VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Ayaan Shrestha (1BM23CS056)

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 **Ayaan Shrestha (1BM23CS056),** 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.

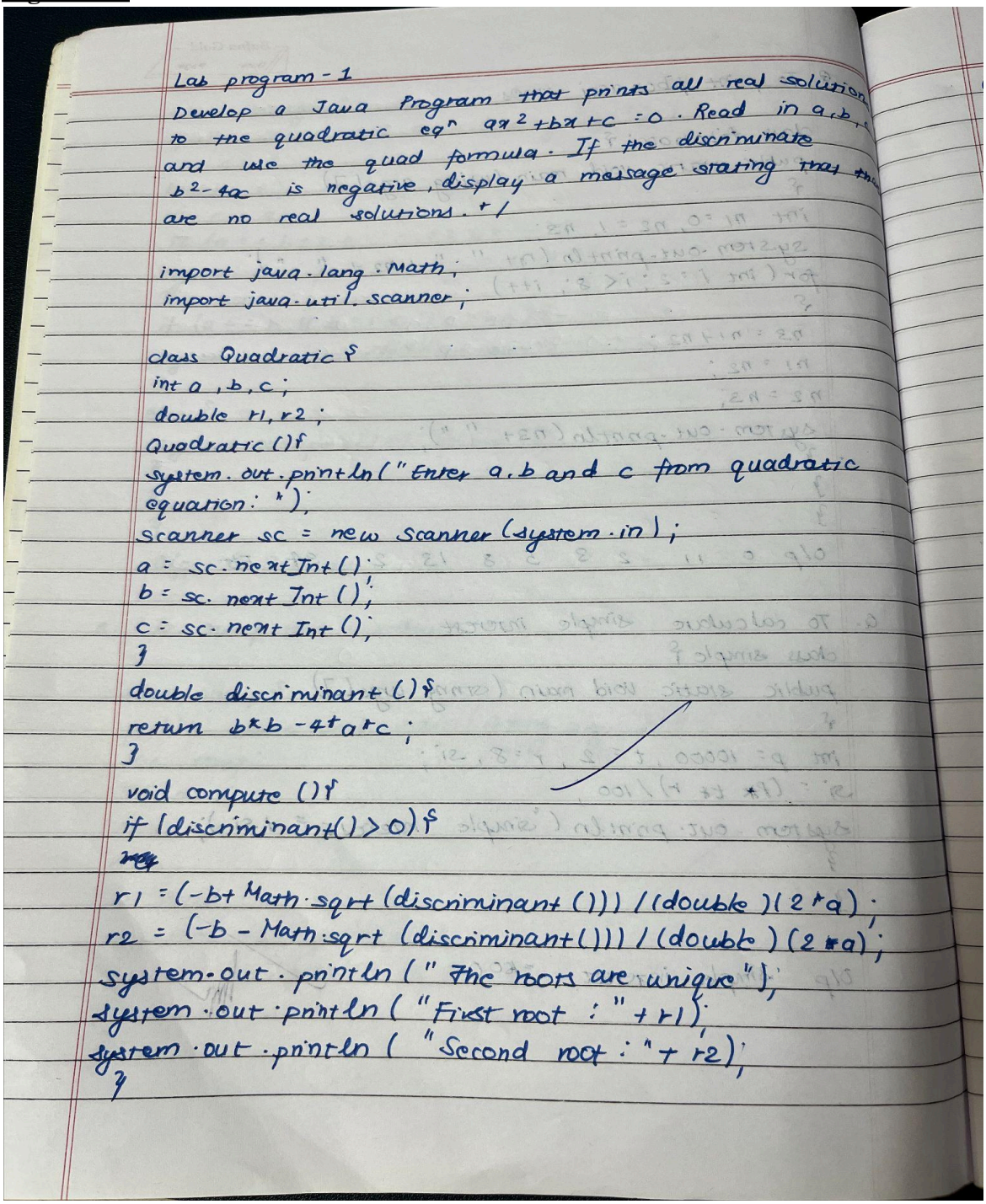
Dr. Prasad Gr Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link:
https://github.com/ayaanstha/JAVA

