VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

JAYANTH GOWDA A (1BM23CS123)

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 **Jayanth Gowda A(1BM23CS123),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

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

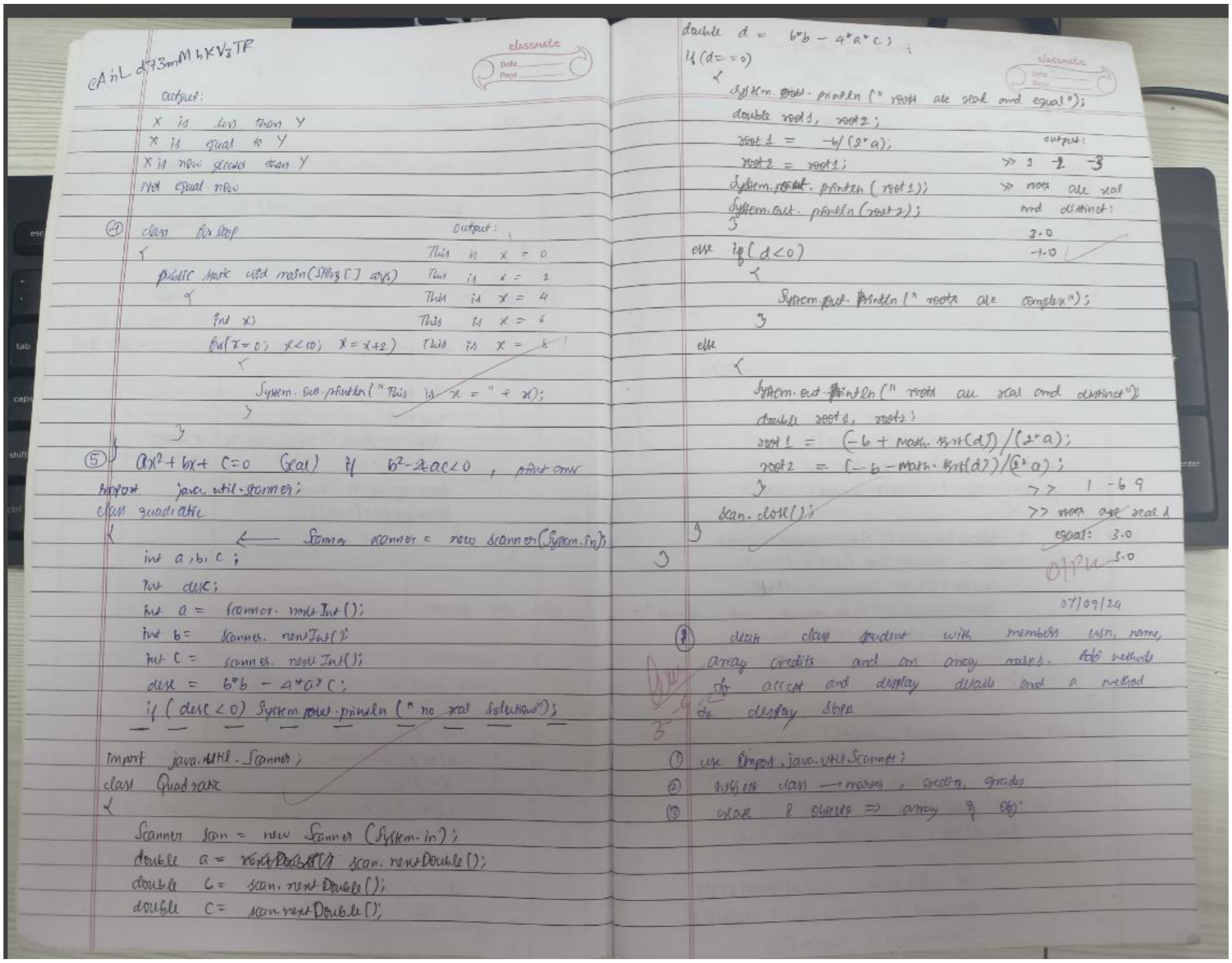
Index

Github Link:
https://github.com/Jayanth0927/java-lab-programs

Program 1

Implement Quadratic Equation

ALGORITHM:



cAnL d73mMbKV3TP

output:

```
X is less than Y
X is equal to Y
X is not greater than Y
not equal now
```

④ class forLoop

```
{
  public static void main(String[] args)
  {
      int x;
      for(x=0; x<10; x=x+2)
      {
          System.out.println("This is x = " + x);
      }
  }
}
```

Output:

```
This is x = 0
This is x = 2
This is x = 4
This is x = 6
This is x = 8
```

⑤ $ax^2+bx+c=0$ (real) if $b^2-4ac<0$, print error

```
import java.util.Scanner;
class quadratic
{
        ← Scanner scanner = new Scanner(System.in);
    int a, b, c;
    int desc;
    int a = scanner.nextInt();
    int b = scanner.nextInt();
    int c = scanner.nextInt();
    desc = b*b - 4*a*c;
    if (desc < 0) System.out.println ("no real solution");

import java.util.Scanner;
class Quadratic
{
    Scanner scan = new Scanner (System.in);
    double a = ~~nextDouble()~~ scan.nextDouble();
    double c = scan.nextDouble();
    double c = scan.nextDouble();
double d = b*b - 4*a*c;
if (d==0)
  {
    System.~~out~~.println(" roots are real and equal");
    double root1, root2;
    root1 = -b/(2*a);
    root2 = root1;
    System.~~out~~.println(root1);
    System.out.println(root2);
  }
else if(d<0)
  {
      System.out.println(" roots are complex");
  }
else
  {
    System.out.println(" roots are real and distinct");
    double root1, root2;
    root1 = (-b + Math.sqrt(d))/(2*a);
    root2 = (-b - Math.sqrt(d))/(2*a);
  }
  scan.close();
}
}
```

output:

```
>> 1 -2 -3
>> roots are real
and distinct:
3.0
-1.0
```

```
>> 1 -6 9
>> roots are real &
equal: 3.0
3.0
```

O/P

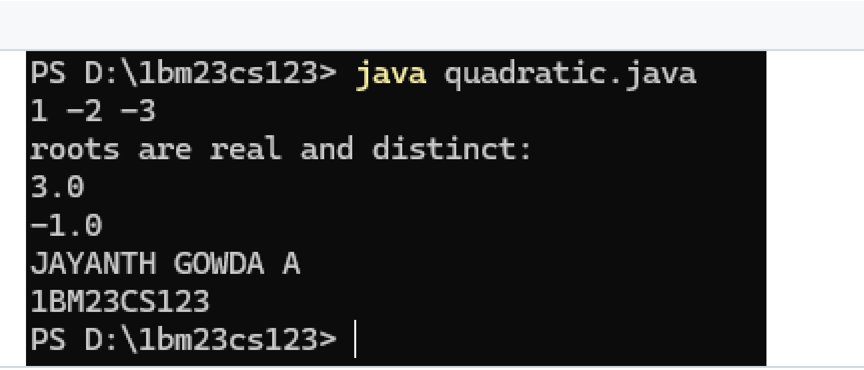
07/09/24

⑧ create class student with members usn, name, array credits and an array marks. Add methods to accept and display details and a method to display sgpa

① use import.java.util.Scanner;
② subject class → marks, credits, grades
③ create 8 objects ⇒ array of obj

CODE:

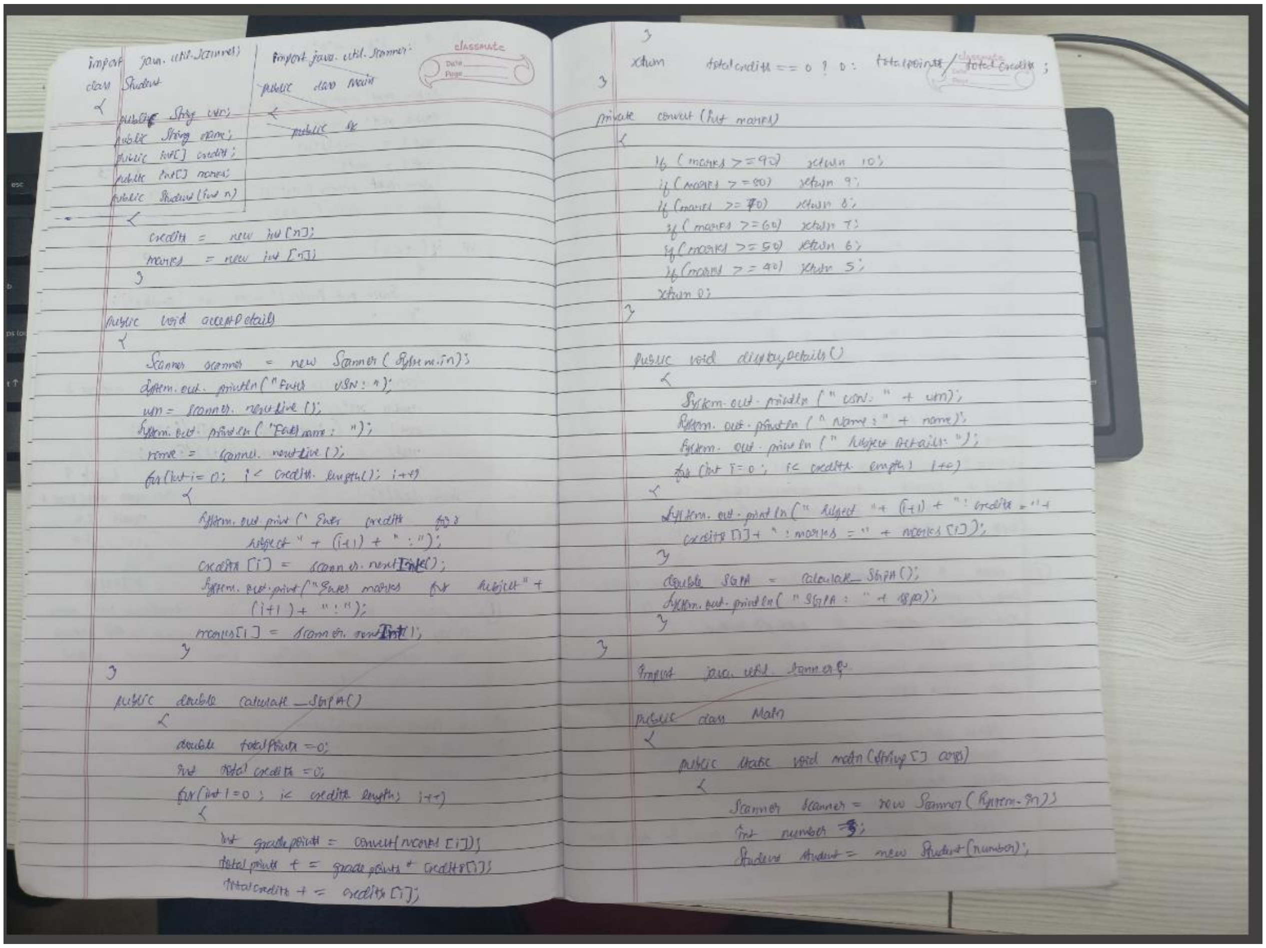
```
import java.util.Scanner;

public class QuadraticEquation {

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

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

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

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

        double discriminant = b * b - 4 * a * c;
        if (discriminant > 0) {
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
            System.out.println("The roots are real and distinct.");
            System.out.println("Root 1: " + root1);
            System.out.println("Root 2: " + root2);
        } else if (discriminant == 0) {
            double root = -b / (2 * a);
            System.out.println("The roots are real and equal.");
            System.out.println("Root: " + root);
        } else {
            double realPart = -b / (2 * a);
            double imaginaryPart = Math.sqrt(-discriminant) / (2 * a);
            System.out.println("The roots are complex and distinct.");
            System.out.println("Root 1: " + realPart + " + " + imaginaryPart + "i");
            System.out.println("Root 2: " + realPart + " - " + imaginaryPart + "i");
        }

        scanner.close();
    }
```



```
PS D:\1bm23cs123> java quadratic.java
1 -2 -3
roots are real and distinct:
3.0
-1.0
JAYANTH GOWDA A
1BM23CS123
PS D:\1bm23cs123>
```

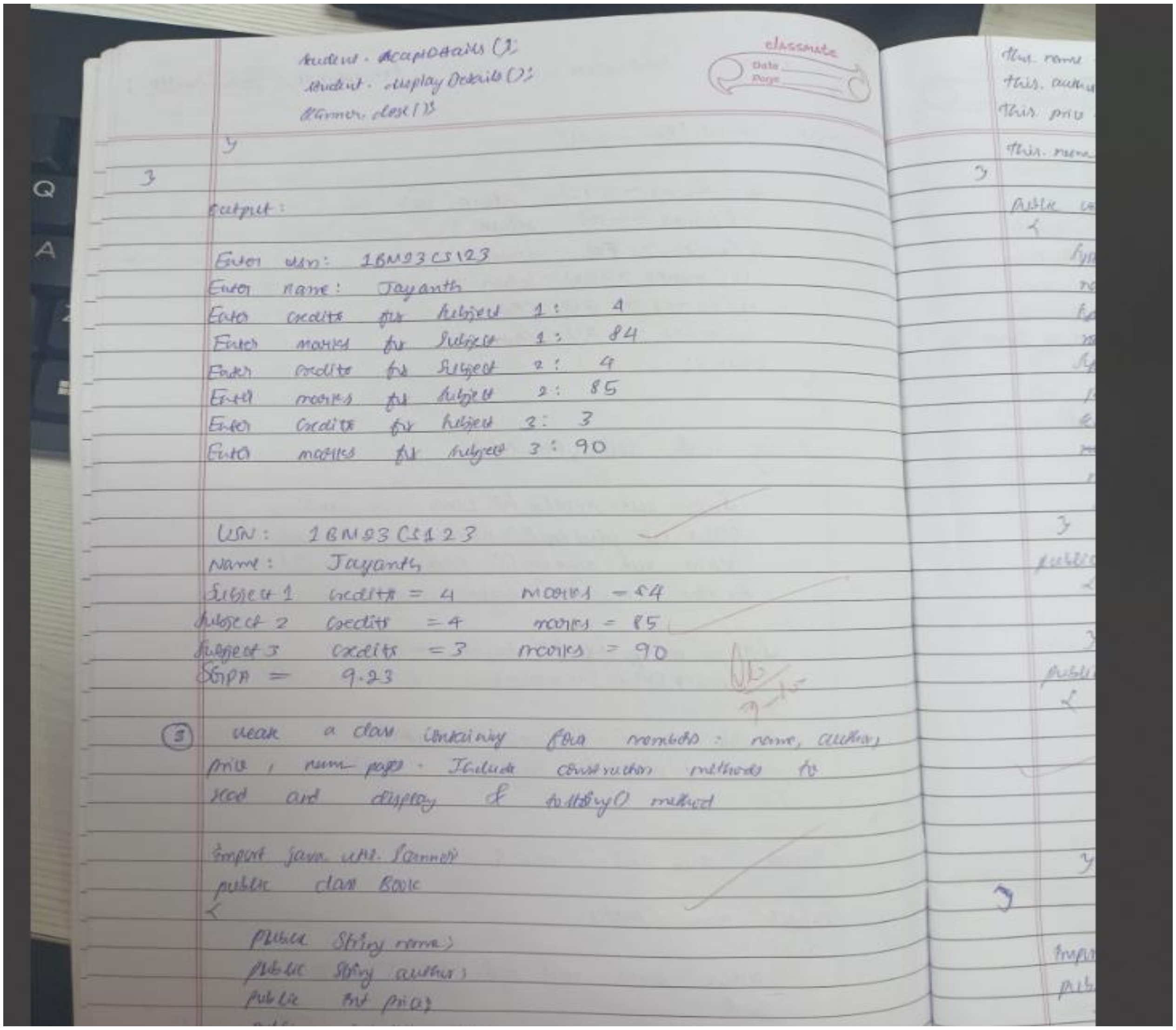
PROGRAM 2: CREATE SGPA:

ALGORITHM:



~~Import java. util. Scanner;~~
~~public class Main~~
~~{~~
~~public st~~

```
import java. util. Scanner;
class Student
{
public String usn;
public String name;
public int[] credits;
public int[] marks;
public Student (int n)
  {
    credits = new int [n];
    marks = new int [n];
  }

public void acceptDetails
  {
    Scanner scanner = new Scanner ( System.in);
    System. out. println ("Enter USN: ");
    usn = scanner. nextLine ();
    System. out. println ("Enter name: ");
    name = scanner. nextLine ();
    for(int i= 0; i< credits. length(); i++)
      {
        System. out. print (" Enter credits for
              subject " + (i+1) + " : ");
        credits [i] = scanner. nextInt();
        System. out. print ("Enter marks for subject" +
              (i+1) + " : ");
        marks[i] = scanner. nextInt();
      }
  }
public double calculate_SGPA()
  {
    double totalPoints = 0;
    int totalcredits = 0;
    for(int i = 0 ; i< credits length; i++)
      {
        int gradepoints = convert(marks [i]);
        totalpoints + = grade points * credits[i];
        totalcredits + = credits [i];
      }
    return totalcredits == 0 ? 0 : totalpoints / total credits ;
  }

private convert (int marks)
  {
    if (marks >= 90)  return 10;
    if (marks >= 80)  return 9;
    if (marks >= 70)  return 8;
    if (marks >= 60)  return 7;
    if (marks >= 50)  return 6;
    if (marks >= 40)  return 5;
    return 0;
  }

public void displayDetails()
  {
    System. out. println (" USN: " + usn);
    System. out. println (" Name : " + name);
    System. out. println (" subject Details: ");
    for (int i = 0 ; i< credits. length; i++)
  {
  System. out. println (" subject "+ (i+1) + " : credits = "+
    credits [i] + " : marks = " + marks [i]);
  }
  double SGPA = calculate_SGPA();
  System. out. println ( " SGPA : " + sgpa);
  }
}

Import java. util. Scanner;

public class Main
 {
   public static void main (String [] args)
     {
       Scanner scanner = new Scanner ( System. in);
       int number = 3;
       Student student = new Student (number);
```



```
        Student. acceptDetails ();
        Student. displayDetails ();
        Scanner. close();
    }
}
```

