance);
        this.interestRate = interestRate;
    }
```

```
void calcInterest(){
    double interest = (super.balance * interestRate)/100;
    System.out.println("Interest is" + interest);
    System.out.println("your new Balance is" +
                        (balance + interest));}}

class CurrentAccount extends Account{
        double MIN_BALANCE = 500.0;
        double SERVICE_CHARGE = 500;

CurrentAccount(String CustomerName, int accountNumber,
                        double balance){
    super(CustomerName, accountNumber, balance);
    }
    void withdraw(double amount){
      if(amount>balance)
        System.out.println("Insufficient funds");
      else if((balance-amount)<MIN_BALANCE){
        System.out.println("Service charge will be imposed");
      balance=balance-SERVICE_CHARGE-amount;
        System.out.println("Balance after service charge
                        imposed=" + balance);
    }
    else{ balance=balance-amount;
        System.out.println("Balance after withdraw=" +balance);
    }}}
Class Bank1
{ Public static void main(String args[]){
    Scanner s= new Scanner(System.in);
    System.out.println("Enter the Name, AccountNum,
                        Balance, InterestRate");
    String CustomerName = s.nextLine();
```

```
int accountNum = S.nextDouble();
double balance = S.nextDouble();
  double interestRate = 5;
SavingsAccount Sa = new SavingAccount(CustomerName,
                    accountNum, balance, interestRate);
CurrentAccount Ca = new CurrentAccount(Customer
                    Name, accountNum, balance);
System.out.Println("which type of account do you
                    have ? (n)
    1) Saving Account \n
    2) Current Account");
  int type = S.nextINT();
    While(true){
    System.out.Println("---- MENU ---- \n
      1. Deposit \n    3. Interest Calculation
      2. withdraw \n   4. Account Details \n
                       5. Exit");
  int choice = S.nextInt();
    int count = 0;
  Switch(choice){
Case 1 : { System.out.Println("Enter the amount:");
      double amount = S.nextDouble();
        if (type == 1)
        Sa.deposit(amount);
        else
        Ca.deposit(amount);
        break;
      }
Case 2 : {
        System.out.Println("Enter the amount:");
        double amount = S.nextDouble();
        if (type == 1)
```

```
Sa.withdraw(amount);
else
Ca.withdraw(amount);
 break;
}
Case 3: {
    if (type == 1)
    Sa.CalcInterest();
    else
    System.out.Println("not possible for
                        Current Account");
        break;
    }
Case 4: {
    System.out.Println("Customer Name"+CustomerName
    System.out.println("Account Number"+account
    System.out.println("Account type"+type);
    if (type == 1)
        Sa.getBalance();
    else
        Ca.getBalance();
        break; }
default:
    System.out.println("Invalid choice");
        count = 1;
        break;
    }
    if (count == 1)
        break;
}
}
}
```

output :

Enter the Name and Account Number and Balance, Interest Rate

Nival. H

21

1000

Which type of account do yo have ?

1) Saving Account

2) Current Account

1

---- MENU ----

1. Deposit

2. withdraw

3. Interest Calculation

4. Account Details

5. Exiet

1

Enter the amount

500

Balance after Deposit = 1500.0

---- MENU ----

1 Deposit

2. withdraw

3. Interest Calculation

4. Account Details

5. Exiet

2

Enter the amount

300

Balance after withdraw = 1200.0

---- MENU ----

1. Deposit
2. withdraw
3. Interest Calculation
4. Account Details
5. Exist
3
Interest is 60.0
Your Balance is 1260.0
---- MENU ----
1. Deposit
2. withdraw
3. Interest Calculate
4. Account Details
5. Exit
4
Customer Name Arjun
Account Number 21
Account Type 1
Balance = 1260.0
---- MENU ----
1. Deposit
2. withdraw
3. Interest calculation
4. Account Details
5. Exit
5
Invalid choice.

10/10

PROGRAM 6
create a package which has two classes

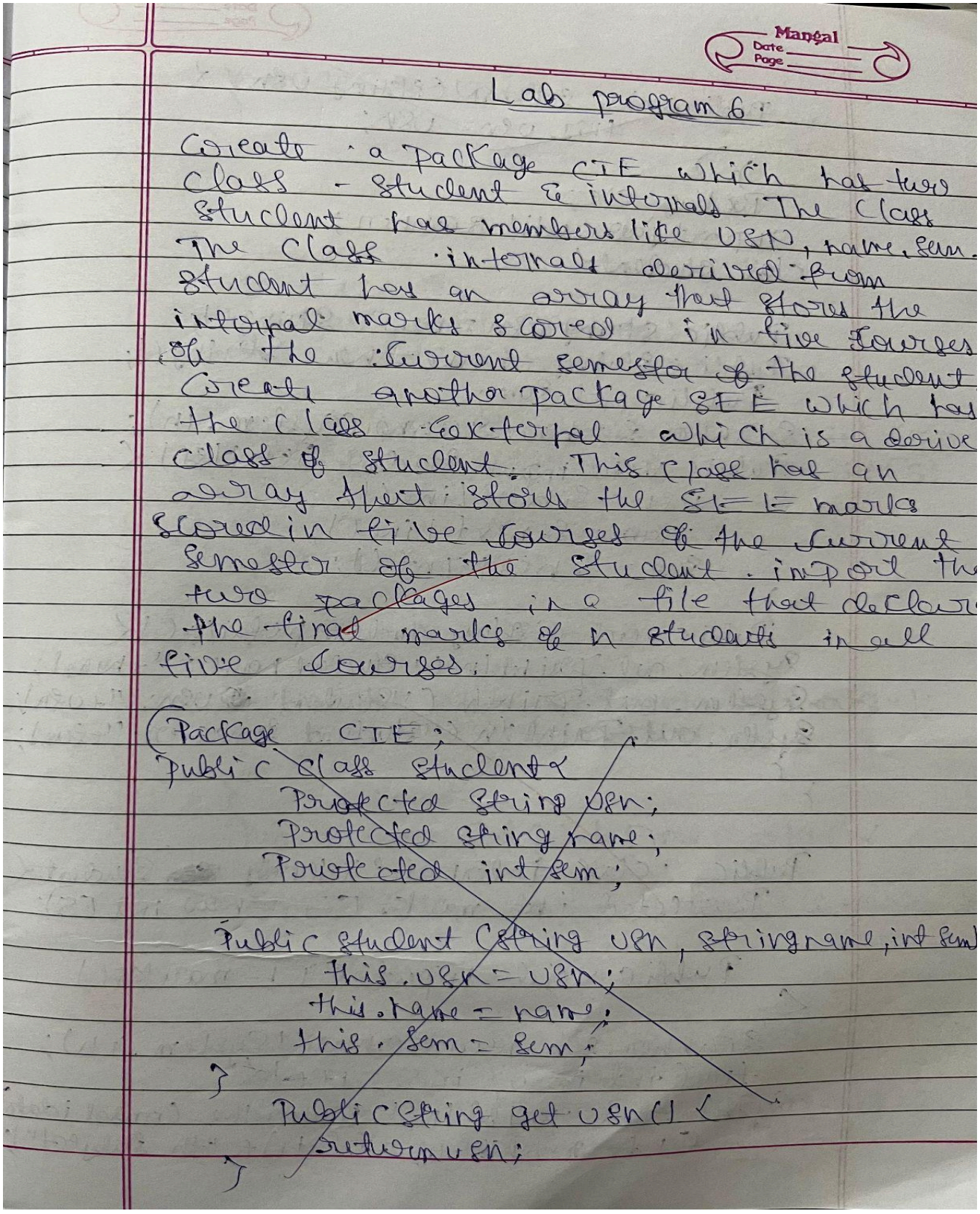
```
package cie;
import java.util.Scanner;

