**VISVESVARAYA TECHNOLOGICAL UNIVERSITY, JNANASANGAMA, BELGAUM - 590014, KARNATAKA**



**LABORATORY REPORT ON**

**Object Oriented Java Programming (23CS3PCOOJ) *Submitted by***

**Dhanush S (1BM22CS086)**

***In partial fulfilment 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**

**December-2023 to April-2024**

**Laboratory Program – 1**

**01) 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.**

**Program=>**

import java.util.Scanner;

public class Pro1\_QuadraticEqun

{

public static void main(String[] args)

{

System.out.println("Name: Dhanush S \n USN: 1BM22CS086");

Scanner sc=new Scanner(System.in);

System.out.println("Enter coefficient a");

double a=sc.nextDouble();

System.out.println("Enter coefficieant B");

double b=sc.nextDouble();

System.out.println("Enter Coefficient C ");

double c =sc.nextDouble();

double discriment=b\*b-4\*a\*c;

if (discriment >=0)

{

double root1=(-b+Math.sqrt(discriment))/(2\*a);

double root2=(-b-Math.sqrt(discriment))/(2\*a);

System.out.println("Real Soluation");

System.out.println("Root 1 "+root1);

System.out.println("Root 2 "+root2);

}else

{

System.out.println("No real Soluation.discriment is negative.");

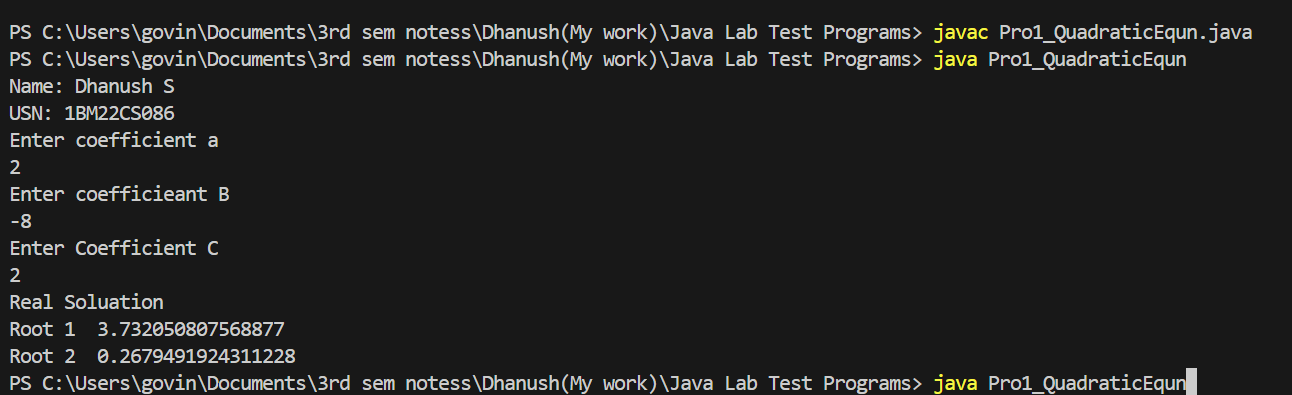
}

sc.close();

}

}

**Output=.>**



**Laboratory Program – 2**

**02) 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.**

**Program=>**

import java.util.Scanner;

class Student {

String USN, Name;

double mark, sum;

Scanner sc = new Scanner(System.in);

double marks[] = new double[6];

Student(String USN, String Name)

{

this.USN = USN;

this.Name = Name;

}

void getMarks()

{

for (int i = 0; i < 6; i++) {

System.out.println("Enter Sub:" + (i + 1) + " Mark (Out of 100) : ");

mark = sc.nextDouble();

marks[i] = mark;

}

}

double totalMarks\_percent()

{

for (int i = 0; i < 6; i++) {

sum = sum + marks[i];

}

return (sum/6);

}

}

class Student\_info

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

System.out.println("Name: Dhanush S\n USN:= 1BM22CS086");

System.out.println("Enter Student USN: ");

String USN = sc.next();

System.out.println("Enter Student Name: ");

String name = sc.next();

Student s1 = new Student(USN, name);

s1.getMarks();

