VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Ananya N Gowda (1BM23CS034)

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 **Ananya N Gowda (1BM23CS034),** who is a 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.

Lab faculty Incharge Sheetal V A Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link: https://github.com/ananyarex/Java-Lab/tree/main/code

Program 1

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 discriminant b2 -4ac is negative, display a message stating that there are no real solutions.

Algorithm:

9/10/24

Date __/__/__
Page

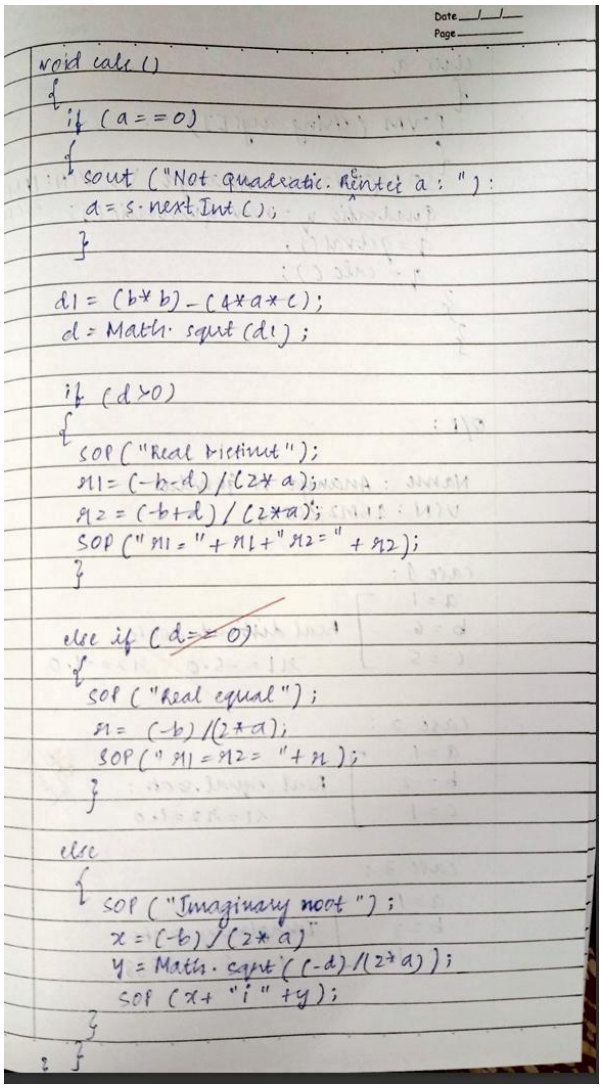
LAB - 1

Github rep - lab
→ extra

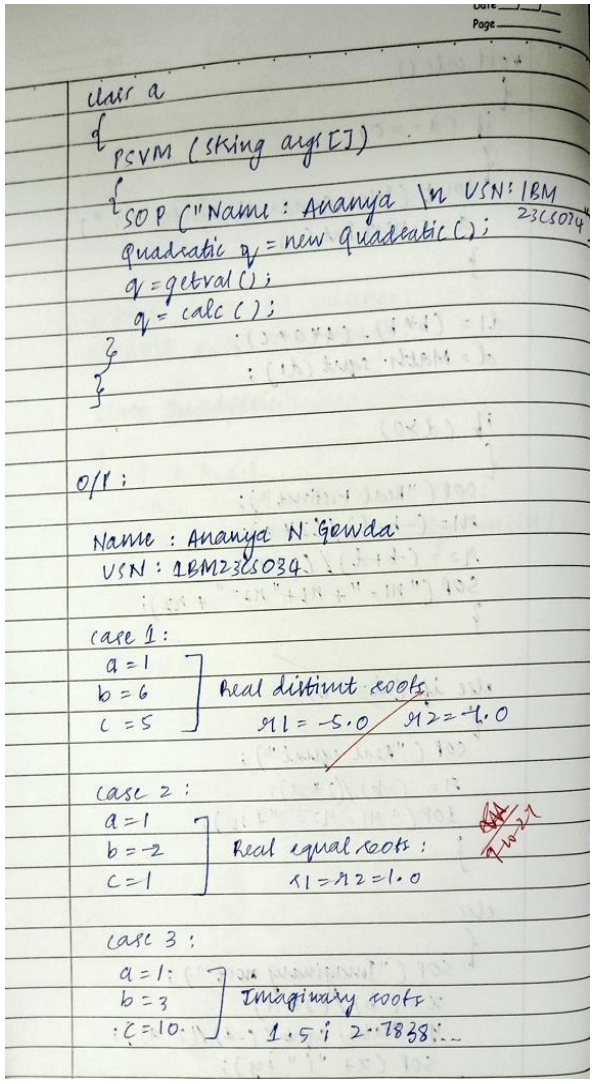
Name & VSN

source code & o/p SS

```
import java.util.scanner;
import java.lang.Math;

class quadratic
{
  int a, b, c;
  double r1, r2, d1, d, r, x, y;
  scanner s = new scanner (System.in);

void getval()
{
 ~~scanner~~
 sout ("Enter coeff a: ");
 a = s.nextInt();

 sout ("Enter coeff b: ");
 b = s. nextInt();

 sout (" Enter coeff c: ");
 c = s.nextInt();
}
```



```
void calc ()
{
  if (a==0)
  {
   sout ("Not quadratic. Enter a : ");
    a = s.nextInt();
  }

  d1 = (b*b) - (4*a*c);
  d = Math.sqrt(d1);

  if (d>0)
  {
   SOP ("Real Distinct");
   r1 = (-b-d) / (2*a);
   r2 = (-b+d) / (2*a);
   SOP ("r1 = " + r1 + " r2 = " + r2);
  }

  else if (d==0)
  {
   SOP ("Real equal");
   r = (-b) / (2*a);
   SOP (" r1 = r2 = " + r);
  }

  else
  {
   SOP ("Imaginary root");
   x = (-b) / (2*a)
   y = Math.sqrt((-d)/(2*a));
   SOP (x + "i" + y);
  }
}
```



```
class a
{
  PSVM (String args[])
  {
   SOP ("Name : Ananya \n USN: 1BM 23CS034"
   Quadratic q = new Quadratic();
   q = getval();
   q = calc();
  }
}
```

O/P :

Name : Ananya N Gowda
USN : 1BM23CS034

case 1 :
a = 1
b = 6
c = 5
Real distinct roots
r1 = -5.0 r2 = -1.0

case 2 :
a = 1
b = -2
c = 1
Real equal roots :
r1 = r2 = 1.0

case 3 :
a = 1
b = 3
c = 10
Imaginary roots
1.5 i 2.7838...

Code:

```
import java.util.Scanner;
import java.lang.Math;

class Quadratic
{
int a,b,c;
double r1,r2,d1,d,r,x,y;
Scanner s=new Scanner(System.in);

void getval()
{
Scanner s=new Scanner(System.in);
```

```
System.out.println("Enter coefficient of a: ");
a=s.nextInt();
System.out.println("Enter coefficient of b: "); b=s.nextInt();

System.out.println("Enter coefficient of c: ");
c=s.nextInt();
} void calc()
{ if(a==0)
{
System.out.println("Not quadratic, Enter a non zero non negative value of a: ");
a=s.nextInt(); }

d1=(b*b)-(4*a*c);
d=Math.sqrt(d1);
if(d>0)
{
System.out.println("Real Distinct roots: ");
r1=(-b-d)/(2*a); r2=(-b+d)/(2*a);
System.out.println("r1= "+r1+ " r2= "+r2);
} else
if(d==0)
{
System.out.println("Real equal roots: ");
r=(-b)/(2*a);
System.out.println("r1=r2= "+r);
}
else
{
System.out.println("Imaginary roots: "); x=(-
b)/(double)(2*a); y=Math.sqrt(-d1)/(double)(2*a);
System.out.println(x+"i"+y);
}
} }
class a
{
public static void main(String args[])
{
System.out.println("Name: Ananya N Gowda\nUSN: 1BM23CS034");
Quadratic q=new Quadratic(); q.getval();
q.calc();
}
}
```

Output:

```
D:\ananya>java a
Name: Ananya N Gowda
USN: 1BM23CS034
Enter coefficient of a:
1
Enter coefficient of b:
6
Enter coefficient of c:
5
Real Distinct roots:
r1= -5.0 r2= -1.0

D:\ananya>javac a.java

D:\ananya>java a
Name: Ananya N Gowda
USN: 1BM23CS034
Enter coefficient of a:
1
Enter coefficient of b:
-2
Enter coefficient of c:
1
Real equal roots:
r1=r2= 1.0

D:\ananya>javac a.java

D:\ananya>java a
Name: Ananya N Gowda
USN: 1BM23CS034
Enter coefficient of a:
1
Enter coefficient of b:
-3
Enter coefficient of c:
10
Imaginary roots:
1.5i2.7838821814150108
```

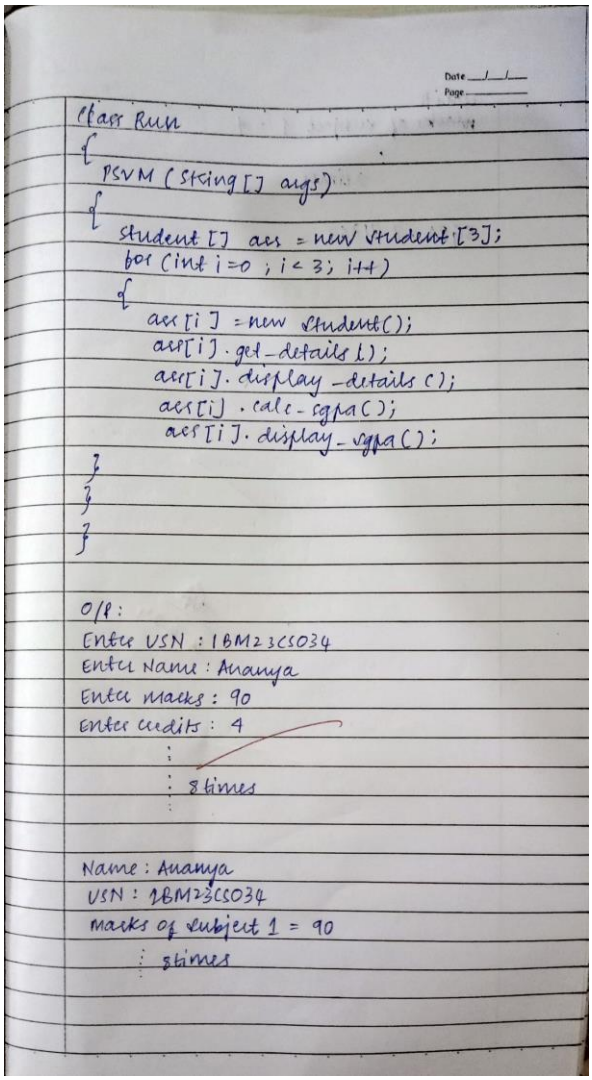
Program 2

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.