class Student {

public String usn = new String();
 public String name = new String();
 public int sem;
Scanner s=new Scanner(System.in);

public void inputStudentDetails() {
 usn=s.nextLine();
 name=s.nextLine();
 sem=s.nextInt();
}

public void displayStudentDetails() {
System.out.println("student name: " + name);
System.out.println("student usn: " + usn);
System.out.println("student semester: " + sem);
  }
}


public class internals extends Student {

protected int marks[] = new int[5];

public void inputCIEmarks()

{
Scanner s = new Scanner(System.in);
for(int i=0;i<5;i++){
System.out.println("enter the CONSOLIDATED CIE marks  of " +(i+1)+"th  subject");
marks[i]=s.nextInt();

   }}}
```

```
package see;

import cie.*;

import java.util.Scanner;

public class externals extends internals {
int i;
protected int seemarks[]=new int[5];

protected int finalMarks[]=new int[5];

Scanner s=new Scanner(System.in);
public void inputSEEmarks()
  {
for(i=0;i<5;i++)
{
System.out.println("Enter see marks");
seemarks[i]=s.nextInt();
  }
}
public void calculateFinalMarks() {
for(i=0;i<5;i++){
finalMarks[i]=(marks[i]+(seemarks[i]/2));


}

 }

public void displayFinalMarks() {

for (i=0;i<5;i++){
System.out.println("Final marks is"+finalMarks[i]);
}}}

import see.*;
import cie.*;
class main{
  public static void main(String args[]){
```

```
externals e=new externals();
System.out.println(" enter your name , usn and semester");
e.inputStudentDetails();
e.displayStudentDetails();
e.inputCIEmarks();
e.inputSEEmarks();
e.calculateFinalMarks();
e.displayFinalMarks();
}
}
```

```
 enter your name , usn and semester
nivas
1bm23cs165
3
student name: 1bm23cs165
student usn: nivas
student semester: 3
enter the CONSOLIDATED CIE marks  of 1th  subject
35
enter the CONSOLIDATED CIE marks  of 2th  subject
40
enter the CONSOLIDATED CIE marks  of 3th  subject
37
enter the CONSOLIDATED CIE marks  of 4th  subject
35
enter the CONSOLIDATED CIE marks  of 5th  subject
39
Enter see marks
85
Enter see marks
89
Enter see marks
90
Enter see marks
92
Enter see marks
95
Final marks is77
Final marks is84
Final marks is82
Final marks is81
Final marks is86
PS C:\1BM23CS165>
```

source code:



Lab program 6:

Create a package CIE which has two class - Student & 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 derive 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 declare the final marks of n students in all five courses.

```
Package CIE;
Public class Student{
    Protected String USN;
    Protected String name;
    Protected int Sem;

Public Student (String USN, String name, int Sem)
    this.USN = USN;
    this.name = name;
    this.Sem = Sem;
}
    Public String getUSN() {
        return USN;
}
```

```
Public void setUSN(String USN) {
        this.usn = usn;
    }
Package cie;
import java.util.Scanner;
class student{

Public String usn = new String();
Public String name = new String();
 Public int sem;
Scanner s = new Scanner(System.in);
Public void inputStudentDetails(){
    usn = s.nextLine();
    name = s.nextLine();
    sem = s.nextInt();
}
Public void displayStudentDetails(){
System.out.println("Student name: "+name);
System.out.println("Student usn: "+usn);
System.out.println("Student Semester: "+sem);
}
}

Public class internal extends Student{
 Protected int marks[] = new int[5];

   Public void inputCIEmarks()
{
   Scanner s = new Scanner(System.in);
   for(int i=0; i<5; i++){
System.out.println("Enter the Consolidated
    CIE marks of "+(i+1)+"th Subject"
```

```
        marks[i] = S.nextINT();
      }
    }
  }

Package see;
 import cie.*;
import java.util.Scanner;
 Public class externals internals{
    int ii;
protected int seemarks[] = new int[5];
protected int finalmarks[] = new int[5];

Scanner S = new Scanner(System.in);
 Public void inputSEEmarks()
  {
  for(i=0;i<5;i++)
  {
  System.out.println("Enter see marks");
  seemarks[i] = S.nextINT();
  }
}
  Public void CalculateFinalMarks(){
  for(i=0;i<5;i++){
  FinalMarks[i] = (marks[i] + (seemarks[i]/2));
  }
}

  Public void displayFinalMarks(){
  for(i=0;i<5;i++){
   System.out.println("Final marks is"
              + FinalMarks[i]);
}}
```