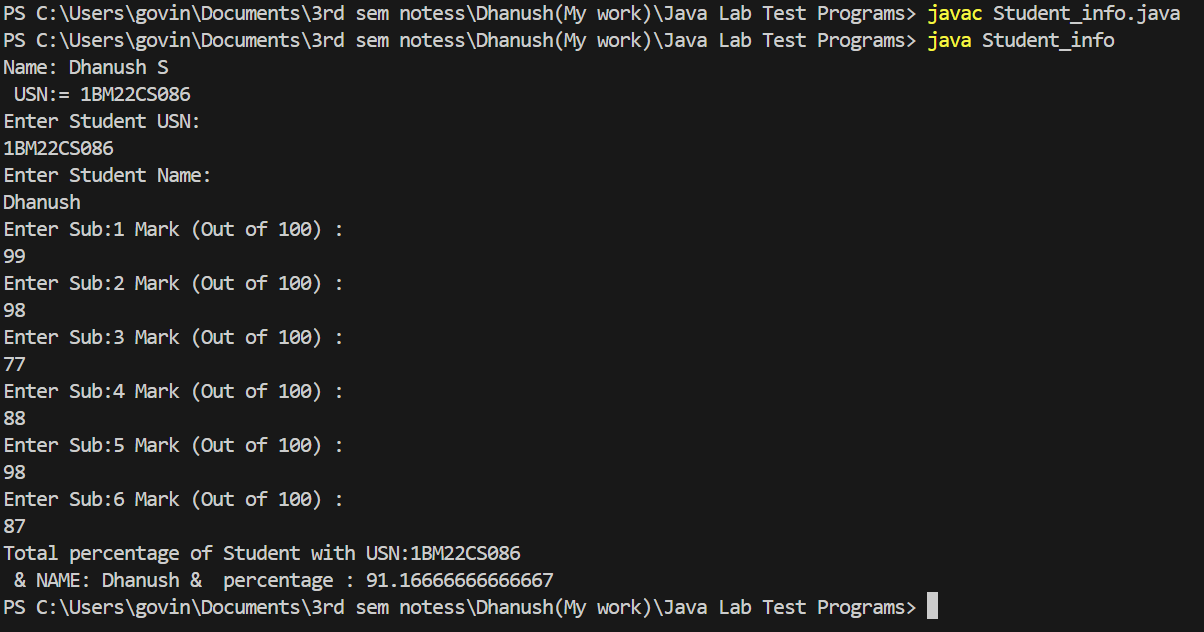
double percentage = s1.totalMarks\_percent();

System.out.println("Total percentage of Student with USN:" + s1.USN + "\n & NAME: " + s1.Name +" is : " + percentage);

}

}

**Output=>**



**Laboratory Program – 3**

**03) 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.**

**Program=>**

import java.util.Scanner;

class Books

{

String name;

String author;

int price;

int numPages;

Books() {};

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 sc = new Scanner(System.in);

System.out.println("Name: Dhanush S\n USN:= 1BM22CS086");

int n;

String name, author;

int price, numPages;

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

n = sc.nextInt();

Books b[];

b = new Books[n];

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

{

System.out.println("Books " + (i + 1) + ": ");

System.out.print("Enter name of the book: ");

name = sc.next();

System.out.print("\nEnter Author: ");

author = sc.next();

System.out.print("\nEnter price: ");

price = sc.nextInt();

System.out.print("\nEnter number of pages: ");

numPages = sc.nextInt();