Output:

```
Enter usn:  1BM23CS123
Enter name:  Jayanth
Enter credits for subject 1:  4
Enter marks for subject 1:  84
Enter credits for subject 2:  4
Enter marks for subject 2:  85
Enter credits for subject 3:  3
Enter marks for subject 3:  90

USN:  1BM23CS123
Name:  Jayanth
Subject 1  credits = 4   marks = 84
Subject 2  credits = 4   marks = 85
Subject 3  credits = 3   marks = 90
SGPA =  9.23
```

(3) Create a class containing four members: name, author, price, num pages. Include construction methods to read and display & toString() method

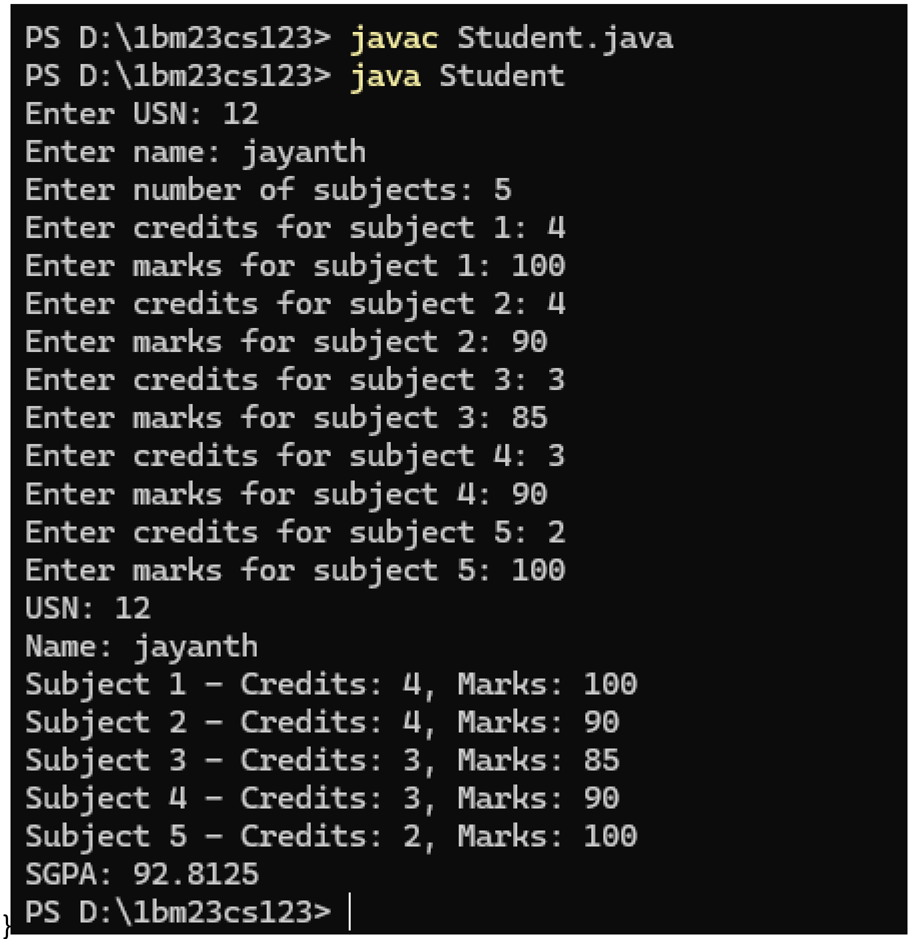
```
import java.util.Scanner
public class Book
{
    public String name;
    public String author;
    public int price;
```

```
this. name
this. autho
this. pric
this. nam
}
public
{
}
public
}
public
{
}
}
}
impo
pub
```

CODE:

```
 import java.util.Scanner;

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

   public void acceptDetails() {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter USN: ");
```

```
        usn = scanner.nextLine();
        System.out.print("Enter name: ");
        name = scanner.nextLine();
        System.out.print("Enter number of subjects: ");
        int n = scanner.nextInt();
        credits = new int[n];
        marks = new int[n];
        for (int i = 0; i < n; i++) {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = scanner.nextInt();
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }

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

    public double calculateSGPA() {
        double totalCredits = 0;
        double weightedMarks = 0;
        for (int i = 0; i < credits.length; i++) {
            weightedMarks += credits[i] * marks[i];
            totalCredits += credits[i];
        }
        return weightedMarks / totalCredits;
    }

    public static void main(String[] args) {
        Student student = new Student();
        student.acceptDetails();
        student.displayDetails();
        double sgpa = student.calculateSGPA();
        System.out.println("SGPA: " + sgpa);
    }
```

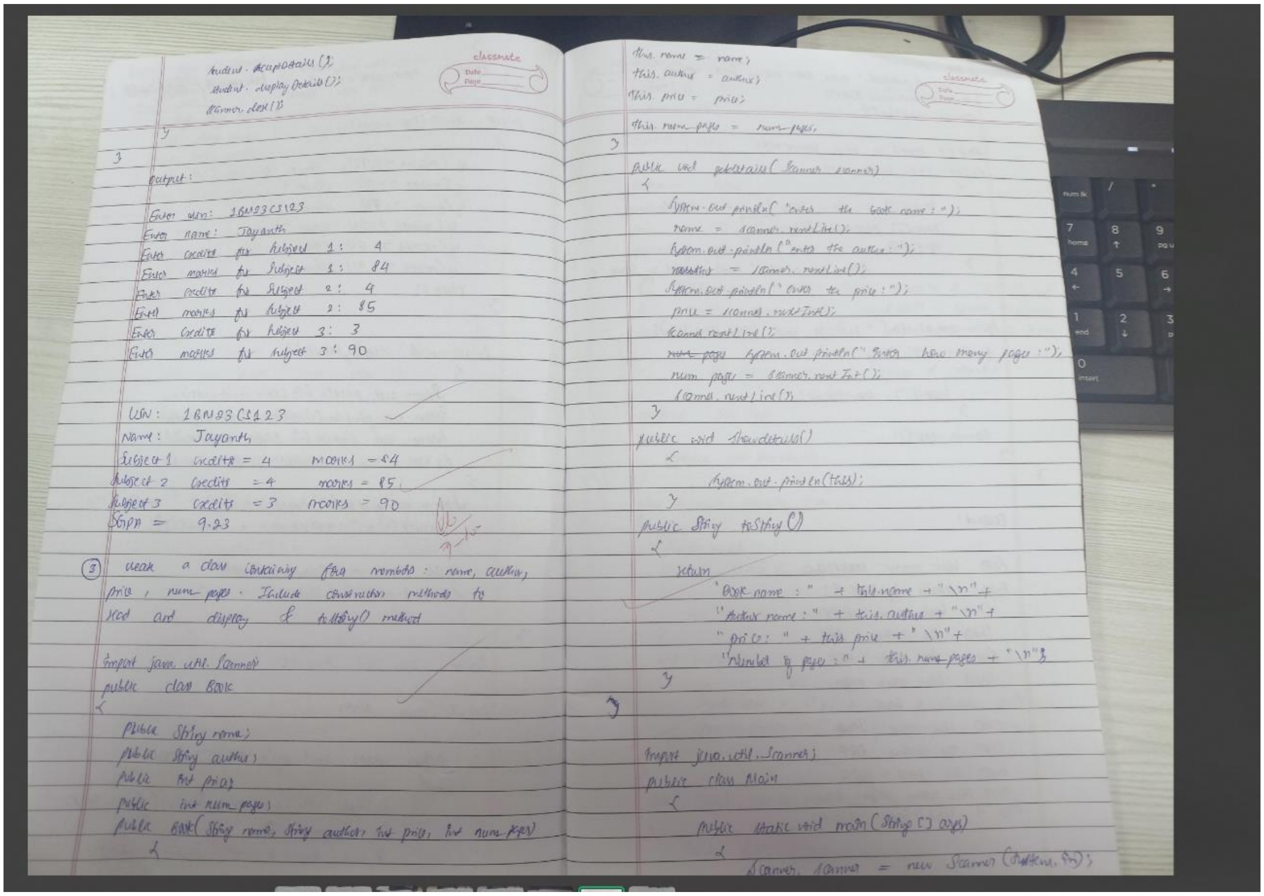


```
PS D:\1bm23cs123> javac Student.java
PS D:\1bm23cs123> java Student
Enter USN: 12
Enter name: jayanth
Enter number of subjects: 5
Enter credits for subject 1: 4
Enter marks for subject 1: 100
Enter credits for subject 2: 4
Enter marks for subject 2: 90
Enter credits for subject 3: 3
Enter marks for subject 3: 85
Enter credits for subject 4: 3
Enter marks for subject 4: 90
Enter credits for subject 5: 2
Enter marks for subject 5: 100
USN: 12
Name: jayanth
Subject 1 - Credits: 4, Marks: 100
Subject 2 - Credits: 4, Marks: 90
Subject 3 - Credits: 3, Marks: 85
Subject 4 - Credits: 3, Marks: 90
Subject 5 - Credits: 2, Marks: 100
SGPA: 92.8125
PS D:\1bm23cs123>
```

}

PROGRAM 3: CLASS BOOK

ALGORITHM:



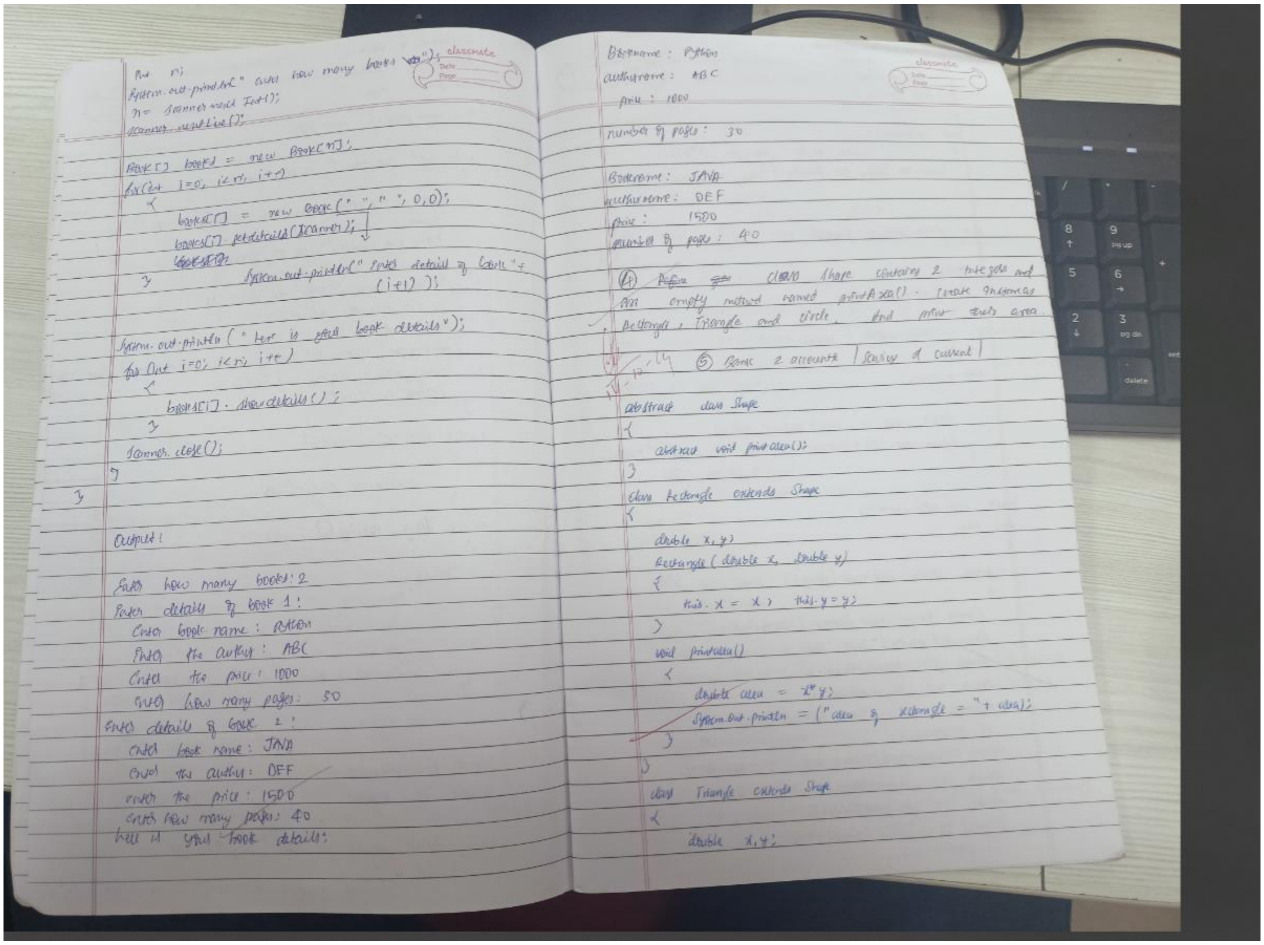
```
        student.acceptDetails();
        student.displayDetails();
        scanner.close();
    }
}
```

output:

```
Enter usn: 1BM23CS123
Enter name: Jayanth
Enter credits for subject 1:  4
Enter marks for subject 1:  84
Enter credits for subject 2:  4
Enter marks for subject 2:  85
Enter credits for subject 3:  3
Enter marks for subject 3: 90


USN:  1BM23CS123
Name:   Jayanth
Subject 1  credits = 4     marks = 84
Subject 2  credits = 4     marks = 85
Subject 3  credits = 3     marks = 90
SGPA =   9.23
```

(3) Create a class containing five members : name, author, price, num_pages. Include constructor methods to read and display & toString() method

```
import java.util.Scanner;
public class Book
{
    public String name;
    public String author;
    public int price;
    public int num_pages;
    public 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 void getDetails(Scanner scanner)
    {
        System.out.println("enter the book name : ");
        name = scanner.nextLine();
        System.out.println("enter the author : ");
        author = scanner.nextLine();
        System.out.println("enter the price : ");
        price = scanner.nextInt();
        scanner.nextLine();
        ~~num_pages~~ System.out.println("Enter how many pages : ");
        num_pages = scanner.nextInt();
        scanner.nextLine();
    }
    public void showdetails()
    {
        System.out.println(this);
    }
    public String toString()
    {
        return
            "Book name : " + this.name + "\n" +
            "Author name : " + this.author + "\n" +
            "price : " + this.price + "\n" +
            "Number of pages : " + this.~~num~~_pages + "\n";
    }
}


import java.util.Scanner;
public class Main
{
    public static void main(String [] args)
    {
        Scanner scanner = new Scanner(System.in);
```



```
int n;
System.out.println(" enter how many books ");
n= scanner.nextInt();
scanner.nextLine();

Book[] books = new Book[n];
for(int i=0; i<n; i++)
  {
      books[i] = new Book(" ", " ", 0,0);
      books[i].getdetails(scanner);
      ~~books[i]~~
  }          System.out.println(" Enter details of Book "+
                                   (i+1) );

System.out.println ( " here is your book details");
for (int i=0; i<n; i++)
  {
      books[i] . showdetails () ;
  }
scanner.close();
}
}
```

Output:

```
Enter how many books: 2
Enter details of book 1:
  Enter book name : PYTHON
  Enter the author : ABC
  Enter the price : 1000
  Enter how many pages: 30
Enter details of Book 2 :
  Enter book name: JAVA
  Enter the author: DEF
  enter the price : 1500
  enter how many pages: 40
here is your book details:
```

```
Bookname : Python
authorname : ABC
 price : 1000
number of pages : 30

Bookname : JAVA
authorname : DEF
price :      1500
number of pages : 40
```

(4) ~~Define~~ ~~an~~ class Shape containing 2 integers and an empty method named printArea() . Create instances Rectangle , Triangle and Circle. And print their area.