```
import see.*;
import cie.*;
class main {
    Public static void main(String args[])
 ex tornals e= new ex-tornals();
System.out.println("Enter your name,
                    usn. & Semester");
e.input Student Details();
e.display Student Details();
e.input CIE marks();
e.input SEE marks();
e.Calculate Final marks();
e.display Final marks();
 }
}
```

output

Enter your name, USN & Semester
Nivas. H
1BM23CS165
3

Student name: Nivas. H
Student USN: 1BM23CS165
Student Sem: 3

Enter the Consolidated CIE marks of 1st subject
44
45
46
47
48

Mangal
Date
Page

Enter Bee marks
85
86
87
88
89
Final marks is 86
Final mark is 88
Final marks is 89
Final marks is 91
Final marks is 92.

Sh

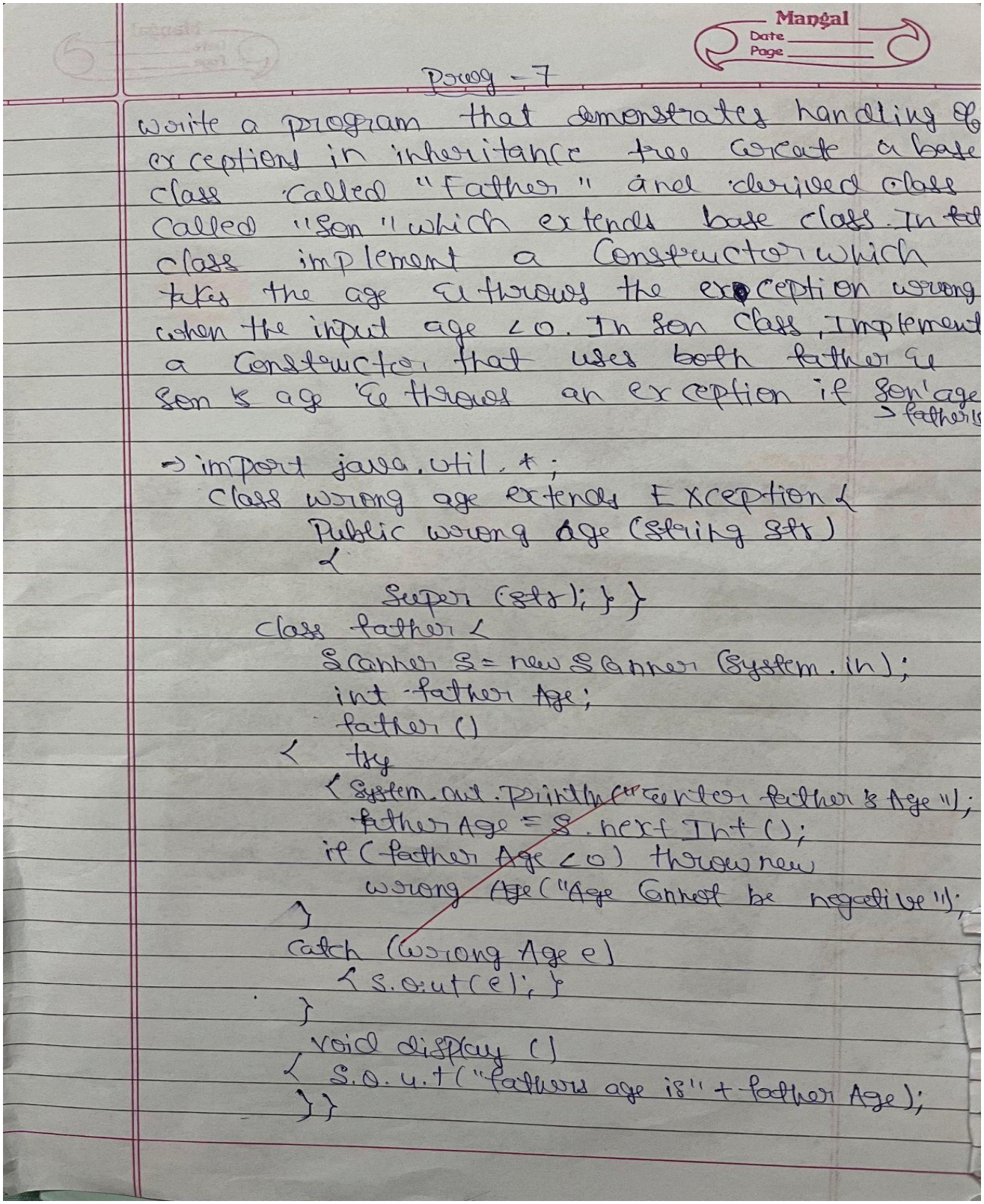
PROGRAM 7:
write a program that demonstrate handling of exceptions in inheritance

```
import java.util.*;
class wrongAge extends Exception
{
	public wrongAge(String str){
super(str);
}

}
class father
{
	Scanner s=new Scanner(System.in);
	int fatherAge;
	father()
	{
		try
		{
			System.out.println("enter the father's age");
			fatherAge=s.nextInt();
			if(fatherAge<0) throw new wrongAge("Age cannot be negative");
		}

	catch(wrongAge e){System.out.println(e);}
	}
	void display()
	{
		System.out.println("father's age is "+fatherAge);
	}
}
class son extends father
{
	Scanner s=new Scanner(System.in);
	int sonAge;
	son()
	{
		super();
		try
		{
```

```
                System.out.println("enter the son's age");
                sonAge=s.nextInt();
                if(sonAge<0) throw new wrongAge("Age cannot be negative");
                if(fatherAge<sonAge)
                throw new wrongAge("father's age cannot be less than son's age");
            }
            catch(wrongAge e){System.out.println(e);}
        }
        void display()
        {
            System.out.println("son's age is "+sonAge);
        }
}
class Program
{
        public static void main(String args[])
        {
            son so=new son();
            System.out.println("Name: LAKSHMINIVAS H");
            System.out.println("USN: 1BM23CS165");
        }
}
```

```
enter the father's age
20
enter the son's age
16
Name: LAKSHMINIVAS H
USN: 1BM23CS165
```

source code:



Prog - 7

Write a program that demonstrates handling of exceptions in inheritance tree create a base class called "Father" and derived class called "Son" which extends base class. In father class implement a Constructor which takes the age & throws the exception wrong when the input age < 0. In Son class, Implement a Constructor that uses both father & Son's age & throws an exception if Son's age > father's