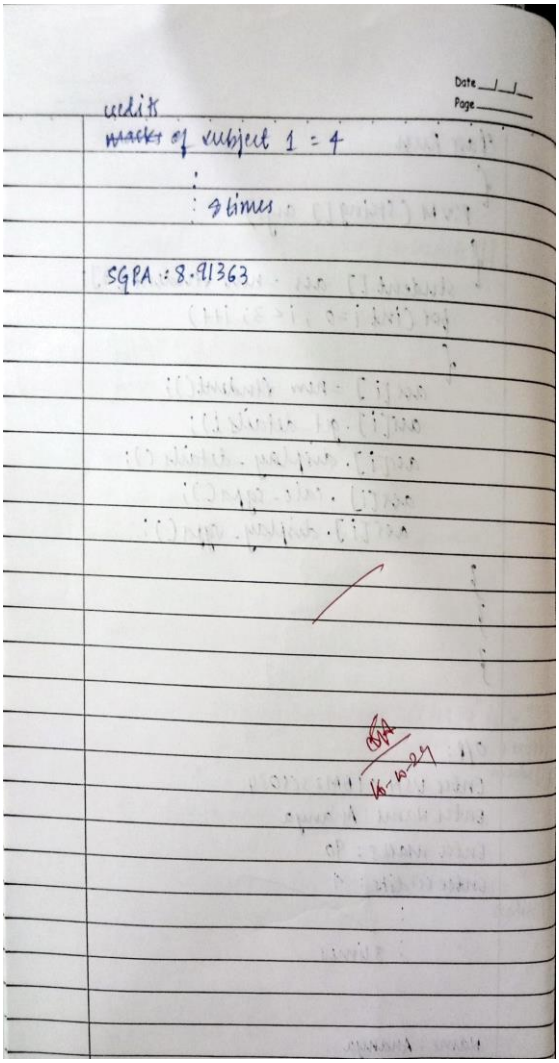
Algorithm:

Lab - 2 16/10/24

```
import java.util.Scanner

class Student
{
  string usn;
  string name;
  int [] marks; int [] credits; double sgpa;
Student(){this.usn=usn; this.name=name;}
void get_details()
{
  Scanner sc = new Scanner(System.in);
  usn = sc.nextLine();
  SOP("Enter usn:");
  SOP("Enter name:");
  name = sc.nextLine();
  credits = new int[8]
  marks = new int[8]

  for(int i=0; i<8; i++)
  {
    SOP("Enter marks and credits:")
    marks[i] = sc.nextInt();
    credits[i] = sc.nextInt();
  } sc.close();
}
void calc_sgpa()
{
  int tot_marks=0;
  int tot_credits=0;
  for(int i=0; i<8; i++)
  {
    tot_marks += marks[i];
    tot_credits += credits[i];
  }
double sgp=0;
for(int i=0; i<8; i++)
{
  sgp+=(marks[i]/10.0)*credits[i];
}

sgpa = sgp/tot_credits;
}

void display_details()
{
  SOP("USN : "+usn);
  SOP("Name : "+name);

for(int i=0; i<8; i++)
{
  SOP("Marks of subject : "+(i+1)+" is = "+marks[i])
}

for(int i=0; i<8; i++)
{
  SOP("Credits of subjects : "+(i+1)+" is = "+credits[i])
}
}

void display_sgpa()
{
  SOP("SGPA : "+sgpa);
}

}
```



```
class Run
{
  PSVM (String[] args)
  {
    Student [] acs = new Student[3];
    for (int i=0 ; i< 3; i++)
    {
      acs[i] = new Student();
      acs[i].get_details();
      acs[i].display_details();
      acs[i].calc_sgpa();
      acs[i].display_sgpa();
    }
  }
}

O/P:
Enter USN : 1BM23CS034
Enter Name : Ananya
Enter Marks : 90
Enter credits : 4
      :
      : 8 times

Name : Ananya
USN : 1BM23CS034
Marks of subject 1 = 90
      : 8 times
```



```
credits
~~marks~~ of subject 1 = 4
      :
      : 8 times

SGPA : 8.91363
```

Code:

```
import java.util.Scanner;

class Student
{
String usn; String
name; int[] marks=new
int[8]; int[] credits=new
int[8]; double sgpa;


Student() {
this.usn=usn;
```

```
this.name=n
ame; }

void get_details()
{
Scanner sc=new Scanner(System.in);
System.out.print("Enter USN: ");
usn=sc.nextLine();
System.out.print("Enter Name: ");
name=sc.nextLine();

for(int i=0;i<8;i++)
{
System.out.print("Enter marks: ");
marks[i]=sc.nextInt();
System.out.print("Enter credits: ");
credits[i]=sc.nextInt();
}
sc.close();
}

void calc_sgpa()
{ int
tot_marks=0; int
tot_credits=0;

for(int i=0; i<8;i++) {
tot_marks+=marks[i];
tot_credits+=credits[i];
} double sgp=0; for(int
i=0;i<8;i++) {
sgp+=(marks[i]/10.0)*credits[i];

} sgpa=sgp/tot_credits; }



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

for(int i=0;i<8;i++)
{
```

```
System.out.println("marks of subject: "+(i+1)+" is = "+marks[i]);
}

for(int i=0;i<8;i++)
{
System.out.println("credits of subject: "+(i+1)+" is = "+credits[i]);
}
}

void display_sgpa()
{
System.out.println("SGPA : "+sgpa);

}



} class
Run {
public static void main(String[] args)
{
System.out.println("Name: Ananya N Gowda ");

System.out.println("USN: 1BM23CS034 ");


Student[] arr=new Student[3];
for(int i=0;i<3;i++)
{ arr[i]=new
 Student();
arr[i].get_details();

arr[i].display_details();
arr[i].calc_sgpa();
arr[i].display_sgpa();
}


}
}
```

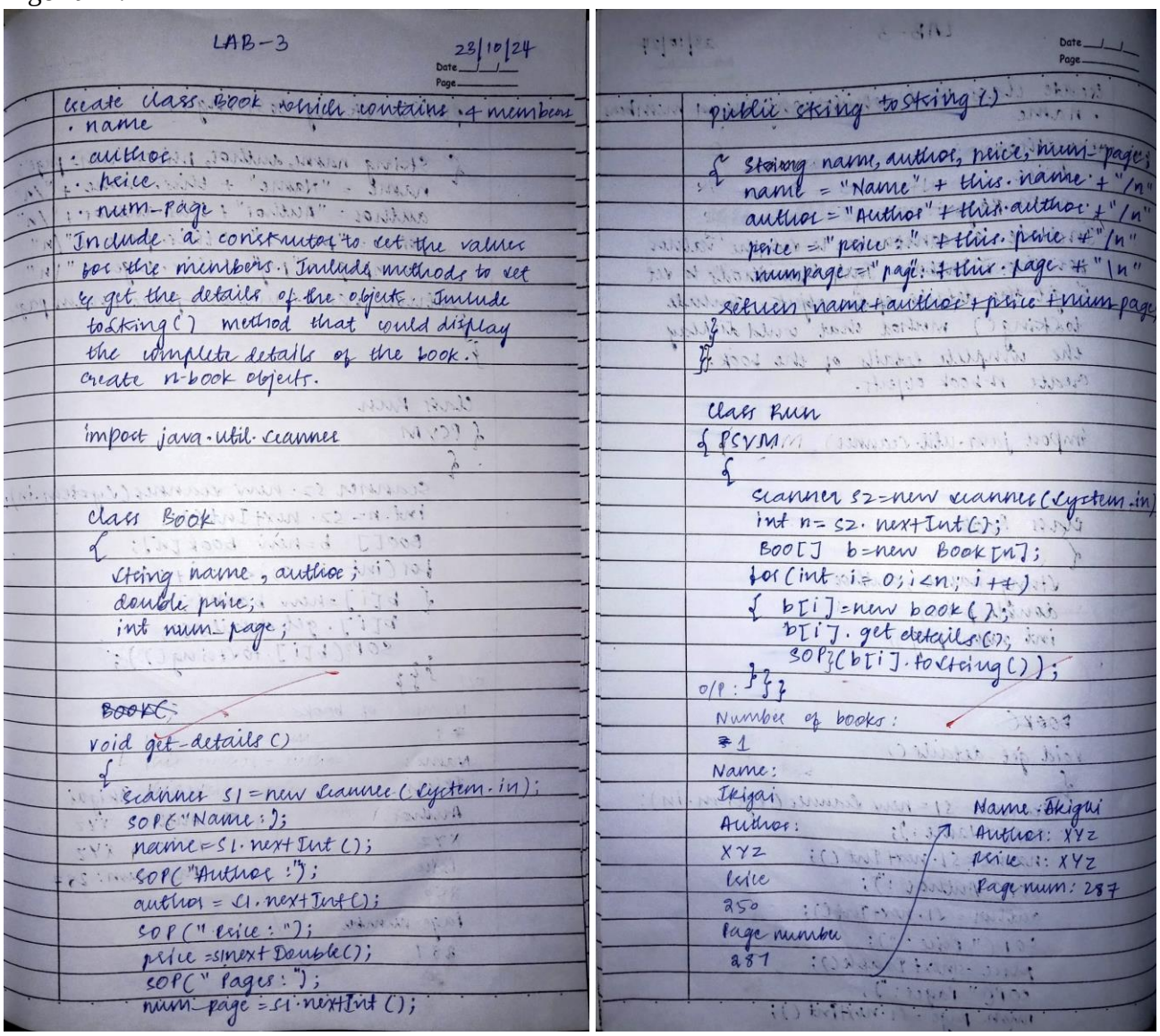
Output:

```
D:\ananya>java Run
Enter USN: 1BM23CS034
Enter Name: Ananya
Enter marks: 90
Enter credits: 4
Enter marks: 90
Enter credits: 4
Enter marks: 80
Enter credits: 4
Enter marks: 85
Enter credits: 3
Enter marks: 87
Enter credits: 3
Enter marks: 79
Enter credits: 2
Enter marks: 88
Enter credits: 1
Enter marks: 88
Enter credits: 1
USN: 1BM23CS034
Name: Ananya
marks of subject: 1 is = 90
marks of subject: 2 is = 90
marks of subject: 3 is = 80
marks of subject: 4 is = 85
marks of subject: 5 is = 87
marks of subject: 6 is = 79
marks of subject: 7 is = 88
marks of subject: 8 is = 88
credits of subject: 1 is = 4
credits of subject: 2 is = 4
credits of subject: 3 is = 4
credits of subject: 4 is = 3
credits of subject: 5 is = 3
credits of subject: 6 is = 2
credits of subject: 7 is = 1
credits of subject: 8 is = 1
SGPA : 8.590909090909092
```