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

}

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

{

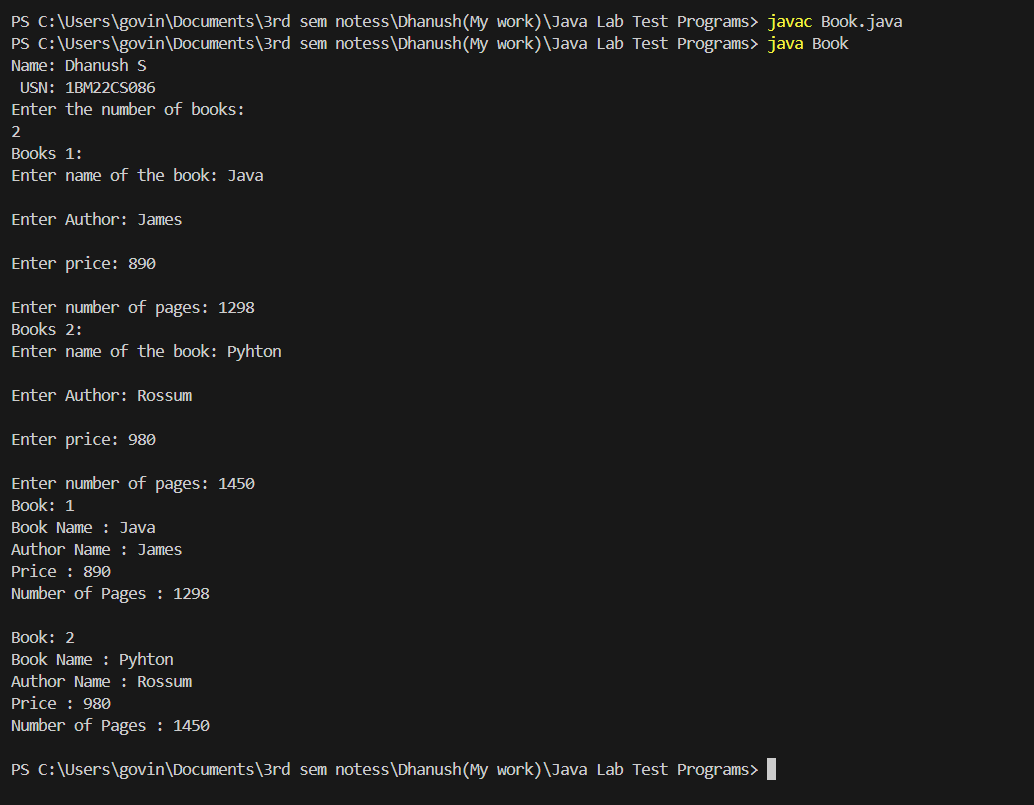
System.out.println("Book: " + (i + 1) + "\n" + b[i]);

}

}

}

**Output=>**

****

**Laboratory Program – 4**

**04) 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.**

**Program=>**

abstract class Shape

{

public int side1, side2;

abstract void printArea();

}

class Rectangle extends Shape

{

Rectangle(int length, int breadth)

{

this.side1 = length;

this.side2 = breadth;

}

void printArea()

{

System.out.println("The Area of Rectangle : " + (side1 \* side2));

}

}

class Triangle extends Shape

{

Triangle(int base, int height)

{

this.side1 = base;

this.side2 = height;

}

void printArea()

{

System.out.println("The Area of Triangle : " + (0.8\* side1 \* side2));

}

}

class Circle extends Shape

{

Circle(int rad)

{

this.side1 = this.side2 = rad;

}

void printArea()

{

System.out.println("The Area of Circle : " + (3.14 \* side1 \* side2));

}

}

class shapeClass

public static void main(String[] args)

{

System.out.println("Name: Dhanush S\n USN:= 1BM22CS086");

Rectangle r = new Rectangle(10, 10);

Triangle t = new Triangle(6, 10);

Circle c = new Circle(6);

r.printArea();

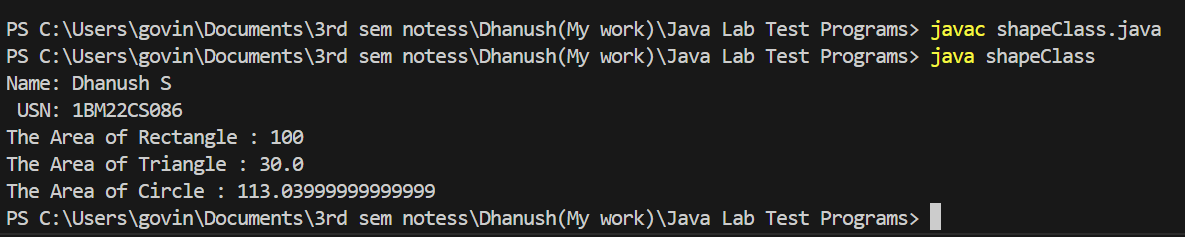
t.printArea();

c.printArea();

}

}

**Output=>**

****

**Laboratory Program – 5**

**05) 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.**

**Program=>**

import java.util.Scanner;

abstract class Account {

String customerName;

int accountNumber;