```
import java.util.*;
class wrong age extends EXception {
    Public wrong Age (String str)
    {
        super (str); } }
class father {
    Scanner S = new Scanner (System.in);
    int father Age;
    father ()
    {  try
    { System.out.println("Enter father's Age");
      father Age = S.nextInt();
    if (father Age < 0) throw new
      wrong Age("Age Cannot be negative");
    }
    catch (wrong Age e)
      { S.out(e); }
    }
    void display ()
    { S.O.u.t("fathers age is" + father Age);
    }}
```

```
class Son extends father {
    Scanner S = new Scanner(System.in);
    int Son Age;
Son () {
    super ();
    try
    { S.o.u.t ("Enter Son's Age");
      Son Age = S.nextInt();
      if (Son Age < 0) throw new
        WrongAge ("Age Cannot be negative");
      if (father Age < Son Age) throw new
          WrongAge ("father's Age Cannot
                be less than Son's Age");
      }
      catch (WrongAge e)
        { S.o.ut (e); }
    }
    void display()
    {
      S.o.u.t ("Son's Age is" + Son Age);
    }}
class program
  {
    Public static void main (String args [])
      {
        Son S = new Son (); }}
```

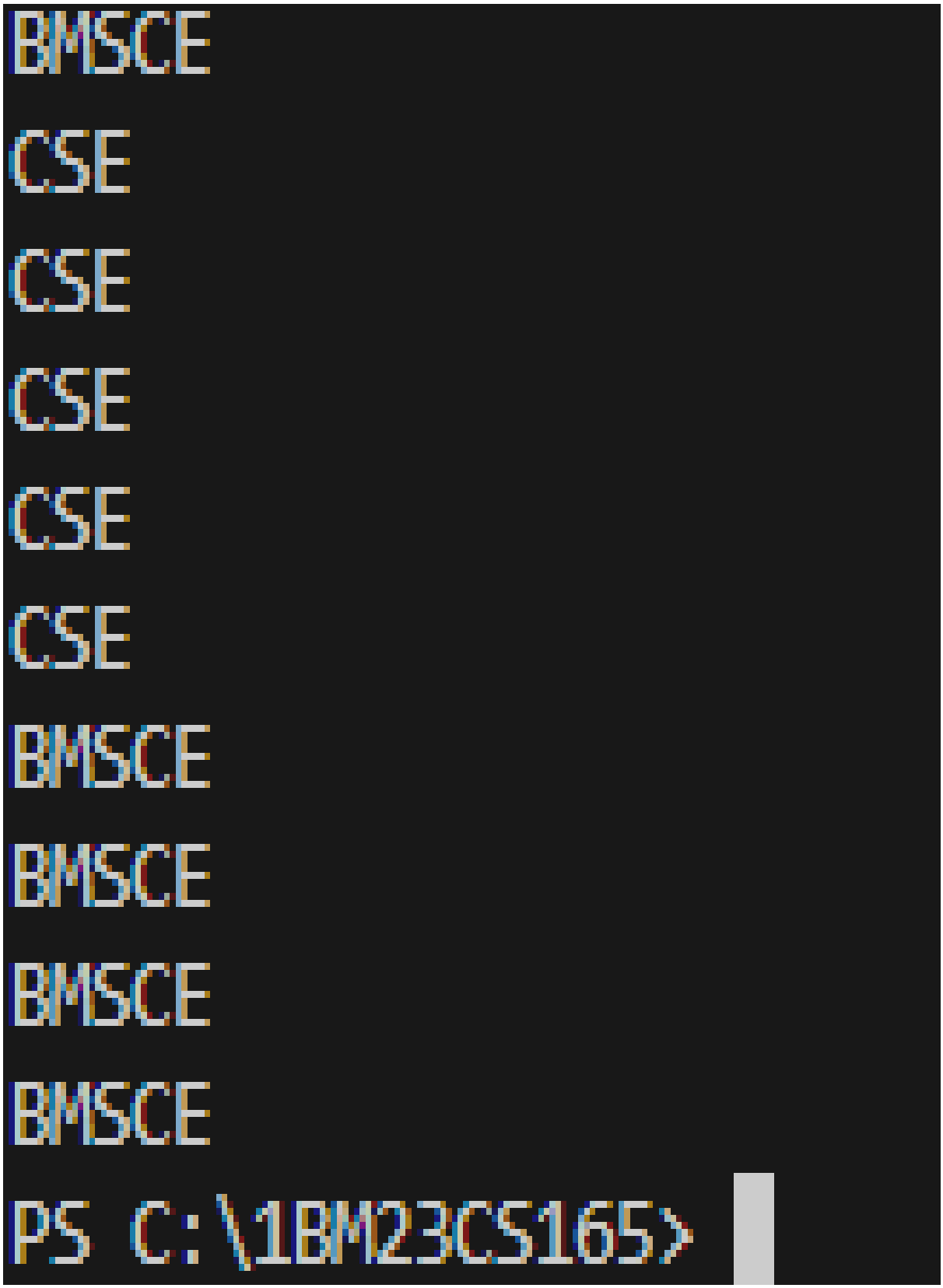
output

Enter the father's Age
40
Enter the Son's Age
50
Wrong Age: father's age Cannot be less than Son's Age

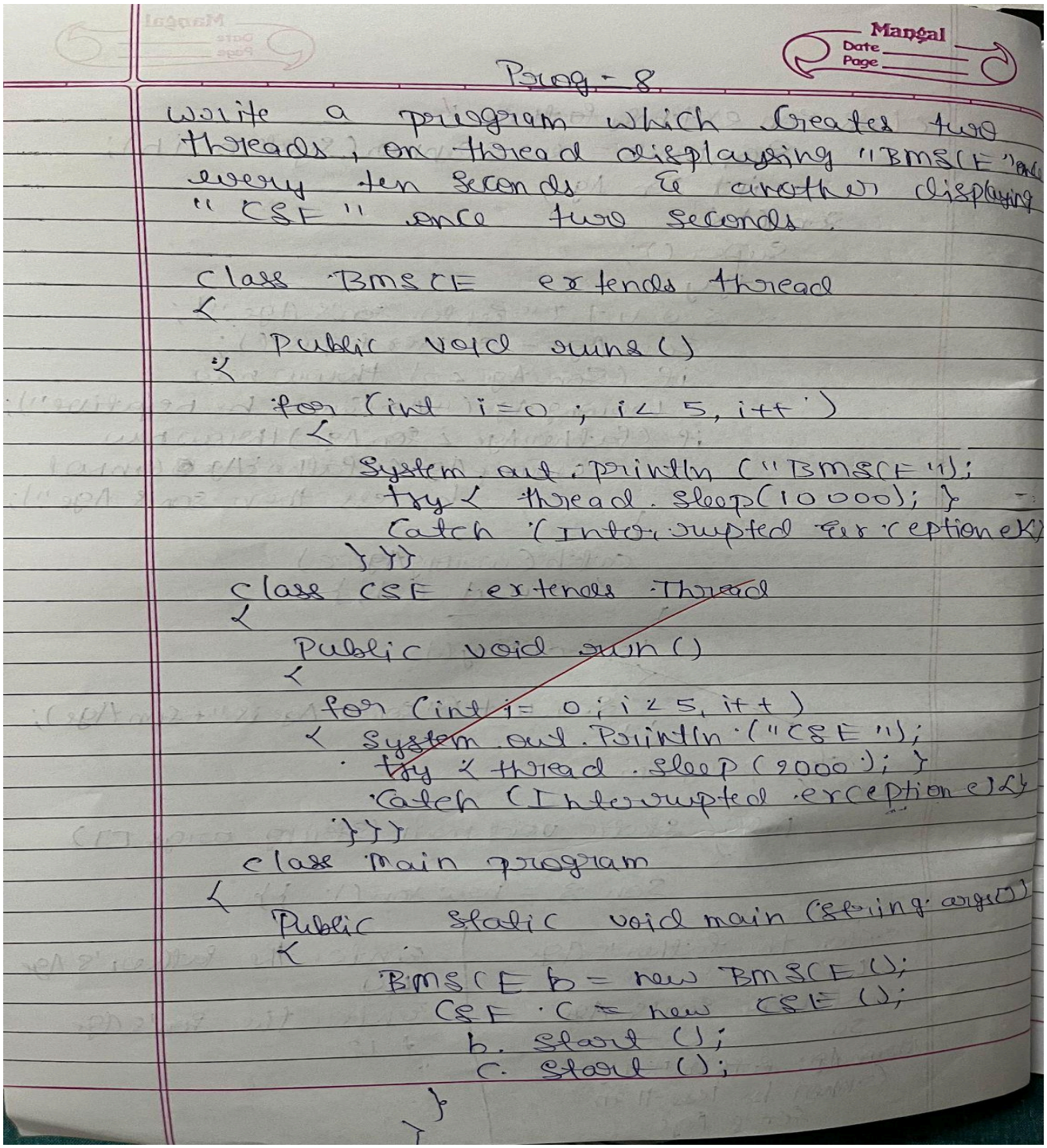
Enter the father's Age
40
Enter the Son's Age
+12
..

PROGRAM 8:
write a program which creates two threads

