VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Aaron B Ajay (1BM23CS003)

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
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 **Aaron B Ajay (1BM23CS003),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfilment 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.

Geetha N Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Sl No.	Date	Experiment Title	Page No.
1	09/10/24	Quadratic Equation Implementation	01
2	16/10/24	Student Class (SGPA Calculator)	06
3	23/10/24	Book Class (Constructors and Methods)	13
4	23/10/24	Shape Class (Abstract Class)	18
5	30/10/24	Bank Class (Inheritance)	23
6	13/11/24	Student Marks (Packages)	30
7	20/11/24	Father/Son Age Class (Exception Handling)	41
8	27/11/24	Multithreading	46
9	27/11/24	User Interface Division Calculator	49
10	27/11/24	IPC and Deadlock	55

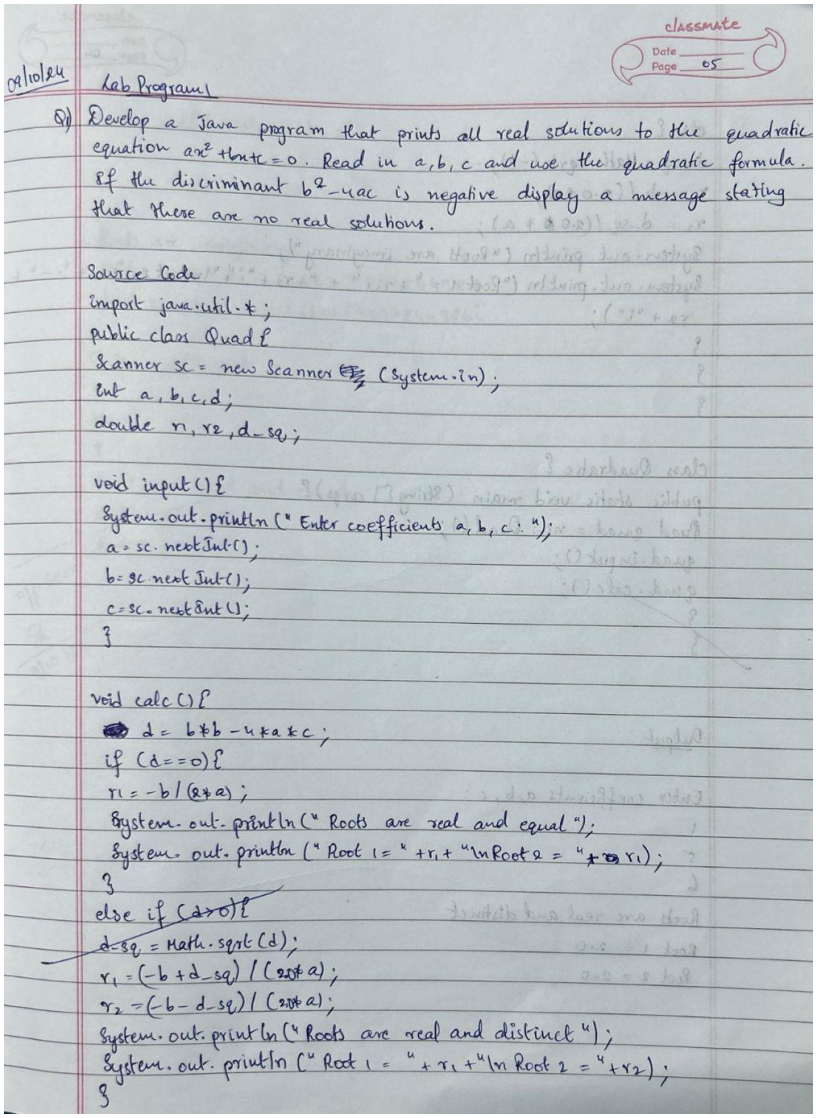
Github Link:

https://github.com/Aronnnn1/JAVA-1BM23CS003

Program 1

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

Observation Book



classmate

Date

Page 05

09/10/24

Lab Program 1

Q1) Develop a Java program that prints all real solutions to the quadratic equation $ax^2+bx+c=0$. Read in a, b, c and use the quadratic formula. If the discriminant b^2-4ac is negative display a message stating that there are no real solutions.

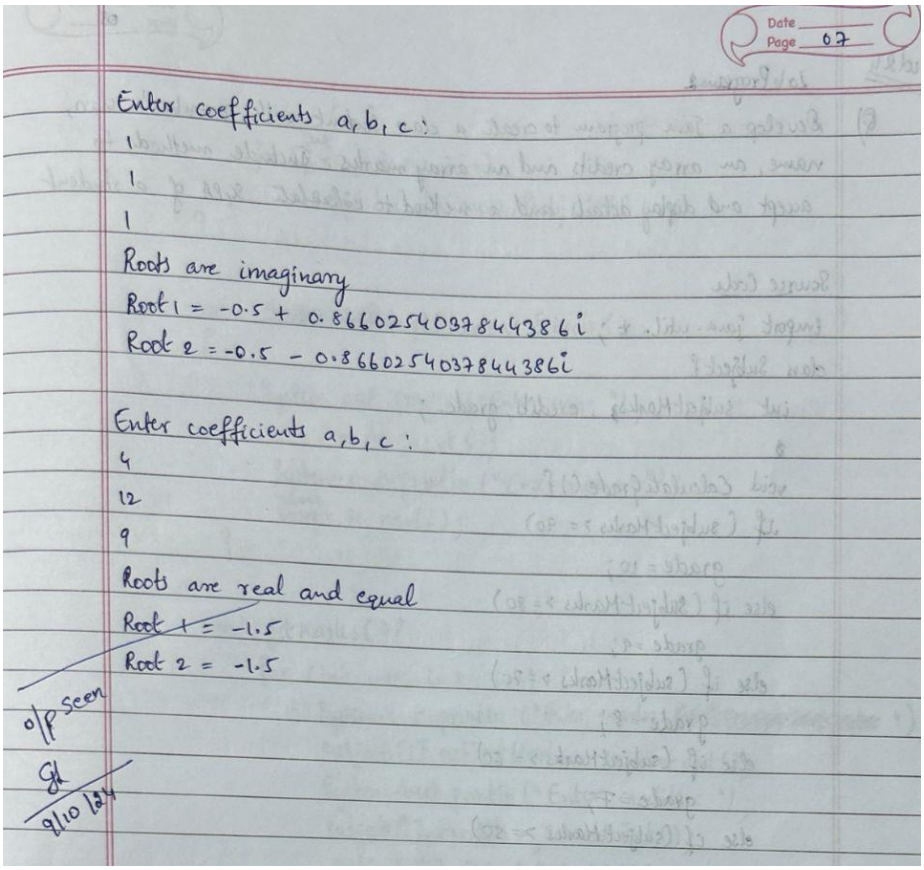
Source Code

```
import java.util.*;
public class Quad{
Scanner sc = new Scanner (System.in);
int a, b, c, d;
double r1, r2, d_sq;

void input(){
System.out.println(" Enter coefficients a, b, c: ");
a = sc.nextInt();
b = sc.nextInt();
c = sc.nextInt();
}

void calc(){
d = b*b - 4*a*c;
if (d==0){
r1 = -b/(2*a);
System.out.println(" Roots are real and equal ");
System.out.println(" Root 1 = " +r1+ "\n Root 2 = " + r1);
}
else if (d>0){
d_sq = Math.sqrt(d);
r1 = (-b + d_sq) / (2*a);
r2 = (-b - d_sq) / (2*a);
System.out.println(" Roots are real and distinct ");
System.out.println(" Root 1 = " + r1 + "\n Root 2 = " + r2);
}
```

```
else {
d_sq = Math.sqrt (-d);
r1 = -b /(2.0 * a);
r2 = d_sq /(2.0 * a);
System.out.println ("Roots are imaginary");
System.out.println ("Root 1 = " + r1 + " + " + r2 + "i" + "\n Root 2 = " + r1 + " - " +
 r2 + "i");
}
}
}

class Quadratic {
public static void main (String[] args) {
Quad quad = new Quad();
quad.input();
quad.calc();
}
}
```

Output

```
Enter coefficients a, b, c :
1
5
6
Roots are real and distinct
Root 1 = 2.0
Root 2 = 2.0
```



Date
Page 07

Enter coefficients a, b, c :
1
1
1
Roots are imaginary
Root 1 = -0.5 + 0.866025403784438 6i
Root 2 = -0.5 - 0.866025403784438 6i

Enter coefficients a, b, c :
4
12
9
Roots are real and equal
Root 1 = -1.5
Root 2 = -1.5

o/p seen
9/10/24

Record

Source Code

```
import java.util.*;

class Quad{
    Scanner sc=new Scanner(System.in);
    int a,b,c,d;
    double r1,r2,d_sq;

    void input(){
        System.out.println("Enter coefficients a,b,c:");
        a=sc.nextInt();
        b=sc.nextInt();
        c=sc.nextInt();
    }

    void calc(){
```

```
        d=b*b-4*a*c;
        if (d==0){
            r1=-b/(2.0*a);
            System.out.println("Roots are real and equal");
            System.out.println("Root 1 = "+r1+"\nRoot 2 = "+r1);
        }
        else if(d>0){
            d_sq=Math.sqrt(d);
            r1=(-b+d_sq)/(2.0*a);
            r2=(-b-d_sq)/(2.0*a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root 1 = "+r1+"\nRoot 2 = "+r2);
        }
        else{
            d_sq=Math.sqrt(-d);
            r1=-b/(2.0*a);
            r2=d_sq/(2.0*a);
            System.out.println("Roots are imaginary");
            System.out.println("Root 1 = "+r1+" + "+r2+"i"+"\nRoot 2 = "+r1+" - "+r2+"i");
        }
    }
}

class Quadratic{
    public static void main(String[] args){
        System.out.println("USN: 1BM23CS003");
        System.out.println("Name: Aaron B Ajay");
        Quad quad=new Quad();
        quad.input();
        quad.calc();
    }
}
```

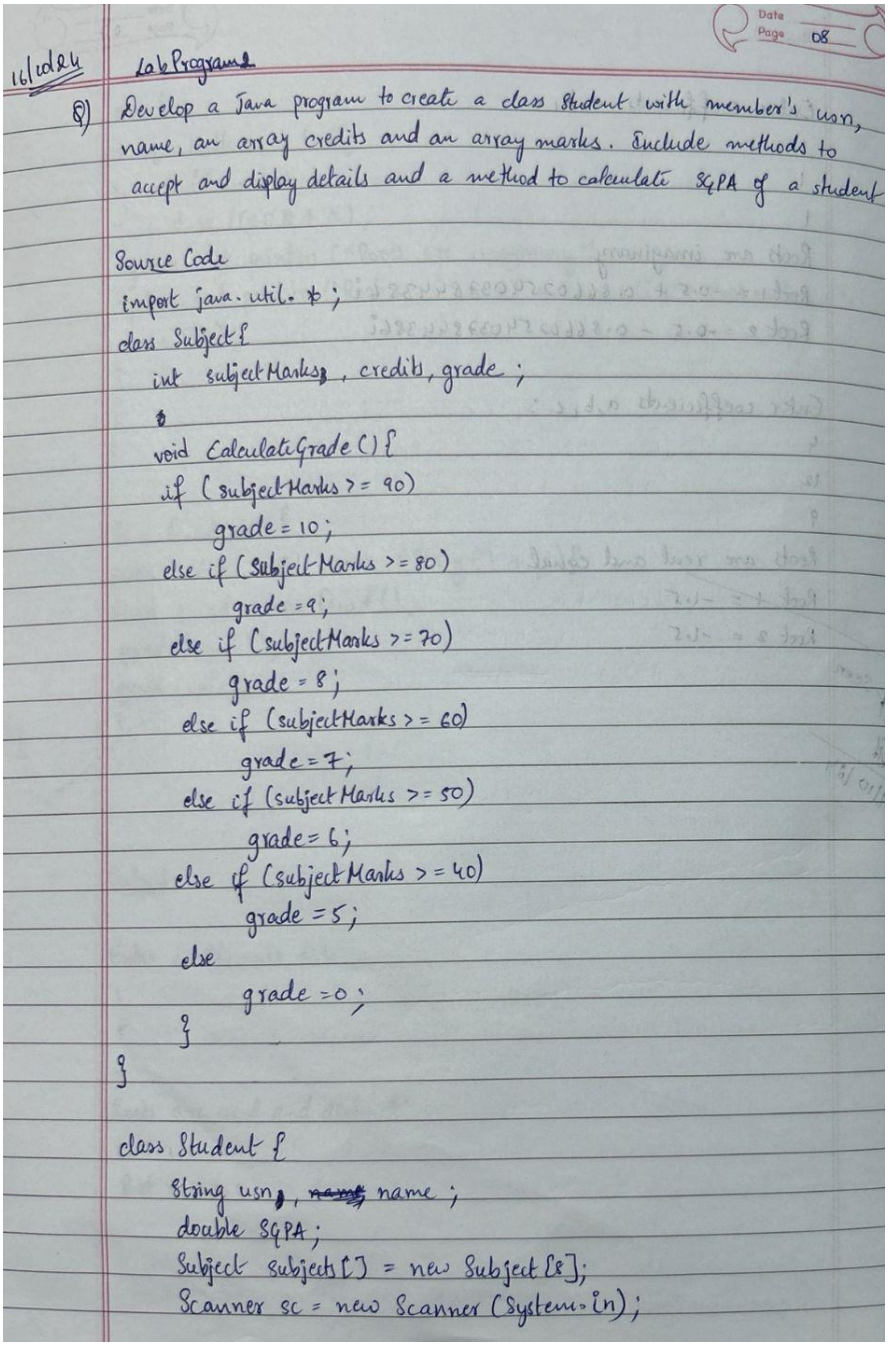
Output

```
C:\Users\Admin\Downloads\1BM23CS003>java Quadratic
USN: 1BM23CS003
Name: Aaron B Ajay
Enter coefficients a,b,c:
1

5
6
Roots are real and distinct
Root 1 = -2.0
Root 2 = -3.0

C:\Users\Admin\Downloads\1BM23CS003>java Quadratic
USN: 1BM23CS003
Name: Aaron B Ajay
Enter coefficients a,b,c:
1
1
1
Roots are imaginary
Root 1 = -0.5 + 0.8660254037844386i
Root 2 = -0.5 - 0.8660254037844386i

C:\Users\Admin\Downloads\1BM23CS003>java Quadratic
USN: 1BM23CS003
Name: Aaron B Ajay
Enter coefficients a,b,c:
4
12
9
Roots are real and equal
Root 1 = -1.5
Root 2 = -1.5
```