Program 1
Implement Quadratic Equation

Algorithm:



```
Lab program - 1
Develop a Java Program that prints all real solutions
to the quadratic eqn ax^2+bx+c = 0. Read in a,b,c
and use the quad formula. If the discriminate
b^2-4ac is negative, display a message stating that
are no real solutions. */

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

class Quadratic {
int a, b, c;
double r1, r2;
Quadratic (){
System.out.println("Enter a,b and c from quadratic
equation: ");
Scanner sc = new Scanner(System.in);
a = sc.nextInt();
b = sc.nextInt();
c = sc.nextInt();
}
double discriminant (){
return b*b - 4*a*c;
}
void compute (){
if (discriminant()>0){
r1 = (-b+Math.sqrt(discriminant()))/(double)(2*a);
r2 = (-b-Math.sqrt(discriminant()))/(double)(2*a);
System.out.println("The roots are unique");
System.out.println("First root : "+r1);
System.out.println("Second root : "+r2);
}
```

```
else if (discriminant() == 0) {
r1 = -b/(2*a);
System.out.println("The roots are equal");
System.out.println("The root is :"+r1);
}
else if (discriminant() < 0) {
r1 = -b/(2*a);
r2 = (-b+Math.sqrt(-discriminant()))/(double)(2*a);
System.out.println("The roots are imaginary");
System.out.println("First root :"+r1+"+i"+r2);
System.out.println("second root : "+r1+" -i"+r2);

}
}
}

class Run {
public static void main(String args[]) {
Quadratic eqn1 = new Quadratic();
eqn1.compute();
Quadratic eq2 = new Quadratic();
eq2.compute();
Quadratic eq3 = new Quadratic();
eq3.compute();
}
}
```

O/P

```
Enter a, b and c from quad equation:
1
2
1
```

The roots are equal
The root is : -1.0

Enter a, b and c
3
4
9
The roots are imaginary
First root : 0.0 + i0.9315438411042892
Second root : 0.0 - i0.9319448411042392

Enter a, b & c
1
5
6

The roots are unique

First root : -2.0
Second root : -3.0

Code:

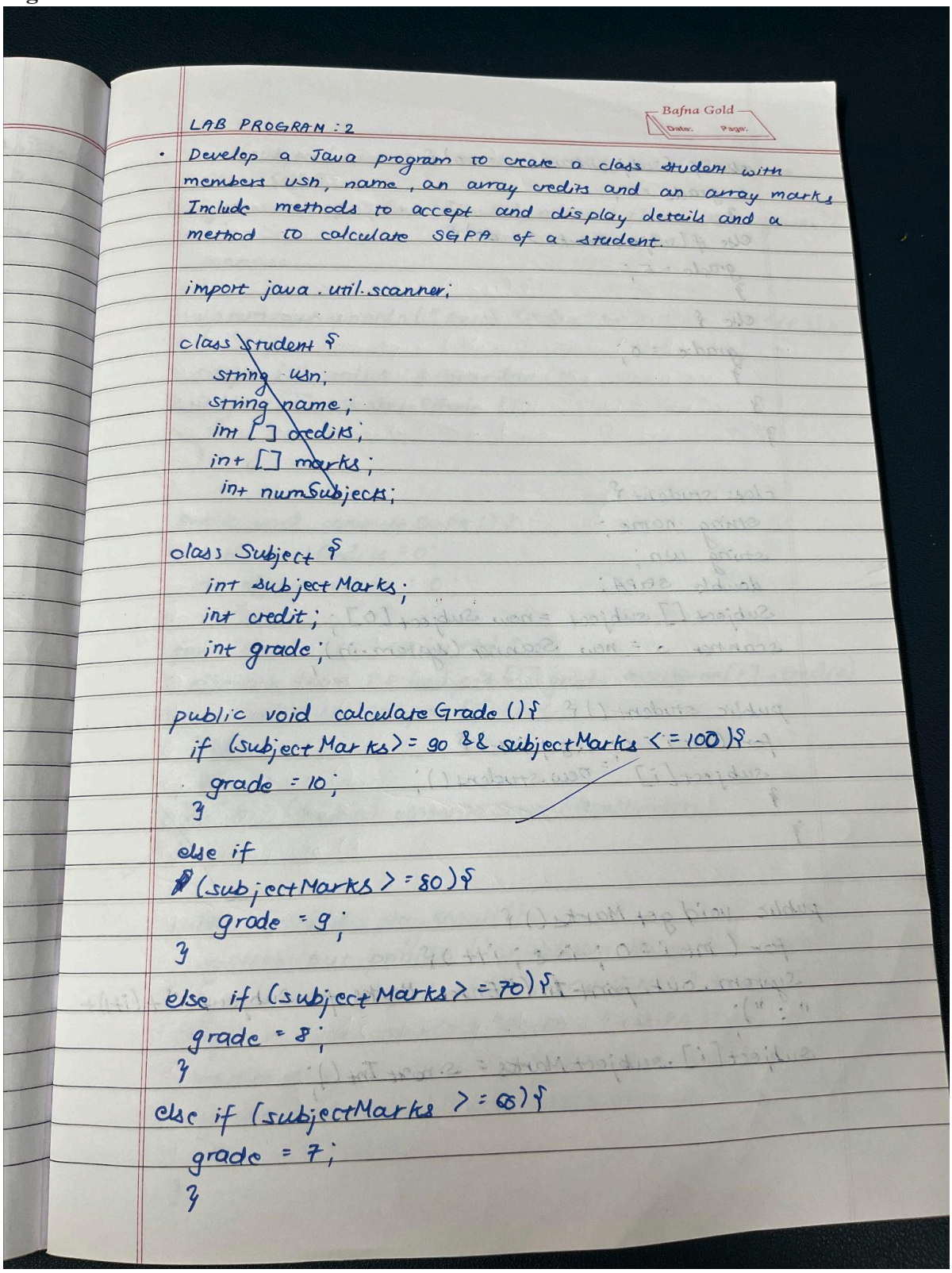
```
import java.util.Scanner;

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

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

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

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

      double discriminant = b * b - 4 * a * c;

      if (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 different:");
         System.out.println("Root 1: " + root1);
         System.out.println("Root 2: " + root2);
      } else if (discriminant == 0) {
         double root = -b / (2 * a);
         System.out.println("The roots are real and the same:");
         System.out.println("Root: " + root);
      } else {
         System.out.println("The roots are complex:");
         double realPart = -b / (2 * a);
         double imaginaryPart = Math.sqrt(-discriminant) / (2 * a);
         System.out.println("Root 1: " + realPart + " + " + imaginaryPart + "i");
         System.out.println("Root 2: " + realPart + " - " + imaginaryPart + "i");
                  System.out.print("Ayaan Shrestha 1BM23CS056");
      }

      scanner.close();
   }
}
```

Output :

```
/Library/Java/JavaVirtualMachines/jdk-22.jdk/Contents/
Enter coefficient a: 1
Enter coefficient b: 1
Enter coefficient c: 1
The roots are complex:
Root 1: -0.5 + 0.8660254037844386i
Root 2: -0.5 - 0.8660254037844386i
Ayaan Shrestha 1BM23CS056
Process finished with exit code 0
```

Program 2 :
Implement SGPA Calculator

Algorithm :