String accountType;

double balance;

Account(String customerName, int accountNumber, String accountType, double balance) {

this.customerName = customerName;

this.accountNumber = accountNumber;

this.accountType = accountType;

this.balance = balance; }

abstract void deposit(double amount);

abstract void displayBalance();

abstract void computeInterest();

abstract void withdraw(double amount);

}

class SavingsAccount extends Account

{

SavingsAccount(String customerName, int accountNumber, String accountType, double balance) {

super(customerName, accountNumber, accountType, balance);

}

void deposit(double amount)

{

balance += amount;

System.out.println("Amount deposited: " + amount);

}

void displayBalance()

{

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

}

void computeInterest()

{

double interestRate = 0.05;

double interest = balance \* interestRate;

balance += interest;

System.out.println("Interest added: " + interest);

}

void withdraw(double amount)

{

if (balance < amount)

{

System.out.println("Insufficient balance");

} else {

balance -= amount;

System.out.println("Amount withdrawn: " + amount);

}

}

}

class CurrentAccount extends Account

{

double minimumBalance = 1000;

double serviceCharge = 50;

CurrentAccount(String customerName, int accountNumber, String accountType, double balance) {

super(customerName, accountNumber, accountType, balance);

}

void deposit(double amount)

{

balance += amount;

System.out.println("Amount deposited: " + amount);

}

void displayBalance()

{

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

}

void computeInterest()

{

System.out.println("Current account does not earn interest");

}

void withdraw(double amount)

{

if (balance - amount < minimumBalance) {

System.out.println("Insufficient balance");

balance -= serviceCharge;

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

}

else {

balance -= amount;

System.out.println("Amount withdrawn: " + amount);

}

}

}

class Bank

{

public static void main(String[] args)

{

System.out.println("Name: Dhanush S\n USN:= 1BM22CS086");

Scanner sc = new Scanner(System.in);

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

String customerName = sc.nextLine();

System.out.print("Enter account number: ");

int accountNumber = sc.nextInt();

System.out.print("Enter account type (savings/current): ");

String accountType = sc.next();

System.out.print("Enter initial balance: ");

double balance = sc.nextDouble();

System.out.println("\n");

Account account;

if (accountType.equals("savings"))

{

account = new SavingsAccount(customerName, accountNumber, accountType, balance);

} else {

account = new CurrentAccount(customerName, accountNumber, accountType, balance);

}

while (true)

{

System.out.println("\n1. Deposit");

System.out.println("2. Display balance");

System.out.println("3. Compute interest");

System.out.println("4. Withdraw");

System.out.println("5. Exit\n");

System.out.print("Enter choice: ");

int choice = sc.nextInt();

switch (choice)

{

case 1:

System.out.print("\nEnter amount to deposit: ");

double amount = sc.nextDouble();

account.deposit(amount);

break;

case 2:

account.displayBalance();

break;

case 3:

account.computeInterest();

break;

case 4:

System.out.print("\nEnter amount to withdraw: ");

amount = sc.nextDouble();

account.withdraw(amount);

break;

case 5:

sc.close();

System.exit(0);

break;

default:

System.out.println("\nInvalid choice");