Program 2

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

Observation Book



Date
Page 08

16/10/24 Lab Program 2

Q) Develop a Java program to create a class Student with member's 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

Source Code

```
import java.util.*;
class Subject{
    int subjectMarks, credits, grade;

    void CalculateGrade(){
    if (subjectMarks >= 90)
        grade = 10;
    else if (subjectMarks >= 80)
        grade = 9;
    else if (subjectMarks >= 70)
        grade = 8;
    else if (subjectMarks >= 60)
        grade = 7;
    else if (subjectMarks >= 50)
        grade = 6;
    else if (subjectMarks >= 40)
        grade = 5;
    else
        grade = 0;
    }
}

class Student {
    String usn, name;
    double SGPA;
    Subject subjects[] = new Subject[8];
    Scanner sc = new Scanner(System.in);
```

Date
Page 09

```
void Student (){
    for (int i=0; i<8; i++)
        subjects[i]= new Subject();
}

void getStudentDetails()
{
    System.out.println("Enter usn: ");
    usn = sc.next();
    System.out.println("Enter name: ");
    name = sc.next();
}

void getMarks(){
    for (int i=0; i<8; i++) {
        System.out.println("Enter marks: ");
        subjects[i].subjectMarks = sc.nextInt();
        System.out.println("Enter credits : ");
        subjects[i].credits = sc.nextInt();
        subjects[i].CalculateGrade();
    }
}

void calcSGPA(){
    double Score = 0;
    int total_credits = 0;
    for (int i=0; i<8; i++) {
        Score += subjects[i].credits * subject[i].grade;
        total_credits += subjects[i].credits;
    }
    SGPA = Score / total_credits;
}
```

```
    void display(){
        System.out.println("USN: "+usn);
        System.out.println("Name: "+name);
        System.out.println("SGPA: "+SGPA);
    }
}

class StudDetails{
    public static void main(String[] args){
        Student students[] = new Student[3];
        for (int j=0; j<3; j++){
            System.out.println("Enter details for student "+(i+1)+":");
            students[j]= new Student();
            students[j].getStudentDetails();
            students[j].getMarks();
            students[j].calcSGPA();
        }
        for (int i=0; i<3; i++){
            students[i].display();
        }
    }
}
```

Seen execute

Output:

```
Enter the details for student 1:
Enter USN: 1BM23CS001
Enter Name: AA
Enter marks: 99
Enter credits: 4
Enter marks: 98
Enter credits: 1
```



```
Enter marks: 97
Enter credits: 4
Enter marks: 96
Enter credits: 1
Enter marks: 78
Enter credits: 1
Enter marks: 86
Enter credits: 3
Enter marks: 90
Enter credits: 3
Enter marks: 91
Enter credits: 3
USN: 1BM23CS001
Name: AA
SGPA: 9.75
```

o/p Seen
Sl
16/10/24

Record

Source Code

```
import java.util.Scanner;
class Subject {
    int subjectMarks;
    int credits;
    int grade;

    void calculateGrade() {
        if (subjectMarks >= 90)
            grade = 10;
        else if (subjectMarks >= 80)
            grade = 9;
        else if (subjectMarks >= 70)
            grade = 8;
        else if (subjectMarks >= 60)
            grade = 7;
        else if (subjectMarks >= 50)
            grade = 6;
        else if (subjectMarks >= 40)
            grade = 5;
        else
            grade = 0;
    }
}

class Student {
    String usn;
    String name;
    double SGPA;
    Subject[] subjects = new Subject[8];
    Scanner sc = new Scanner(System.in);

    Student() {
        for (int i = 0; i < 8; i++)
            subjects[i] = new Subject();
    }

    void getStudentDetails() {
        System.out.print("Enter the USN: ");
```

```
        usn = sc.next();
        System.out.print("Enter the Name: ");
        name = sc.next();
    }

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

    void calcSGPA() {
        double score = 0;
        int total_credits = 0;

        for (int i = 0; i < 8; i++) {
            score += (subjects[i].grade * subjects[i].credits);
            total_credits += subjects[i].credits;
        }
        SGPA = score/total_credits;
    }

    void display() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("SGPA: " + SGPA);
    }
}

public class StudDetails {
    public static void main(String[] args) {
        System.out.println("USN: 1BM23CS003");
        System.out.println("Name: Aaron B Ajay");
        Student[] students = new Student[3];

        for (int j = 0; j < 3; j++) {
            System.out.println("Enter the details for student " + (j + 1) + ":");
```

```
            students[j] = new Student();
            students[j].getStudentDetails();
            students[j].getMarks();
            students[j].calcSGPA();
        }

        for (int i=0; i<8; i++) {
            students[i].display();
        }
    }
}
```

Output

```
C:\Users\Admin\Desktop\1BM23CS003>java StudDetails
USN: 1BM23CS003
Name: Aaron B Ajay
Enter the details for student 1:
Enter the USN: 1BM23CS001
Enter the Name: AA
Enter marks for subject 1: 99
Enter credits for subject 1: 4
Enter marks for subject 2: 98
Enter credits for subject 2: 1
Enter marks for subject 3: 97
Enter credits for subject 3: 4
Enter marks for subject 4: 96
Enter credits for subject 4: 1
Enter marks for subject 5: 78
Enter credits for subject 5: 1
Enter marks for subject 6: 86
Enter credits for subject 6: 3
Enter marks for subject 7: 90
Enter credits for subject 7: 3
Enter marks for subject 8: 91
Enter credits for subject 8: 3
```

```
Enter the USN: 1BM23CS002
Enter the Name: AB
Enter marks for subject 1: 89
Enter credits for subject 1: 1
Enter marks for subject 2: 78
Enter credits for subject 2: 1
Enter marks for subject 3: 89
Enter credits for subject 3: 1
Enter marks for subject 4: 90
Enter credits for subject 4: 4
Enter marks for subject 5: 98
Enter credits for subject 5: 4
Enter marks for subject 6: 99
Enter credits for subject 6: 3
Enter marks for subject 7: 87
Enter credits for subject 7: 3
Enter marks for subject 8: 89
Enter credits for subject 8: 3
```

```
Enter the details for student 3:
Enter the USN: 1BM23CS004
Enter the Name: AD
Enter marks for subject 1: 90
Enter credits for subject 1: 4
Enter marks for subject 2: 95
Enter credits for subject 2: 4
Enter marks for subject 3: 79
Enter credits for subject 3: 1
Enter marks for subject 4: 89
Enter credits for subject 4: 1
Enter marks for subject 5: 88
Enter credits for subject 5: 1
Enter marks for subject 6: 90
Enter credits for subject 6: 3
Enter marks for subject 7: 99
Enter credits for subject 7: 3
Enter marks for subject 8: 93
Enter credits for subject 8: 3
```

```
USN: 1BM23CS001
Name: AA
SGPA: 9.75
USN: 1BM23CS002
Name: AB
SGPA: 9.5
USN: 1BM23CS004
Name: AD
SGPA: 9.8
```

Program 3

Create a class Book which contains four members: name, author, price and 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.

Observation Book

23/10/24 Lab Program 3

Q) 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 & 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.

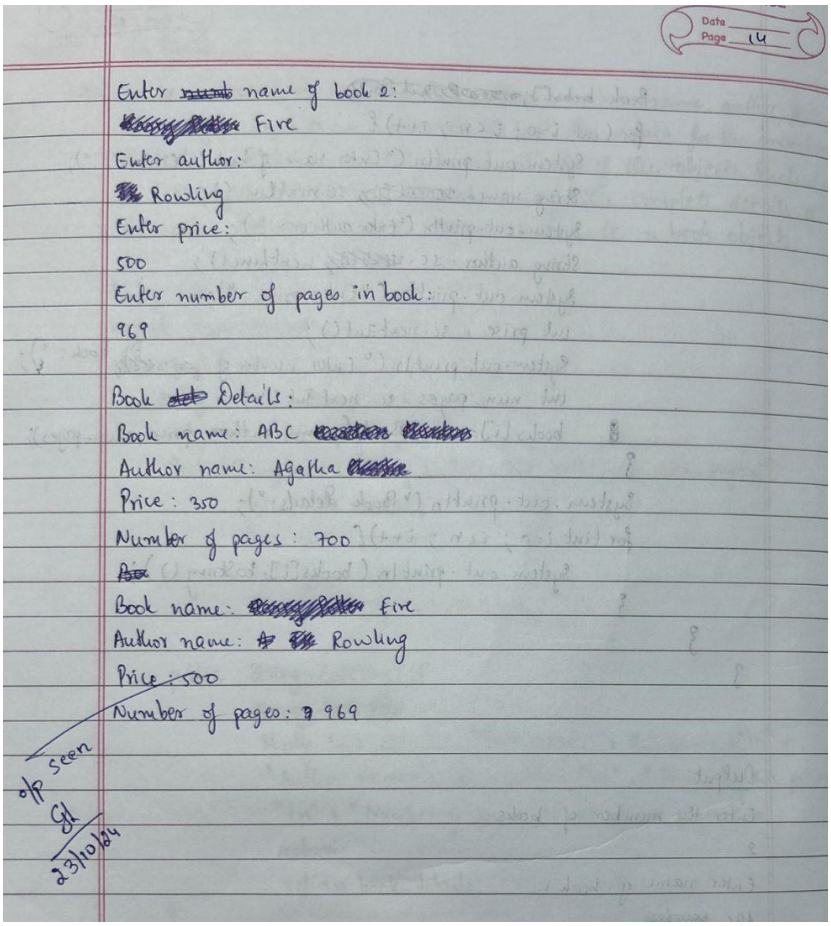
Source Code

```
import java.util.*;
class Book{
    String name, author;
    int price, num_pages;
    Book (String name, String author, int price, int num_pages){
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;

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

class Run{
    public static void main (String[] args) {
        Scanner sc = new Scanner (System.in);
        System.out.println("Enter number of books: ");
        int n = sc.nextInt();
```

```
        Book books[];
        for (int i=0; i<n; i++) {
            System.out.println("Enter name of book "+(i+1)+": ");
            String name= sc.nextLine();
            System.out.println("Enter author: ");
            String author = sc.nextLine();
            System.out.println("Enter price: ");
            int price = sc.nextInt();
            System.out.println("Enter number of pages in book: ");
            int num_pages = sc.nextInt();
            books[i]= new Book(name, author, price, num_pages);
        }
        System.out.println("Book details:");
        for (int i=0; i<n; i++){
            System.out.println(books[i].toString());
        }
    }
}
```