LAB 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.

```
import java.util.scanner;

class student {
    string usn;
    string name;
    int [] credits;
    int [] marks;
    int numSubjects;

class Subject {
    int subjectMarks;
    int credit;
    int grade;

public void calculateGrade (){
  if (subjectMarks >= 90 && subjectMarks <= 100){
    grade = 10;
  }
  else if
  (subjectMarks >= 80){
    grade = 9;
  }
  else if (subjectMarks >= 70){
    grade = 8;
  }
  else if (subjectMarks >= 60){
    grade = 7;
  }
```

```
else if (subjectMarks >= 50) {
    grade = 6;
  }
  else if (subjectMarks >= 40) {
    grade = 5;
  }
  else {
    grade = 0;
  }
 }
}

class student {
  string name;
  string usn;
  double sGPA;
  Subject [] subject = new Subject [0];
  scanner s = new Scanner (System.in);

  public student () {
   for (int i = 0; i<8; i++) {
    subject[i] = new student ();
   }
  }

public void getMarks() {
   for ( int i = 0; i<8; i++ ) {
  System.out.println ("Enter Marks for Subject " + (i+1) +
" : ");
subject[i].subjectMarks = s.nextInt();
```

```
if (subject[i].subjectMarks > 100 || subject[i].subject
Marks < 0) {
System.out.println("Invalid marks! Please enter again.");
i--;
continue;
}
System.out.println("Enter Credits for Subject " + (i+1) +
": ");
subject[i].credits = s.nextInt();
subject[i].calculateGrade();
}
}

public void computeSGPA() {
int totalCredits = 0;
int effectiveScore = 0;

for (int i = 0; i < 8; i++) {
effectiveScore += (subject[i].grade * subject[i].credits);
totalCredits += subject[i].credits;
}

SGPA = (double) effectiveScore / totalCredits;
}

public void displayResult() {
System.out.println("\n Student Name : " + name);
System.out.println("Student USN : " + usn);
System.out.println("SGPA : " + SGPA);
System.out.println("\n Ayaan Shrestha");
System.out.println("\n 1BM23CS056");
}
}
```

```
public class Main {
    public static void main (String [] args) {
    Student student = new Student();
    student.getStudentDetails();
    student.getMarks();
    student.computeSGPA();
    student.displayResult();
    }
}
```

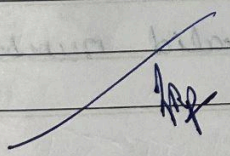
O/P:

```
Enter student Name:
Ayaan Shrestha

Enter Student USN:
1BM23CS056

Enter Marks for Subject 1:
90
Enter Credits for Subject 1:
4
Enter Marks for Subject 2:
89
Enter credits for subject 2:
4
Enter Marks for subject 3:
78
Enter credits for subject 3:
3
Enter Marks for Subject 4:
69
```

Bafna Gold
Date: Page:

Enter credits for subject 4 :
3
Enter Marks for subject 5 :
89
Enter credits for subject 5 :
4
Enter Marks for subject 6 :
92
Enter credits for subject 6 :
2
Enter Marks for subject 7 :
94
Enter credits for subject 7 :
1
Enter Marks for subject 8 :
87
Enter credits for subject 8 :
1

Student name : Ayaan Shrestha
Student USN : 123
SGPA : 9.090909090.92



Code :

```
import java.util.Scanner;

class Subject {
    int subjectMarks;
    int credits;
    int grade;

    // Method to calculate grade based on marks
    public void calculateGrade() {
        if (subjectMarks >= 90 && subjectMarks <= 100) {
            grade = 10;
        } else if (subjectMarks >= 80) {
            grade = 9;
        } else if (subjectMarks >= 70) {
            grade = 8;
        } else if (subjectMarks >= 60) {
            grade = 7;
        } else if (subjectMarks >= 50) {
            grade = 6;
        } else if (subjectMarks >= 40) {
            grade = 5;
        } else {
            grade = 0;  // Failed the subject
        }
    }
}

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


    public Student() {
        for (int i = 0; i < 8; i++) {
            subject[i] = new Subject(); // Create subject objects
        }
    }


    public void getStudentDetails() {
        System.out.println("Enter Student Name: ");
        name = s.nextLine();
```

```
        System.out.println("Enter Student USN: ");
        usn = s.nextLine();
    }


    public void getMarks() {
        for (int i = 0; i < 8; i++) {
            System.out.println("Enter Marks for Subject " + (i + 1) + ": ");
            subject[i].subjectMarks = s.nextInt();


            if (subject[i].subjectMarks > 100 || subject[i].subjectMarks < 0) {
                System.out.println("Invalid marks! Please enter again.");
                i--;
                continue;
            }

            System.out.println("Enter Credits for Subject " + (i + 1) + ": ");
            subject[i].credits = s.nextInt();

            subject[i].calculateGrade(); // Calculate grade based on marks
        }
    }


    public void computeSGPA() {
        int totalCredits = 0;
        int effectiveScore = 0;

        for (int i = 0; i < 8; i++) {
            effectiveScore += (subject[i].grade * subject[i].credits);
            totalCredits += subject[i].credits;
        }

        SGPA = (double) effectiveScore / totalCredits;
    }


    public void displayResult() {
        System.out.println("\nStudent Name: " + name);
        System.out.println("Student USN: " + usn);
        System.out.println("SGPA: " + SGPA);
    }
}

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