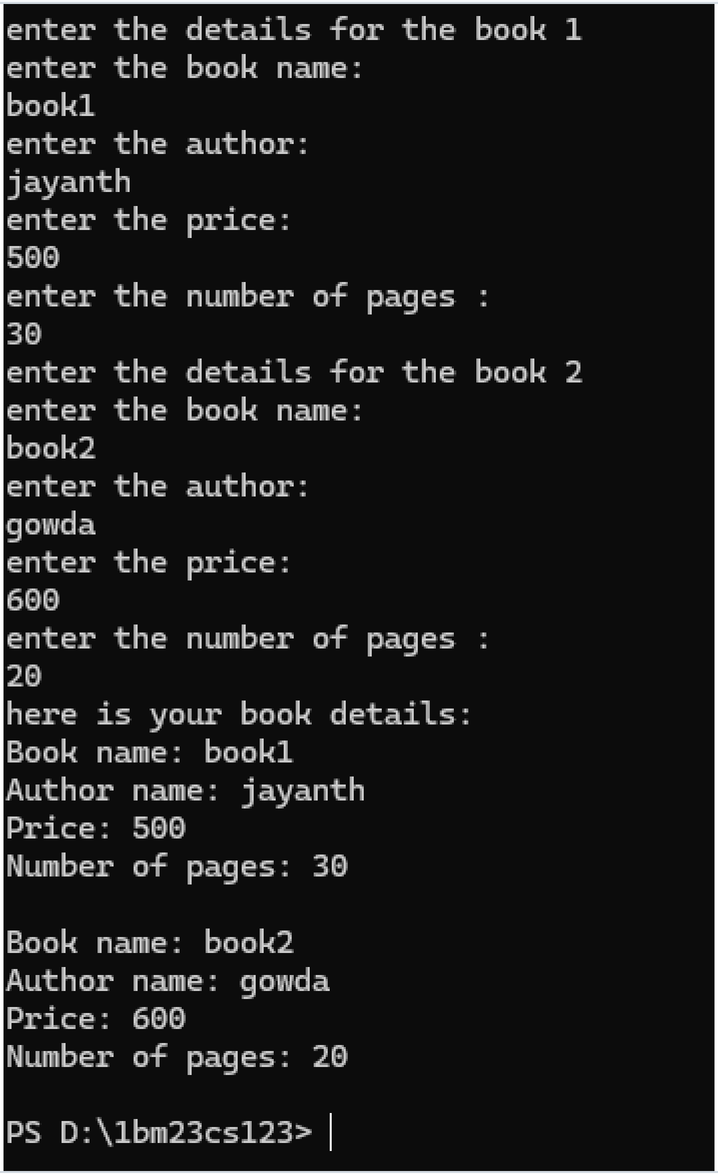
(5) Bank 2 accounts | Saving & current |

```
abstract  class Shape
{
    abstract void printArea();
}
class Rectangle extends Shape
{
    double x, y;
    Rectangle ( double x, double y)
    {
        this. x = x ;  this. y = y;
    }
    void printarea()
     {
        double area = x* y;
        System.out.println = ("area of rectangle = "+ area);
     }
}
class Triangle extends Shape
{
    double x, y;
```

CODE: import java.util.Scanner;

```
public class Book
{
  public String name;
  public String author;
  public int  price;
  public int num_pages;
  public 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 void getdetails(Scanner scanner)

 {

   System.out.println("enter the book name: ");

   name = scanner.nextLine();

   System.out.println("enter the author: ");

   author = scanner.nextLine();

   System.out.println("enter the price: ");

   price = scanner.nextInt();

   scanner.nextLine();

   System.out.println("enter the number of pages : ");

   num_pages = scanner.nextInt();

   scanner.nextLine();

 }

 public void showdetails()

 {

  System.out.println(this);

 }

 public String toString()

 {

   return

    "Book name: " + this.name + "\n" +

    "Author name: " + this.author + "\n" +

    "Price: " + this.price + "\n" +

    "Number of pages: " +this.num_pages + "\n";

  }

}import java.util.Scanner;
```

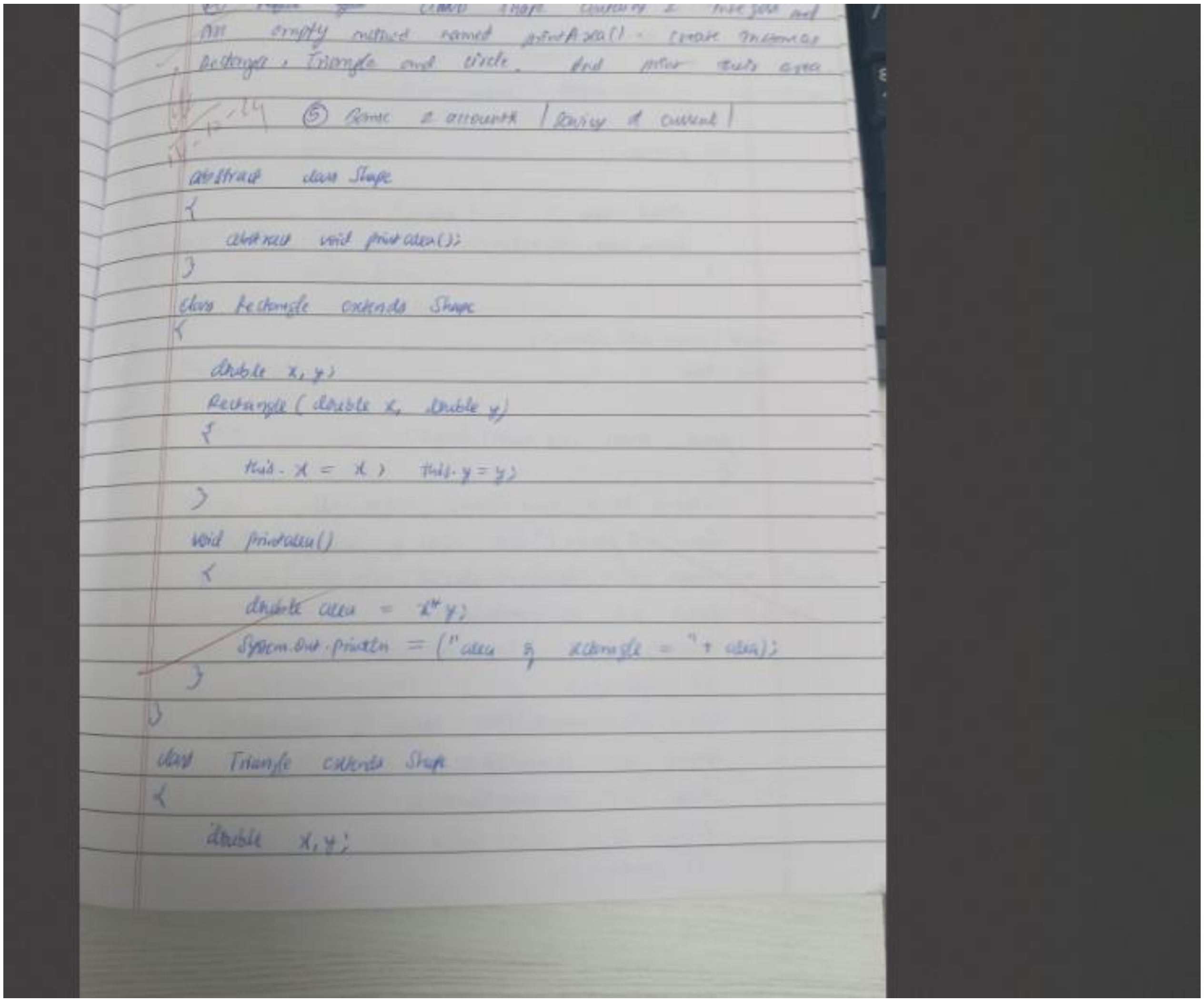
```
public class Main
{
  public static void main(String[] args)
  {
    Scanner scanner = new Scanner(System.in);
    int n;
    System.out.println("enter how many books");
    n = scanner.nextInt();
    scanner.nextLine();
    Book[] books = new Book[n];
    for(int i=0;i<n;i++)
    {
         books[i] = new Book(" "," ",0,0);
         System.out.println("enter the details for the book " + (i+1));
         books[i].getdetails(scanner);
    }
    System.out.println("here is your book details: ");
      for(int i=0;i<n;i++)
     {
        books[i].showdetails();
     }
        scanner.close();
  }
}
```



```
enter the details for the book 1
enter the book name:
book1
enter the author:
jayanth
enter the price:
500
enter the number of pages :
30
enter the details for the book 2
enter the book name:
book2
enter the author:
gowda
enter the price:
600
enter the number of pages :
20
here is your book details:
Book name: book1
Author name: jayanth
Price: 500
Number of pages: 30

Book name: book2
Author name: gowda
Price: 600
Number of pages: 20

PS D:\1bm23cs123> 
```

PROGRAM 4: ABSTRACT CLASS

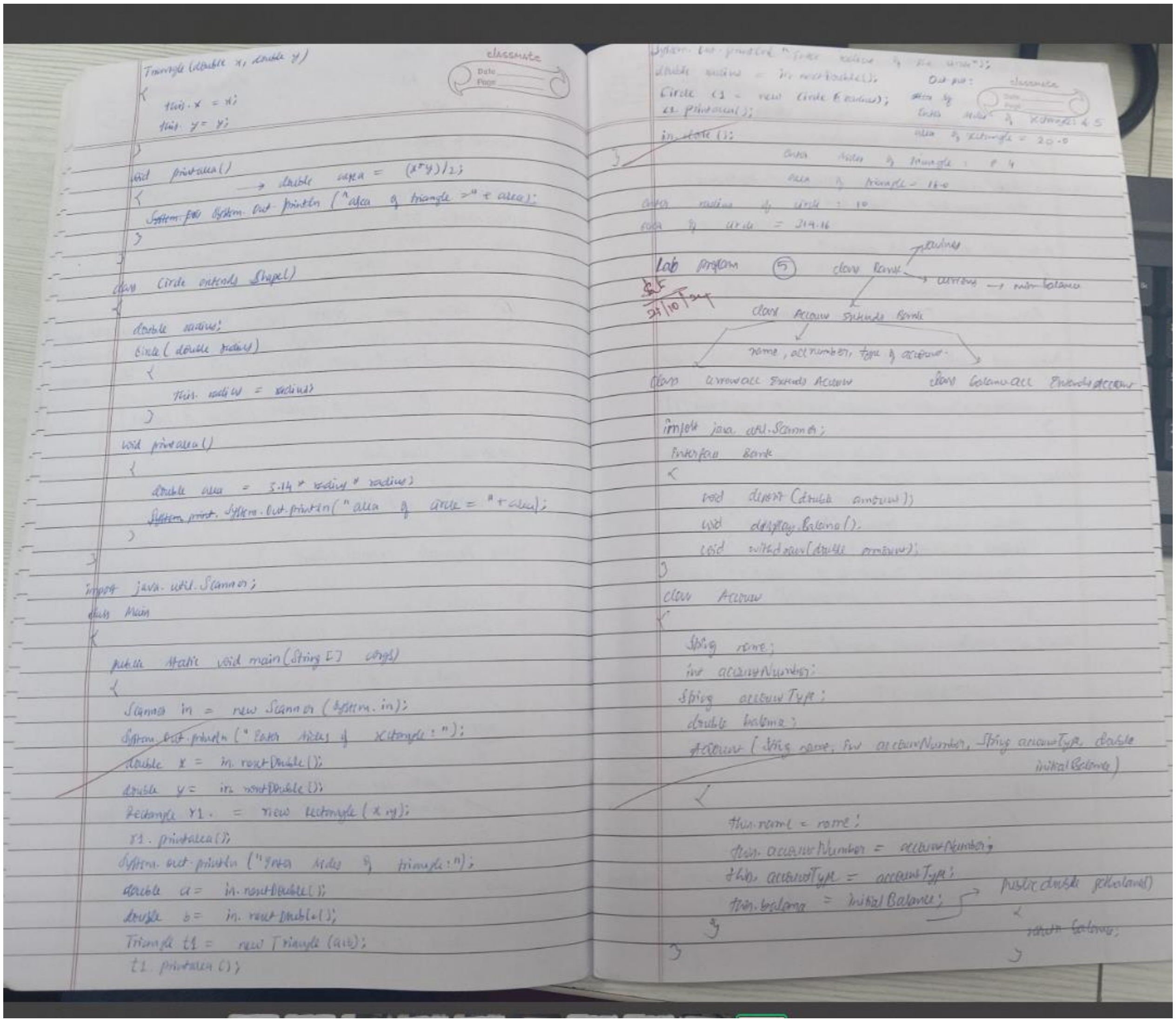
ALGORITHM :



[illegible] [illegible] [illegible] class Shape containing 2 [illegible] and
An empty method named printArea(). Create instances
Rectangle, Triangle and circle. And print their area

⑤ Some 2 accounts / Saving & current /

```
abstract class Shape
{
    abstract void printArea();
}
class Rectangle extends Shape
{
    double x, y;
    Rectangle ( double x, double y)
    {
        this.x = x; this.y = y;
    }
    void printArea()
    {
        double area = x* y;
        System.out.println = ("area of rectangle = " + area);
    }
}
class Triangle extends Shape
{
    double x, y;
```



```
Triangle(double x, double y)
{
  this.x = x;
  this.y= y;
}
void printarea()
{             → double area = (x*y)/2;
  ~~System.pri~~ System.out.println ("area of triangle ="+ area);
}
}
class Circle extends Shape()
{
  double radius;
  Circle( double radius)
   {
     This.radius = radius;
   }
  void printarea()
   {
     double area = 3.14 * radius * radius;
     ~~System.print.~~ System.out.println(" area of circle = "+area);
   }
}
import java.util.Scanner;
class Main
{
  public static void main(String [] args)
  {
   Scanner in = new Scanner (System.in);
   System.out.println (" Enter sides of rectangle : ");
   double x = in.nextDouble();
   double y= in.nextDouble();
   Rectangle r1 = new Rectangle (x,y);
   r1.printarea();
   System.out.println ("Enter sides of triangle:");
   double a= in.nextDouble();
   double b= in.nextDouble();
   Triangle t1 = new Triangle (a,b);
   t1.printarea();
   System.out.println ("Enter radius of the circle");
   double radius = in.nextDouble();
   Circle c1 = new Circle (radius);
   c1.printarea();
   in.close();
}
```

Output:

```
Enter sides of rectangle: 4 5
area of rectangle = 20.0
Enter sides of triangle : 8 4
area of triangle = 16.0
Enter radius of circle : 10
area of circle = 314.16
```

Lab program (5)

24/10/24

class Bank → savings, current → min balance

class Account extends Bank

name, acc number, type of account

class currentacc extends Account   class savingsacc extends account

```
import java.util.Scanner;
interface Bank
{
   void deposit(double amount);
   void displayBalance();
   void withdraw(double amount);
}
class Account
{
  String name;
  int accountNumber;
  String accountType;
  double balance;
  Account ( String name, int accountNumber, String accountType, double
                                                        initialBalance)
   {
      this.name = name;
      this.accountNumber = accountNumber;
      this.accountType = accountType;
      this.balance = initialBalance;  →  public double getBalance()
   }                                      {
}                                            return balance;
                                          }
```

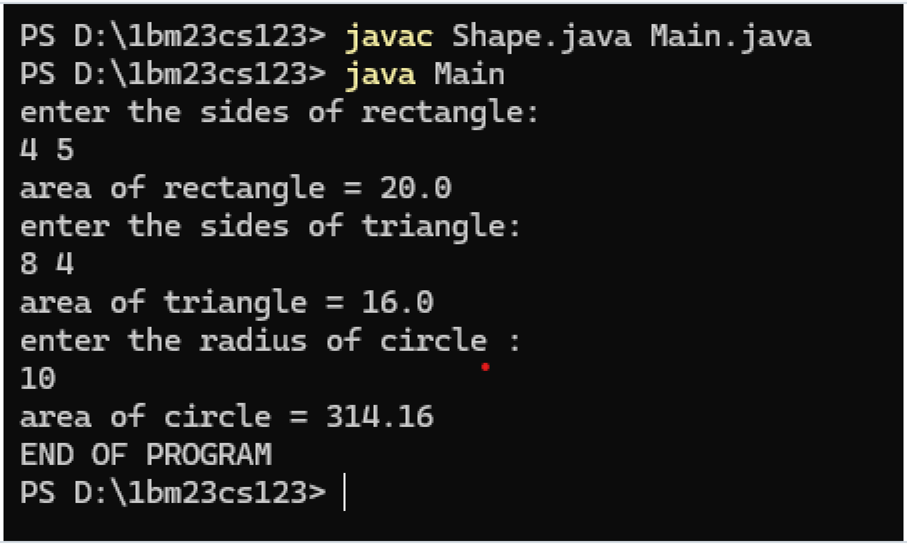
CODE:

```
abstract class Shape
{
  double x,y;
  abstract void printarea();
}
class Rectangle extends Shape
{
  Rectangle(double a,double b)
```

```
 {
   this.x = a;
   this.y = b;
 }
 void printarea()
 {
   double area = x*y;
   System.out.println("area of rectangle = " + area);
 }
}
class Triangle extends Shape
{
 Triangle(double a,double b)
 {
   this.x = a;
   this.y = b;
 }
 void printarea()
 {
   double area = (x*y)/2;
   System.out.println("area of triangle = " + area);
 }
}
class Circle extends Shape
{
 Circle(double radius)
 {
   this.x = radius;
 }
```

```
  void printarea()
  {
    double area = 3.1416*x*x;
    System.out.println("area of circle = " + area);
  }
}
import java.util.Scanner;
class Main
{
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.println("enter the sides of rectangle: ");
    double x = in.nextDouble();
    double y = in.nextDouble();
    Rectangle r1 = new Rectangle(x,y);
    r1.printarea();
    System.out.println("enter the sides of triangle: ");
    double a = in.nextDouble();
    double b = in.nextDouble();
    Triangle t1 = new Triangle(a,b);
    t1.printarea();
    System.out.println("enter the radius of circle : ");
    double radius = in.nextDouble();
    Circle c1 = new Circle(radius);
    c1.printarea();
    System.out.println("END OF PROGRAM");
    in.close();
  }
```

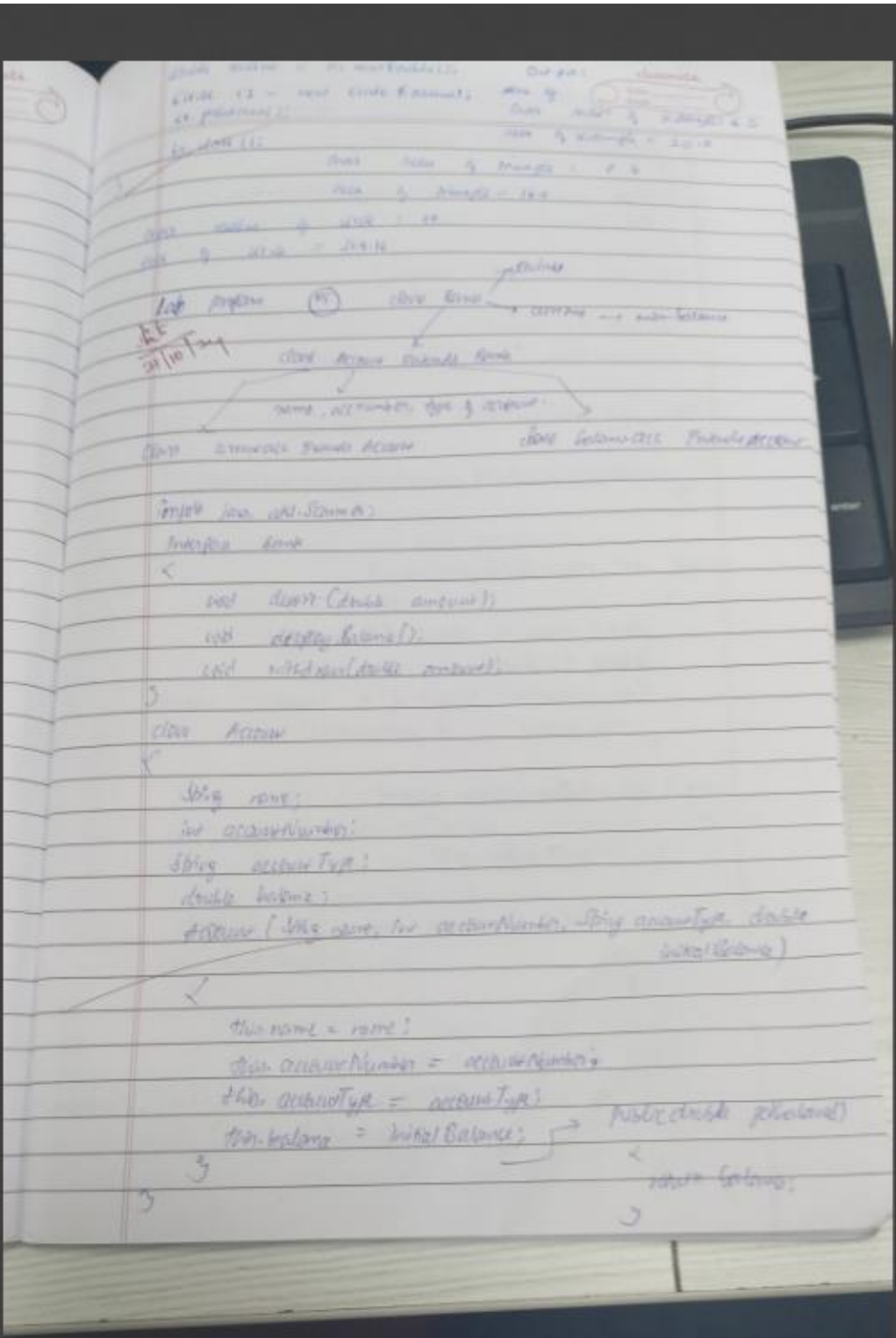
```
}
```



```
PS D:\1bm23cs123> javac Shape.java Main.java
PS D:\1bm23cs123> java Main
enter the sides of rectangle:
4 5
area of rectangle = 20.0
enter the sides of triangle:
8 4
area of triangle = 16.0
enter the radius of circle :
10
area of circle = 314.16
END OF PROGRAM
PS D:\1bm23cs123>
```