Output

```
Enter the number of books:
2
Enter name of book 1:
ABC
Enter author:
Agatha
Enter price:
350
Enter number of pages in book:
700
```



Date
Page 14

Enter name of book 2:
Fire
Enter author:
Rowling
Enter price:
500
Enter number of pages in book:
969

Book Details:
Book name: ABC
Author name: Agatha
Price : 350
Number of pages : 700

Book name: Fire
Author name: Rowling
Price : 500
Number of pages: 969

o/p seen
23/10/24

Record

Source Code

```
import java.util.Scanner;
class Book {
    String name;
    String author;
    int price;
    int num_pages;
    Book(String name, String author, int price, int num_pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;
    }
    public String toString() {
        String book_details = "Book name: " + this.name + "\n" +
```

```
        "Author name: " + this.author + "\n" +
        "Price: " + this.price + "\n" +
        "Number of pages: " + this.num_pages + "\n";
        return book_details;
    }
}

class Run {
    public static void main(String[] args) {
        System.out.println("Name: Aaron B Ajay");
        System.out.println("USN: 1BM23CS003 ");
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of books: ");
        int n = sc.nextInt();
        Book[] books=new Book[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter name of book " + (i + 1) + ": ");
            String name = sc.next();
            sc.nextLine();
            System.out.println("Enter author: ");
            String author = sc.next();
            sc.nextLine();
            System.out.println("Enter price: ");
            int price = sc.nextInt();
            System.out.println("Enter number of pages in book: ");
            int num_pages = sc.nextInt();
            books[i] = new Book(name, author, price, num_pages);
        }
        System.out.println("\nBook Details:");
        for (int i=0; i<n;i++) {
            System.out.println(books[i].toString());
        }
    }
}
```

Output

```
Name: Aaron B Ajay
USN: 1BM23CS003
Enter the number of books:
3
Enter name of book 1:
ABC
Enter author:
Agatha
Enter price:
350
Enter number of pages in book:
700
```

```
Enter name of book 2:
Fire
Enter author:
Rowling
Enter price:
500
Enter number of pages in book:
969
```

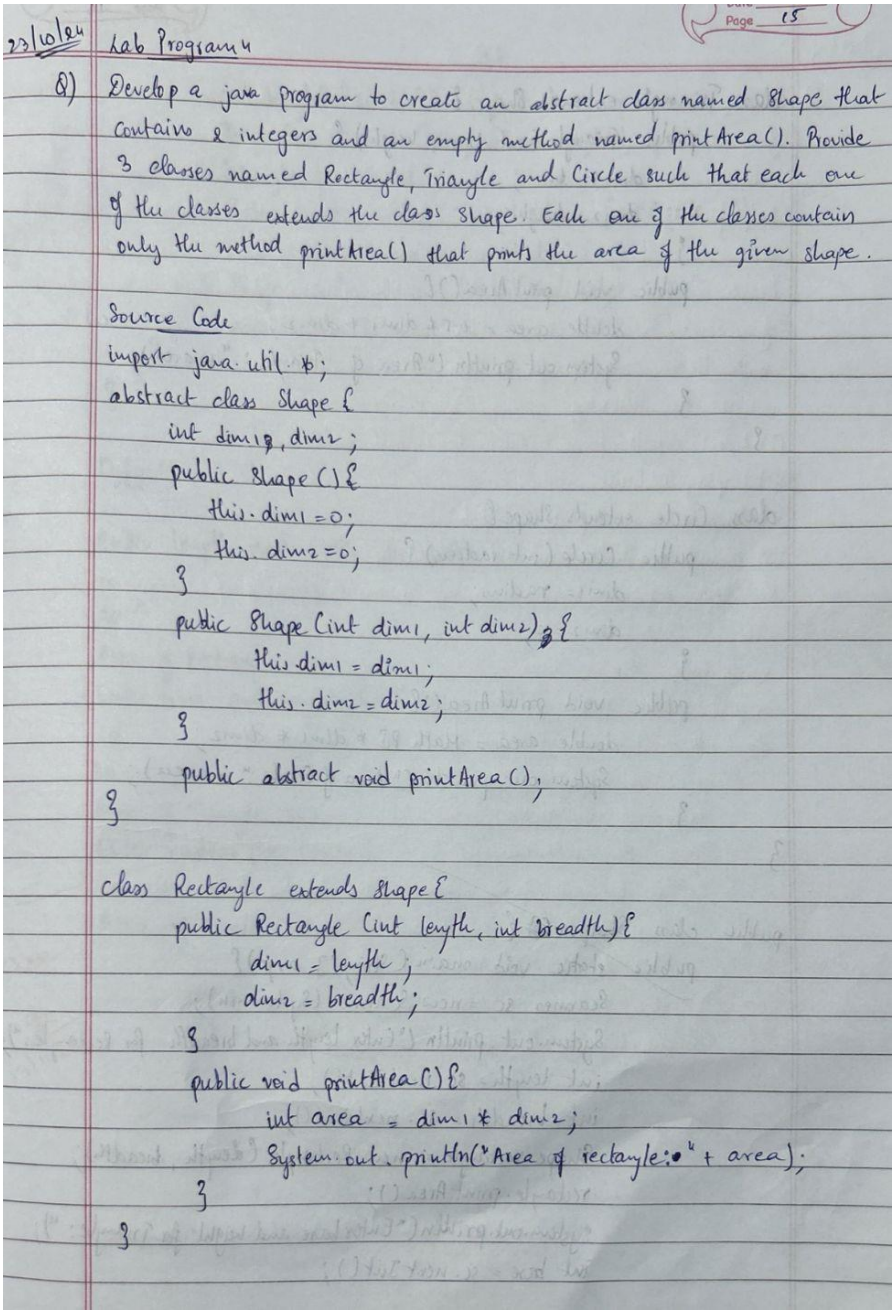
```
Enter name of book 3:
LOTR
Enter author:
Tolkein
Enter price:
680
Enter number of pages in book:
970
```

```
Book Details:
Book name: ABC
Author name: Agatha
Price: 350
Number of pages: 700

Book name: Fire
Author name: Rowling
Price: 500
Number of pages: 969

Book name: LOTR
Author name: Tolkein
Price: 680
Number of pages: 970
```

Program 4

Develop a Java program to create an abstract class named Shape that contains 2 integers and an empty method named printArea(). Provide 3 classes names 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.

Observation Book



23/10/24 Lab Program 4

Q) Develop a java program to create an abstract class named Shape that contains 2 integers and an empty method named printArea(). Provide 3 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.

Source Code

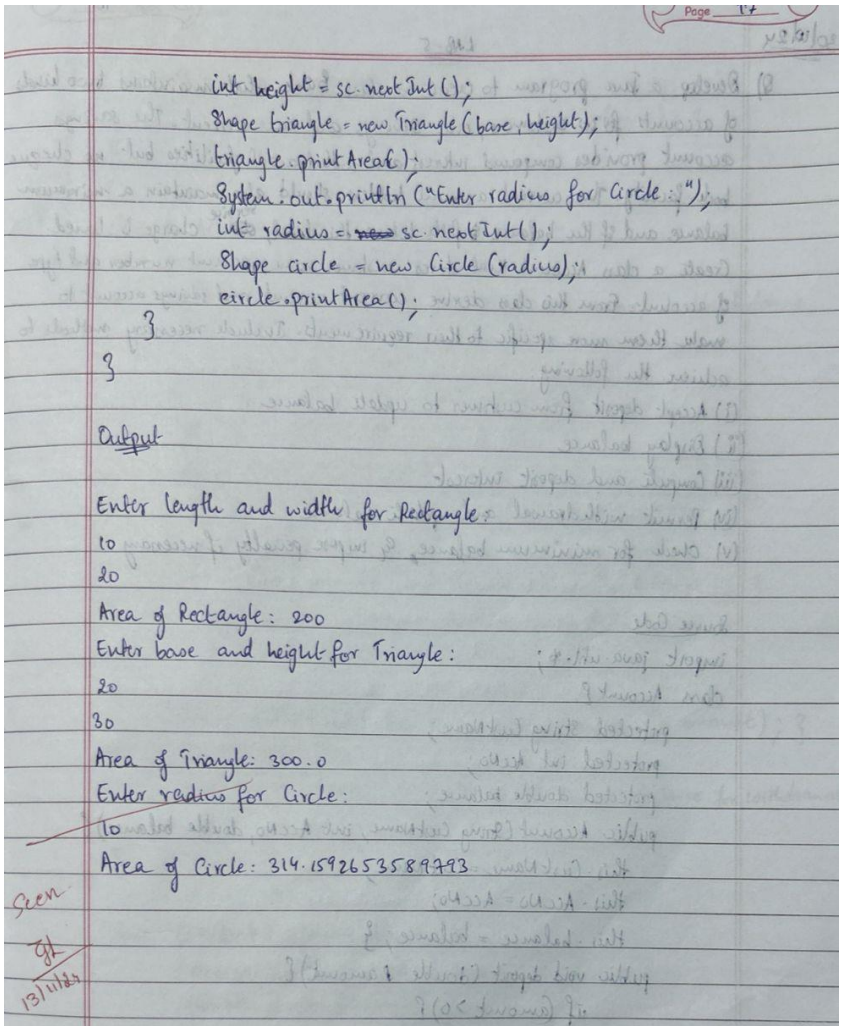
```
import java.util.*;
abstract class Shape {
    int dim1, dim2;
    public Shape(){
        this.dim1=0;
        this.dim2=0;
    }
    public Shape(int dim1, int dim2){
        this.dim1 = dim1;
        this.dim2 = dim2;
    }
    public abstract void printArea();
}

class Rectangle extends Shape{
    public Rectangle(int length, int breadth){
        dim1 = length;
        dim2 = breadth;
    }
    public void printArea(){
        int area = dim1 * dim2;
        System.out.println("Area of rectangle:" + area);
    }
}
```

23/10/24 Lab Program 4

Q) Develop a java program to create an abstract class named Shape that contains 2 integers and an empty method named printArea(). Provide 3 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.