```
        Scanner s = new Scanner(System.in);
           System.out.println("\nayaan shrestha");



        Student[] students = new Student[3];


        for (int i = 0; i < 3; i++) {
           System.out.println("\nEnter details for Student " + (i + 1) + ": ");
           students[i] = new Student(); // Create a new student object
           students[i].getStudentDetails(); // Get name and usn
           students[i].getMarks(); // Get marks and credits for subjects
           students[i].computeSGPA(); // Compute SGPA
        }


        System.out.println("\n\nResults for all students:");
        for (int i = 0; i < 3; i++) {
           students[i].displayResult();

        }
    }
}
```

Output :

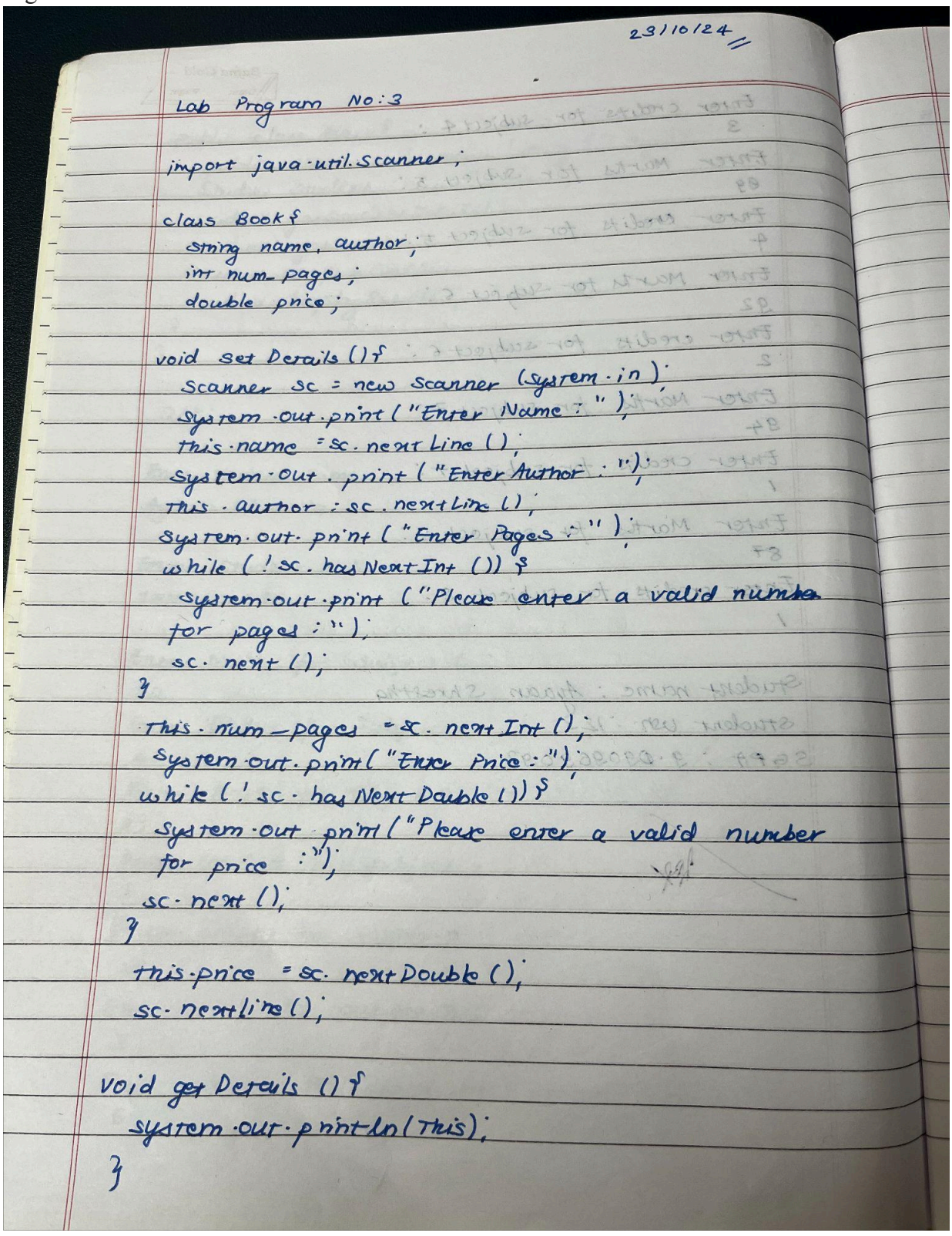
```
ayaan shrestha