PROGRAM 5: BANK ACCOUNT

ALGORITHM:



```
[illegible] = [illegible];                Output:
Circle c2 = new Circle [illegible];       Area of
c1.[illegible]();                         Circle [illegible] of [illegible] = [illegible]
[illegible]();                            Area of Rectangle = [illegible]
}
                 [illegible] Area of Triangle = [illegible]
                 Area of Triangle = [illegible]
[illegible] of circle = [illegible]
[illegible] of circle = [illegible]
```

Lab program (11)

[illegible] 21/10/24

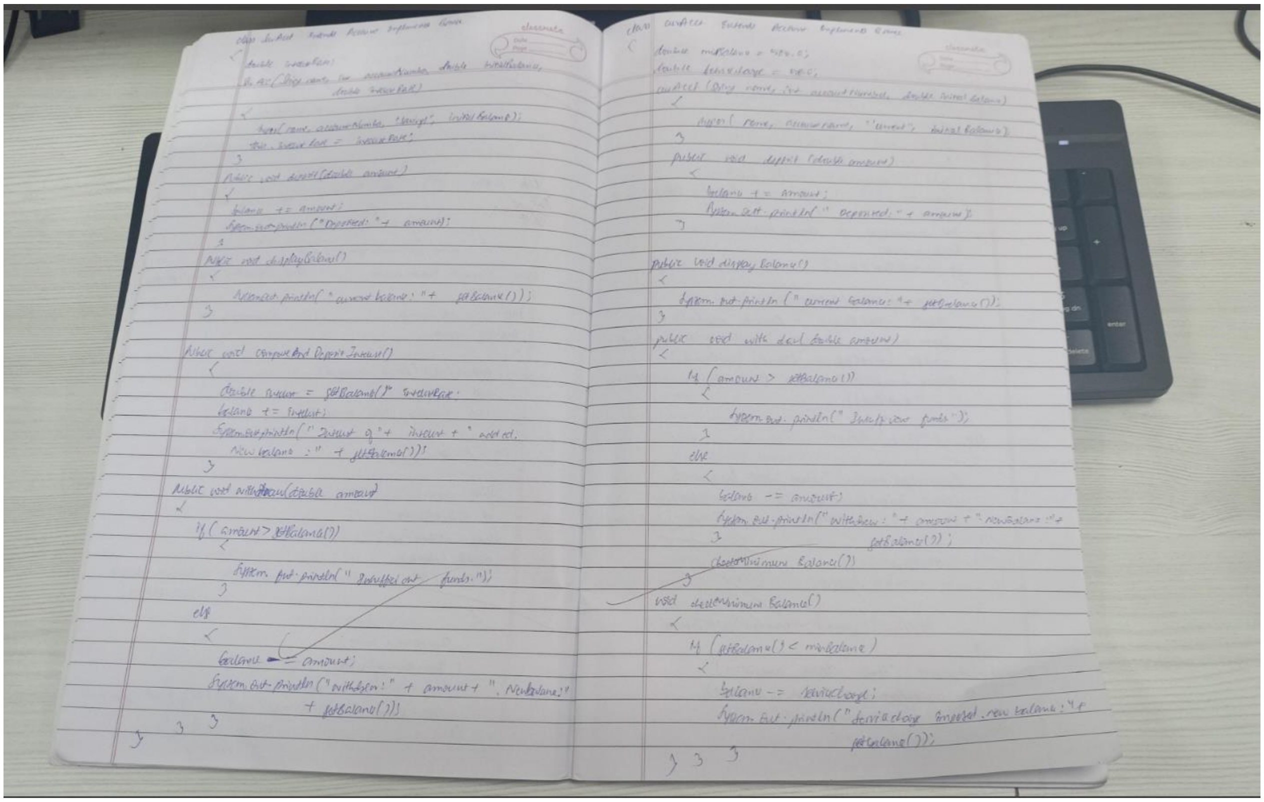
class Bank → deposit, withdraw → [illegible] balance

class Account extends Bank

name, acc number, type of account

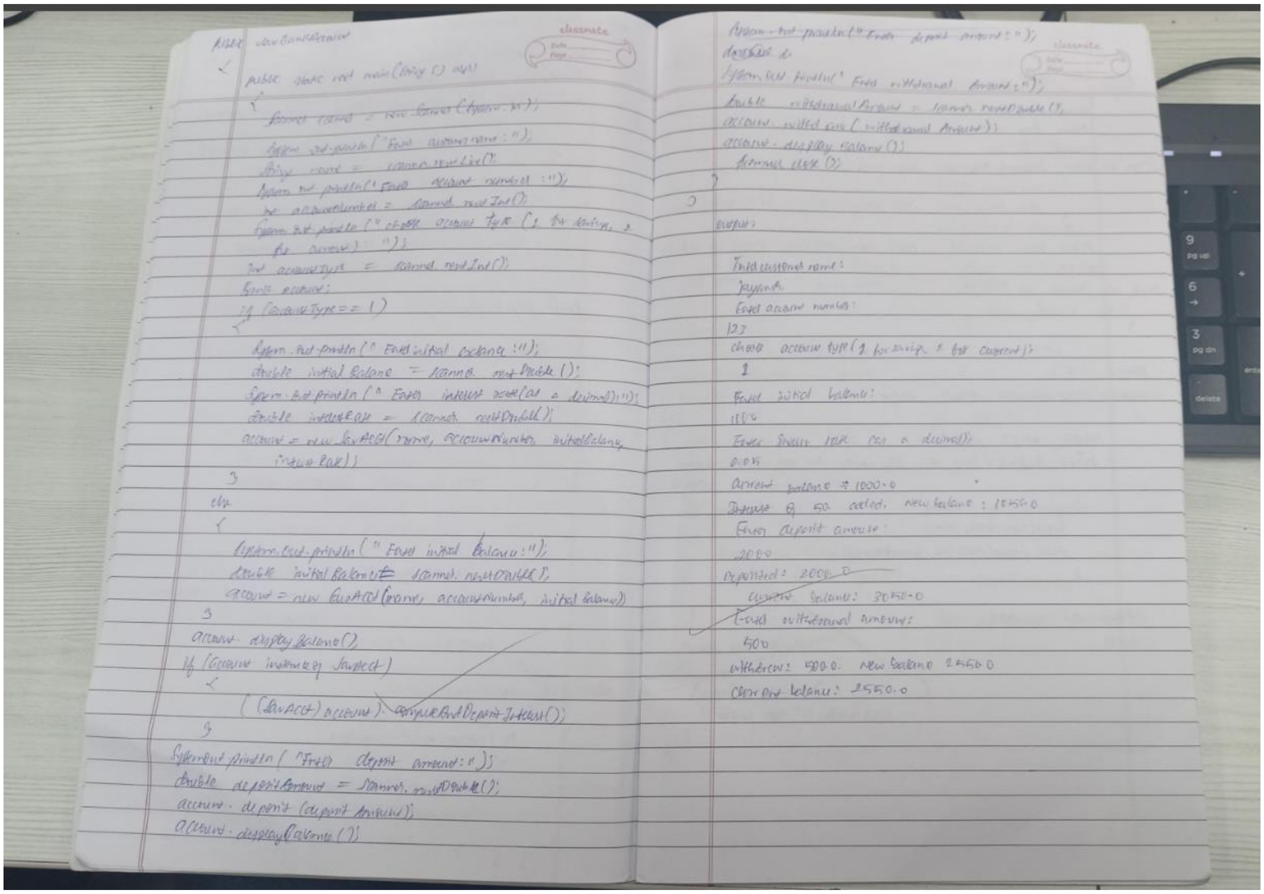
class [illegible] extends Account   class [illegible] extends Account

```
import java.util.Scanner;
interface Bank
{
    void deposit(double amount);
    void displayBalance();
    void withdraw(double amount);
}
class Account
{
  String name;
  int accountNumber;
  String accountType;
  double balance;
  Account (String name, int accountNumber, String accountType, double
                                                     initialBalance)
  {
      this.name = name;
      this.accountNumber = accountNumber;
      this.accountType = accountType;
      this.balance = initialBalance;      public double getBalance()
  }                                       {
}                                           return balance;
                                          }
```



```
class SavAcct extends Account implements Bank
{
    double interestRate;
    SavAcc(String name, int accountNumber, double initialBalance,
                    double interestRate)
    {
        super(name, accountNumber, "Savings", initialBalance);
        this. interestRate = interestRate;
    }
    public void deposit(double amount)
    {
        balance += amount;
        System.out.println("Deposited: "+ amount);
    }
    public void displayBalance()
    {
        System.out.println(" current balance: "+  getBalance());
    }

    public void computeAndDepositInterest()
    {
        double interest = getBalance()* interestRate;
        balance += interest;
        System.out.println(" Interest of "+ interest + " added.
          New balance : " + getBalance());
    }
public void withdraw(double amount
{
    if( amount> getBalance())
    {
        System. out. println(" Insufficient funds.");
    }
    else
    {
        balance -= amount;
        System.out.println("withdrew:" + amount + ". NewBalance:"
                    + getBalance());
    }
  }
}
```

```
class CurAcct Extends Account Implements Bank
{
    double minBalance = 500.0;
    double serviceCharge = 150.0;
    CurAcct (String name, int accountNumber, double initial Balance)
    {
        super( name, accountName, "current", initial Balance);
    }
    public void deposit (double amount)
    {
        balance += amount;
        System. out. println(" deposited: " + amount);
    }

public void display Balance()
{
    System. out. println (" current balance: "+ getBalance());
}
public void withdraw( double amount)
{
    if (amount > getBalance())
    {
        System. out. println(" Insufficient funds");
    }
    else
    {
        balance -= amount;
        System. out. println(" withdrew: " + amount + ". NewBalance :"+
        }                                   getBalance());
        checkMinimum Balance();
    }
void checkMinimum Balance()
{
    if (getBalance() < minBalance)
    {
        balance -= serviceCharge;
        System. out. println(" service charge imposed. new balance: "+
                                      getBalance());
} } }
```



```
public class BankAccount
{
public static void main(String [] args)
{
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter customer name:");
    String name = scanner.nextLine();
    System.out.println("Enter account number:");
    int accountNumber = scanner.nextInt();
    System.out.println("choose account type (1 for savings, 2
      for current): ");
  int accountType = scanner.nextInt();
  Bank account;
  if (accountType == 1)
  {
    System.out.println("Enter initial balance:");
    double initialBalance = scanner.nextDouble();
    System.out.println("Enter interest rate(as a decimal):");
    double interestRate = scanner.nextDouble();
    account = new SavAcct(name, accountNumber, initialBalance,
       interestRate);
  }
  else
  {
    System.out.println("Enter initial Balance:");
    double initialBalance = scanner.nextDouble();
    account = new CurAcct(name, accountNumber, initialBalance);
  }
  account.displayBalance();
  if (account instanceof SavAcct)
  {
     ((SavAcct) account).computeAndDepositInterest();
  }
System.out.println("Enter deposit amount:");
double depositAmount = scanner.nextDouble();
account.deposit(depositAmount);
account.displayBalance();
```

```
~~System.out.println("Enter deposit amount:");~~
~~double d~~
System.out.println("Enter withdrawal Amount:");
double withdrawalAmount = scanner.nextDouble();
account.withdraw(withdrawalAmount);
account.display Balance();
 scanner.close();
}
}
```

output:

```
Enter customer name:
 jayanth
Enter account number:
123
choose account type(1 for savings, 2 for current):
 1
Enter initial balance:
1000
Enter interest rate (as a decimal):
0.05
Current balance : 1000.0
Interest of 50 added. New balance : 1050.0
 Enter deposit amount:
 2000
Deposited: 2000.0
  current balance: 3050.0
 Enter withdrawal amount:
 500
withdrew: 500.0. New balance 2550.0
Current balance: 2550.0
```

CODE : import java.util.Scanner;

```
interface Bank
{
   void deposit(double amount);
   void displayBalance();
   void withdraw(double amount);
}

class Account
{
   String name;
   int accountNumber;
```

```
    String accountType;
    double balance;

    Account(String name, int accountNumber, String accountType, double initialBalance)
    {
        this.name = name;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = initialBalance;
    }

    // Method to get the balance
    public double getBalance()
    {
        return balance;
    }
}

class SavAcct extends Account implements Bank
{
    double interestRate;

    SavAcct(String name, int accountNumber, double initialBalance, double interestRate)
    {
        super(name, accountNumber, "Savings", initialBalance);
        this.interestRate = interestRate;
    }

    public void deposit(double amount)
```

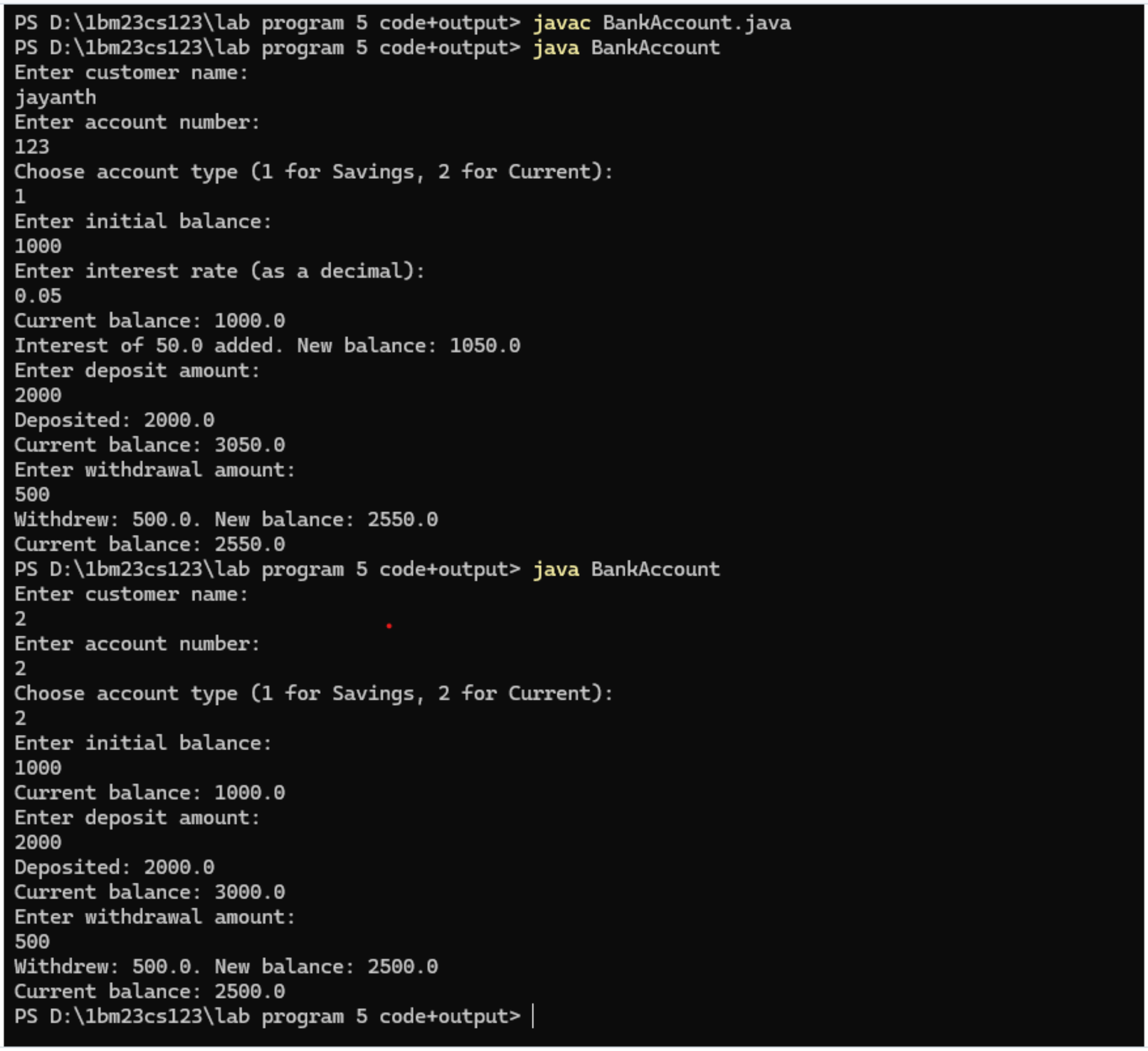
```
{
    balance += amount;
    System.out.println("Deposited: " + amount);
}

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

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

public void withdraw(double amount)
{
    if (amount > getBalance())
    {
        System.out.println("Insufficient funds.");
    }
    else
    {
        balance -= amount;
        System.out.println("Withdrew: " + amount + ". New balance: " + getBalance());
    }
}
```

```
}

class CurAcct extends Account implements Bank
{
   double minBalance = 500.0;
   double serviceCharge = 50.0;

   CurAcct(String name, int accountNumber, double initialBalance)
   {
      super(name, accountNumber, "Current", initialBalance);
   }

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

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

   public void withdraw(double amount)
   {
      if (amount > getBalance())
      {
         System.out.println("Insufficient funds.");
      }
```

```
        else
        {
            balance -= amount;
            System.out.println("Withdrew: " + amount + ". New balance: " + getBalance());
        }
        checkMinimumBalance();
    }

    void checkMinimumBalance()
    {
        if (getBalance() < minBalance)
        {
            balance -= serviceCharge;
            System.out.println("Service charge imposed. New balance: " + getBalance());
        }
    }
}

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

        System.out.println("Enter customer name: ");
        String name = scanner.nextLine();

        System.out.println("Enter account number: ");
        int accountNumber = scanner.nextInt();
```

```
System.out.println("Choose account type (1 for Savings, 2 for Current): ");
int accountType = scanner.nextInt();


Bank account;
if (accountType == 1)
{
   System.out.println("Enter initial balance: ");
   double initialBalance = scanner.nextDouble();
   System.out.println("Enter interest rate (as a decimal): ");
   double interestRate = scanner.nextDouble();
   account = new SavAcct(name, accountNumber, initialBalance, interestRate);
}
else
{
   System.out.println("Enter initial balance: ");
   double initialBalance = scanner.nextDouble();
   account = new CurAcct(name, accountNumber, initialBalance);
}


account.displayBalance();


// Immediate interest calculation for savings account
if (account instanceof SavAcct)
{
   ((SavAcct) account).computeAndDepositInterest();
}


System.out.println("Enter deposit amount: ");
```