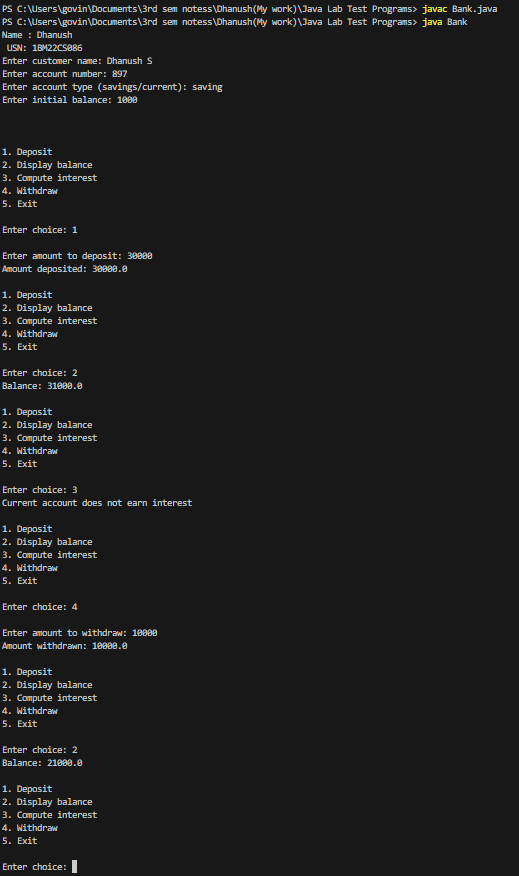
}

}

}

}

**Output=>**

****

**Laboratory Program – 6**

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

1. **Create a folder CIE and save the programs Student.java and Internals.java within it.**
2. **Create a folder SEE and save the program External.java within it.**
3. **Save the Main program outside these two folders.**
4. **Compile Main.java and Execute the Main.class**

**Program:--**

**Student .java(Inside CIE Folder)**

package CIE;

public class Student {

private String usn, name;

private int sem;

// Constructor

public Student(String usn, String name, int sem) {

this.usn = usn;

this.name = name;

this.sem = sem;

}

// Getter and Setter methods

public String getUsn() {

return usn;

}

public void setUsn(String usn) {

this.usn = usn;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getSem() {

return sem;

}

public void setSem(int sem) {

this.sem = sem;

}

}

**Internals .java(Inside CIE Folder)**

**Program:--**

package CIE;

public class Internals extends Student {

private int[] internalMarks;

// Constructor

public Internals(String usn, String name, int sem, int[] internalMarks) {

super(usn, name, sem);

this.internalMarks = internalMarks;

}

// Getter and Setter method for internalMarks

public int[] getInternalMarks() {

return internalMarks;

}

public void setInternalMarks(int[] internalMarks) {

this.internalMarks = internalMarks;

}

}

**External.java(Inside SEE Folder)**

**Program:--**

package SEE;

import CIE.Student;

public class External extends Student {

private int[] seeMarks;

// Constructor

public External(String usn, String name, int sem, int[] seeMarks) {

super(usn, name, sem);

this.seeMarks = seeMarks;

}

// Getter and Setter method for seeMarks

public int[] getSeeMarks() {

return seeMarks;

}

public void setSeeMarks(int[] seeMarks) {

this.seeMarks = seeMarks;

}

}

**Main.java(Outside these two folders)**

**Program:--**

import CIE.Internals;

import SEE.External;

import java.util.Scanner;

public class Main {

public static void main(String[] args)

{

System.out.println("Name: Dhanush S\n USN:= 1BM22CS086"); Scanner scanner = new Scanner(System.in);

int n = 2; // set to 2 for two students

// Arrays to store CIE and SEE students

Internals[] cieStudents = new Internals[n];

External[] seeStudents = new External[n];

// Input details for CIE students

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

{

System.out.println("Enter details for CIE student " + (i + 1));

System.out.print("USN: ");

String usn = scanner.next();

System.out.print("Name: ");

String name = scanner.next();

System.out.print("Semester: ");

int sem = scanner.nextInt();

// Input internal marks for 5 courses

int[] internalMarks = new int[5];

System.out.println("Enter internal marks for 5 courses:");

for (int j = 0; j < 5; j++)

{

System.out.print("Course " + (j + 1) + ": ");

internalMarks[j] = scanner.nextInt();

}

// Create CIE student object

cieStudents[i] = new Internals(usn, name, sem, internalMarks);

}

// Input details for SEE students

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

System.out.println("\nEnter details for SEE student " + (i + 1));

System.out.print("USN: ");

String usn = scanner.next();

System.out.print("Name: ");

String name = scanner.next();

System.out.print("Semester: ");

int sem = scanner.nextInt();

// Input SEE marks for 5 courses

int[] seeMarks = new int[5];

System.out.println("Enter SEE marks for 5 courses:");

for (int j = 0; j < 5; j++)

{

System.out.print("Course " + (j + 1) + ": ");

seeMarks[j] = scanner.nextInt();

}

// Create SEE student object

seeStudents[i] = new External(usn, name, sem, seeMarks);

}

// Display student details, internal marks, and final marks

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

System.out.println("\nStudent: " + cieStudents[i].getName());

System.out.println("USN: " + cieStudents[i].getUsn());