Program 3

Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to create n book objects.

Algorithm:



LAB-3 23/10/24

Create class Book which contains 4 members
- name
- author
- price
- num_page

"Include a constructor to set the values for the members. Include methods to set & get the details of the objects. Include toString() method that could display the complete details of the book. Create n-book objects.

```
import java.util.Scanner

class Book
{
  String name, author;
  double price;
  int num_page;

  ~~Book();~~
  void get_details()
  {
    Scanner s1=new Scanner(System.in);
    SOP("Name:");
    name=s1.nextInt();
    SOP("Author:");
    author = s1.nextInt();
    SOP("price:");
    price =s1.nextDouble();
    SOP("Pages:");
    num_page =s1.nextInt();
```

```
public String toString()
{ String name, author, price, num_page;
  name = "Name" + this.name + "/n"
  author = "Author" + this.author + "/n"
  price = "price:" + this.price + "/n"
  numpage = "page:" + this.page + "\n"
  return name+author+price+num_page;
}
}

class Run
{ PSVM
  {
    Scanner s2=new Scanner(System.in)
    int n= s2.nextInt();
    Boo[] b=new Book[n];
    for(int i=0;i<n; i++)
    { b[i]=new book();
      b[i].get details();
      SOP(b[i].toString());
}}}
```

O/P:

```
Number of books:
1
Name:
Ikigai
Author:
XYZ
Price
250
Page number
287
```

Name: Ikigai
Author: XYZ
Price: XYZ
Page num: 287

Code**:**

```
import java.util.Scanner;

class Book
{
```

```
String name,author;
double price; int
num_page,book_count;

void get_details()
{
Scanner s1=new Scanner(System.in);
System.out.println("Name: ");
name=s1.nextLine();
System.out.println("Author: ");
author=s1.nextLine();
System.out.println("Price: ");
price=s1.nextDouble();
System.out.println("Page number: ");
num_page=s1.nextInt();
}
public String toString()
{
String name,author,price,num_page; name="Name:
"+this.name+"\n"; author = "Author: " + this.author +
"\n"; price = "Price: " + this.price + "\n"; num_page =
"Number of pages: " + this.num_page + "\n"; return name
+ author + price + num_page;
} } class
dontrun {
public static void main(String[] args)
{
System.out.println("Name: Ananya N Gowda");
System.out.println("USN: 1BM23CS034");
Scanner s2=new Scanner(System.in); int n;
System.out.println("Number of books: ");
n=s2.nextInt();
Book[] b=new Book[n];
for(int i=0;i<n;i++) {
b[i]=new Book();
b[i].get_details();
}
for(int i=0;i<n;i++)
{
System.out.println(b[i].toString());
}
}
}
```

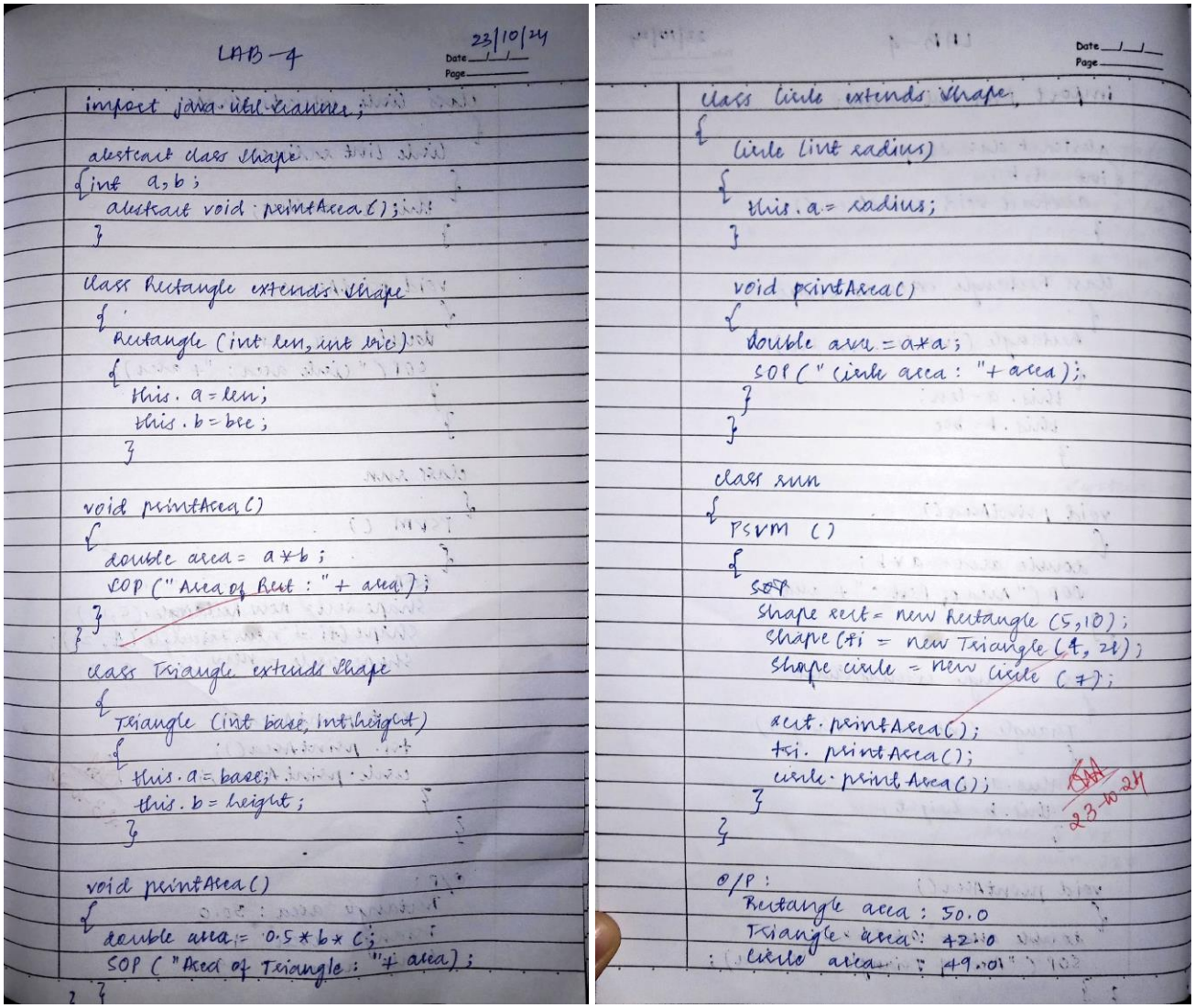
Output:

```
D:\ananya>javac dontrun.java

D:\ananya>java dontrun
Name: Ananya N Gowda
USN: 1BM23CS034
Number of books:
2
Name:
Ikigai
Author:
Shuri
Price:
250
Page number:
300
Name:
Aretmis
Author:
Zaeke
Price:
345
Page number:
287
Name: Ikigai
Author: Shuri
Price: 250.0
Number of pages: 300

Name: Aretmis
Author: Zaeke
Price: 345.0
Number of pages: 287
```

Program 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 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.