```
class BMSCE extends Thread
{
	public void run()
	{
		for(int i=0;i<5;i++)
		{
			System.out.println("BMSCE");
			try{Thread.sleep(10000);}
			catch(InterruptedException e){}
		}
	}
}
class CSE extends Thread
{
	public void run()
	{
		for(int i=0;i<5;i++)
		{
			System.out.println("CSE");
			try{Thread.sleep(2000);}
			catch(InterruptedException e){}
		}
	}
}
class mainprogram
{
public static void main(String args[])
	{
		BMSCE b=new BMSCE();
		CSE c=new CSE();
		b.start();
		c.start();
	}
}
```


BMSCE
CSE
CSE
CSE
CSE
CSE
CSE
BMSCE
BMSCE
BMSCE
BMSCE
PS C:\1BM23CS165>

source code:



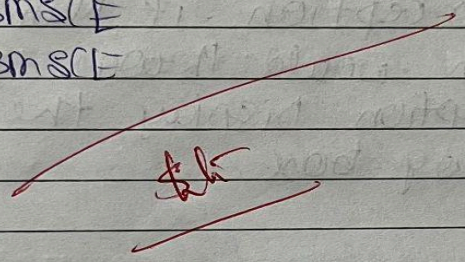
Prog - 8.

Write a program which creates two threads, on thread displaying "BMSCE" every ten seconds & another displaying "CSE" once two seconds.