Source Code

```
import java.util.*;
abstract class Shape {
    int dim1, dim2;
    public Shape(){
        this.dim1=0;
        this.dim2=0;
    }
    public Shape(int dim1, int dim2){
        this.dim1 = dim1;
        this.dim2 = dim2;
    }
    public abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int breadth){
        dim1 = length;
        dim2 = breadth;
    }
    public void printArea(){
        int area = dim1 * dim2;
        System.out.println("Area of rectangle:" + area);
    }
}
```



```
        int height = sc.nextInt();
        Shape triangle = new Triangle(base, height);
        triangle.printArea();
        System.out.println("Enter radius for Circle:");
        int radius = sc.nextInt();
        Shape circle = new Circle(radius);
        circle.printArea();
    }
}
```

Output

```
Enter length and width for Rectangle:
10
20
Area of Rectangle: 200
Enter base and height for Triangle:
20
30
Area of Triangle: 300.0
Enter radius for Circle:
10
Area of Circle: 314.159265358979 3
```

Seen

13/11/24

Record

Source Code

```
import java.util.Scanner;

abstract class Shape {
    int dim1;
    int dim2;
    public Shape() {
        this.dim1 = 0;
        this.dim2 = 0;
    }
    public Shape(int dim1, int dim2) {
        this.dim1 = dim1;
        this.dim2 = dim2;
```

```
    }
    public abstract void printArea();

}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        dim1 = length;
        dim2 = width;
    }
    public void printArea() {
        int area = dim1 * dim2;
        System.out.println("Area of Rectangle: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        dim1 = base;
        dim2 = height;
    }
    public void printArea() {
        double area = 0.5 * dim1 * dim2;
        System.out.println("Area of Triangle: " + area);
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        dim1 = radius;
        dim2 = 0;
    }

    public void printArea() {
        double area = Math.PI * dim1 * dim1;
        System.out.println("Area of Circle: " + area);
    }
}

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

```
        System.out.println("USN: 1BM23CS003 \nName: Aaron B Ajay");
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter length and width for Rectangle:");
        int length = sc.nextInt();
        int width = sc.nextInt();
        Shape rectangle = new Rectangle(length, width);
        rectangle.printArea();
        System.out.println("Enter base and height for Triangle:");
        int base = sc.nextInt();
        int height = sc.nextInt();
        Shape triangle = new Triangle(base, height);
        triangle.printArea();
        System.out.println("Enter radius for Circle:");
        int radius = sc.nextInt();
        Shape circle = new Circle(radius);
        circle.printArea();
    }
}
```

Output

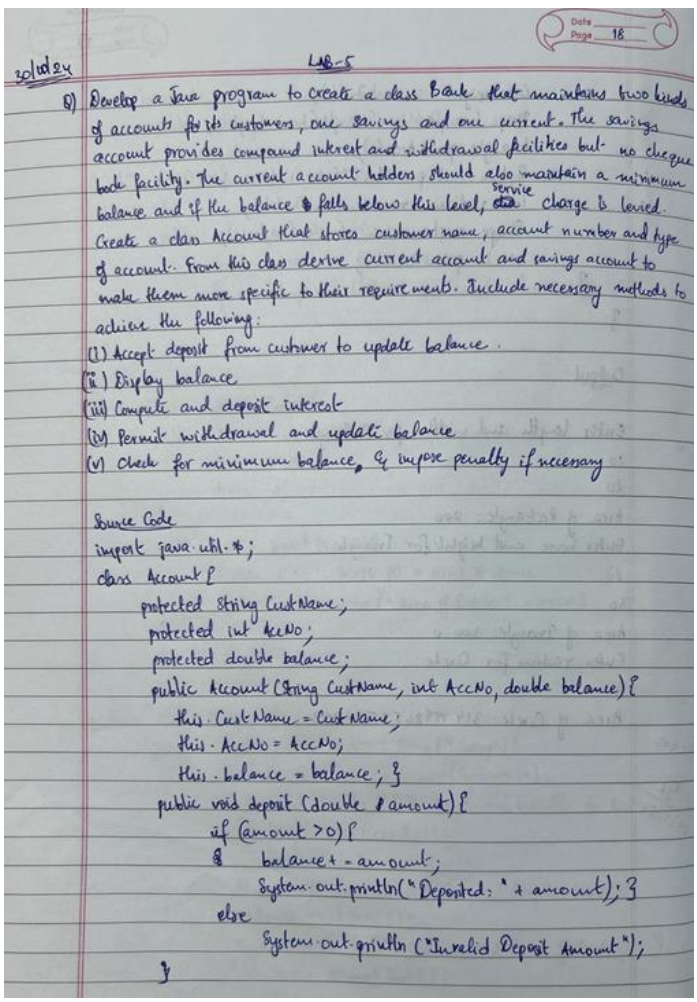
```
USN: 1BM23CS003
Name: Aaron B Ajay
Enter length and width for Rectangle:
10
20
Area of Rectangle: 200
Enter base and height for Triangle:
20
30
Area of Triangle: 300.0
Enter radius for Circle:
10
Area of Circle: 314.1592653589793
```

Program 5

Develop a Java program to create a class Bank that maintains two kinds of accounts for its customers, one savings and one current. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account holders should also maintain a minimum balance and if the balance falls below this level, service charge is levied. Create a class Account that stores customer name, account number and type of account. From this class, derive current account and savings account to make them more specific to their requirements. Include necessary methods to achieve the following:

(i) Accept deposit from customer to update balance.
(ii) Display balance
(iii) Compute and deposit interest
(iv) Permit withdrawal and update balance
(v) Check for minimum balance and impose penalty if necessary

Observation Book



Date
Page 18

30/10/24 LAB-5

Q) Develop a Java program to create a class Bank that maintains two kinds of accounts for its customers, one savings and one current. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account holders should also maintain a minimum balance and if the balance falls below this level, service charge is levied. Create a class Account that stores customer name, account number and type of account. From this class derive current account and savings account to make them more specific to their requirements. Include necessary methods to achieve the following:

(i) Accept deposit from customer to update balance.
(ii) Display balance
(iii) Compute and deposit interest
(iv) Permit withdrawal and update balance
(v) Check for minimum balance, & impose penalty if necessary

Source Code

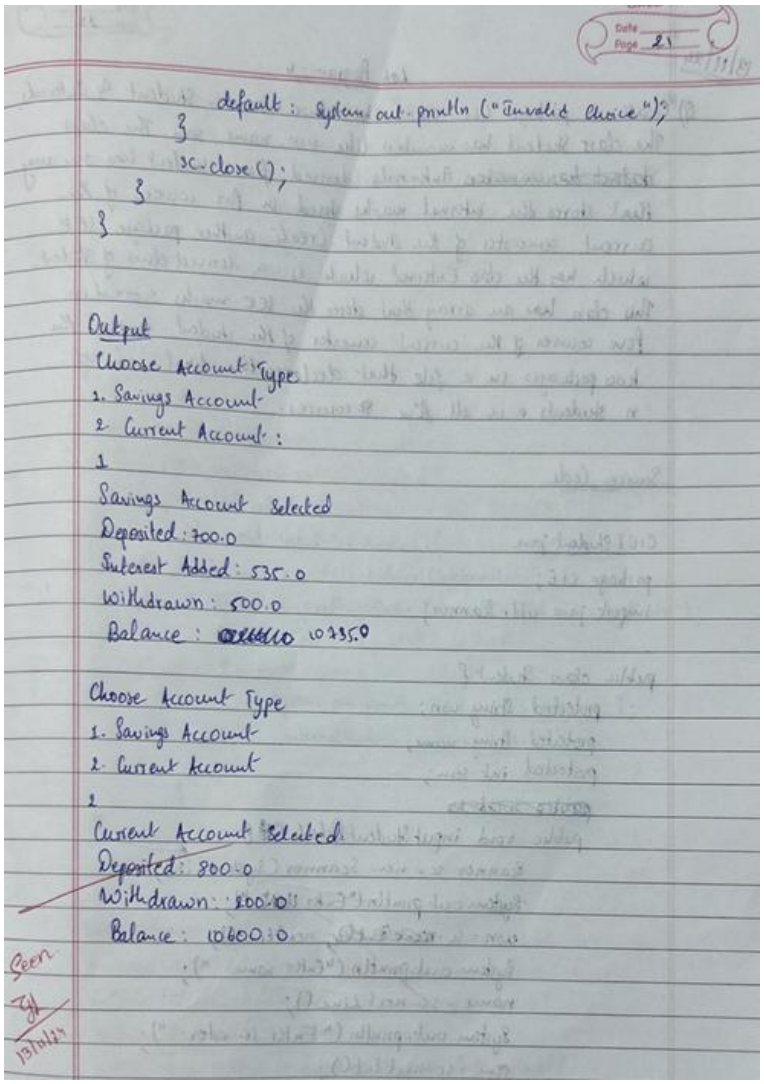
```
import java.util.*;
class Account {
    protected String CustName;
    protected int AccNo;
    protected double balance;
    public Account (String CustName, int AccNo, double balance) {
        this.CustName = CustName;
        this.AccNo = AccNo;
        this.balance = balance; }
    public void deposit (double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount); }
        else
            System.out.println ("Invalid Deposit Amount");
    }
```

Date
Page 19

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

class SavAcct extends Account{
    private double interestRate;
    public SavAcct (String CustomerName, int accno, double Balance,
    double interestRate){
        super (CustomerName, accno, Balance);
        this.interestRate = interestRate;
    }
    public void CompInterest (){
        double interest = balance * (interestRate/100);
        balance += interest;
        System.out.println("Interest Added: " + interest); }
    public void withdraw (double amount){
        if (amount < balance)
        {   balance -= amount;
            System.out.println("Withdrawn: " + amount); }
        else
            System.out.println("Insufficient Balance for withdrawal");
    }
}

class CurrAcct extends Account{
    private double minimumBalance;
    private double servicecharge;
    public CurrAcct (String CustomerName, int accno, double Balance,
    double minimumBalance, double servicecharge){
        super (CustomerName, accno, Balance);
        this.minimumBalance = minimumBalance;
        this.servicecharge = servicecharge; }
```

Date
Page 20

```
    public void withdraw (double amount){
        if (amount <= balance){
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
            if (balance < minimumBalance){
                balance -= serviceCharge;
                System.out.println("Service charge imposed: "
                + serviceCharge); }
        }
    }
}

public class Bank {
    public static void main (String[] args) {
        Scanner sc = new Scanner (System.in);
        SavAcct SavAcc = new SavAcct ("Aaron", 12345, 10000, 5);
        CurrAcct CurrAcc = new CurrAcct ("Aaron", 12345, 10000, 5000, 500);
        System.out.println ("Choose Account Type \n1. Savings Account
        \n2. Current Account: ");
        int ch = sc.nextInt ();
        switch (ch) {
            case 1: System.out.println ("Savings Account Selected");
                    SavAcc.deposit (700);
                    SavAcc.CompInterest ();
                    SavAcc.withdraw (500);
                    SavAcc.displayBalance ();
                    break;
            case 2: System.out.println ("Current Account Selected");
                    CurrAcc.deposit (600);
                    CurrAcc.withdraw (200);
                    CurrAcc.displayBalance ();
                    break;
```



```
        default : System.out.println ("Invalid Choice");
        }
        sc.close();
    }
}
```

Output

```
Choose Account Type
1. Savings Account
2. Current Account :
1
Savings Account Selected
Deposited: 700.0
Interest Added: 535.0
Withdrawn: 500.0
Balance : 10235.0

Choose Account Type
1. Savings Account
2. Current Account
2
Current Account Selected
Deposited: 800.0
Withdrawn: 800.0
Balance: 10600.0
```

Record

Source Code

```
import java.util.Scanner;

class Account {
    protected String CustName;
    protected int AccNo;
    protected double balance;

    public Account(String CustName, int AccNo, double balance) {
        this.CustName = CustName;
        this.AccNo = AccNo;
        this.balance = balance;
```

```
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount);
        }
        else {
            System.out.println("Invalid Deposit Amount");
        }
    }

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

class SavAcct extends Account {
    private double interestRate;

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

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

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

```
class CurAcct extends Account {
    private double minimumBalance;
    private double serviceCharge;

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

    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);

            if (balance < minimumBalance) {
                balance -= serviceCharge;
                System.out.println("Service charge imposed: " + serviceCharge);
            }
        }
        else {
            System.out.println("Insufficient balance for withdrawal");
        }
    }
}

public class Bank {
    public static void main(String[] args) {
            System.out.println("USN: 1BM23CS003 \nName: Aaron B Ajay");
        Scanner sc = new Scanner(System.in);
        SavAcct savAcc = new SavAcct("Aaron", 12345, 10000, 5);
        CurAcct curAcc = new CurAcct("Aaron", 12345, 10000, 5000, 500);

        System.out.println("Choose Account Type:\n1. Savings Account\n2. Current Account");
        int choice = sc.nextInt();

        switch (choice) {
            case 1:
                System.out.println("Savings Account Selected");
```