Algorithm:



LAB-4 23/10/24

```
import java.util.Scanner;

abstract class Shape
{int a,b;
  abstract void printArea();
}

class Rectangle extends Shape
{
  Rectangle (int len, int bre)
  {
    this.a = len;
    this.b = bre;
  }

void printArea()
{
  double area = a*b;
  SOP ("Area of Rect : " + area);
}
}

class Triangle extends Shape
{
  Triangle (int base, int height)
  {
    this.a = base;
    this.b = height;
  }

void printArea()
{
  double area = 0.5*b*c;
  SOP ("Area of Triangle : " + area);
}
}

class Circle extends Shape
{
  Circle (int radius)
  {
    this.a = radius;
  }

  void printArea()
  {
    double area = a*a;
    SOP (" Circle area : " + area);
  }
}

class run
{
  PSVM ()
  {
    SOP
    Shape rect = new Rectangle (5,10);
    Shape tri = new Triangle (4, 21);
    Shape circle = new Circle (7);

    rect.printArea();
    tri.printArea();
    circle.printArea();
  }
}

O/P :
  Rectangle area : 50.0
  Triangle area : 42.0
  Circle area : 49.01
```

23-10-24

Code:

```
import java.util.Scanner;

abstract class Shape
{
int a,b; abstract void
printArea(); }
```

```
class Rectangle extends Shape
{
Rectangle(int len,int breadth)
{
this.a=len;
this.b=breadth;
}


void printArea()
{ double
area=a*b;
System.out.println("Rectangle area: "+area);
}
}

class Triangle extends Shape
{
Triangle(int base,int height)
{
this.a=base;
this.b=height;
}


void printArea() {
double area=0.5*a*b;
System.out.println("Triangle area: "+area);
}

}

class Circle extends Shape
{
Circle(int radius)
{
this.a=radius;

}


void printArea()
{ double
area=a*a;
```

```
System.out.println("Circle area: "+area);
}

}


class run { public static void
main(String[] args)
{
System.out.println("Name: Ananya N Gowda");
System.out.println("USN: 1BM23CS034");

Shape rect=new Rectangle(5,10);
rect.printArea();

Shape tri=new Triangle(4,21);
tri.printArea();

Shape circle=new Circle(7);
circle.printArea();
}
}
```

Output:

```
D:\ananya>javac run.java

D:\ananya>java run
Name: Ananya N Gowda
USN: 1BM23CS034
Rectangle area: 50.0
Triangle area: 42.0
Circle area: 49.0
```

Program 5

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.

Algorithm:

LAB - 5

Date / / Page 30/10/24

```
import java.util.Scanner;

class Account
{
  protected string cust name;
  protected int acno;
  protected double balance;

  public account (string custname, int acno,
                  double balance)
  {
    this.custname = custname;
    this.acno = acno;
    this.balance = balance;
  }

  public void deposit (double amount)
  {
    if (amount > 0)
      {
        balance += amount;
        SOP("deposited" + amount);
      }
    else
      {
        SOP ("invalid deposit amount");
      }
  }
public void display_balance ()
  {
    SOP ("balance: " + balance);
  }
}
```

Date / / Page / /

```
class savac extends account
{
  private double interestrate;
  public savac (string custname, int ac
    double balance, double interestrate)
  {
    super(custname, acno, balance)
    this.interestrate = interestrate;
  }

public void computeand deposit interest()
  {
    double interest = balance * (interestrate/
    balance+=interest;
    SOP("interest added : "+ interest);
  }

public void withdraw (double amount)
  {
    if (amount <= balance)
      {
        balance -= amount;
        SOP("withdrawn : "+ amount);
      }
    else
      {SOP ("insufficient balance ");
      }
  }
}
```

```
class currAcc extends account
{
    private double minimumBalance;
    private double serviceCharge;

    public currAcc (string customerName, int accNum, double minimumBalance, double
    serviceCharge)
    {
        super (customerName, accNo, balance);
        this.minBalance = minBalance;
        this.service charge = serviceCharge;
    }

public void withdraw (double amount)
    {
        if (amount <= balance)
        {
            balance -= amount;
            SOP (withdrawn: " amount);
        }
        if (balance < minBalance)
        {
            balance -= service charge;
            SOP ("service charge imposed "+ service charge);
        }
    }
else
    { SOP ("insufficient balance for withdrawal)
    }
}
}
```

```
public class Bank
{
    psvm (String [] args)
    {
        Scanner sc = new Scanner (System.in);
        SavAcc savAcc = new SavAcc
                ("Ananya", 12345,
                1000,5);

        currAcc currAcc = new CurrAcc
                ('Jam', 67890, 2000,
                500, 50);

        SOP ("choose acc type : \n 1:
            Savings Account \n
            2: current Account ");

        int choice = sc.nextInt();

    switch (choice)
    {
        case 1 :
            SOP ("Savings Account selected");
            savAcc.deposit(500);
            savAcc.compdepInterest();
            savAcc.withdraw(300);
            savAcc.displayBalance();
            break;

        case 2 :
            SOP ("current Account select");
            currAcc.deposit(500);
            currAcc.withdraw(1800);
            currAcc.displayBalance();
            break;
```

```
        default :
            SOP ("Invalid choice")
        }
        sc.close()
    }
}
```

O/P :

```
1> Savings Account
2. Current Account
1
Savings account selected
deposited : 500.0
interest added : 75.0
withdrawn : 300.0
Balance : 1275.0

choose Account type
1. Savings Account
2. Current Account
2
current Account selected
Deposited : 500.0
withdrawn : 1800.0
Balance : 7000.
```

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) {
        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:

```
PS C:\Users\Dell\Desktop\Ananya\bms sem3\OOPS in Java>  & 'C:\
lsInExceptionMessages' '-cp' 'C:\Users\Dell\AppData\Roaming\Co
OPS in Java_9d75c0f4\bin' 'Bank'
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
PS C:\Users\Dell\Desktop\Ananya\bms sem3\OOPS in Java>  & 'C:\
lsInExceptionMessages' '-cp' 'C:\Users\Dell\AppData\Roaming\Co
OPS in Java_9d75c0f4\bin' 'Bank'
Choose Account Type:
1. Savings Account
2. Current Account

2
Current Account Selected
Deposited: 800.0
Withdrawn: 200.0
Balance: 10600.0
PS C:\Users\Dell\Desktop\Ananya\bms sem3\OOPS in Java>
```

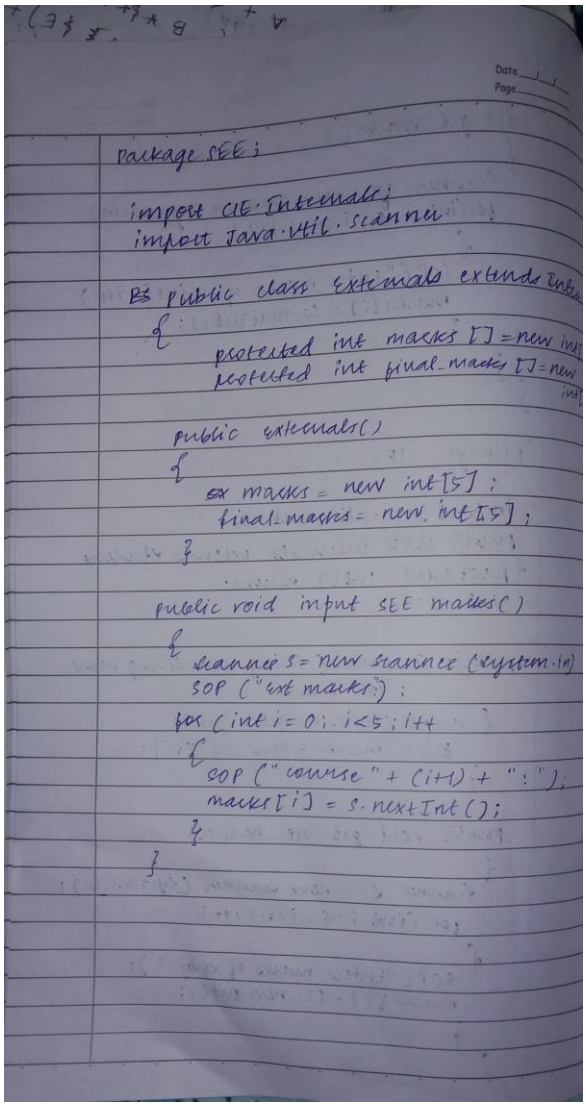
Program 6

Create a package CIE which has two classes - Personal and Internals. The class Personal has members like usn, name, sem. The class Internals 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 Personal. 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.