```
class BMSCE extends thread
{
  Public void runs()
  {
    for (int i=0 ; i< 5, i++ )
    {
      System.out.println ("BMSCE");
        try { thread.sleep(10000); }
        Catch (Interrupted Exception eK)
    }}
class CSE extends Thread
{
   Public void run()
   {
     for (int i= 0; i < 5, it+ )
     { System.out.println ("CSE");
       try { thread.sleep (2000); }
       Catch (Interrupted exception e){}
     }}}
class main program
{
  Public static void main (String args[])
  {
      BMSCE b = new BMSCE ();
       CSE c = new CSE ();
        b.start ();
        c.start ();
  }
}
```

output

BMSCE
CSE
CSE
CSE
CSE
CSE
BMSCE
BMSCE
BMSCE
BMSCE



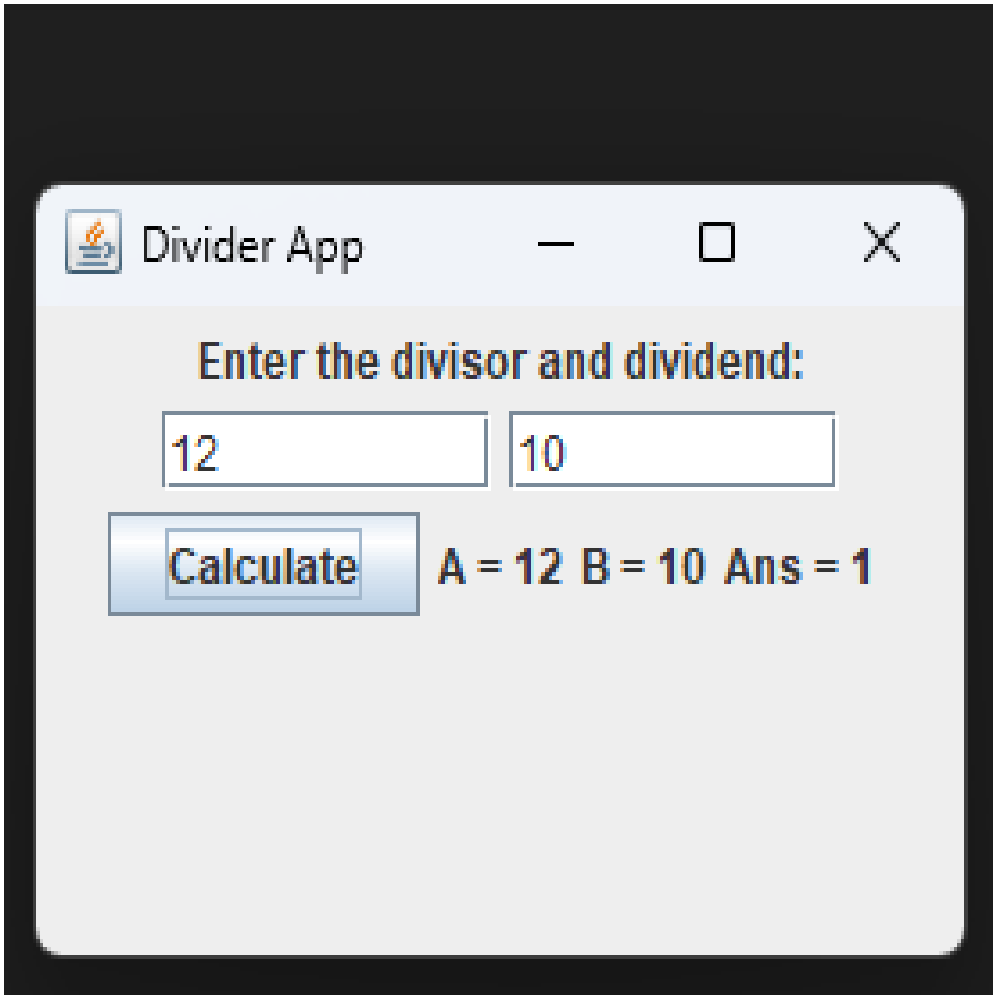
PROGRAM 9:

write a program that creates a user interface to perform integer divisions

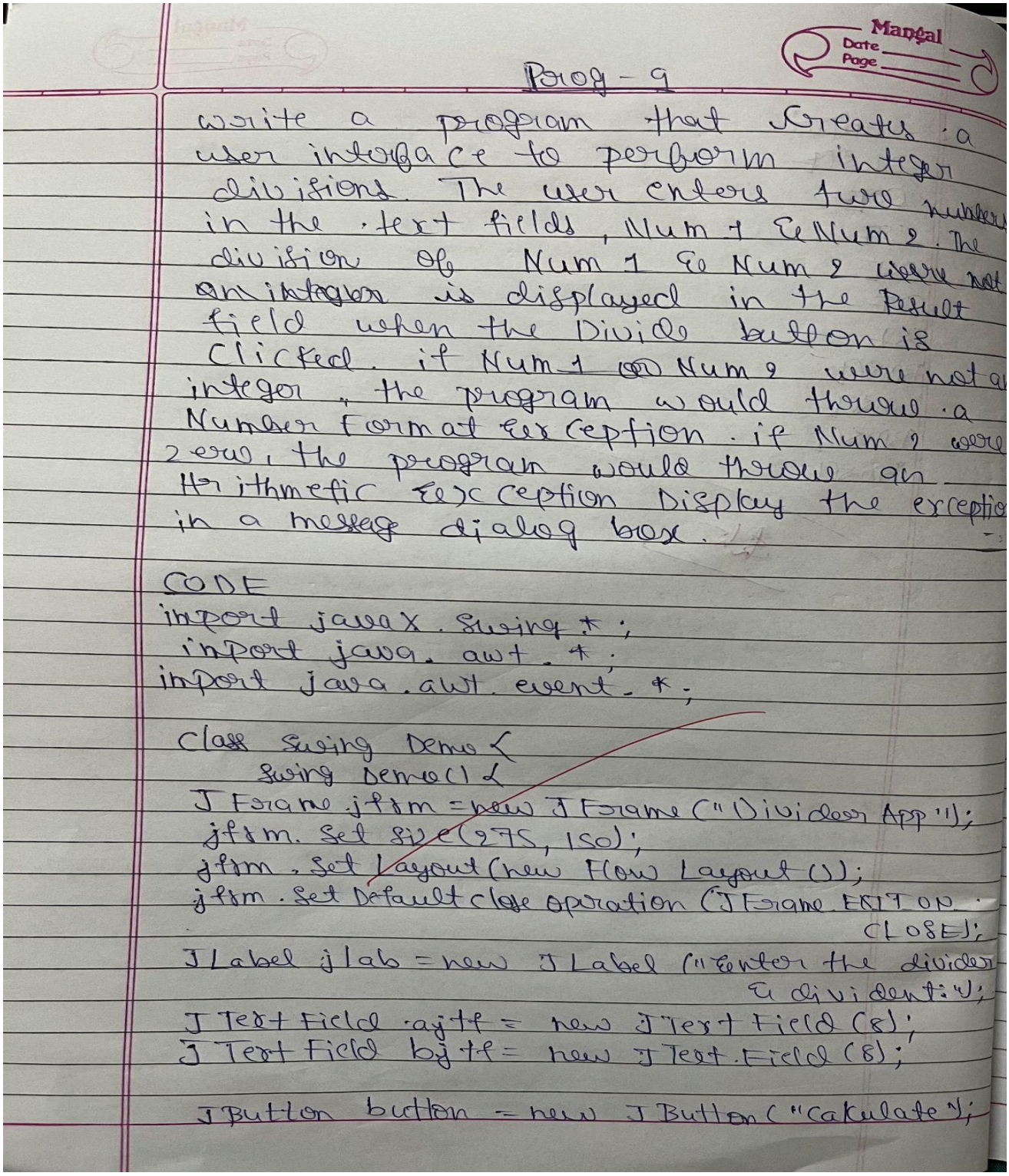
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class SwingDemo {
   SwingDemo() {
      // Create JFrame container
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 200);
      jfrm.setLayout(new FlowLayout());

      // To terminate the program when the window closes
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // Label prompting user
      JLabel jlab = new JLabel("Enter the divisor and dividend:");

      // Create text fields for input
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);

      // Create button to trigger calculation
      JButton button = new JButton("Calculate");

      // Labels to display output and errors
      JLabel err = new JLabel();
      JLabel alab = new JLabel();
      JLabel blab = new JLabel();
      JLabel anslab = new JLabel();

      // Add components to the frame
      jfrm.add(jlab);
      jfrm.add(ajtf);
      jfrm.add(bjtf);
      jfrm.add(button);
```

```
    jfrm.add(alab);
    jfrm.add(blab);
    jfrm.add(anslab);
    jfrm.add(err);

    // Action listener for button click
    button.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
            try {
                // Parse integers from text fields
                int a = Integer.parseInt(ajtf.getText());
                int b = Integer.parseInt(bjtf.getText());

                // Perform division
                int ans = a / b;

                // Display results
                alab.setText("A = " + a);
                blab.setText("B = " + b);
                anslab.setText("Ans = " + ans);
                err.setText("");  // Clear error message
            } catch (NumberFormatException e) {
                // Handle non-integer input
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("Enter Only Integers!");
            } catch (ArithmeticException e) {
                // Handle division by zero
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("B should be NON-zero!");
            }
        }
    });

    // Display the frame
    jfrm.setVisible(true);
}
```

```
    public static void main(String[] args) {
        // Create the frame on the Event Dispatch Thread
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new SwingDemo();
            }
        });
    }
}
```