System.out.println("Semester: " + cieStudents[i].getSem());

// Display internal marks

System.out.print("Internal Marks: ");

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

System.out.print(cieStudents[i].getInternalMarks()[j] + " ");

}

// Assuming you have some logic to calculate final marks

int[] finalMarks = cieStudents[i].getInternalMarks();

System.out.println("\nFinal Marks: ");

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

System.out.println("Course " + (j + 1) + ": " + finalMarks[j]);

}

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

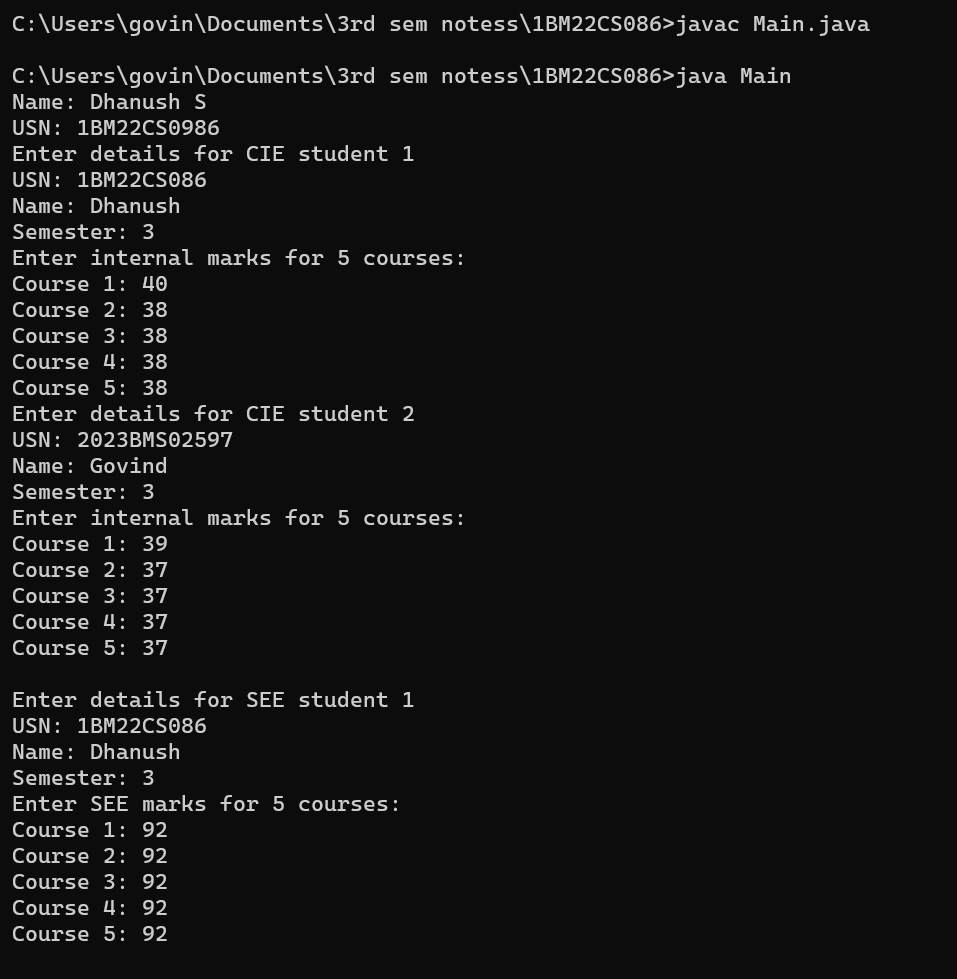
}

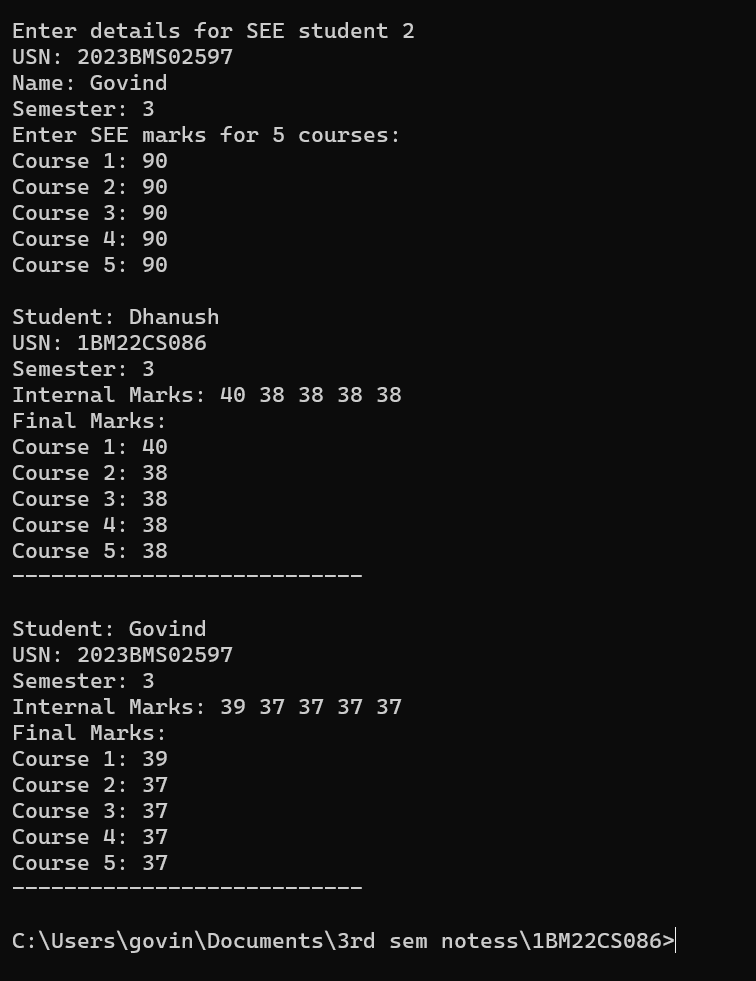
// Close the scanner

scanner.close();

}

}

****

****

**Laboratory Program – 7**

**07) 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<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is >=father’s age.**

**Program=>**

import java.util.Scanner;

class WrongAge extends Exception {

public WrongAge() {

super("Invalid age!");

}

}

class Father {

private int age;

public Father(int age) throws WrongAge {

if (age < 0) {

throw new WrongAge();

}

this.age = age;

}

public int getAge() {

return age;

}

}

class Son extends Father {

private int sonAge;

public Son(int fatherAge, int sonAge) throws WrongAge {

super(fatherAge);

if (sonAge >= fatherAge) {

throw new WrongAge();

}

this.sonAge = sonAge;

}

public int getSonAge() {

return sonAge;

}

}

public class WrongAgeExceptionProgram{

public static void main(String[] args)

{

System.out.println("Name: Dhanush S\n USN:= 1BM22CS086"); Scanner scanner = new Scanner(System.in);

try {

System.out.print("Enter father's age: ");

int fatherAge = scanner.nextInt();

System.out.print("Enter son's age: ");

int sonAge = scanner.nextInt();

Father father = new Father(fatherAge);

System.out.println("Father's age: " + father.getAge());

Son son = new Son(fatherAge, sonAge);

System.out.println("Son's age: " + son.getSonAge());

} catch (WrongAge e) {

System.out.println(e.getMessage());

} catch (Exception e) {

System.out.println("Invalid input-------.");