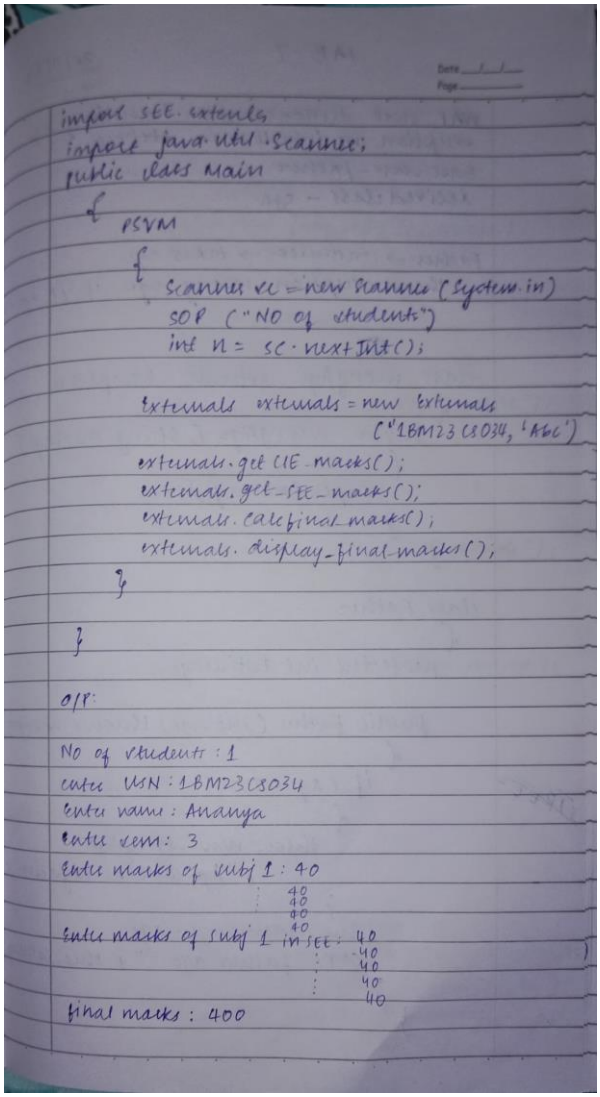
Algorithm:

```
LAB
LAB-6
13/11/24

package CIE;

import java.util.scanner;

class student
{
    string usn, name;
    int sem;

    student()
    {
     this.usn = usn;
     this.name = name;
     this.sem = sem;
    }

  void get_details()
  {
    Scanner S1 = new Scanner(System.in);
    SOP ("Enter USN : ");
    usn = S1.nextLine();
    SOP ("Enter name ");
    name = S1.nextLine();
    SOP ("Enter sem : ");
    sem = S1.nextInt();
  }
}
class Internals extends student
{
    int[] marks; int n = 5;
    ~~super()~~
    Internals (string usn, string name)
    { super(usn, name);
      this.marks = new marks }
```

```
void get_marks()
{
  Scanner S2 = new Scanner(System.in)
  for(int i = 0; i<n; i++)
  {
    SOP ("Enter marks of subject" + i+1)
    marks[i] = S2.nextInt();
  }
}
}

package CIE;

import java.util.scanner;
public class Internals extends student
{protected int[] marks;
 int n=5;

public Internals( string usn, string name,
                  int sem)
 { super(usn, name, sem);
   this.marks = new int [n];
 }

public void get_cie_marks()
 {
  Scanner S2 = new Scanner(System.in);
   for (int i=0; i<n; i++)
  {
   SOP ("Enter marks of subj ");
   marks[i] = S2.nextInt();
  }
 }
}
```



```
package SEE;

import CIE.Internals;
import Java.util.scanner

public class externals extends Inter
{
    protected int marks[] = new int
    protected int final_marks[] = new
                                  int

    public externals()
    {
        marks = new int[5];
        final_marks = new int[5];
    }

    public void input SEE marks()
    {
        scanner s = new scanner (system.in)
        SOP ("ext marks") :
        for (int i = 0; i<5; i++
        {
            SOP ("course " + (i+1) + ":");
            marks[i] = s.nextInt();
        }
    }
```



```
import SEE.externals
import java.util.Scanner;
public class Main
{
    PSVM
    {
        Scanner sc = new Scanner (System.in)
        SOP ("NO of students")
        int n = sc.nextInt();

        Externals externals = new Externals
                                ("1BM23CS034, 'ABC')
        externals.get CIE_marks();
        externals.get_SEE_marks();
        externals.calcfinal_marks();
        externals.display_final_marks();
    }

}

o/p:

No of students : 1
enter USN : 1BM23CS034
enter name : Ananya
enter sem : 3
Enter marks of subj 1 : 40
                        40
                        40
                        40
                        40
Enter marks of subj 1 in SEE : 40
                               40
                               40
                               40
                               40
final marks : 400
```

Code:

```
package CIE; import

java.util.Scanner;

public class Internals extends Student {
    protected int[] marks;
    int n = 5; // assuming there are 5 subjects for now

    // Constructor to initialize the inherited fields and the marks array
    public Internals(String usn, String name) {
        super(); // Calling the parent constructor (Student)
        this.usn = usn; this.name = name;
```

```
        this.marks = new int[n]; // Initialize the marks array with size 5
    }

    // Method to input marks public void
    get_CIE_marks() { Scanner s2 = new
    Scanner(System.in); for (int i = 0; i < n;
    i++) {
            System.out.println("Enter marks of subject " + (i + 1) + ": ");
            marks[i] = s2.nextInt();
        }
    }

    // Method to display marks
    public void display_marks() {
        System.out.println("Marks for the student " + name + ":");
        for (int i = 0; i < n; i++) {
            System.out.println("Subject " + (i + 1) + ": " + marks[i]);
        }
    }

    // Main method for testing public
    static void main(String[] args) {
        Internals intern = new Internals("1BS19CS001", "John Doe");
        intern.get_CIE_marks();
        intern.display_marks();
    }
} package CIE; import

java.util.Scanner;

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

    // Constructor
    public Student() {
    this.usn = "";
    this.name = "";
    this.sem = 0;
    }

    // Method to input details
    public void get_details() {
        Scanner s1 = new Scanner(System.in);
```

```
        System.out.println("Enter USN: ");
        usn = s1.nextLine();
        System.out.println("Enter Name: ");
        name = s1.nextLine();
        System.out.println("Enter Semester: ");

        // Handle input for integer value for semester
        while (!s1.hasNextInt()) {
            System.out.println("Please enter a valid integer for Semester: ");
            s1.next(); // Consume the invalid input
        }
        sem = s1.nextInt();
    }

    // Method to display details
    public void display_details() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }

    // Main method for testing public
    static void main(String[] args) {
    Student student = new Student();
    student.get_details();
        student.display_details();
    }
} package

SEE;

import CIE.Internals;
import java.util.Scanner;

public class Externals extends Internals {
    protected int[] marks;
    protected int[] final_marks; int
    x = 5; // Assuming 5 subjects

    // Constructor to initialize marks and final_marks arrays
    public Externals(String usn, String name) {
        super(usn, name); // Call the constructor of Internals (and Student)
        this.marks = new int[x]; // Initialize marks array this.final_marks =
        new int[x]; // Initialize final_marks array
```

```
    }

    // Method to input external exam marks
    public void get_SEE_marks() { Scanner s3 =
    new Scanner(System.in); for (int i = 0; i < x;
    i++) {
            System.out.println("Enter marks of external exam for subject " + (i + 1) + ": ");
            marks[i] = s3.nextInt();
        }
    }

    // Method to calculate final marks
    public void calc_final_marks() {
        for (int i = 0; i < x; i++) {
            // Assuming final_marks is the sum of internal and external marks
            final_marks[i] = marks[i] + super.marks[i];
        }
    }

    // Method to display final marks
    public void display_final_marks() {
    // Display the basic student details
    display_details();

        // Display the final marks for each subject
        System.out.println("Final marks for each subject: ");
        for (int i = 0; i < x; i++) {
            System.out.println("Subject " + (i + 1) + ": " + final_marks[i]);
        }
    }

    // Main method for testing
    public static void main(String[] args) {
        Externals externals = new Externals("1BS19CS001", "John Doe");
    externals.get_CIE_marks(); // Get internal marks
    externals.get_SEE_marks(); // Get external marks
    externals.calc_final_marks(); // Calculate final marks
    externals.display_final_marks(); // Display final marks }
}

import SEE.Externals;
import java.util.Scanner;
```

```
class Run { public static void
   main(String args[]) { int num;
      Scanner s4 = new Scanner(System.in);

      System.out.println("Total Students: ");
      num = s4.nextInt();
      // Array of Externals objects (since Externals has all the required methods)
      Externals[] students = new Externals[num];

      // Loop through each student
      for (int i = 0; i < num; i++) {
         // Prompt for student details (USN and Name)
         Scanner input = new Scanner(System.in); // To handle input for each student
         System.out.println("Enter USN for student " + (i + 1) + ": ");
         String usn = input.nextLine();
         System.out.println("Enter Name for student " + (i + 1) + ": ");
         String name = input.nextLine();

         // Create an Externals object for each student
         students[i] = new Externals(usn, name);

         // Get details and marks for each student
      students[i].get_details(); // Gets general details
      students[i].display_details(); // Displays general details
      students[i].get_CIE_marks(); // Get internal marks (CIE)
      students[i].get_SEE_marks(); // Get external marks (SEE)
      students[i].calc_final_marks(); // Calculate final marks
      students[i].display_final_marks(); // Display final marks }
   }
}
```