source code:



Prog - 9

write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num 1 & Num 2. The division of Num 1 & Num 2 is displayed in the Result field when the Divide button is clicked. if Num 1 & Num 2 were not an integer, the program would throw a NumberFormatException. if Num 2 were zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

CODE

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

class SwingDemo {
    SwingDemo() {
JFrame jfrm = new JFrame("DivideApp");
jfrm.setSize(275, 150);
jfrm.setLayout(new FlowLayout());
jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_
                                CLOSE);
JLabel jlab = new JLabel("Enter the divider
                          & dividend:");
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);

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

```
JLabel eror = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();

jfrm.add(eror);
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

ActionListener l = new ActionListener() {
  Public void actionPerformed(ActionEvent e
  System.out.Println("Action event from a text field
};
ajtf.addActionListener();
bjtf.addActionListener();

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("InA = "+a);
    blab.SetText("InB = "+b);
    anslab.SetText("InAns = "+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[]) {
    Frame on event dispatching thread
    SwingUtilities.invokeLater(new Runnable()
    Public void run() {
    new SwingDemo();
    }});
    }
    }
```

Output

Divide

Num1: [20] Num2: [4] [Divide]

Result: [5]

PROGRAM 10(a&b):

demonstrate inter process communication and deadlock

```
class A {

synchronized void foo(B b) {

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

System.out.println(name + " entered A.foo");

try {

Thread.sleep(1000);

} catch(Exception e) {

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

}

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

b.last();

}

void last() {

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

}

}

class B {

synchronized void bar(A a) {
```

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

System.out.println(name + " entered B.bar");

try {

Thread.sleep(1000);

} catch(Exception e) {

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

}

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

a.last();

}

void last() {

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

}

}

class Deadlock implements Runnable
{

A a = new A();

B b = new B();

Deadlock() {

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

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

t.start();

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

System.out.println("Back in mainthread");

}

public void run() {

b.bar(a);
System.out.println("Back in otherthread");

}

public static void main(String args[]) {
System.out.println("Name: LAKSHMINIVAS H, USN: 1BM23CS165");
new Deadlock();

}

}


class Q {

int n;

boolean valueSet = false;

synchronized int get() {

while(!valueSet)