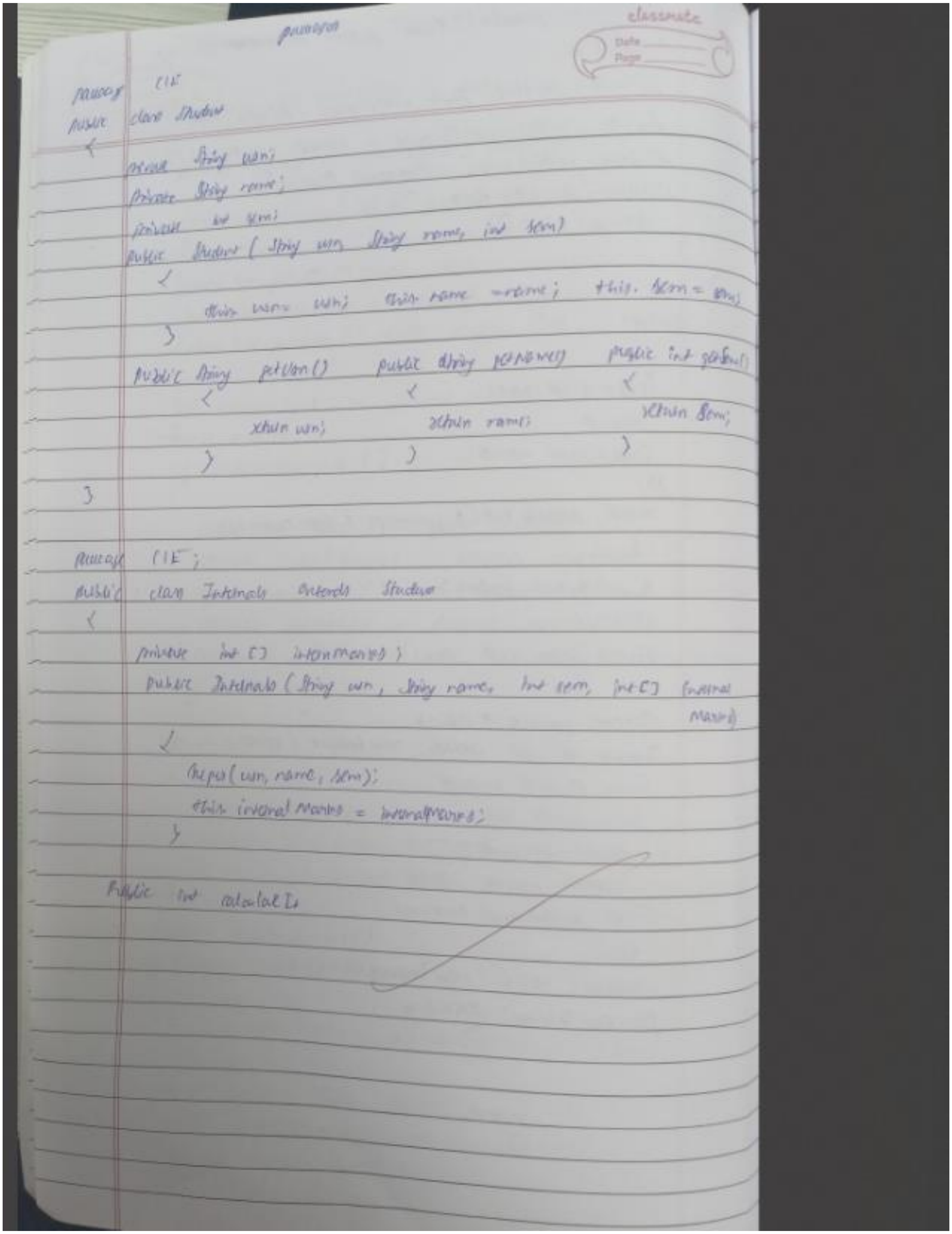
```
        double depositAmount = scanner.nextDouble();
        account.deposit(depositAmount);
        account.displayBalance();

        System.out.println("Enter withdrawal amount: ");
        double withdrawalAmount = scanner.nextDouble();
        account.withdraw(withdrawalAmount);
        account.displayBalance();

        scanner.close();
    }
}
```



```
PS D:\1bm23cs123\lab program 5 code+output> javac BankAccount.java
PS D:\1bm23cs123\lab program 5 code+output> java BankAccount
Enter customer name:
jayanth
Enter account number:
123
Choose account type (1 for Savings, 2 for Current):
1
Enter initial balance:
1000
Enter interest rate (as a decimal):
0.05
Current balance: 1000.0
Interest of 50.0 added. New balance: 1050.0
Enter deposit amount:
2000
Deposited: 2000.0
Current balance: 3050.0
Enter withdrawal amount:
500
Withdrew: 500.0. New balance: 2550.0
Current balance: 2550.0
PS D:\1bm23cs123\lab program 5 code+output> java BankAccount
Enter customer name:
2
Enter account number:
2
Choose account type (1 for Savings, 2 for Current):
2
Enter initial balance:
1000
Current balance: 1000.0
Enter deposit amount:
2000
Deposited: 2000.0
Current balance: 3000.0
Enter withdrawal amount:
500
Withdrew: 500.0. New balance: 2500.0
Current balance: 2500.0
PS D:\1bm23cs123\lab program 5 code+output>
```

PROGRAM 6: PACKAGES

ALGORITHM :



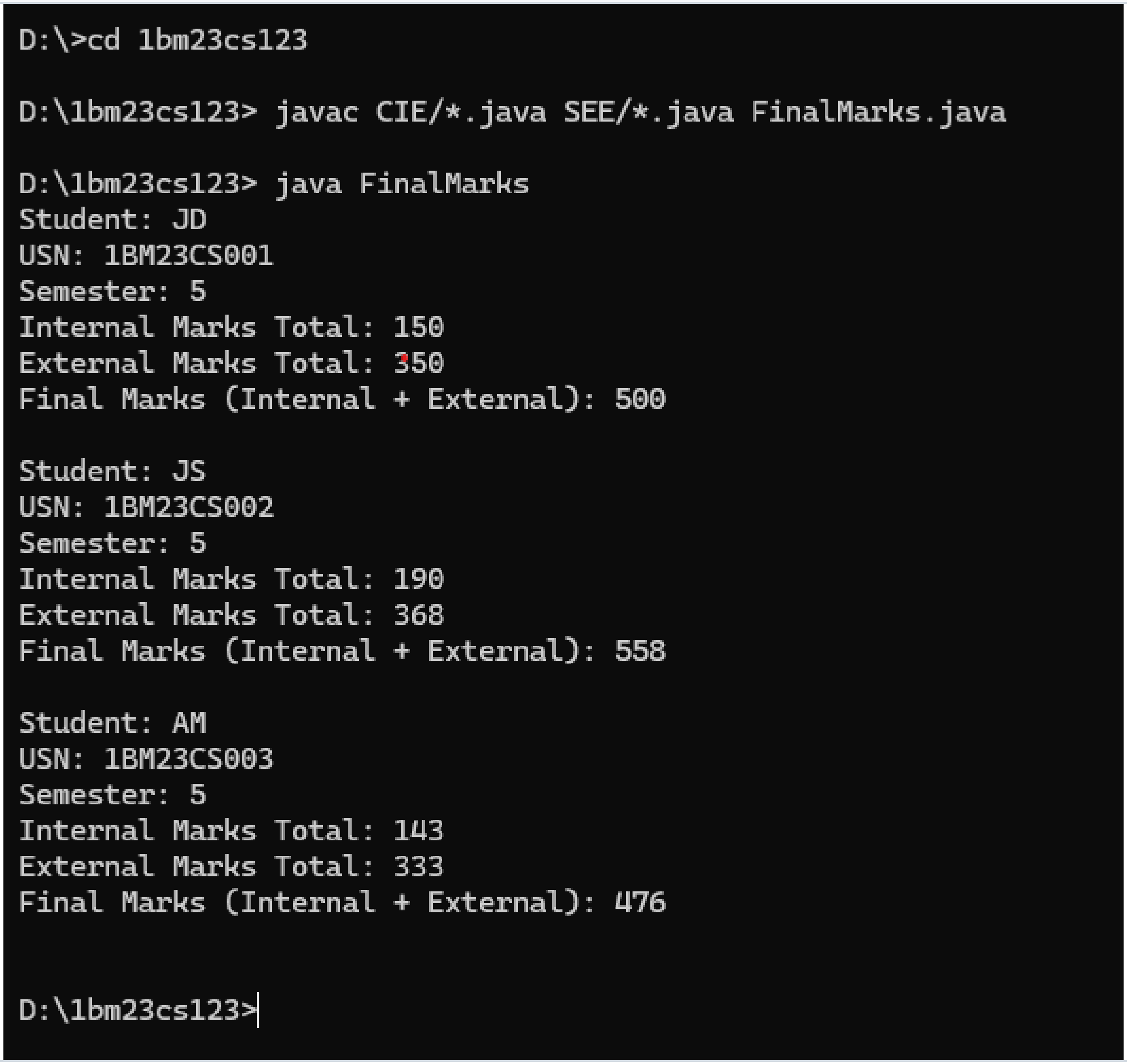
```
                    [illegible]
package   CIE
public  class Student
{
    private String usn;
    private String name;
    private  int sem;
    public  Student ( String usn, String name, int sem)
      {
          this. usn= usn;   this. name =name;   this. sem = sem;
      }
    public String  getUsn()     public String getName()     public int getSem()
         {                          {                          {
              return usn;              return name;              return Sem;
         }                          }                          }
}

package  CIE ;
public  class  Internals  extends  Student
{
    private  int []  internMarks )
    public Internals ( String usn, String name,  int sem, int[] internal
                                                                  Marks)
      {
          super(usn, name, sem);
          this. internal Marks = internalMarks;
      }

  public  int  calculateI
```

CODE: package CIE;

```
package CIE;

public class Internals extends Student {
    private int[] internalMarks;

    // Constructor to initialize internal marks and student details
    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);  // Calling the parent constructor
        this.internalMarks = internalMarks;
    }

    // Method to calculate the total internal marks
    public int calculateInternalTotal() {
        int total = 0;
        for (int i = 0; i < internalMarks.length; i++) {
            total += internalMarks[i];
        }
        return total;
    }

    // Get the internal marks array
    public int[] getInternalMarks() {
        return internalMarks;
    }
}
package CIE;

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

    // Constructor
    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }

    // Getters
    public String getUsn() {
        return usn;
    }

    public String getName() {
        return name;
    }

    public int getSem() {
        return sem;
    }
}
package SEE;

import CIE.Student;

public class External extends Student {
    private int[] externalMarks;


    // Constructor to initialize external marks and student details
    public External(String usn, String name, int sem, int[] externalMarks) {
        super(usn, name, sem);  // Calling the parent constructor
        this.externalMarks = externalMarks;
    }


    // Method to calculate the total external marks
    public int calculateExternalTotal() {
        int total = 0;
        for (int i = 0; i < externalMarks.length; i++) {
            total += externalMarks[i];
        }
        return total;
    }


    // Get the external marks array
    public int[] getExternalMarks() {
        return externalMarks;
    }
}
import CIE.Internals;
import SEE.External;


public class FinalMarks {
    public static void main(String[] args) {
        int[] internalMarks1 = {20, 30, 40, 35, 25};
        int[] externalMarks1 = {60, 70, 75, 80, 65};
        int[] internalMarks2 = {22, 33, 45, 50, 40};
        int[] externalMarks2 = {65, 68, 78, 72, 85};

        int[] internalMarks3 = {19, 25, 38, 33, 28};
        int[] externalMarks3 = {58, 65, 70, 80, 60};

        Internals[] internalsStudents = new Internals[3];
        External[] externalStudents = new External[3];

        internalsStudents[0] = new Internals("1BM23CS001", "JD", 5, internalMarks1);
        externalStudents[0] = new External("1BM23CS001", "JD", 5, externalMarks1);

        internalsStudents[1] = new Internals("1BM23CS002", "JS", 5, internalMarks2);
        externalStudents[1] = new External("1BM23CS002", "JS", 5, externalMarks2);

        internalsStudents[2] = new Internals("1BM23CS003", "AM", 5, internalMarks3);
        externalStudents[2] = new External("1BM23CS003", "AM", 5, externalMarks3);

        for (int i = 0; i < 3; i++) {
            displayFinalMarks(internalsStudents[i], externalStudents[i]);
        }
    }

    public static void displayFinalMarks(Internals internals, External external) {
        int internalTotal = internals.calculateInternalTotal();
        int externalTotal = external.calculateExternalTotal();
        int finalTotal = internalTotal + externalTotal;
```

```
        System.out.println("Student: " + internals.getName());
        System.out.println("USN: " + internals.getUsn());
        System.out.println("Semester: " + internals.getSem());
        System.out.println("Internal Marks Total: " + internalTotal);
        System.out.println("External Marks Total: " + externalTotal);
        System.out.println("Final Marks (Internal + External): " + finalTotal);
        System.out.println();
    }
}
```

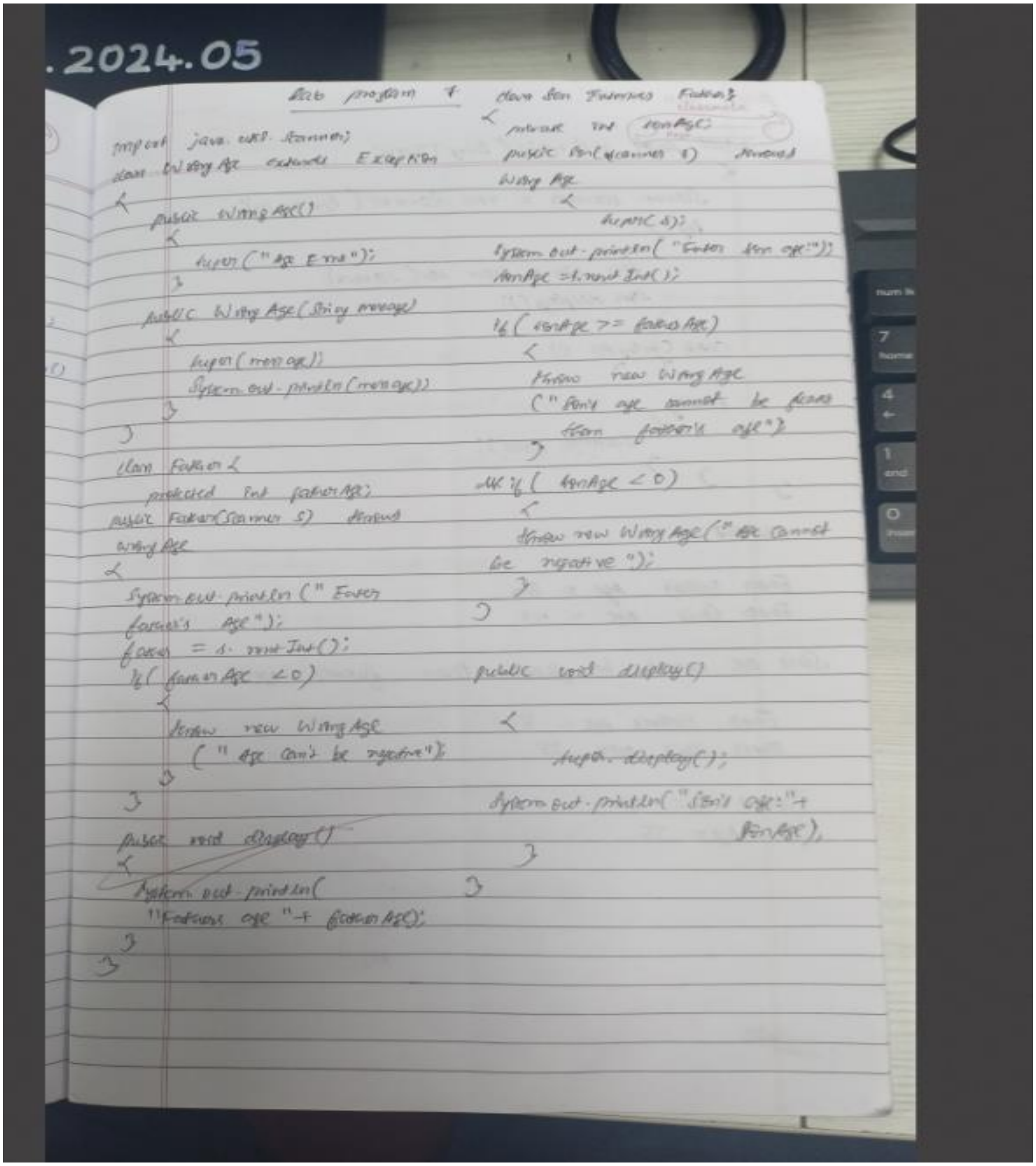
Pr d



```
D:\>cd 1bm23cs123

D:\1bm23cs123> javac CIE/*.java SEE/*.java FinalMarks.java

D:\1bm23cs123> java FinalMarks
Student: JD
USN: 1BM23CS001
Semester: 5
Internal Marks Total: 150
External Marks Total: 350
Final Marks (Internal + External): 500

Student: JS
USN: 1BM23CS002
Semester: 5
Internal Marks Total: 190
External Marks Total: 368
Final Marks (Internal + External): 558

Student: AM
USN: 1BM23CS003
Semester: 5
Internal Marks Total: 143
External Marks Total: 333
Final Marks (Internal + External): 476


D:\1bm23cs123>
```