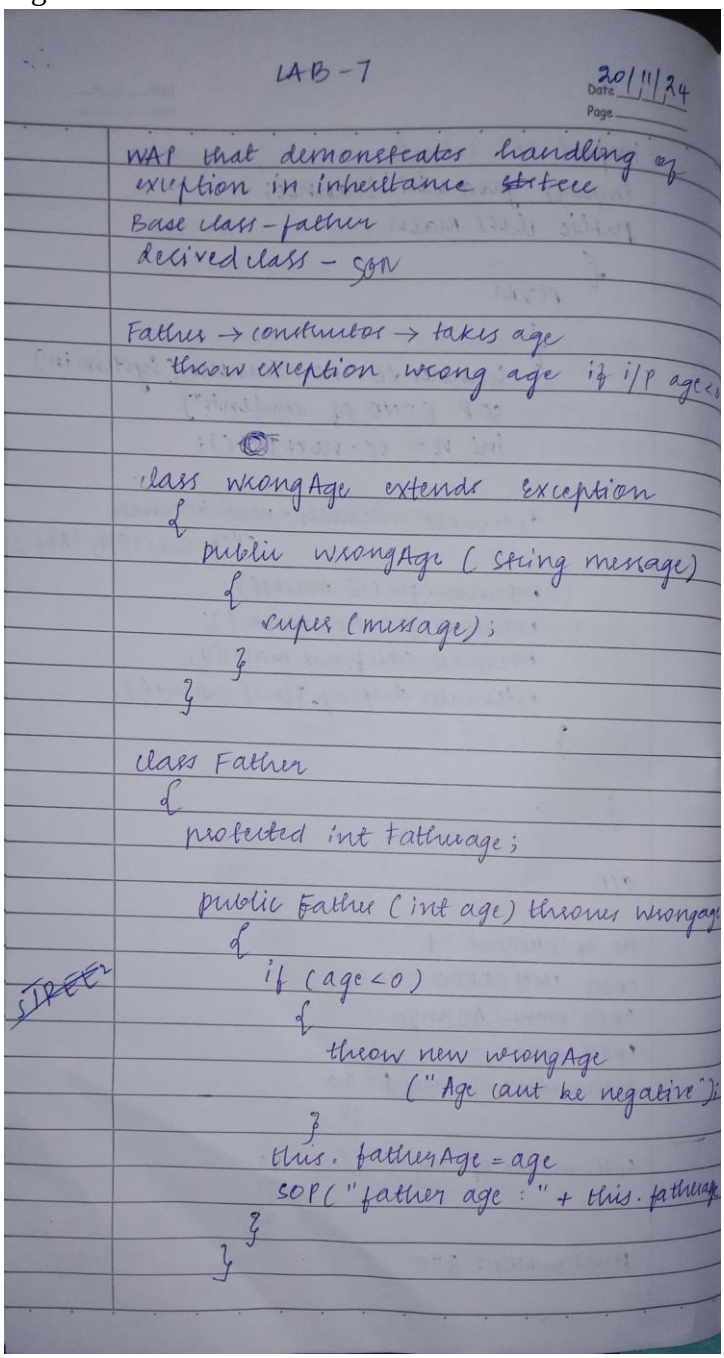
Output:

```
D:\ananya>java Run
Total Students:
2
Enter USN for student 1:
001
Enter Name for student 1:
ABC
Enter USN:
002
Enter Name:
XYZ
Enter Semester:
3
USN: 002
Name: XYZ
Semester: 3
Enter marks of subject 1:
50
Enter marks of subject 2:
50
Enter marks of subject 3:
50
Enter marks of subject 4:
50
Enter marks of subject 5:
50
Enter marks of external exam for subject 1:
50
Enter marks of external exam for subject 2:
50
Enter marks of external exam for subject 3:
50
Enter marks of external exam for subject 4:
50
Enter marks of external exam for subject 5:
50
USN: 002
Name: XYZ
Semester: 3
Final marks for each subject:
Subject 1: 100
Subject 2: 100
Subject 3: 100
Subject 4: 100
Subject 5: 100
Enter USN for student 2:
```

Program 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 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 uses both father and son’s age and throws an exception if son’s age is >=father’s age.

Algorithm:



LAB-7

Date 20/11/24

Page

WAP that demonstrates handling of exception in inheritance ~~stree~~

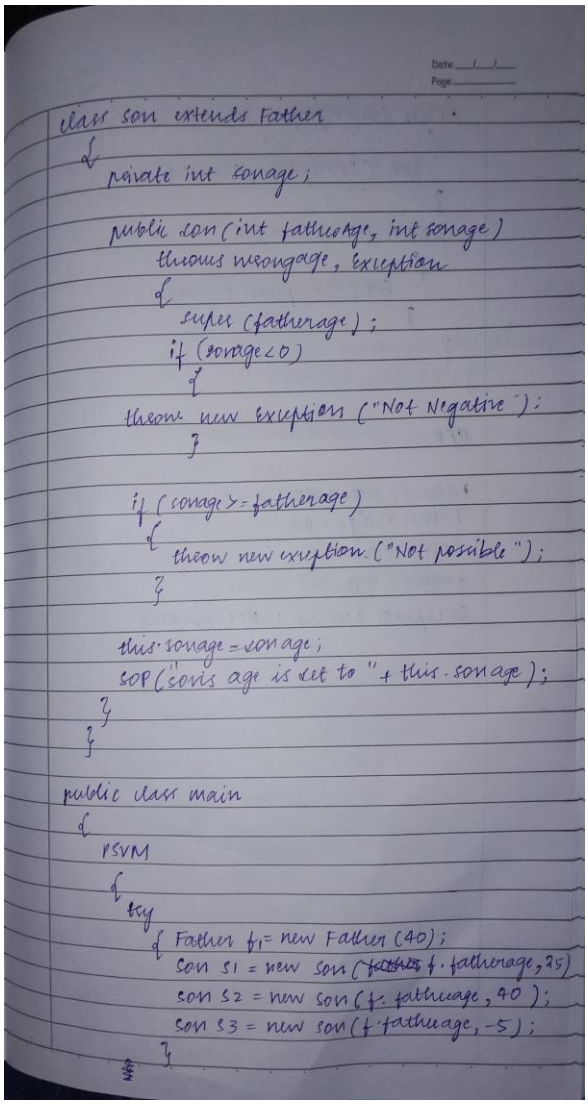
Base class - father

Derived class - son

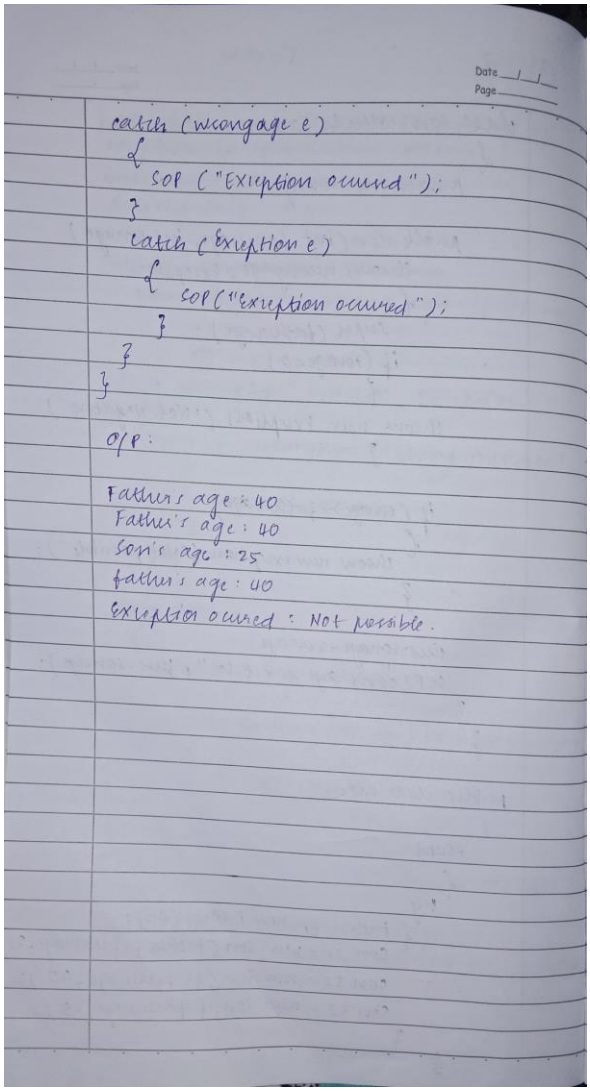
Father → constructor → takes age

throw exception wrong age if i/p age<0

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

class Father
{
    protected int fatherAge;

    public Father (int age) throws WrongAge
    {
        if (age<0)
        {
            throw new WrongAge
                ("Age cant be negative");
        }
        this. fatherAge = age
        SOP("father age : " + this. fatherAge
    }
}
```



```
class son extends Father
{
  private int sonage;

  public son(int fatherAge, int sonage)
      throws wrongage, Exception
      {
        super (fatherage);
        if (sonage<0)
        {
      throw new Exception ("Not Negative");
        }

      if (sonage>=fatherage)
      {
        throw new exception ("Not possible");
      }

      this.sonage = sonage;
      SOP("son's age is set to " + this.sonage);
  }
}

public class main
{
  PSVM
  {
    try
      { Father f1 = new Father (40);
        son s1 = new son (f.fatherage, 25)
        son s2 = new son (f.fatherage, 40);
        son s3 = new son (f.fatherage, -5);
      }
```



```
    catch (wrongage e)
    {
      SOP ("Exception ocurred");
    }
    catch (Exception e)
      {
        SOP("exception ocurred");
      }
  }
}

O/P:

Father's age : 40
Father's age : 40
Son's age : 25
father's age : 40
Exception ocurred : Not possible.
```

Code:

```
import java.util.Scanner;
class WrongAge extends Exception{
int a;
WrongAge(int a)
{
this.a=a;} public String
toString(){ return a+" is a
invalid Age";
}}
class SonAgeExceedsFatherAge extends Exception{
int fa,a;
SonAgeExceedsFatherAge(int fa,int a)
{
```

```
this.fa=fa;
this.a=a;
}
public String toString(){ return "father's("+fa+") age cannot be lesser
than that of son("+a+")";
}} class
Father { int
fage;
Father(int a) {
fage=a; }
public void fathervalidage() throws WrongAge{
if(fage<0){
throw new WrongAge(fage);}
}}
class Son extends Father{
int age;
Son(int fa,int a)
{
super(fa); age=a;} public void sonvalidage() throws
SonAgeExceedsFatherAge{ if(fage<age){ throw new
SonAgeExceedsFatherAge(fage,age);
}} void
display()
{
System.out.println("Father's age:"+fage+"\nSon's age:"+age);
}
}class FatherSon{ public static void
main(String args[]){
Scanner sc=new Scanner(System.in);
System.out.println("Ananya N Gowda R\n1BM23CS034");
System.out.println("Enter Father's age:");
int fage=sc.nextInt();
System.out.println("Enter Son's age:");
int age=sc.nextInt();
Son child=new Son(fage,age);
try{
child.fathervalidage();
child.sonvalidage();
System.out.println("Ages are valid");
child.display();}
catch(WrongAge e){
System.out.println(e);}
catch(SonAgeExceedsFatherAge e){
```