try {

System.out.println("\nConsumer waiting\n");
```

```
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
System.out.println("Got: " + n);
valueSet = false;
System.out.println("\nIntimate Producer\n");
notify();
return n;
}
synchronized void put(int n) {
while(valueSet)
try {
System.out.println("\nProducer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
this.n = n;
valueSet = true;
```

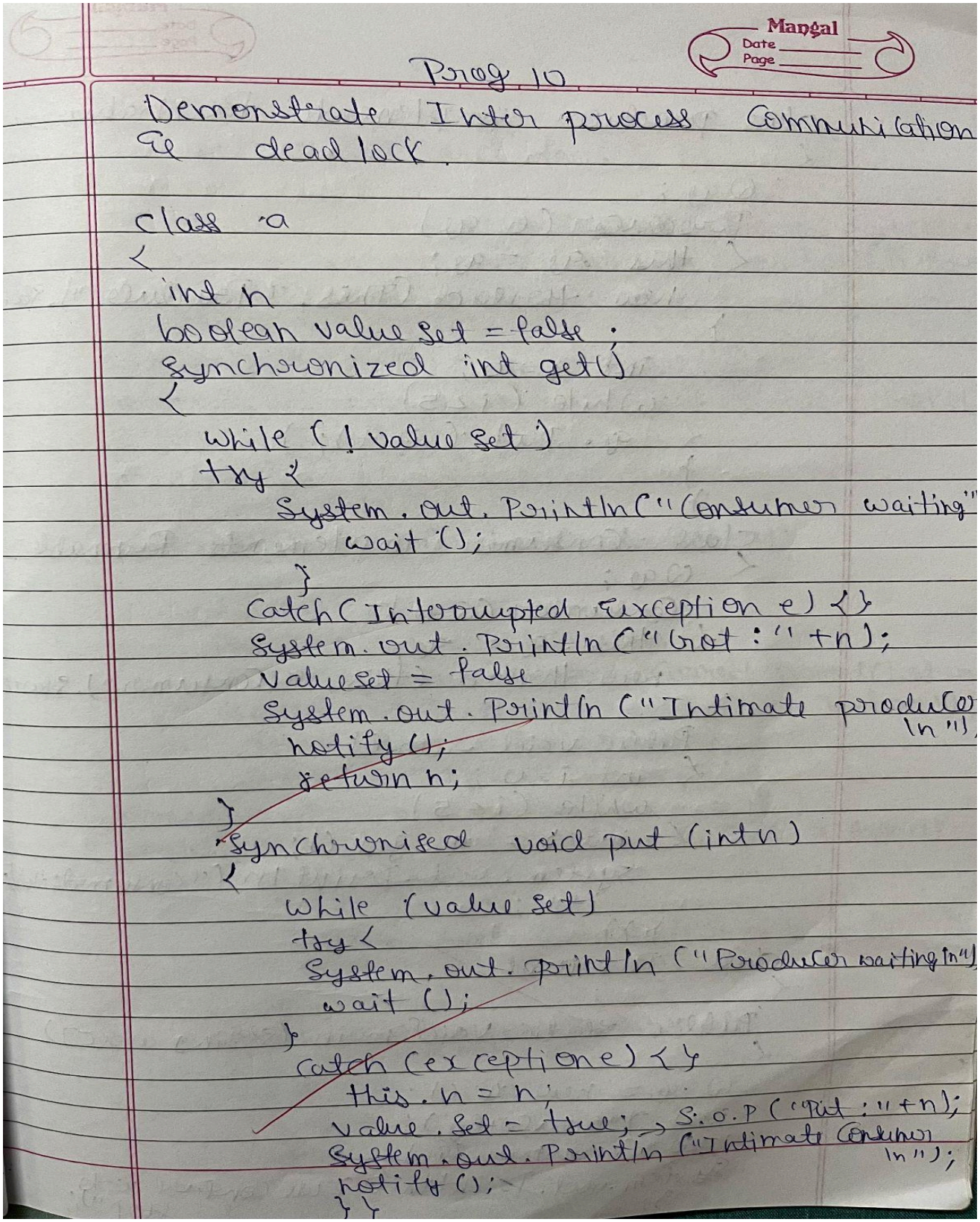
```
System.out.println("Put: " + n);
System.out.println("\nIntimate Consumer\n");
notify();
}
}
class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
public void run() {
int i = 0;
while(i<15) {
q.put(i++);
}
}
}
class Consumer implements Runnable {
Q q;
```

```
Consumer(Q q) {

this.q = q;

new Thread(this, "Consumer").start();

}

public void run() {

int i=0;

while(i<15) {

int r=q.get();

System.out.println("consumed:"+r);

i++;

}

}

}

class PCFixed {

public static void main(String args[]) {
System.out.println("Name: Kavana M A, USN: 1BM23CS145");

Q q = new Q();

new Producer(q);

new Consumer(q);

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

```
PS D:\1BM23CS170\OOJ> javac Deadlock.java
PS D:\1BM23CS170\OOJ> java Deadlock
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
Inside A.last
Back in other thread
MainThread trying to call B.last()
Inside A.last
Back in main thread
```

```
PS D:\1BM23CS170\OOJ> javac PCFixed.java
PS D:\1BM23CS170\OOJ> java PCFixed
Press Control-C to stop.
Put: 0
Intimate Consumer

Producer waiting

Got: 0
Intimate Producer

Put: 1
Intimate Consumer

Producer waiting

consumed:0
Got: 1
Intimate Producer

consumed:1
Put: 2
Intimate Consumer

Producer waiting

Got: 2
Intimate Producer

consumed:2
Put: 3
Intimate Consumer

Producer waiting

Got: 3
Intimate Producer

consumed:3
Put: 4
Intimate Consumer

Got: 4
Intimate Producer

consumed:4
```

source code:



Prog 10

Demonstrate Inter process Communication & deadlock.

```
class a
{
 int n
 boolean valueSet = false ;
 Synchronized int get()
 {
   while ( ! valueSet )
   try {
       System.out.println("Consumer waiting"
           wait();
       }
     Catch( Interrupted Exception e) {}
     System.out.println("Got : " +n);
     valueSet = false
     System.out.println("Intimate producer
                                    \n");
      notify();
      return n;
 }
 Synchronised void put (int n)
 {
     While (valueSet)
     try {
      System.out.println("Producer waiting\n");
       wait();
 }
 catch (exception e) { }
      this.n = n;
      valueSet = true;  -> S.O.P("Put : " +n);
      System.out.println("Intimate Consumer
                                    \n");
      notify();
 } }
```

```
class Producer implements Runnable
{
    Q q;
    Producer(Q q)
    { this.q = q;
      new Thread(this, "Producer").start();
    }

        int i = 0;
        while (i < 5)
      { 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 < 5)
          { int r = q.get();
            System.out.println("Consumed: " +
              i++;
          }}
    class PCFixed
    {
      Public static void main(String args[])
      {
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to
                                    stop");
      }}
```

```
class A
{
  synchronised void foo(B b)
  {
    String name = Thread.currentThread().getName();
    System.out.println(name + "entered A foo");
    try { Thread.sleep(1000); }
    catch(Exception e){
System.out.println("A Interrupted");}
System.out.println(name + "trying to call B.last()");
      b.last();
    }
    void last(){
System.out.println("Inside A last");
    }}

  class B{
    synchronised void bar(A a)
    { String name = Thread.currentThread().getName();
    System.out.println(name + "entered B.bar");
     try { Thread.sleep(1000); }
     catch(Exception e){
System.out.println("B interrupted");}
System.out.println(name + "trying to call A last()");
      a.last();}
      void last(){
      System.out.println("Inside A.last");
      }
    }

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

```
{
    Thread.currentThread().setName("main Thread");
    Thread t = new Thread(this, "Racing Thread");
    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:

Press Control + C to stop

Put = 0

Intimate Consumer

producer waiting

Got: 0

Intimate producer

Producer waiting

Consumed: 0

Got: 1

Intimate producer

Consumed: 1

Put 2

Intimate Consumer

Producer waiting

Got : 2
Intimate producer
Consumed : 2
Put : 3
Intimate Consumer
producer waiting
Got : 3
Intimate producer
Consumed : 3
Put : 4
Intimate Consumer
Got : 4
Intimate producer
Consumed : 4

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

Sat
2/12/24

10/10