```
            savAcc.deposit(700);
            savAcc.computeAndDepositInterest();
            savAcc.withdraw(500);
            savAcc.displayBalance();
            break;

        case 2:
            System.out.println("Current Account Selected");
            curAcc.deposit(800);
            curAcc.withdraw(200);
            curAcc.displayBalance();
            break;

        default:
            System.out.println("Invalid choice");
    }

    sc.close();
  }
}
```

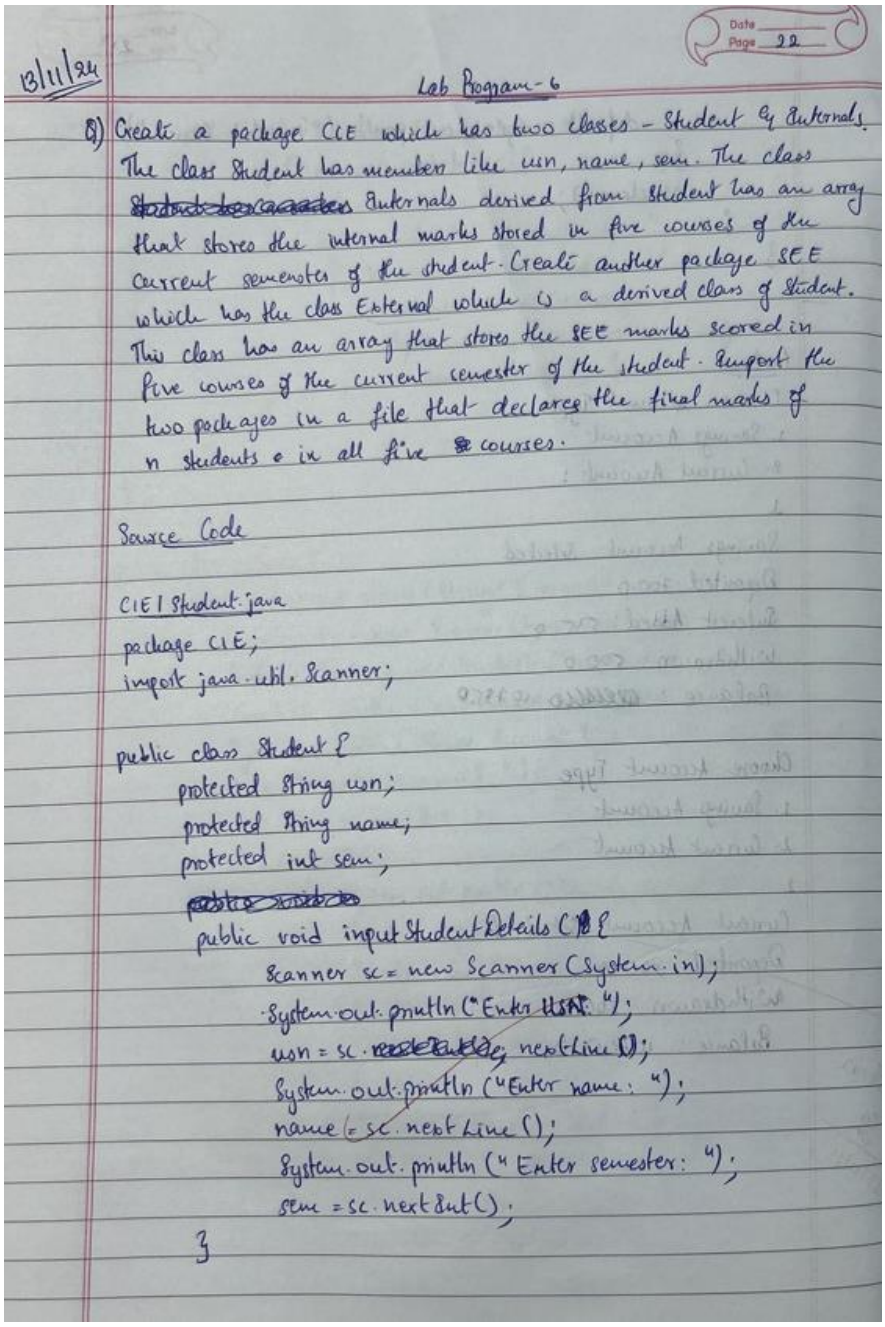
Output

```
USN: 1BM23CS003
Name: Aaron B Ajay
Choose Account Type:
1. Savings Account
2. Current Account
1
Savings Account Selected
Deposited: 700.0
Interest added: 535.0
Withdrawn: 500.0
Balance: 10735.0

C:\Users\Asha\Desktop\Aaron\python\Lab 6 student Marks Calculation>java Bank
USN: 1BM23CS003
Name: Aaron B Ajay
Choose Account Type:
1. Savings Account
2. Current Account
2
Current Account Selected
Deposited: 800.0
Withdrawn: 200.0
Balance: 10600.0
```

Program 6

Create a package CIE which has two classes – Student and Internals. The class Student has members like usn, name and 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 Externals which is a derived class of Student. This class 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.

Observation Book



13/11/24

Date
Page 22

Lab Program-6

Q) Create a package CIE which has two classes - Student & Internals. The class Student has member like usn, name, sem. The class Internals derived from Student has an array that stores the internal marks stored 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.

Source Code

CIE/Student.java

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

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

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

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

CIE/Internals.java

```
package CIE;
import java.util.Scanner;
public class Internals extends Student{
    protected int[] internalMarks = new int[5];
    public void inputCIEMarks(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Internal Marks for 5 Subjects:");
        for (int i=0; i<5; i++)
        {
            System.out.println("Subject " + (i+1) + ":");
            internalMarks[i] = sc.nextInt();
        }
    }
}
```

SEE/Externals.java

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals(){
    protected int[] seeMarks = new int[5];
    protected int[] finalMarks = new int[5];
```

```
public void inputSEEmarks(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter SEE marks for 5 subjects: ");
        for (int i=0; i<5; i++){
                System.out.println("Subject "+(i+1)+": ");
                seeMarks[i] = sc.nextInt();
        }
}
public void calculateFinalMarks(){
        for (int i=0; i<5; i++){
                finalMarks[i]= internalMarks[i] + seeMarks[i];
        }
}
public void displayFinalMarks(){
        displayStudentDetails();
        System.out.println("Final Marks for 5 subjects: ");
        for (int i=0; i<5; i++){
                System.out.println("Subject "+(i+1)+":"+ finalMarks[i]);
        }
}
}


import java.util.Scanner;
import SEE.Externals;
class Main {
    public static void main(String[] args){
        Scanner sc= new Scanner(System.in);
        System.out.print("Enter number of students: ");
        int n= sc.nextInt();
        Externals[] students = new Externals[n];
```

```
for (int i=0; i< n; i++) {
        System.out.println("\n Enter details for student "+(i+1)+":");
        students[i] = new Externals();
        students[i].inputStudentDetails();
        students[i].input CIEmarks();
        students[i].input SEEmarks();
        students[i].calculateFinalMarks();
    }
    System.out.println("\nFinal Marks of Students:");
    for (int i=0; i<n; i++) {
        System.out.println("\nStudent " + (i+1) + ":");
        students[i].displayFinalMarks();
    }
  }
}
```

Output

```
Enter number of students: 1
Enter USN: 1BM23CS003
Enter Name: Aaron
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 45
Subject2: 46
Subject3: 47
Subject 4: 48
Subject 5: 49
Enter SEE Marks for 5 subjects:
Subject 1: 45
Subject 2: 46
Subject 3: 47
```



```
Subject 4: 48
Subject 5: 49

Final Marks of Students:

Student 1:
USN: 1BM23CS003
Name: Aaron
Semester: 3
Final Marks for 5 subjects:
Subject 1: 90
Subject 2: 92
Subject 3: 94
Subject 4: 96
Subject 5: 98
```

13/11/24

Record

Source Code

(i) CIE/Student.java

```
package CIE;
import java.util.Scanner;
public class Student {
    protected String usn;
    protected String name;
    protected int sem;

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

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

(ii) CIE/Internals.java

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

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

    public void inputCIEmarks() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Internal Marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
```

```
            internalMarks[i] = sc.nextInt();
        }
    }
}
```

(iii) SEE/Externals.java

```
package SEE;
import CIE.Internals;
import java.util.Scanner;

public class Externals extends Internals {
    protected int[] seeMarks = new int[5];
    protected int[] finalMarks = new int[5];

    public void inputSEEmarks() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter SEE Marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
            seeMarks[i] = sc.nextInt();
        }
    }

    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = internalMarks[i] + seeMarks[i];
        }
    }

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

(iv) Main function

```
import SEE.Externals;
```

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
            System.out.println("USN: 1BM23CS003 \nName: Aaron B Ajay");
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of students: ");
        int n = sc.nextInt();

        Externals[] students = new Externals[n];

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

        System.out.println("\nFinal Marks of Students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            students[i].displayFinalMarks();
        }
    }
}
```

Output

```
USN: 1BM23CS003
Name: Aaron B Ajay
Enter number of students: 5

Enter details for student 1:
Enter USN: 1BM23CS001
Enter Name: AA
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 45
Subject 2: 46
Subject 3: 47
Subject 4: 48
Subject 5: 49
Enter SEE Marks for 5 subjects:
Subject 1: 45
Subject 2: 46
Subject 3: 47
Subject 4: 48
Subject 5: 49
```

```
Enter details for student 2:
Enter USN: 1BM23CS002
Enter Name: AB
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 49
Subject 2: 48
Subject 3: 47
Subject 4: 46
Subject 5: 45
Enter SEE Marks for 5 subjects:
Subject 1: 49
Subject 2: 48
Subject 3: 47
Subject 4: 46
Subject 5: 45
```

```
Enter details for student 3:
Enter USN: 1BM23CS003
Enter Name: AC
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 41
Subject 2: 42
Subject 3: 43
Subject 4: 44
Subject 5: 45
Enter SEE Marks for 5 subjects:
Subject 1: 41
Subject 2: 42
Subject 3: 43
Subject 4: 44
Subject 5: 45
```

```
Enter details for student 4:
Enter USN: 1BM23CS004
Enter Name: AD
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 45
Subject 2: 44
Subject 3: 43
Subject 4: 42
Subject 5: 41
Enter SEE Marks for 5 subjects:
Subject 1: 45
Subject 2: 44
Subject 3: 43
Subject 4: 42
Subject 5: 41
```

```
Enter details for student 5:
Enter USN: 1BM23CS005
Enter Name: AE
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 43
Subject 2: 48
Subject 3: 49
Subject 4: 42
Subject 5: 47
Enter SEE Marks for 5 subjects:
Subject 1: 48
Subject 2: 49
Subject 3: 45
Subject 4: 42
Subject 5: 41
```

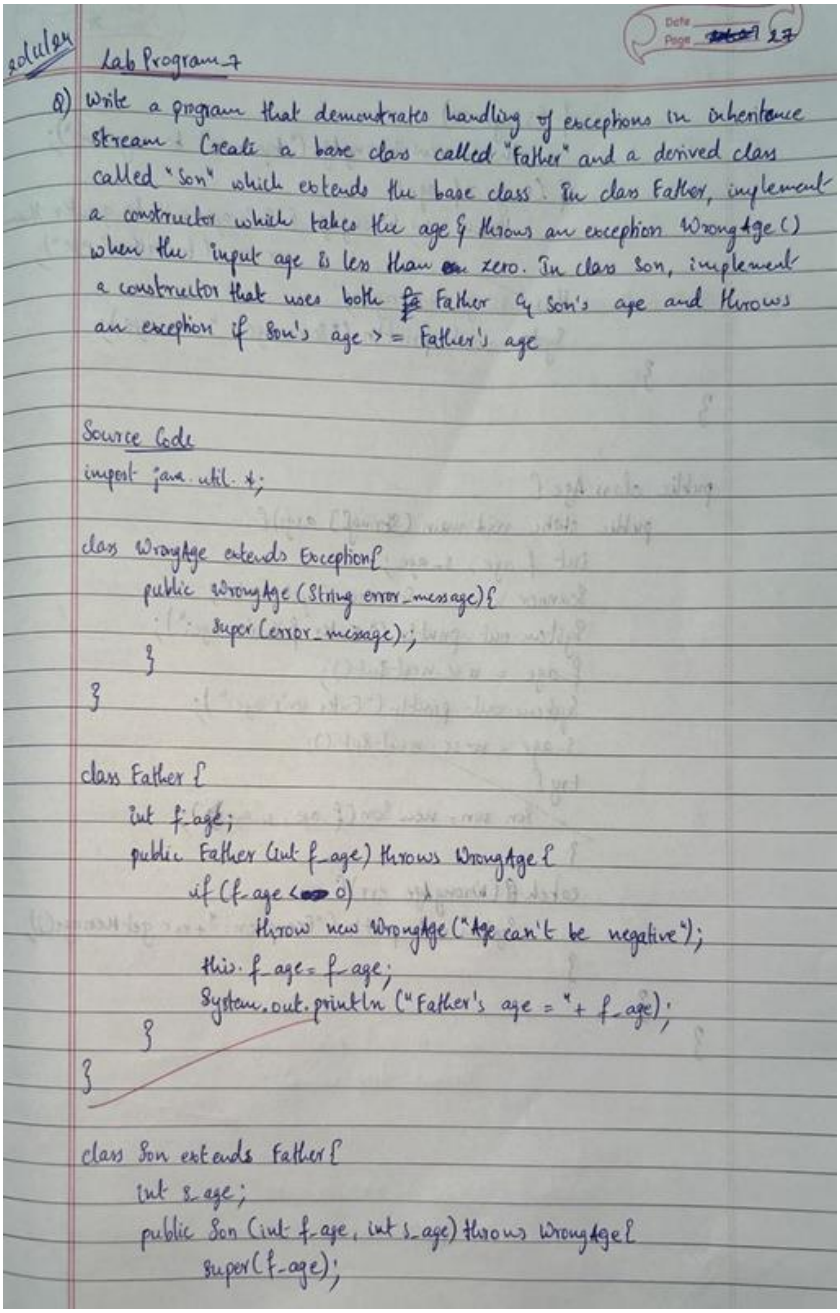
```
Final Marks of Students:

Student 1:
USN: 1BM23CS001
Name: AA
Semester: 3
Final Marks for 5 subjects:
Subject 1: 90
Subject 2: 92
Subject 3: 94
Subject 4: 96
Subject 5: 98

Student 2:
USN: 1BM23CS002
Name: AB
Semester: 3
Final Marks for 5 subjects:
Subject 1: 98
Subject 2: 96
Subject 3: 94
Subject 4: 92
Subject 5: 90
```

```
Student 3:
USN: 1BM23CS003
Name: AC
Semester: 3
Final Marks for 5 subjects:
Subject 1: 82
Subject 2: 84
Subject 3: 86
Subject 4: 88
Subject 5: 90

Student 4:
USN: 1BM23CS004
Name: AD
Semester: 3
Final Marks for 5 subjects:
Subject 1: 90
Subject 2: 88
Subject 3: 86
Subject 4: 84
Subject 5: 82

Student 5:
USN: 1BM23CS005
Name: AE
Semester: 3
Final Marks for 5 subjects:
Subject 1: 91
Subject 2: 97
Subject 3: 94
Subject 4: 84
Subject 5: 88
```