PROGRAM 7 : HANDLING OF EXCEPTION IN INHERITANCE TREE

ALGORITHM :

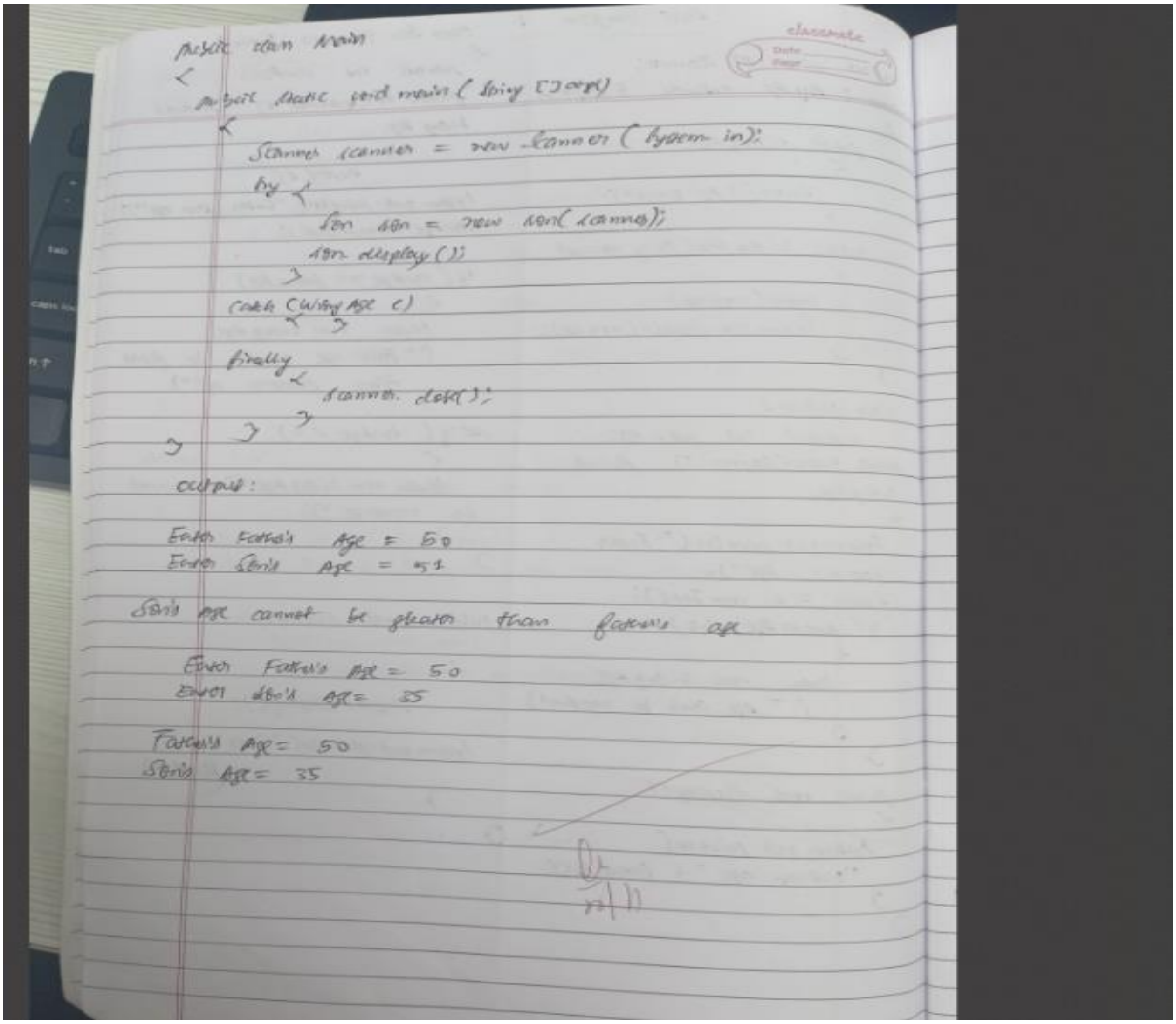


.2024.05

Lab program 7

```
import java.util.Scanner;
class WrongAge extends Exception
{
   public WrongAge()
     {
       super("Age Error");
     }
   public WrongAge(String message)
     {
       super(message);
       System.out.println(message);
     }
}
class Father{
    protected int fatherAge;
public Father(Scanner s) throws
WrongAge
{
  System.out.println(" Enter
  father's Age");
  father = s.nextInt();
   if( fatherAge <0)
     {
       throw new WrongAge
        (" Age can't be negative");
     }
 }
public void display()
{
  System.out.println(
   "Fathers age "+ fatherAge);
 }
}
```

```
class Son extends Father{
{ private int sonAge;
  public Son(Scanner s) throws
  WrongAge
      {
          super(s);
System.out.println( "Enter son age:");
sonAge =s.nextInt();

if( sonAge >= fatherAge)
   {
     throw new WrongAge
     (" Son's age cannot be greater
        than father's age");
   }
else if( sonAge < 0)
   {
     throw new WrongAge(" Age cannot
be negative");
   }
}

public void display()

  {
       super.display();
System.out.println(" Son's age:"+
                      sonAge);
   }
}
```



```
public class Main
{
 public static void main ( String [] args)
  {
    Scanner scanner = new Scanner ( System.in);
    try {
          Son son = new Son( scanner);
          son.display ();
       }
   catch (WrongAge e)
       { }
   finally
       {
          scanner. close();
       }
   }
}
```

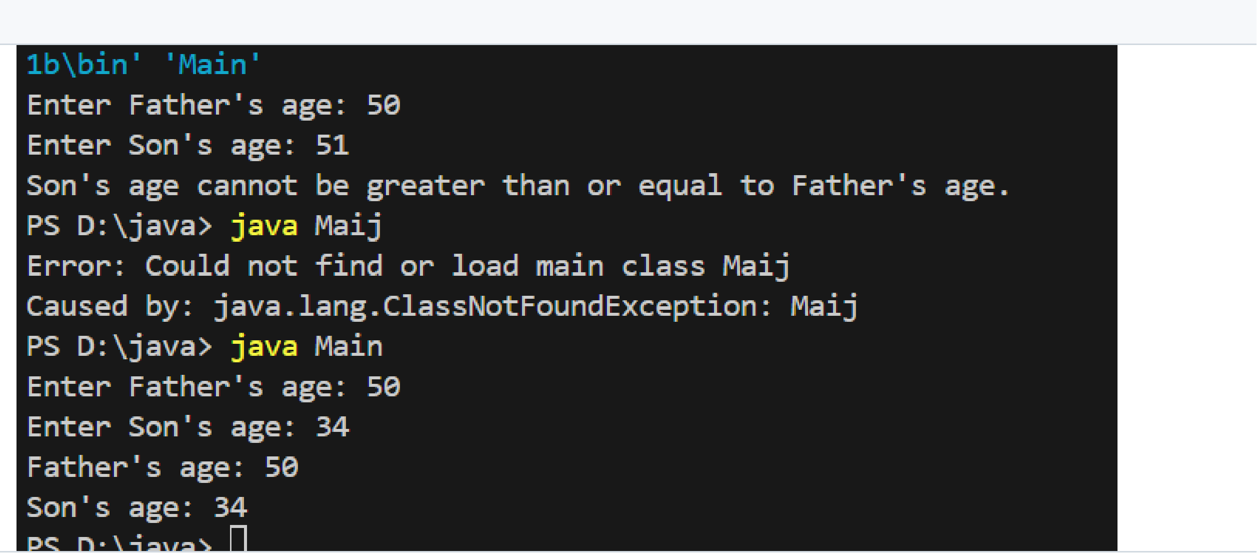
output :

```
Enter Father's  Age = 50
Enter Son's  Age = 51

Son's age cannot be greater than father's age

 Enter Father's Age = 50
 Enter Son's Age= 35

Father's Age= 50
Son's Age= 35
```

CODE :

```
import java.util.Scanner;



class WrongAge extends Exception {


    public WrongAge() {
        super("Age Error");
    }

    public WrongAge(String message) {
        super(message);
        System.out.println(message);
    }
}

class Father {
    protected int fatherAge;

    public Father(Scanner s) throws WrongAge {
        System.out.print("Enter Father's age: ");
        fatherAge = s.nextInt();

        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative.");
        }
    }

    public void display() {
        System.out.println("Father's age: " + fatherAge);
    }
}
```

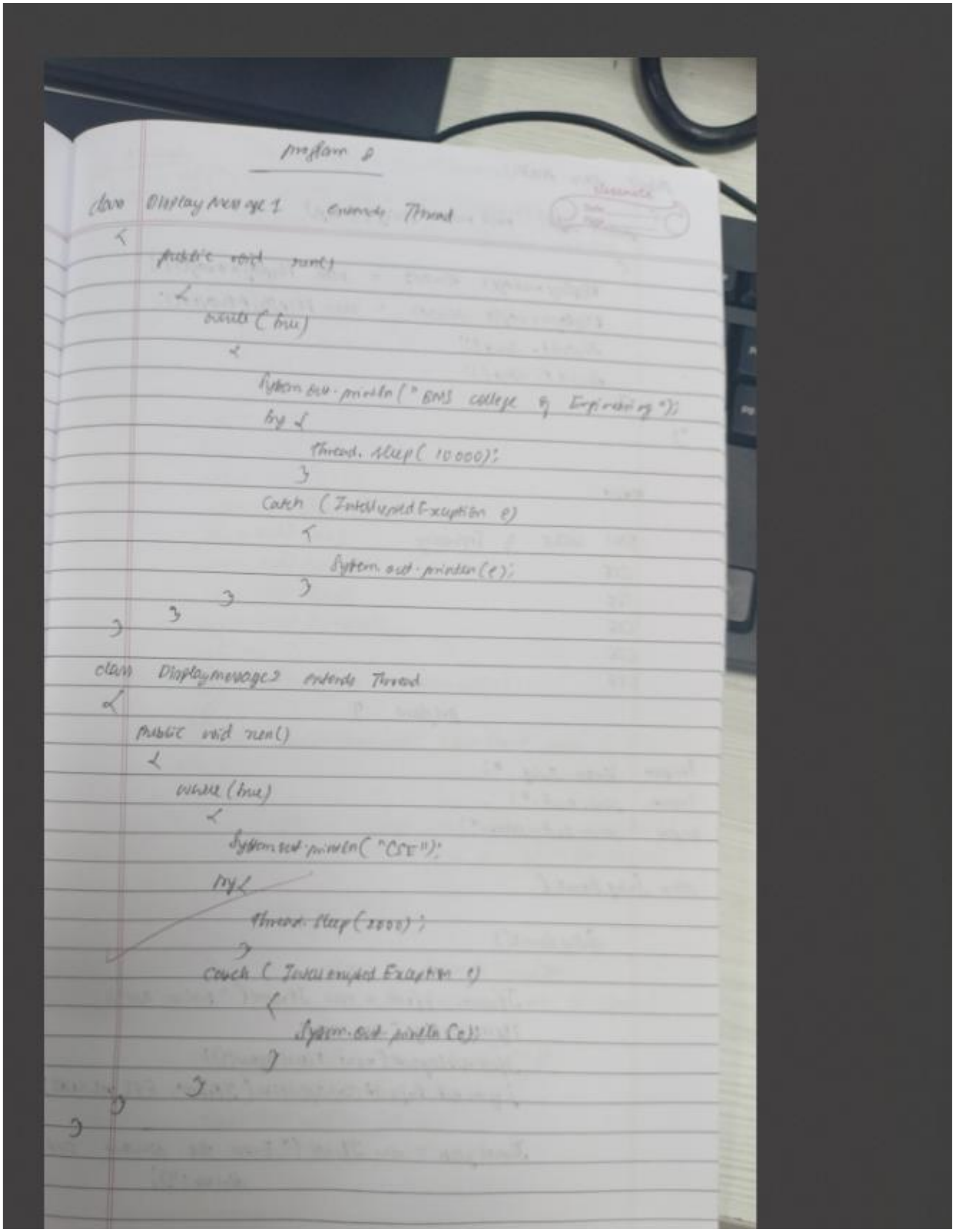
```
class Son extends Father {
    private int sonAge;



    public Son(Scanner s) throws WrongAge {
        super(s);
        System.out.print("Enter Son's age: ");
        sonAge = s.nextInt();

        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to Father's age.");
        } else if (sonAge < 0) {
            throw new WrongAge("Age cannot be negative.");
        }
    }



    public void display() {
        super.display();
        System.out.println("Son's age: " + sonAge);
    }
}



public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            Son son = new Son(scanner);
            son.display();
        } catch (WrongAge e) {

        } finally {
            scanner.close();
    }
}}
```



```
1b\bin' 'Main'
Enter Father's age: 50
Enter Son's age: 51
Son's age cannot be greater than or equal to Father's age.
PS D:\java> java Maij
Error: Could not find or load main class Maij
Caused by: java.lang.ClassNotFoundException: Maij
PS D:\java> java Main
Enter Father's age: 50
Enter Son's age: 34
Father's age: 50
Son's age: 34
PS D:\java>
```

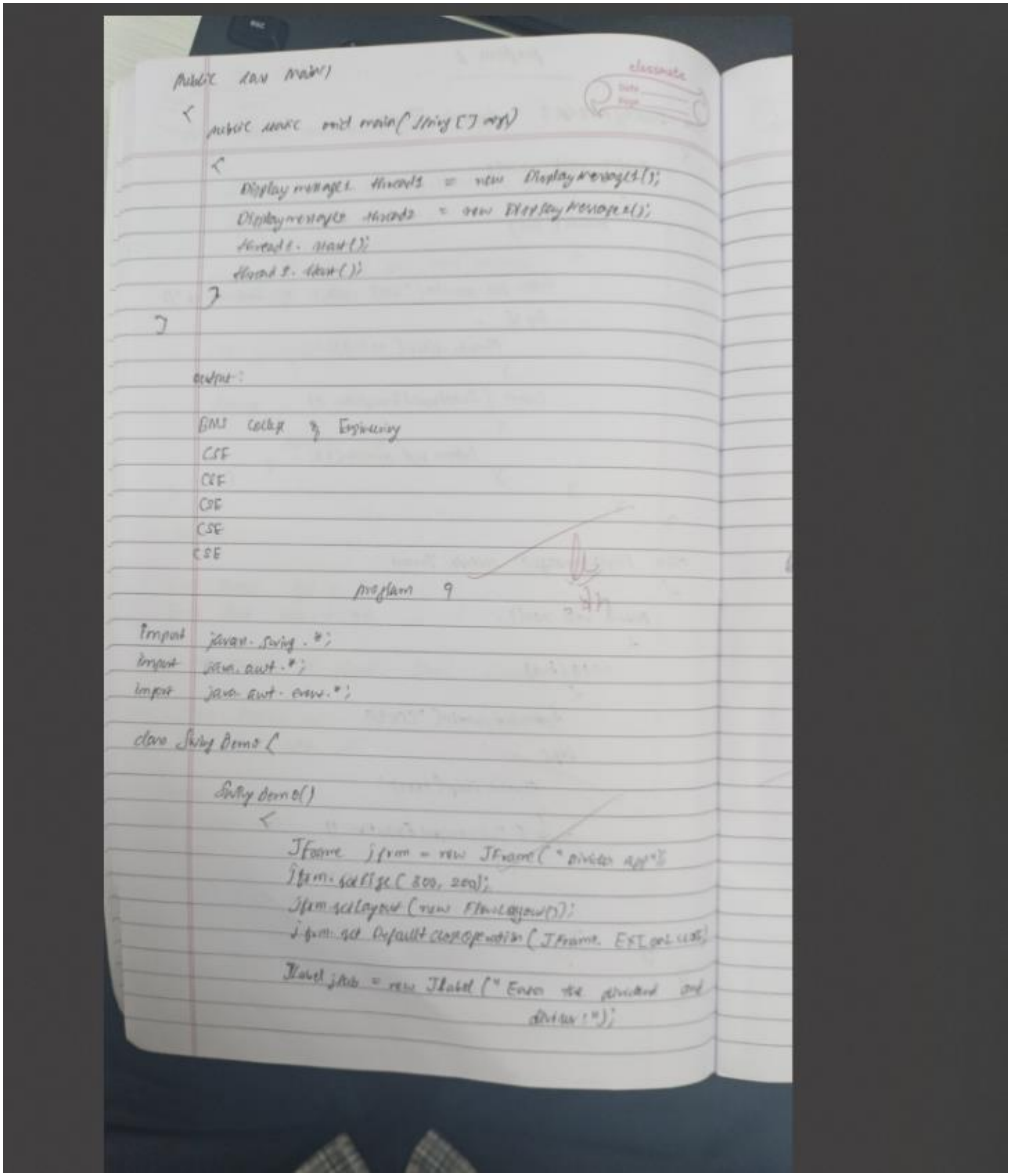
PROGRAM 8: PRINT BMSCE AND CSE (USING SLEEP).