```
System.out.println(e);
}}}
```

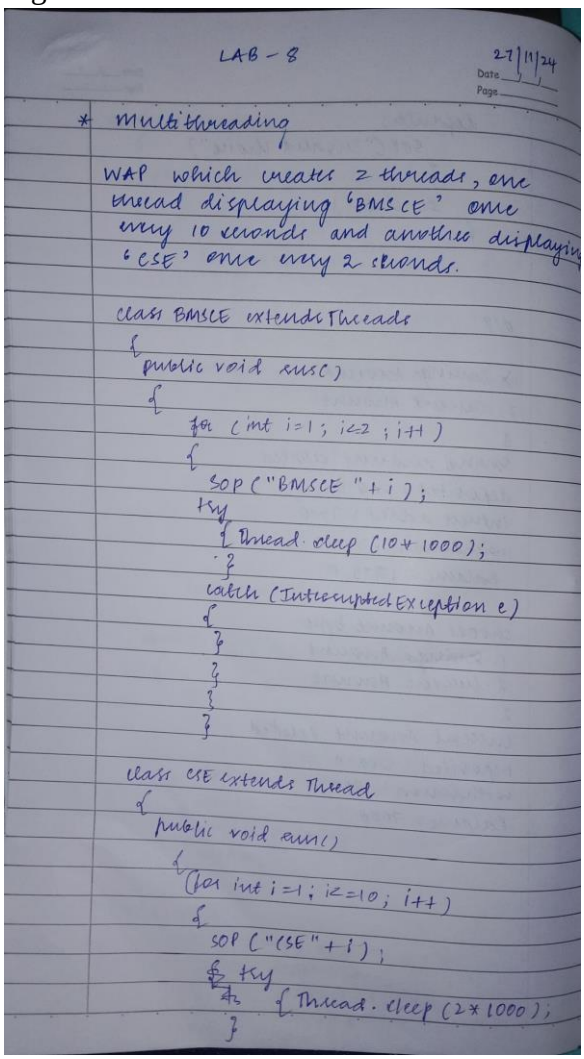
Output:

```
D:\ananya>javac FatherSon.java

D:\ananya>java FatherSon
Ananya N Gowda R
1BM23CS034
Enter Father's age:
45
Enter Son's age:
25
Ages are valid
Father's age:45
Son's age:25

D:\ananya>java FatherSon
Ananya N Gowda R
1BM23CS034
Enter Father's age:
25
Enter Son's age:
26
father's(25) age cannot be lesser than that of son(26)

D:\ananya>
```

Program 8

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.

Algorithm:

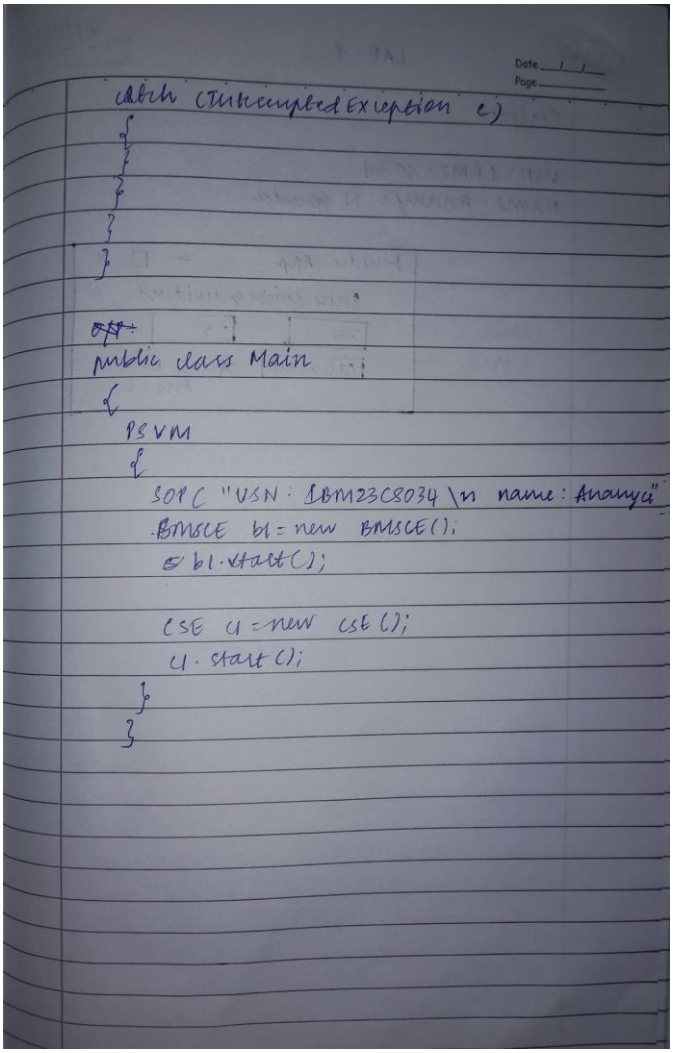


LAB - 8 27/11/24

* Multithreading

WAP which creates 2 threads, one thread displaying 'BMSCE' once every 10 seconds and another displaying 'CSE' once every 2 seconds.

```
class BMSCE extends Threads
{
 public void run()
 {
    for (int i=1; i<=2 ; i++)
    {
     SOP ("BMSCE " + i);
     try
      { Thread.sleep (10* 1000);
      }
     catch (InterruptedException e)
     {
      }
      }
      }
      }

class CSE extends Thread
{
 public void run()
 {for int i=1; i<=10; i++)
  {
   SOP ("CSE" + i);
   try
    { Thread.sleep (2* 1000);
    }
```



```
catch (InterruptedException e)
{
}
}
}
}

public class Main
{
 PSVM
 {
  SOP ("USN: 1BM23CS034 \n name: Ananya")
  BMSCE b1 = new BMSCE();
  b1.start();

  CSE c1 = new CSE();
  c1.start();
 }
}
```

Code:

```
class BMSCE extends Thread
{
public void run()
{
for(int i=1; i<=2; i++)
{
System.out.println("BMSCE " + i);
try
{Thread.sleep(10*1000);
```

```
}
catch(InterruptedException e)
{
}
}
}
}

class CSE extends Thread
{
public void run()
{
for(int i=1; i<=10; i++)
{
System.out.println("CSE " + i);
try
{Thread.sleep(2*1000);
}
catch(InterruptedException e)
{
}
}
}
}



public class Main
{
public static void main(String args[])
{
System.out.println("USN: 1BM23CS034\nName: Ananya N Gowda\n");
BMSCE b1 = new BMSCE();
b1.start();

CSE c1 = new CSE();
c1.start();

}
}
```

Output:

```
D:\ananya>java Main
USN: 1BM23CS034
Name: Ananya N Gowda

CSE 1
BMSCE 1
CSE 2
CSE 3
CSE 4
CSE 5
BMSCE 2
CSE 6
CSE 7
CSE 8
CSE 9
CSE 10

D:\ananya>
```

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

Algorithm:

LAB - 9

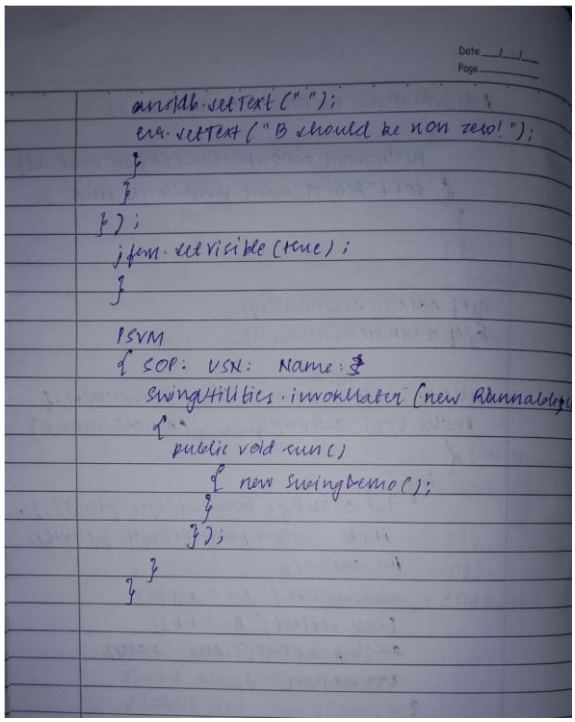
Date 27/11/24

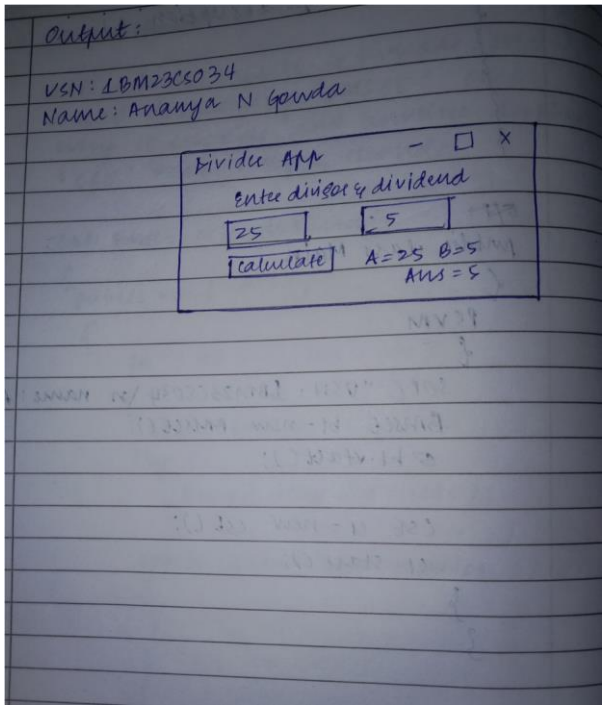
```
# import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class 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 divisor
                    and Dividend ");