Program 7

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

Observation Book



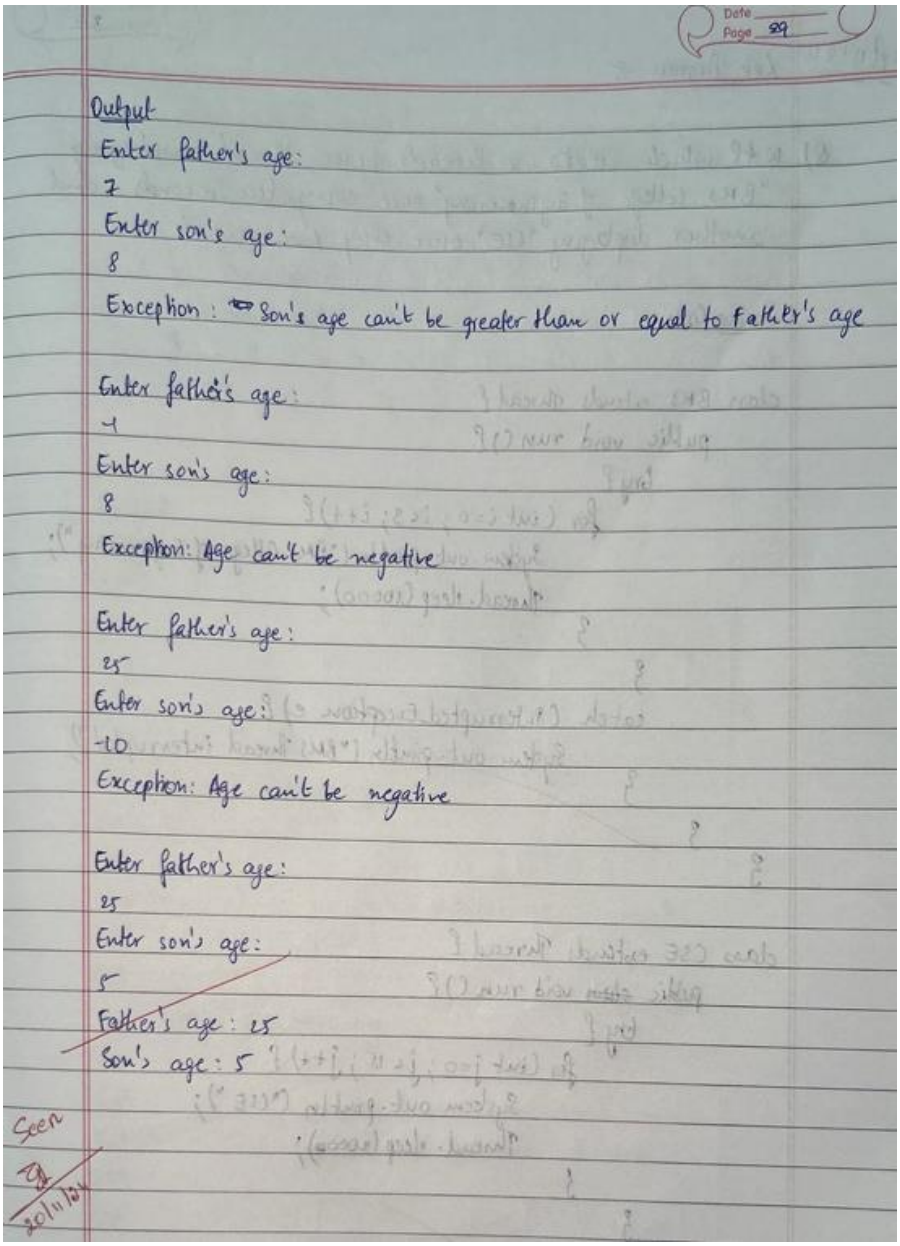
Lab Program 7

Q) Write a program that demonstrates handling of exceptions in inheritance stream. Create a base class called "Father" and a derived class called "Son" which extends the base class. In class Father, implement a constructor which takes the age & throws an exception WrongAge() when the input age is less than zero. In class Son, implement a constructor that uses both Father & Son's age and throws an exception if Son's age >= Father's age

Source Code

```
import java.util.*;

class WrongAge extends Exception{
    public WrongAge(String error_message){
        super(error_message);
    }
}

class Father {
    int f_age;
    public Father(int f_age) throws WrongAge {
        if (f_age < 0)
            throw new WrongAge("Age can't be negative");
        this.f_age = f_age;
        System.out.println("Father's age = "+ f_age);
    }
}

class Son extends Father{
    int s_age;
    public Son(int f_age, int s_age) throws WrongAge{
        super(f_age);
```

```
        if (s_age <0)
            throw new WrongAge ("Age can't be negative");
        if (s_age > f_age)
            throw new WrongAge ("Son's age can't be greater than
                                or equal to Father's age");
        this. s_age = s_age;
        System.out.println ("Son's age: " + s_age);
    }
}

public class Age {
    public static void main (String[] args){
        int f_age, s_age;
        Scanner sc = new Scanner (System.in);
        System.out.println ("Enter father's age:");
        f_age = sc.nextInt();
        System.out.println ("Enter son's age:");
        s_age = sc.nextInt ();
        try {
            Son son = new Son (f_age, s_age);
        }
        catch (WrongAge err){
            System.out.println ("Exception: " + err.getMessage());
        }
    }
}
```



Output

```
Enter father's age:
7
Enter son's age:
8
Exception : Son's age can't be greater than or equal to Father's age

Enter father's age:
-1
Enter son's age:
8
Exception: Age can't be negative

Enter father's age:
25
Enter son's age:
-10
Exception: Age can't be negative

Enter father's age:
25
Enter son's age:
5
Father's age : 25
Son's age : 5
```

Seen

Record

Source Code

```
import java.util.*;

class WrongAge extends Exception{
   public WrongAge(String error_message){
      super(error_message);
   }
```

```
}

class Father{
    int f_age;
    public Father(int f_age) throws WrongAge{
        if (f_age < 0)
            throw new WrongAge("Age can't be negative");
        this.f_age=f_age;
        System.out.println("Father's age = " + f_age);
    }
}

class Son extends Father {
    int s_age;
    public Son(int f_age, int s_age) throws WrongAge {
        super(f_age);
        if (s_age < 0) {
            throw new WrongAge("Age can't be negative");
        }
        if (s_age>=f_age) {
            throw new WrongAge("Son's age can't be greater than or equal to Father's age");
        }
        this.s_age = s_age;
        System.out.println("Son's age: " + s_age);
    }
}

public class Age {
    public static void main(String[] args) {
        System.out.println("USN: 1BM23CS003 \nName: Aaron B Ajay");
            int f_age, s_age;
            Scanner sc=new Scanner(System.in);
            System.out.println("Enter father's age: ");
            f_age=sc.nextInt();
            System.out.println("Enter son's age: ");
            s_age=sc.nextInt();

            try {
            Son son=new Son(f_age,s_age);
        }
            catch (WrongAge err) {
```

```
            System.out.println("Exception: " + err.getMessage());
        }
    }
}
```

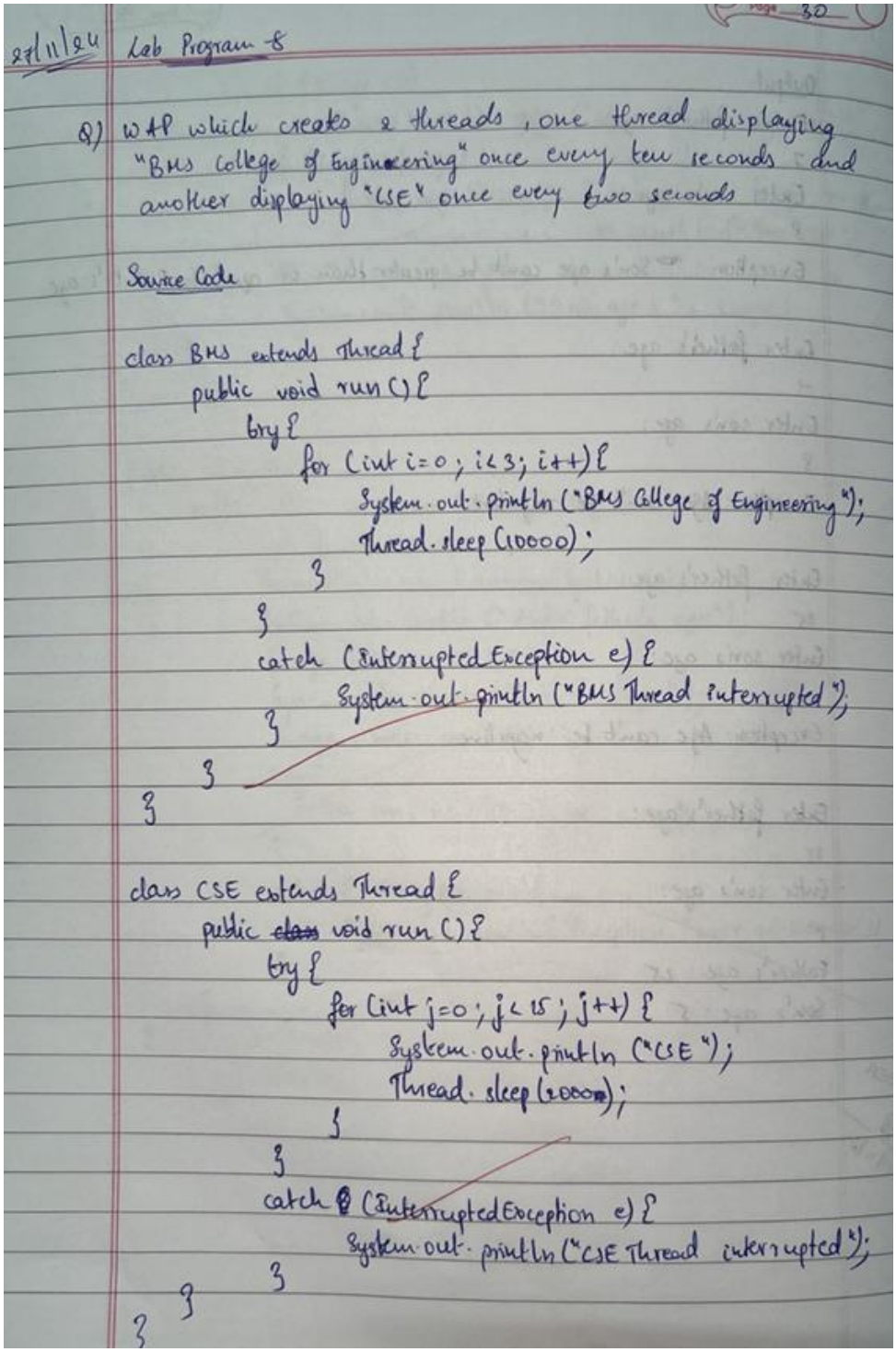
Output

```
C:\Users\Asha\Downloads>java Age
USN: 1BM23CS003
Name: Aaron B Ajay
Enter father's age:
7
Enter son's age:
8
Father's age = 7
Exception: Son's age can't be greater than or equal to Father's age

C:\Users\Asha\Downloads>java Age
USN: 1BM23CS003
Name: Aaron B Ajay
Enter father's age:
-1
Enter son's age:
8
Exception: Age can't be negative

C:\Users\Asha\Downloads>java Age
USN: 1BM23CS003
Name: Aaron B Ajay
Enter father's age:
25
Enter son's age:
-10
Father's age = 25
Exception: Age can't be negative

C:\Users\Asha\Downloads>java Age
USN: 1BM23CS003
Name: Aaron B Ajay
Enter father's age:
25
Enter son's age:
5
Father's age = 25
Son's age: 5
```

Program 8

Write a program which creates 2 threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds.