ALGORITHM:



program 8

```
class DisplayMessage1   extends Thread
<
    public void run()
    <
        while ( true)
        <
            System.out.println(" BMS college of Engineering ");
            try <
                Thread. sleep( 10000);
                }
            Catch (InterruptedException e)
                <
                System.out.println(e);
                }
        }
    }
}

class DisplayMessage2  extends Thread
<
   public void run()
    <
      while (true)
        <
          System.out.println( "CSE");
          try<
              Thread.sleep(2000);
          }
          catch ( InterruptedException e)
              <
              System.out.println(e);
              }
        }
    }
}
```



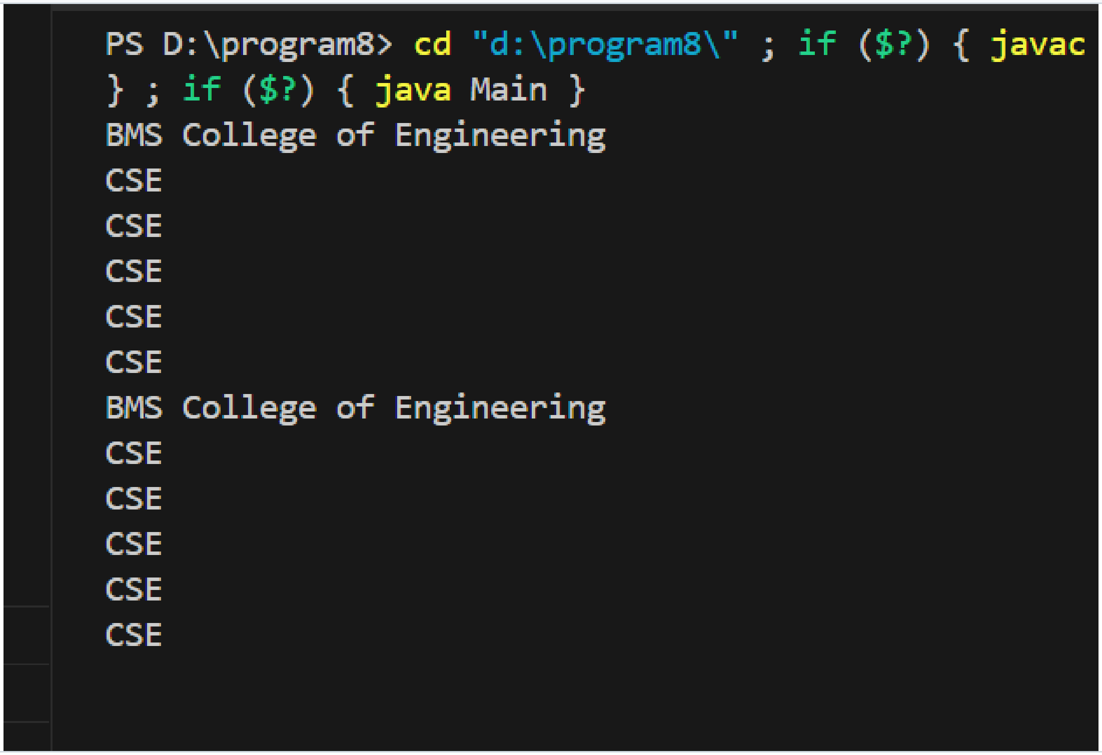
```
public class main()
{
    public static void main(String [] args)
    {
        Displaymessage1 thread1 = new Displaymessage1();
        Displaymessage2 thread2 = new Displaymessage2();
        thread1. start();
        thread2. start();
    }
}
```

output:

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

program 9

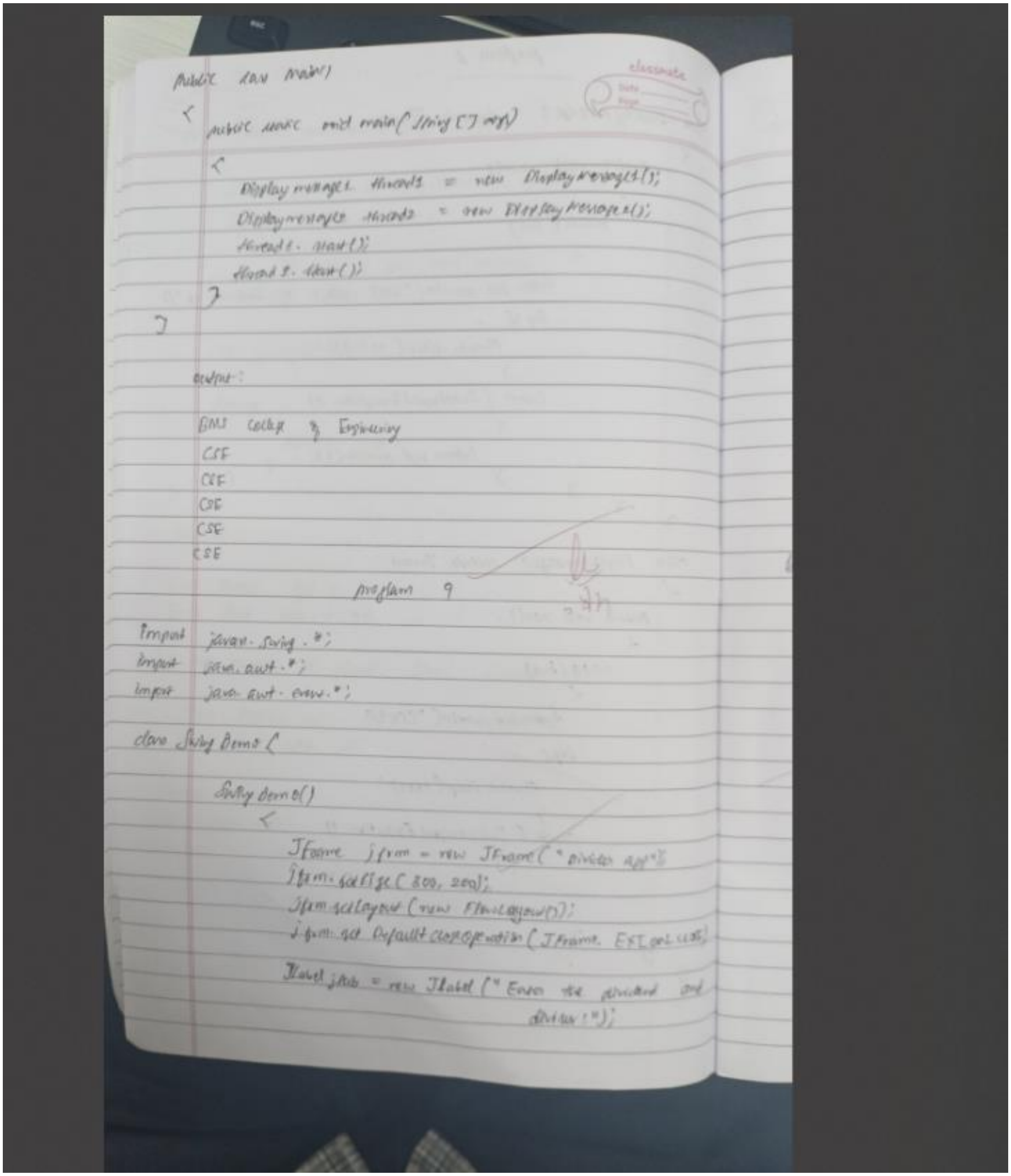
```
import javax. swing .*;
import java. awt .*;
import java. awt. event.*;

class SwingDemo {

    SwingDemo()
    {
        JFrame jfrm = new JFrame(" Divider App");
        jfrm. setSize( 300, 200);
        jfrm. setLayout (new FlowLayout());
        jfrm. set DefaultCloseOperation( JFrame. EXIT_ON_CLOSE);

        JLabel jlab = new JLabel (" Enter the dividend and
                                   divisor :"));
```



```
PS D:\program8> cd "d:\program8\" ; if ($?) { javac
} ; if ($?) { java Main }
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
```

PROGRAM 9 : UI TO PERORM INTEGER DIVISION

ALGORITHM :



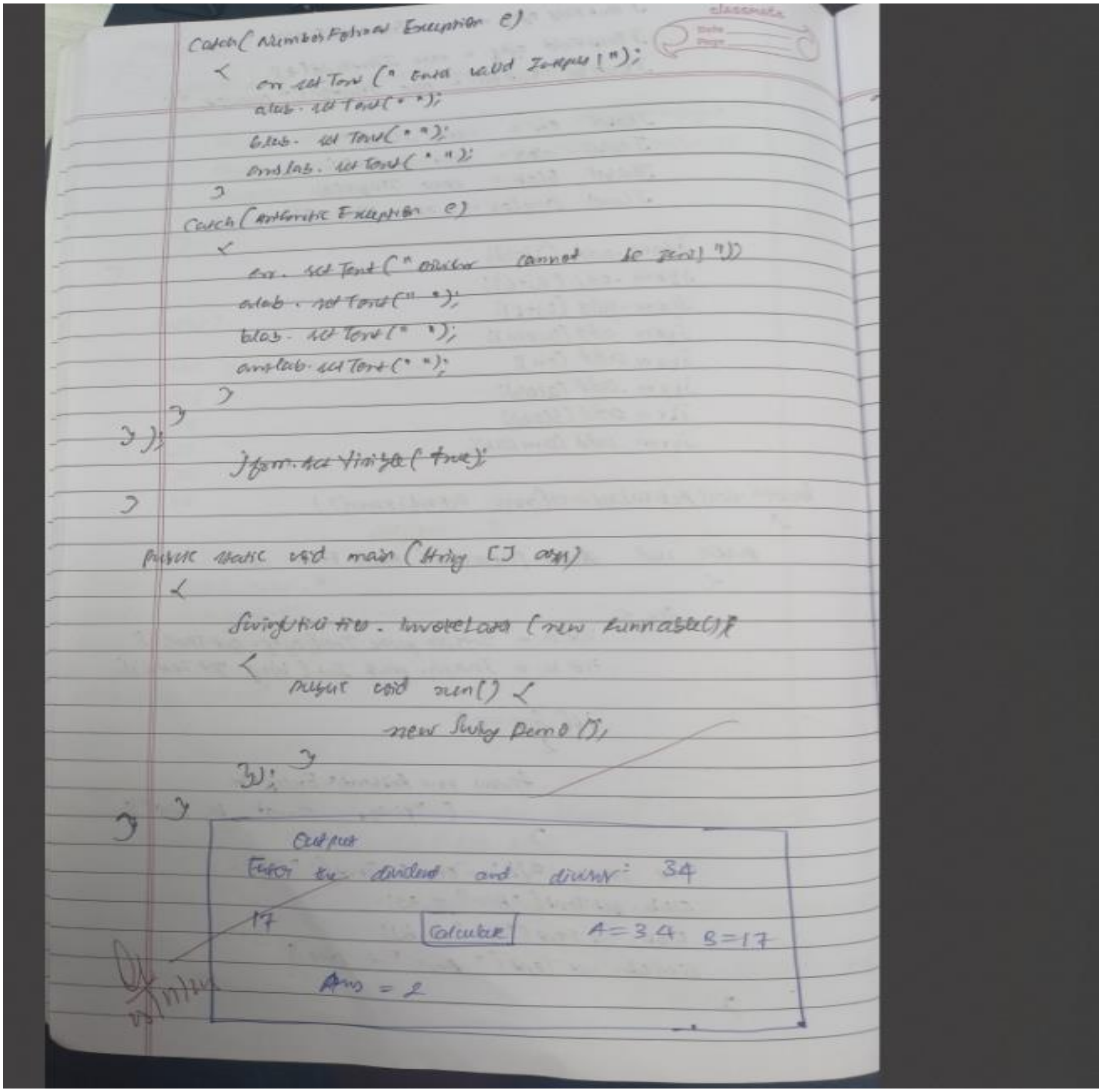
```
public class main()
 {
   public static void main(String [] args)
   {
      Displaymessage1 thread1 = new Displaymessage1();
      Displaymessage2 thread2 = new Displaymessage2();
      thread1.start();
      thread2.start();
   }
}
```

output:

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

program 9

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

class SwingDemo {

    SwingDemo()
        {
            JFrame jfrm = new JFrame("Divider App");
            jfrm.setSize(300, 200);
            jfrm.setLayout(new FlowLayout());
            jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

            JLabel jlab = new JLabel("Enter the dividend and
                                     divisor:");
```



```
Catch( NumberFormat Exception e)
  {
      err.setText ("Enter valid Integer !");
      alab.setText(" ");
      blab.setText(" ");
      anslab.setText(" ");
  }
Catch(Arithmetic Exception e)
  {
      err.setText("Divisor cannot be zero!"))
      alab.setText(" ");
      blab.setText(" ");
      anslab.setText(" ");
  }
 }
});
      jform.setVisible(true);
}

public static void main(String [] args)
  {
      SwingUtilities.invokeLater(new Runnable(){
      {
          public void run() {
                  new SwingDemo();
      }});
  }
}
```

```
Output
Enter the dividend and divisor: 34
17              Calculate       A=34  B=17
      Ans = 2
```

CODE :

import javax.swing.*;

import java.awt.*;

import java.awt.event.*;

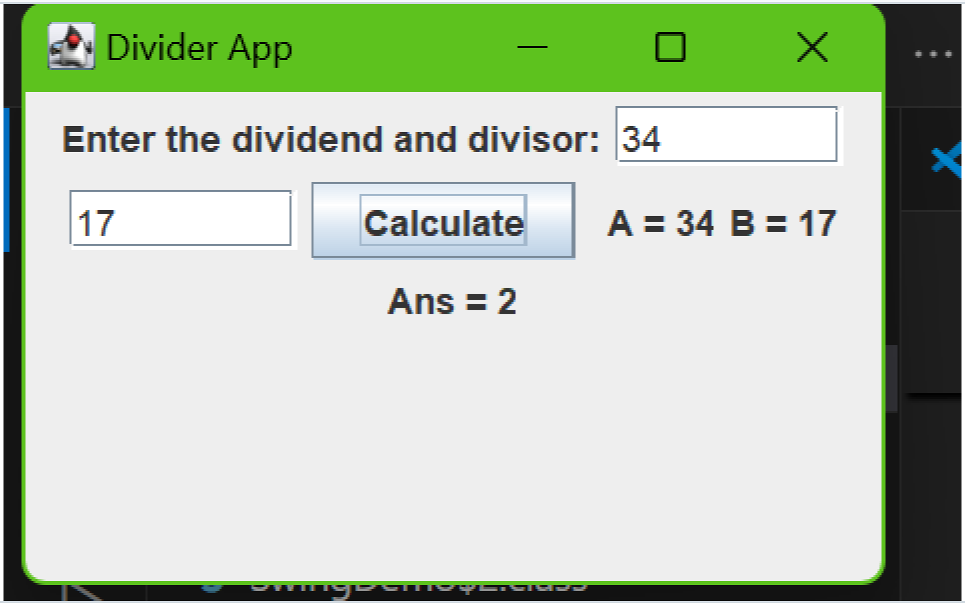
class SwingDemo {

SwingDemo() {

```
// Create JFrame container
JFrame jfrm = new JFrame("Divider App");
jfrm.setSize(300, 200);
jfrm.setLayout(new FlowLayout());

// Terminate program on close
jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

// Create components
JLabel jlab = new JLabel("Enter the dividend and divisor:");
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
JButton button = new JButton("Calculate");

JLabel err = new JLabel(); // Error label
JLabel alab = new JLabel(); // A value label
JLabel blab = new JLabel(); // B value label
JLabel anslab = new JLabel(); // Result label