} finally {

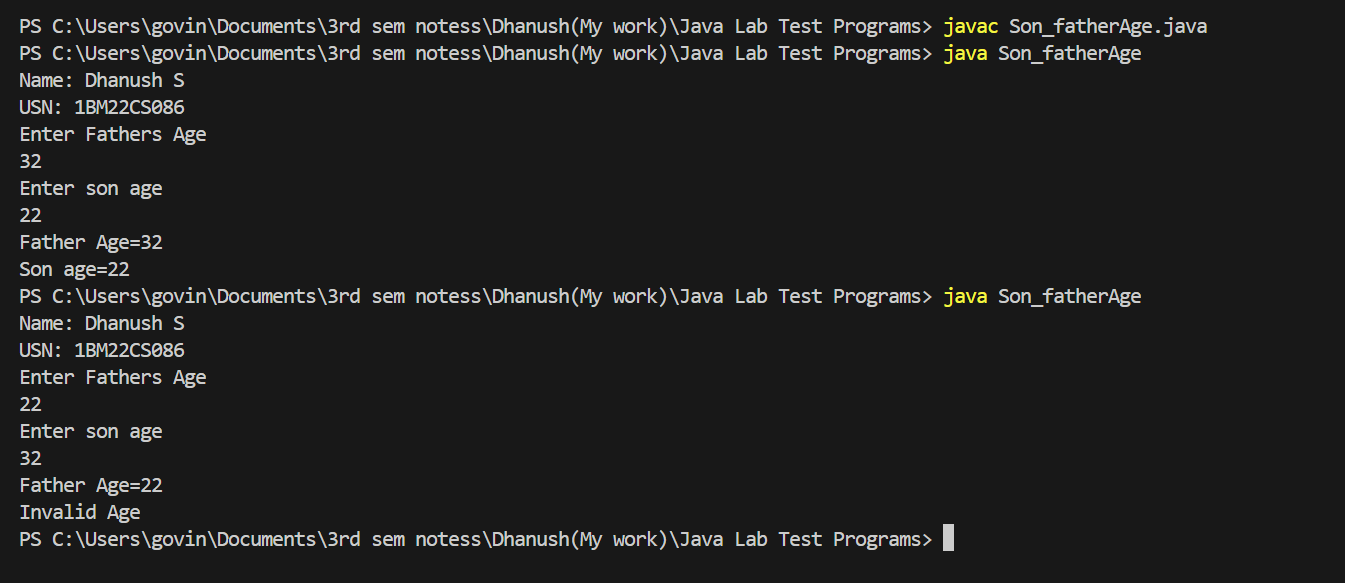
scanner.close();

}

}

}

**Output=>**

****

**Laboratory Program – 8**

**07) 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.**

**Program=>**

class DisplayThread extends Thread {

private String message;

private int interval;

public DisplayThread(String message, int interval) {

this.message = message;

this.interval = interval;

}

public void run() {

try {

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

System.out.println(message);

Thread.sleep(interval \* 1000);

}

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

class Multi\_Threading

{

public static void main(String[] args)

{

System.out.println("Name: Dhanush S\n USN:= 1BM22CS086");

DisplayThread thread1 = new DisplayThread("BMS College of Engineering", 10);

thread1.start();

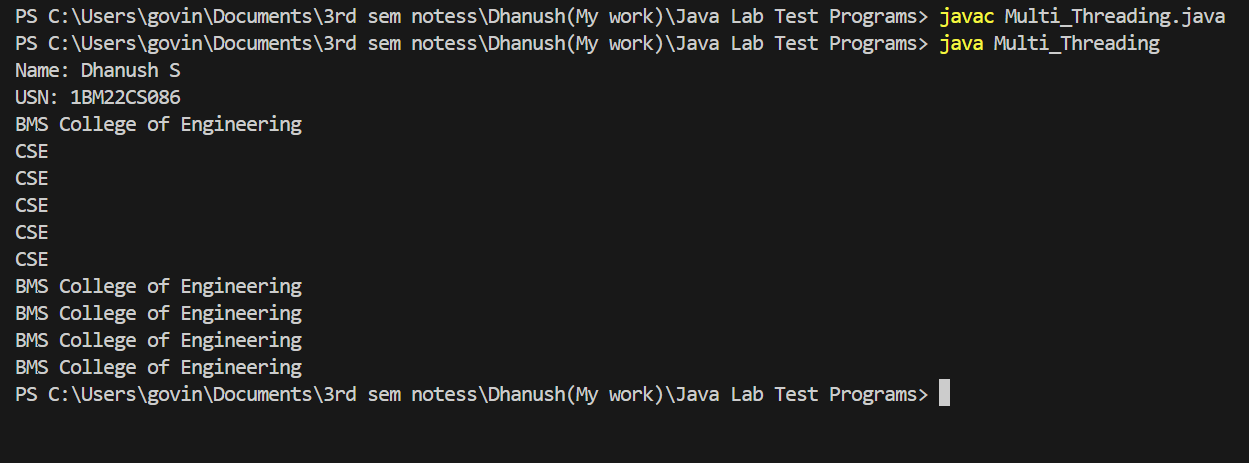
DisplayThread thread2 = new DisplayThread("CSE", 2);

thread2.start();

}

}

**Output=>**

****