Observation Book

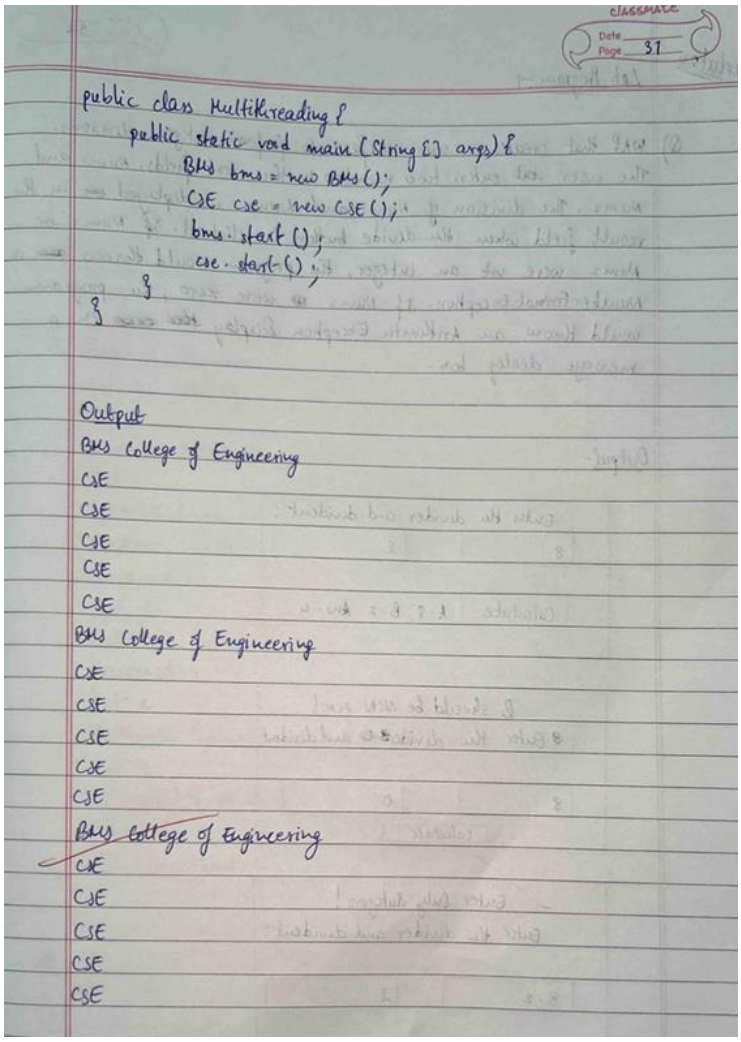


27/11/24 Lab Program - 8

Q) WAP which creates 2 threads, one thread displaying "BMS college of Engineering" once every ten seconds and another displaying "CSE" once every two seconds

Source Code

```
class BMS extends Thread {
    public void run () {
        try {
            for (int i=0; i<3; i++) {
                System.out.println ("BMS College of Engineering");
                Thread.sleep (10000);
            }
        }
        catch (InterruptedException e) {
            System.out.println ("BMS Thread interrupted");
        }
    }
}

class CSE extends Thread {
    public void run () {
        try {
            for (int j=0; j<15; j++) {
                System.out.println ("CSE");
                Thread.sleep (2000);
            }
        }
        catch (InterruptedException e) {
            System.out.println ("CSE Thread interrupted");
        }
    }
}
```



```
public class Multithreading {
    public static void main (String[] args) {
        BMS bms = new BMS();
        CSE cse = new CSE();
        bms.start();
        cse.start();
    }
}
```

Output

```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
```

Record

Source Code

```
class BMS extends Thread {
   public void run() {
      try {
         for(int i=0;i<3;i++) {
            System.out.println("BMS College of Engineering");
            Thread.sleep(10000); // Sleep for 10 seconds
         }
      }
      catch (InterruptedException e) {
         System.out.println("BMS Thread interrupted");
      }
   }
```

```
}
class CSE extends Thread {
    public void run() {
        try {
            for(int j=0;j<15;j++) {
                System.out.println("CSE");
                Thread.sleep(2000); // Sleep for 2 seconds
            }
        }
        catch (InterruptedException e) {
            System.out.println("CSE Thread interrupted");
        }
    }
}
public class Multithreading{
    public static void main(String[] args) {
        System.out.println("USN: 1BM23CS003 \nName: Aaron B Ajay");
        BMS bms = new BMS();
        CSE cse = new CSE();
        bms.start();
        cse.start();
    }
}
```

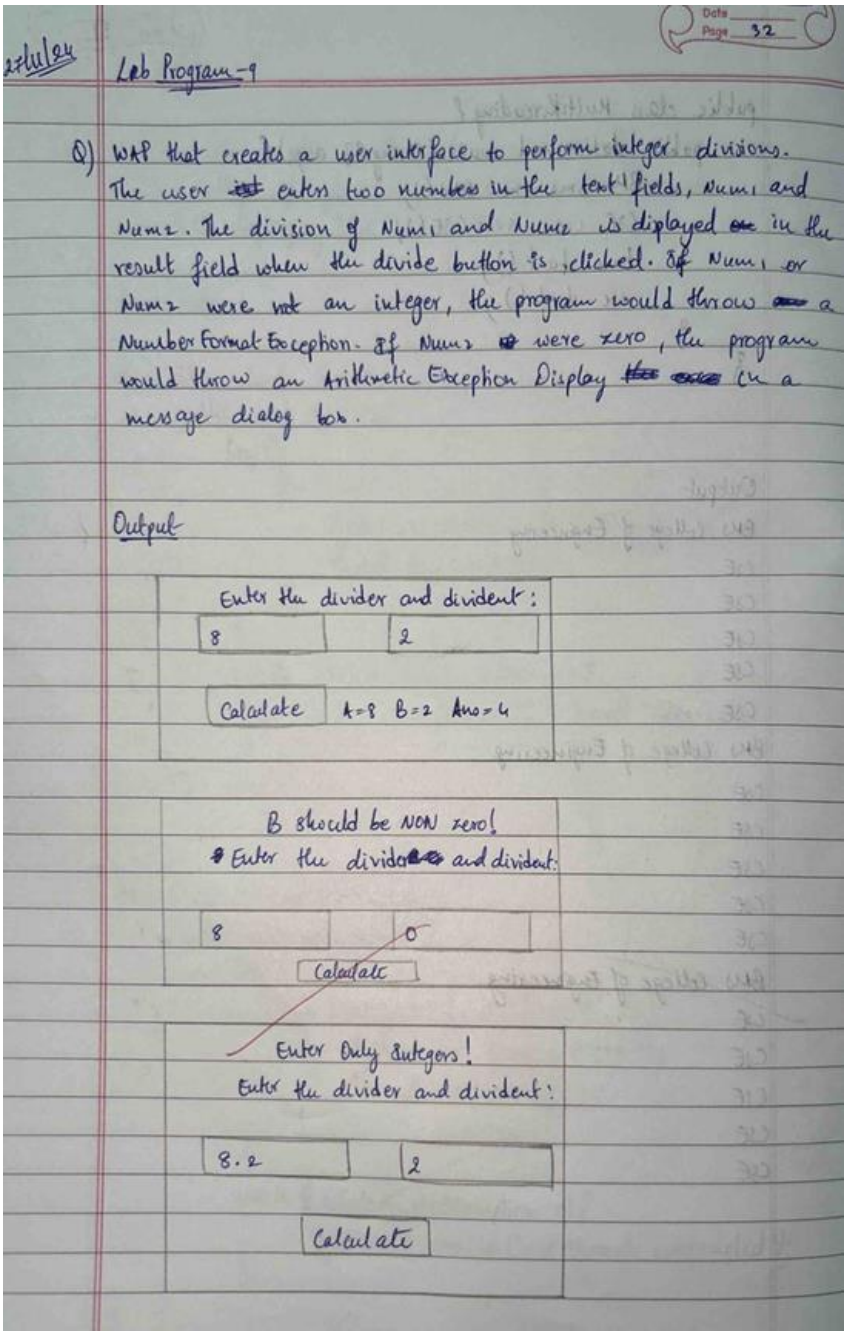
Output

```
USN: 1BM23CS003
Name: Aaron B Ajay
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
```

Program 9

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

Observation Book



Date
Page 32

27/11/24

Lab Program-9

Q) WAP 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 in a message dialog box.

Output

Enter the divider and divident:
8 2
Calculate A=8 B=2 Ans=4

B should be NON zero!
Enter the divider and divident:
8 0
Calculate

Enter Only Integers!
Enter the divider and divident:
8.2 2
Calculate

Lab Program-9

Source Code

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo {
    SwingDemo() {
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275, 150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel jlab = new JLabel("Enter the divider and divident:");
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);
        JButton button = new JButton("Calculate");
        JLabel err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();
        jfrm.add(err);
        jfrm.add(jlab);
        jfrm.add(ajtf);
        jfrm.add(bjtf);
        jfrm.add(button);
        jfrm.add(alab);
        jfrm.add(blab);
        jfrm.add(anslab);
        ActionListener l = new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Action event from a text field");
            }
        };
```