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

```
// Add action listener for the button
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        try {
            int a = Integer.parseInt(ajtf.getText());
            int b = Integer.parseInt(bjtf.getText());

            // Check for division by zero
            if (b == 0) {
                throw new ArithmeticException("Divisor cannot be zero.");
            }

            int ans = a / b;
            err.setText(""); // Clear error messages
            alab.setText("A = " + a);
            blab.setText("B = " + b);
            anslab.setText("Ans = " + ans);
        } catch (NumberFormatException e) {
            err.setText("Enter valid integers!");
            alab.setText("");
            blab.setText("");
            anslab.setText("");
        } catch (ArithmeticException e) {
            err.setText("Divisor cannot be zero!");
            alab.setText("");
            blab.setText("");
            anslab.setText("");
        }
    }
```

```
        });

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

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



Enter the dividend and divisor: 34

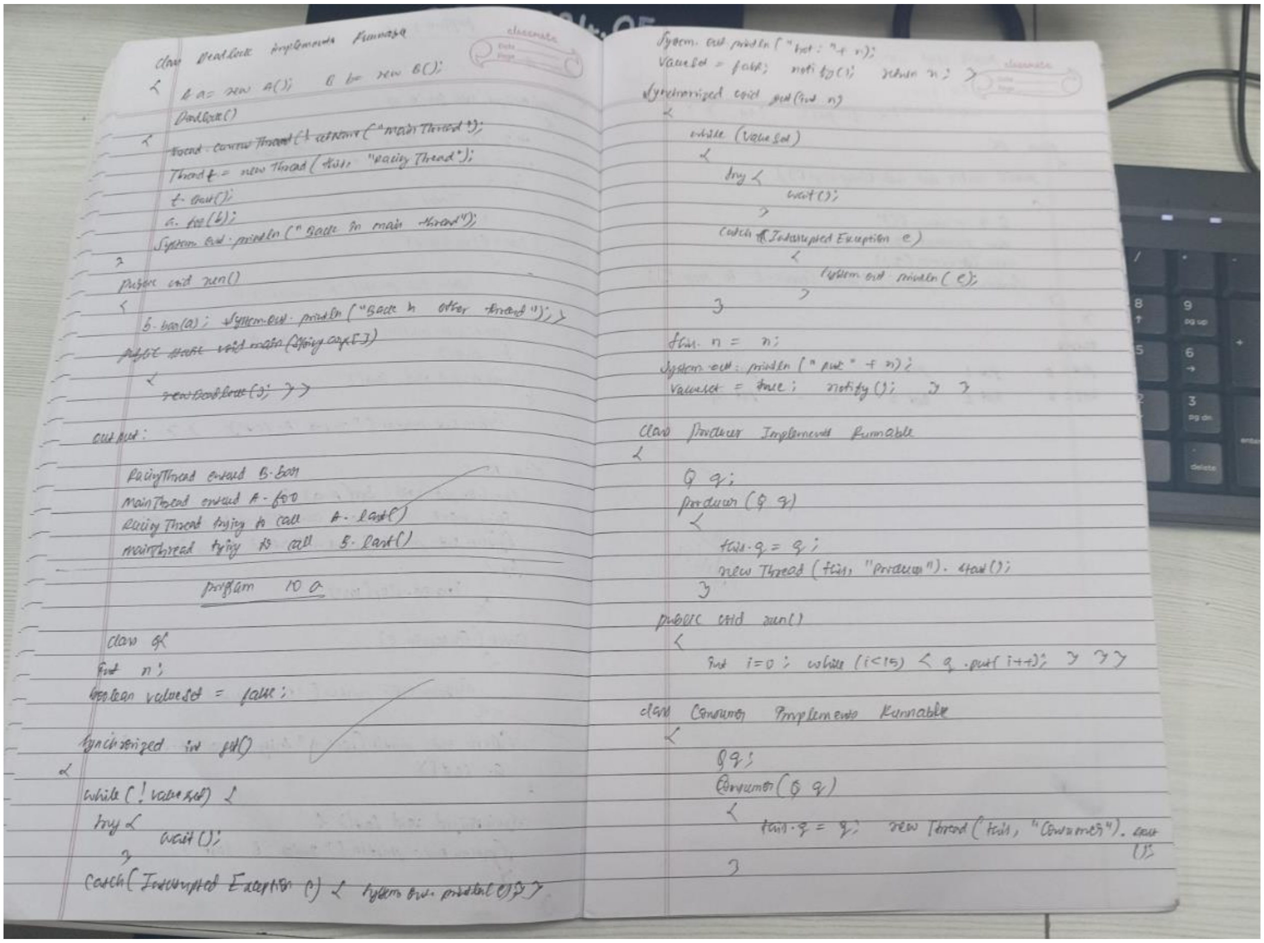
17

A = 34 B = 17

Ans = 2

PROGRAM 10 A: PRODUCER AND CONSUMER

ALGORITHM :

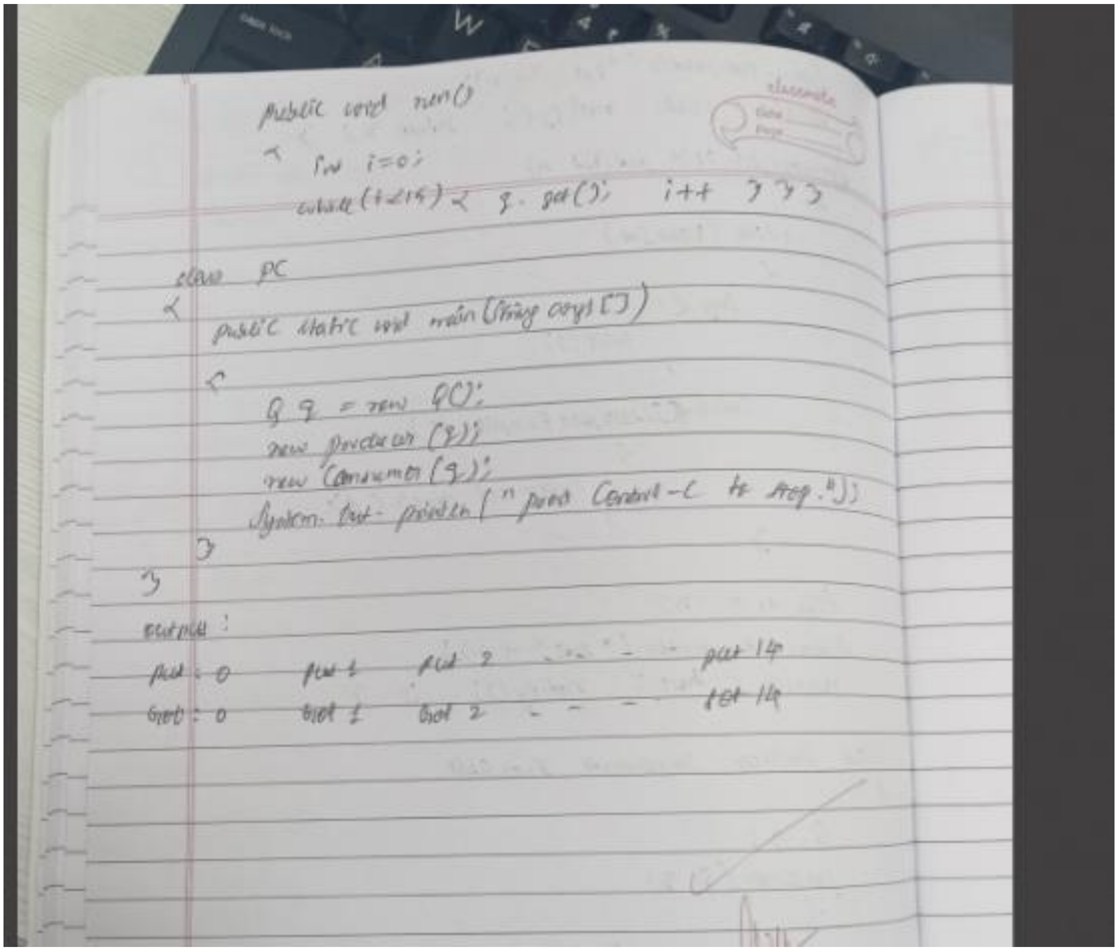


```
class Deadlock implements Runnable
{ A a= new A();   B b= new B();
Deadlock()
{ ~~Thread.currentThread~~().setName("main Thread");
Thread t = new Thread( this, "Racing Thread");
t. start();
a. foo(b);
System.out.println(" Back in main thread");
}
public void run()
{
b.bar(a); System.out.println("Back in other thread"); }
~~public static void main(String args[])~~
{
~~new Deadlock(); }}~~

output:

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

program 10 a

```
class Q{
int n;
boolean valueset = false;

synchronized int get()
{
while (! valueset) {
try {
     wait();
}
catch( InterruptedException e) { System.out.println(e);} }
```

```
System.out.println("Got: "+ n);
valueSet = false; notify(); return n; }
synchronized void put(int n)
{
   while (valueSet)
   {
      try {
            wait();
          }
      catch (InterruptedException e)
          {
             System.out.println( e);
          }
   }

this.n = n;
System.out.println(" put" + n);
valueset = true; 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
                                                        ();
     }
```

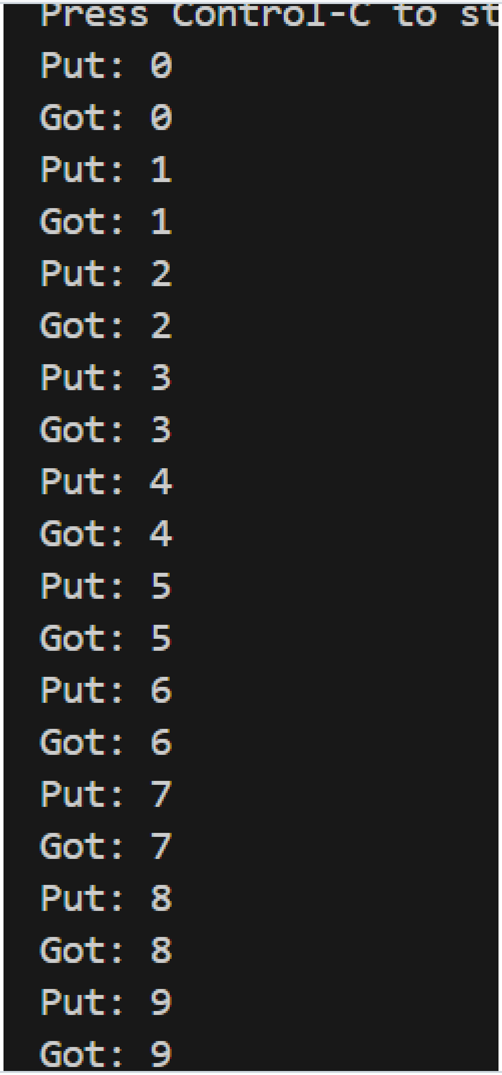


```
public void run()
{  int i=0;
   while(i<15) { q.get();   i++ } } }

class PC
{
  public static void main(String args[])
  {
      Q q = new Q();
      new Producer(q);
      new Consumer(q);
      System.out.println("press Control-C to stop.");
  }
}
output :
put : 0    put 1    put 2  . . . .  put 14
Got : 0    Got 1    Got 2  - - -    Got 14
```

CODE:

```
class Q {
    int n;
    boolean valueSet = false; // Flag to check if value is set

    synchronized int get() {
        // Wait until a value is set by the producer
        while (!valueSet) {
            try {
                wait(); // Wait until a value is put
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
        System.out.println("Got: " + n);
        valueSet = false; // Reset the flag
        notify(); // Notify the producer that value is consumed
        return n;
    }

    synchronized void put(int n) {
        // Wait until the consumer has consumed the previous value
        while (valueSet) {
            try {
                wait(); // Wait until consumer gets the value
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
        this.n = n;
        System.out.println("Put: " + n);
        valueSet = true; // Set the flag indicating value is set
        notify(); // Notify the consumer that a new value is available
    }
}

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) {
            q.get(); // Call get() without storing the result
            i++;
        }
    }
}
```

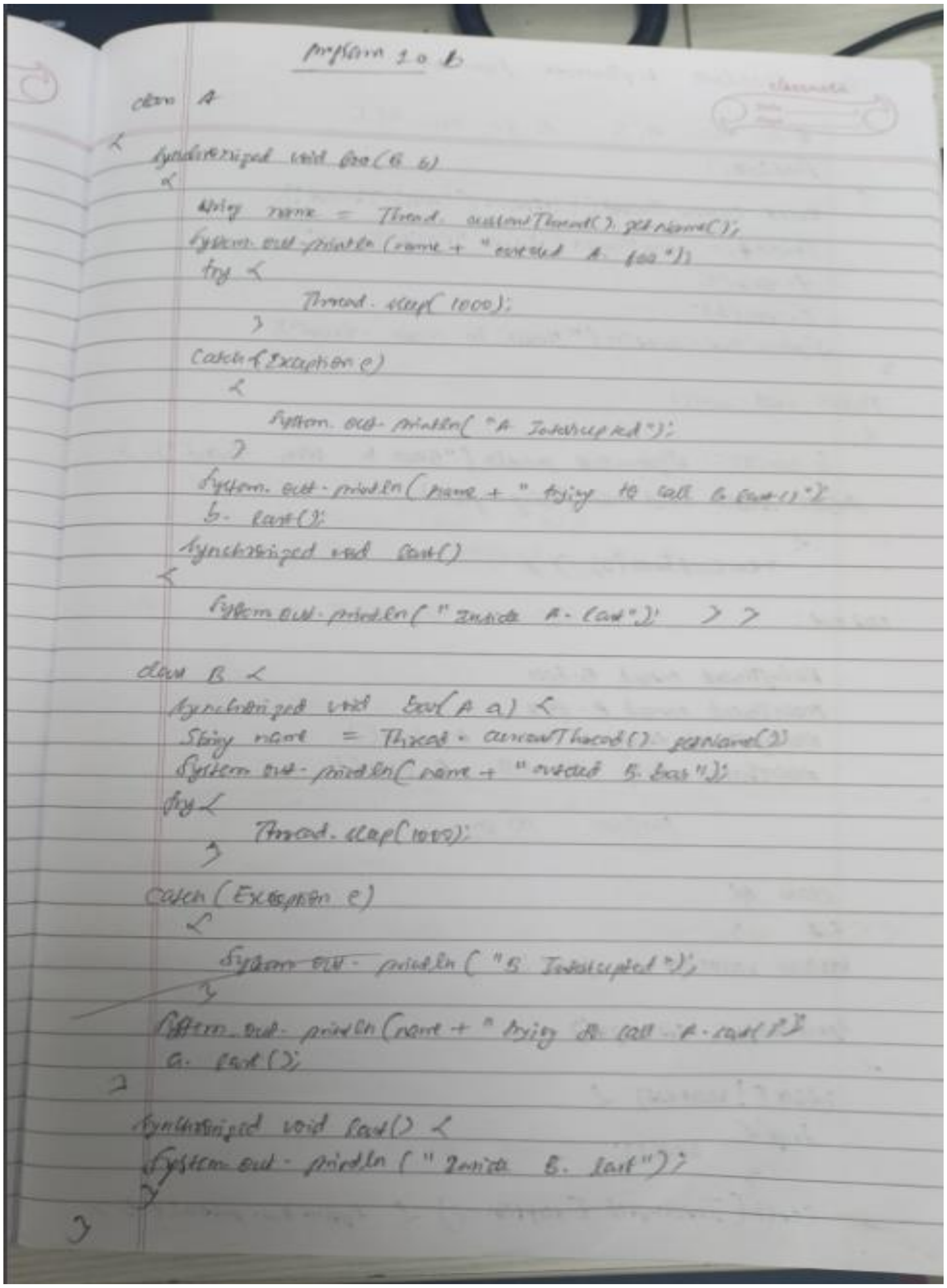
```
class PC {
   public static void main(String args[]) {
      Q q = new Q();


      new Producer(q);
      new Consumer(q);


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



```
Press Control-C to st
Put: 0
Got: 0
Put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
Got: 4
Put: 5
Got: 5
Put: 6
Got: 6
Put: 7
Got: 7
Put: 8
Got: 8
Put: 9
Got: 9
```

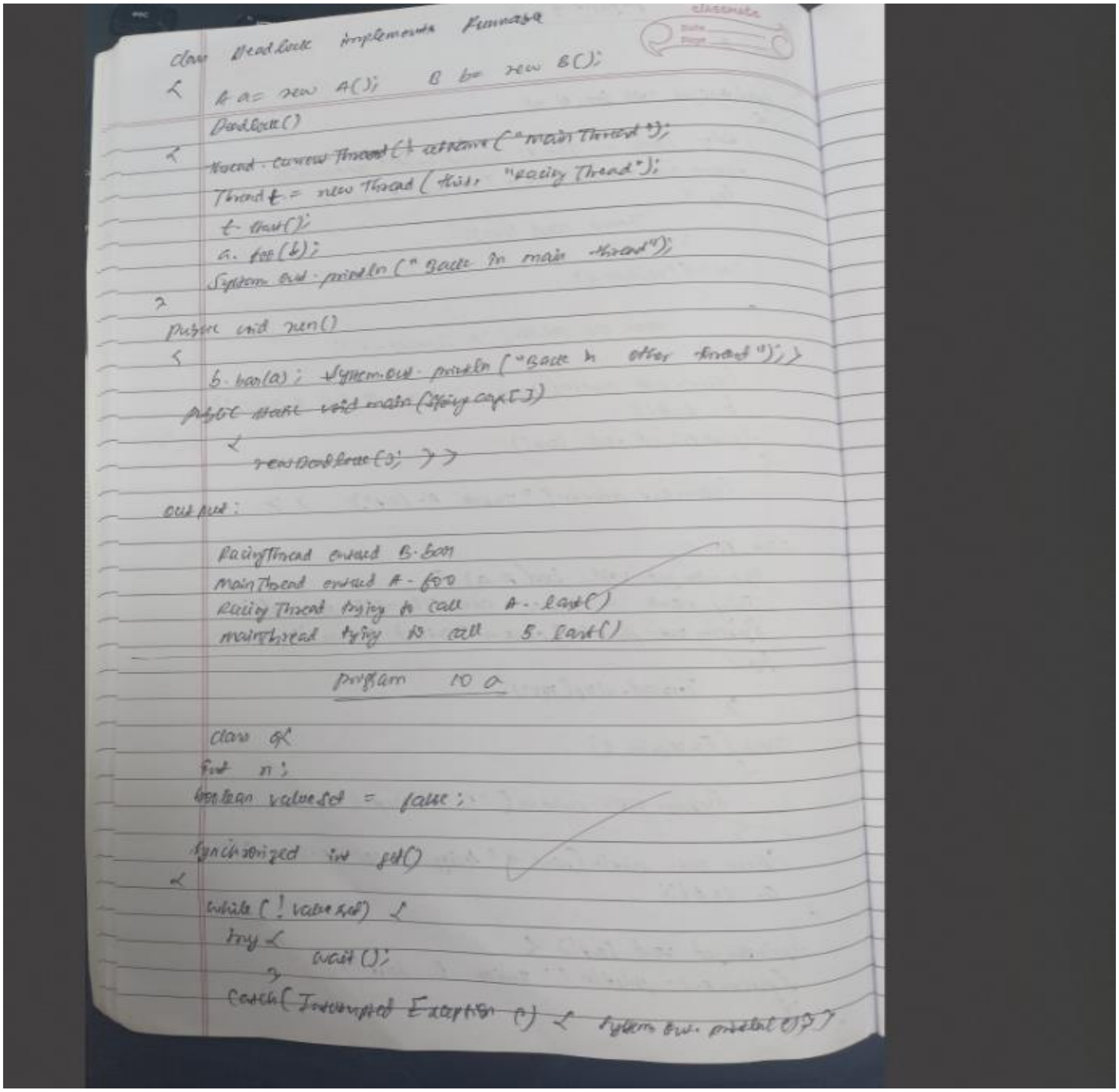
PROGRAM 10B : DEADLOCK

ALGORITHM:



Program 10 b

```
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("main Thread");
    Thread t = new Thread(this, "Racing Thread");
    t.start();
    a.foo(b);
    System.out.println("Back in main thread");
}
public void run()
{
    b.bar(a); System.out.println("Back in other thread");}
public static void main(String args[])
   {
      new DeadLock(); }}

output:

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

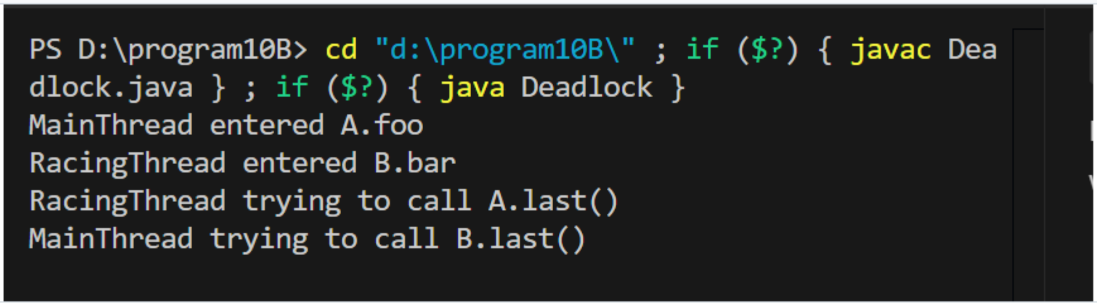
program 10 a

```
    class Q{
   int n;
   boolean valueSet = false;

   synchronized int get()
{
    while(!valueSet) {
     try{
            wait();
        }
     catch(InterruptedException e) { System.out.println(e);}}
```

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();

        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[]) {
        new Deadlock();
    }
}
```



```
PS D:\program10B> cd "d:\program10B\" ; if ($?) { javac Dea
dlock.java } ; if ($?) { java Deadlock }
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()
```