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);
```

Date / /

Page

```
ActionListener l = new ActionListener()
    {
    public void actionPerformed (ActionEvent evt)
    { SOP ("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 ("A = " +a);
    blab.setText ("B = " +b);
    anslab.setText (" Ans = "+ans);
    err.setText (" ");
}
catch (NumberFormatException e)
{
    alab.setText ("");
    blab.setText (" ");
    anslab.setText (" ");
    err.setText (" Enter only integer! ");
}
catch (ArithmeticException e)
{
    alab.setText ("");
    blab.setText (" ");
```





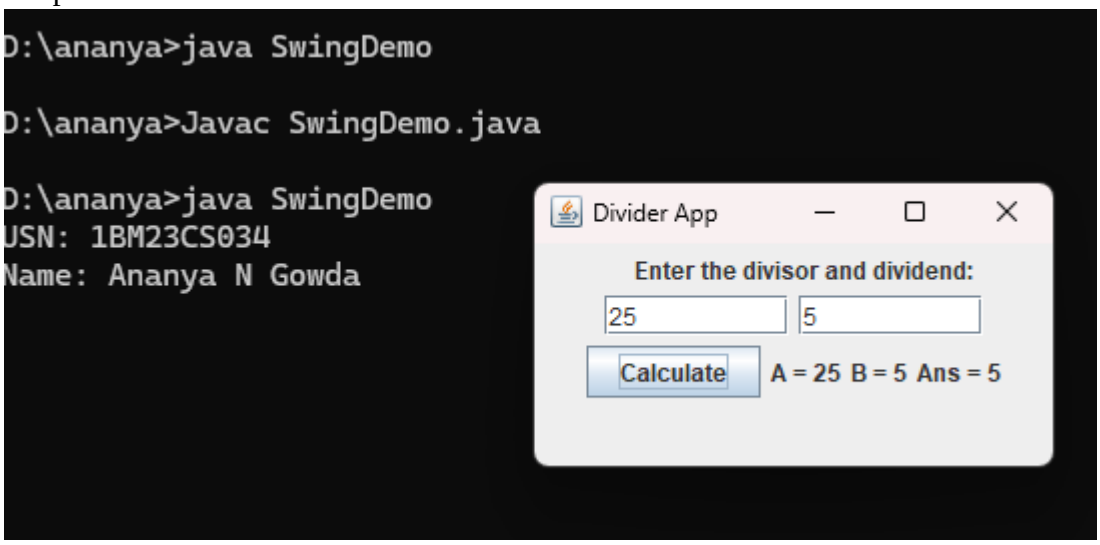
Code:

```
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 divisor and dividend:");
        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("A = " + a);
                    blab.setText("B = " + b);
                    anslab.setText("Ans = " + ans);
                    err.setText("");
                } 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: 1BM23CS034\nName: Ananya N Gowda\n");
```

```
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new SwingDemo();
            }
        });
    }
}
```

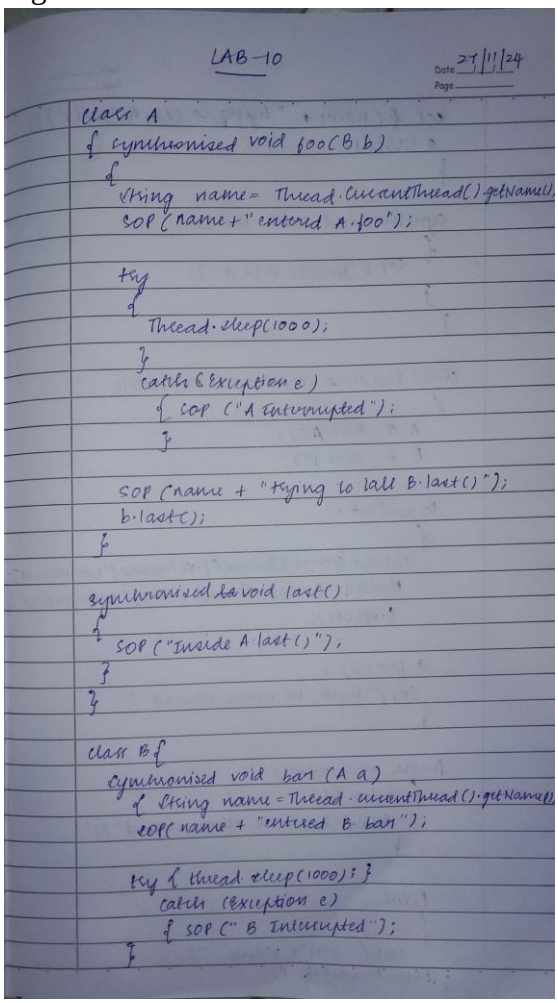
Output:



Program 10

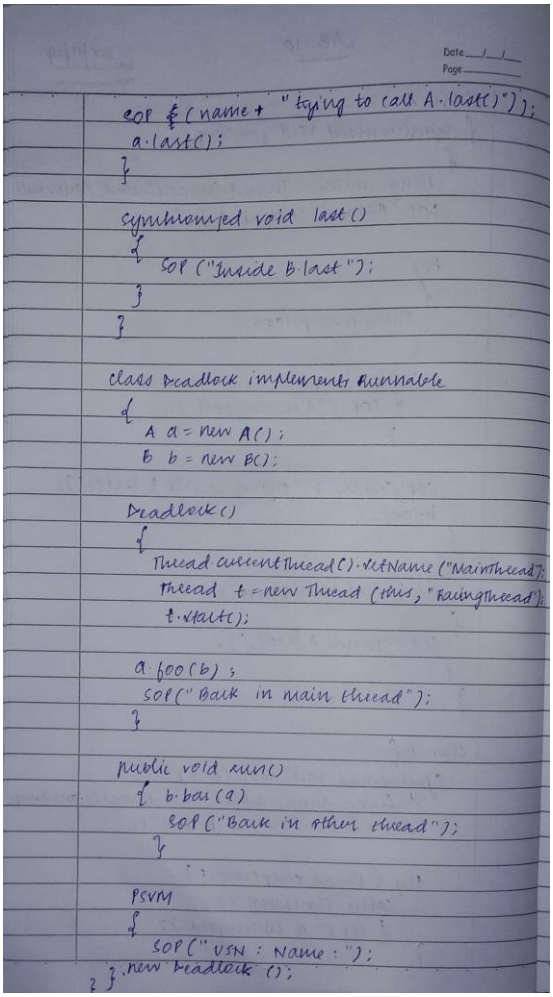
Demonstrate Inter process Communication and deadlock

Algorithm:

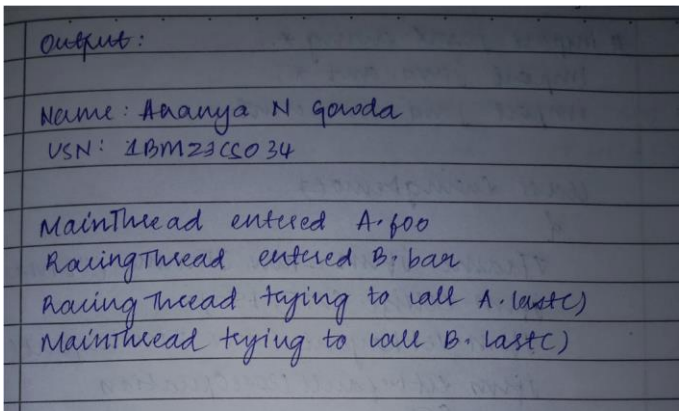


LAB-10 Date 27/11/24

```
class A
{ synchronized void foo(B b)
  {
    String name= Thread.CurrentThread().getName();
    SOP(name+" entered A.foo");

    try
    {
      Thread.sleep(1000);
    }
    catch(Exception e)
      { SOP ("A Interrupted");
      }

    SOP (name + "trying to call B.last()");
    b.last();
  }

synchronized void last()
{
  SOP ("Inside A.last()");
}
}

class B{
  synchronized void bar (A a)
    { String name = Thread.currentThread().getName();
    SOP( name + "entered B.bar");

    try { thread.sleep(1000); }
      catch (Exception e)
        { SOP (" B Interrupted");
      }
```



```
    SOP ( name + "trying to call A.last()");
    a.last();
  }

  synchronized void last()
  {
    SOP ("Inside B.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);
    SOP("Back in main thread");
  }

  public void run()
  { b.bar(a)
    SOP("Back in other thread");
  }

  PSVM
  {
    SOP(" USN : Name : ");
  } new Deadlock ();
}
```



Output:

Name : Ananya N Gowda
USN : 1BM23CS034

MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()

Code:

```
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();
    }

    synchronized 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();
    }

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

        // Get lock on 'a' in this thread
        a.foo(b);
        System.out.println("Back in main thread");
    }

    public void run() {
        // Get lock on 'b' in other thread
        b.bar(a);
        System.out.println("Back in other thread");
    }

    public static void main(String[] args) {
System.out.println("USN: 1BM23CS034\nName: Ananya N Gowda\n");

        new Deadlock();
    }
}
```

Output:

```
PS C:\Users\Admin> d:
PS D:\> cd ananya
PS D:\ananya> javac Deadlock.java
PS D:\ananya> java Deadlock
USN: 1BM23CS034
Name: Ananya N Gowda

MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()
```