```
ajtf.addActionListener(d);
bjtf.addActionListener(d);
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("In A = "+a);
            blab.setText("In B = "+b);
            anslab.setText("In Ans = "+ans);
        }
        catch (NumberFormatException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("Enter Only Integers!");
        }
        catch (ArithmeticException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("B should be NON zero!");
        }
    }
});
frm.setVisible(true);
}
public static void main(String args[]){
    System.out.println("USN: 1BM23CS003 In Name: Aaron B Joy");
    SwingUtilities.invokeLater(new Runnable(){
        public void run(){
            new SwingDemo();
        }
    });
}
}
```

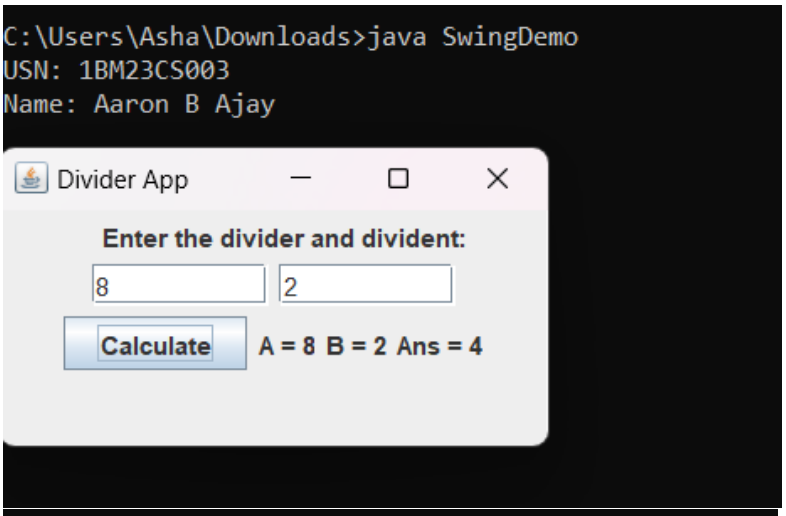
Record

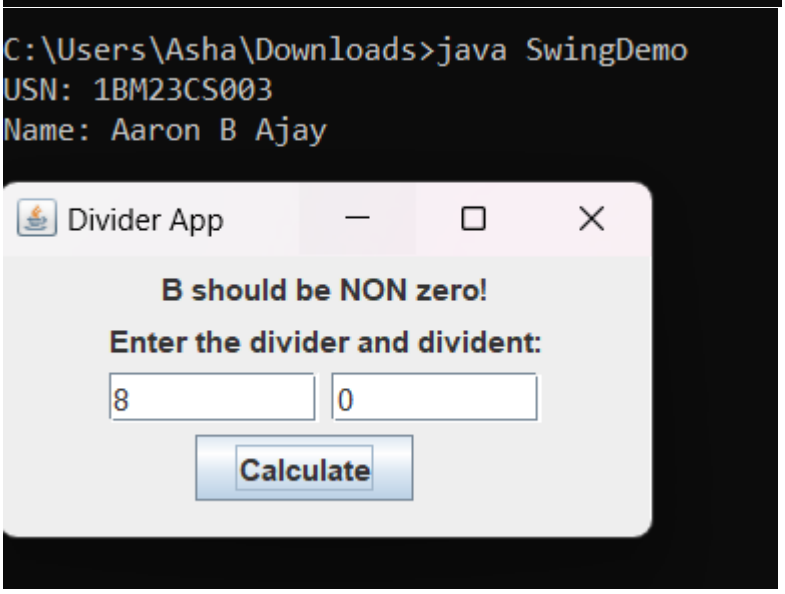
Source Code

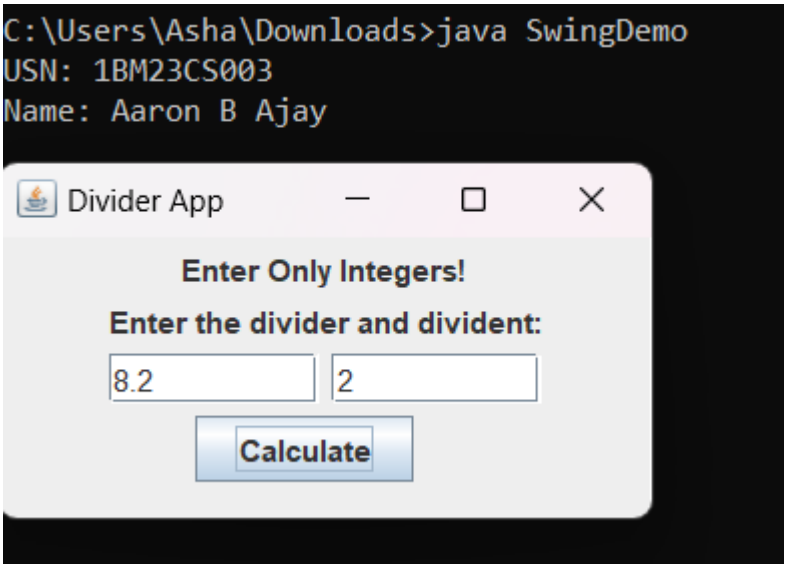
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo{
    SwingDemo(){
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275, 150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel jlab = new JLabel("Enter the divider and divident:");
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);
        JButton button = new JButton("Calculate");
        JLabel err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();
        jfrm.add(err);
        jfrm.add(jlab);
        jfrm.add(ajtf);
        jfrm.add(bjtf);
        jfrm.add(button);
        jfrm.add(alab);
        jfrm.add(blab);
        jfrm.add(anslab);
        ActionListener l = new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Action event from a text field");
            }
        };
        ajtf.addActionListener(l);
        bjtf.addActionListener(l);
        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("\nA = " + a);
                blab.setText("\nB = " + b);
                anslab.setText("\nAns = "+ 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[]){
    System.out.println("USN: 1BM23CS003 \nName: Aaron B Ajay");
    SwingUtilities.invokeLater(new Runnable(){
        public void run(){
            new SwingDemo();
        }
    });
  }
}
```

Output


C:\Users\Asha\Downloads>java SwingDemo
USN: 1BM23CS003
Name: Aaron B Ajay
Divider App
Enter the divider and divident:
8
2
Calculate
A = 8 B = 2 Ans = 4

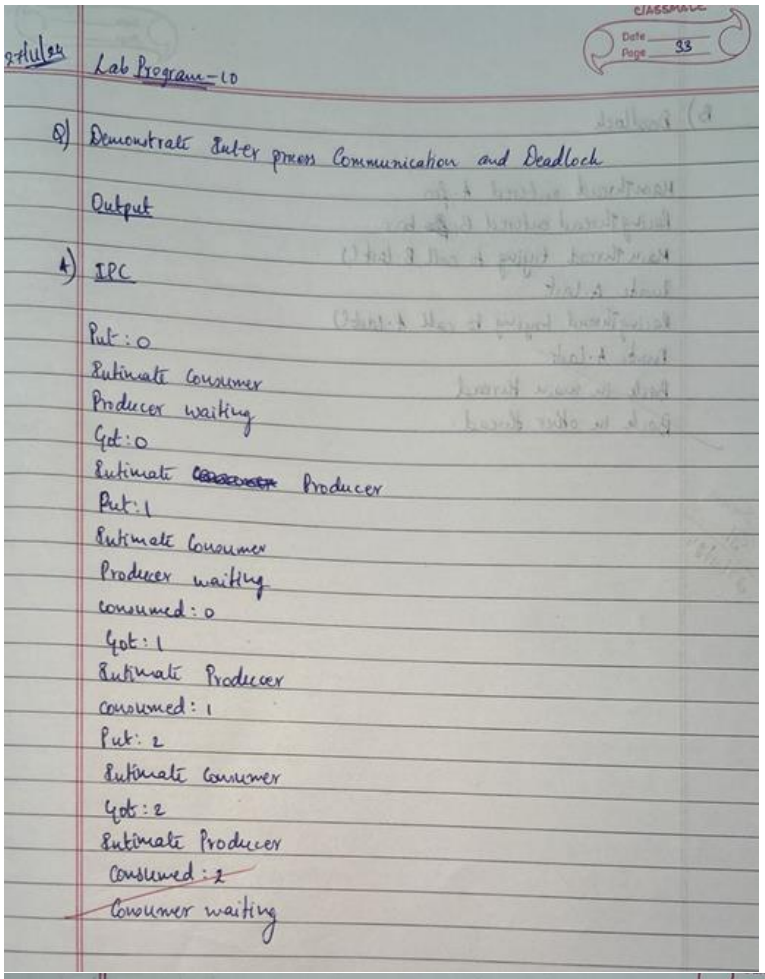

C:\Users\Asha\Downloads>java SwingDemo
USN: 1BM23CS003
Name: Aaron B Ajay
Divider App
B should be NON zero!
Enter the divider and divident:
8
0
Calculate


C:\Users\Asha\Downloads>java SwingDemo
USN: 1BM23CS003
Name: Aaron B Ajay
Divider App
Enter Only Integers!
Enter the divider and divident:
8.2
2
Calculate

Program 10

Demonstrate Inter Process Communication (IPC) and Deadlock.

Observation Book



27/11/24 Lab Program-10

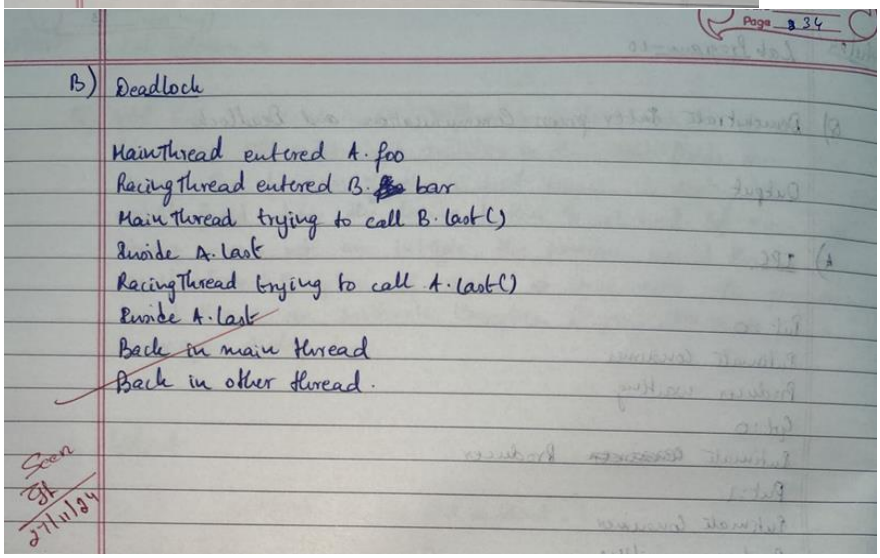
Date Page 33

Q) Demonstrate Inter process Communication and Deadlock

Output

A) IPC

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
Got: 2
Intimate Producer
~~Consumed: 2~~
~~Consumer waiting~~



Page 34

B) Deadlock

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

Seen
27/11/24

Lab Program -10

Source Code

A) IPC

```
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("Interrupted Exception caught");
        }
        System.out.println("Got: " +n);
        valueSet = false;
        System.out.println("\nInformate Producer\n");
        notify();
        return;
    }
    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("Interrupted 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 n = q.get();
```

```
        System.out.println("consumed:" + x);
        i++;
    }
  }
}

class PCFixed {
  public static void main(String args[]) {
      System.out.println("USN: 1BM23CS003 InName: Aaron B Ajay");
      Q q = new Q();
      new Producer(q);
      new Consumer(q);
      System.out.println("Press Control-C to stop.");
  }
}
```

B) 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("Main Thread");
        Thread t = new Thread(this, "RacingThread");
        t.start();
        a.foo(b);
        System.out.println("Back in main thread");
    }
```

```
    public void run() {
        b.bar(a);
        System.out.println("Back in other thread");
    }
    public static void main(String args[]) {
        System.out.println(" USN: 1BM23CS003 In Name: Aaron B Ajay");
        new Deadlock();
    }
}
```

Record

Source Code

(i) IPC

```
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("USN: 1BM23CS003 \nName: Aaron B Ajay");
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
    }
}
```

(ii) 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 main thread");
    }
    public void run() {
        b.bar(a); // get lock on b in other thread.
        System.out.println("Back in other thread");
    }
    public static void main(String args[]) {
        System.out.println("USN: 1BM23CS003 \nName: Aaron B Ajay");
        new Deadlock();
    }
}
```

Output

(i) IPC

```
USN: 1BM23CS003
Name: Aaron B Ajay
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

Put: 3

Intimate Consumer


Producer waiting

consumed:2
Got: 3

Intimate Producer

consumed:3
Put: 4

Intimate Consumer


Producer waiting

Got: 4

Intimate Producer

consumed:4
Put: 5

Intimate Consumer


Producer waiting

Got: 5

Intimate Producer

consumed:5
Put: 6
```

```
Intimate Consumer


Producer waiting

Got: 6

Intimate Producer

consumed:6
Put: 7

Intimate Consumer


Producer waiting

Got: 7

Intimate Producer

consumed:7
Put: 8

Intimate Consumer


Producer waiting

Got: 8

Intimate Producer

consumed:8
Put: 9

Intimate Consumer


Producer waiting

Got: 9

Intimate Producer

consumed:9
Put: 10
```

```
Intimate Consumer


Producer waiting

Got: 10

Intimate Producer

consumed:10
Put: 11

Intimate Consumer


Producer waiting

Got: 11

Intimate Producer

consumed:11
Put: 12

Intimate Consumer


Producer waiting

Got: 12

Intimate Producer

consumed:12
Put: 13

Intimate Consumer


Producer waiting

Got: 13

Intimate Producer

consumed:13
Put: 14
```

```
Intimate Consumer

Got: 14

Intimate Producer

consumed:14
```

(ii) Deadlock

```
C:\Users\Asha\Downloads>java Deadlock
USN: 1BM23CS003
Name: Aaron B Ajay
RacingThread entered B.bar
MainThread entered A.foo
MainThread trying to call B.last()
Inside A.last
RacingThread trying to call A.last()
Back in main thread
Inside A.last
Back in other thread
```