Enter details for Student 1:
Enter Student Name:
Rob Wheeler
Enter Student USN:
12346969
Enter Marks for Subject 1:
100
Enter Credits for Subject 1:
4
Enter Marks for Subject 2:
98
Enter Credits for Subject 2:
4
Enter Marks for Subject 3:
87
Enter Credits for Subject 3:
4
Enter Marks for Subject 4:
99
Enter Credits for Subject 4:
3
Enter Marks for Subject 5:
70
Enter Credits for Subject 5:
3
Enter Marks for Subject 6:
89
Enter Credits for Subject 6:
2
Enter Marks for Subject 7:
88
Enter Credits for Subject 7:
1
Enter Marks for Subject 8:
78
Enter Credits for Subject 8:
1

Enter details for Student 2:
Enter Student Name:
avc
Enter Student USN:
877745
Enter Marks for Subject 1:
100
Enter Credits for Subject 1:
4
Enter Marks for Subject 2:
56
Enter Credits for Subject 2:
4
Enter Marks for Subject 3:
67
```

```
88
Enter Credits for Subject 6:
2
Enter Marks for Subject 7:
90
Enter Credits for Subject 7:
1
Enter Marks for Subject 8:
99
Enter Credits for Subject 8:
1

Enter details for Student 3:
Enter Student Name:
timthetanman
Enter Student USN:
89989898
Enter Marks for Subject 1:
100
Enter Credits for Subject 1:
4
Enter Marks for Subject 2:
90
Enter Credits for Subject 2:
4
Enter Marks for Subject 3:
38
Enter Credits for Subject 3:
4
Enter Marks for Subject 4:
70
Enter Credits for Subject 4:
3
Enter Marks for Subject 5:
67
Enter Credits for Subject 5:
3
Enter Marks for Subject 6:
90
Enter Credits for Subject 6:
2
Enter Marks for Subject 7:
92
Enter Credits for Subject 7:
1
Enter Marks for Subject 8:
100
En  Credits for Subject 8:
1
```

```
Results for all students:

Student Name: Rob Wheeler
Student USN: 12346969
SGPA: 9.318181818181818

Student Name: avc
Student USN: 877745
SGPA: 8.227272727272727

Student Name: timthetanman
Student USN: 89989898
SGPA: 7.5

C:\Users\Admin\Desktop>
```

Program 3 :
Create Objects for Books

Algorithm:



```
23/10/24

Lab Program No:3

import java.util.Scanner;

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

void setDetails(){
  Scanner sc = new Scanner(System.in);
  System.out.print("Enter Name : ");
  this.name = sc.nextLine();
  System.out.print("Enter Author : ");
  this.author = sc.nextLine();
  System.out.print("Enter Pages : ");
  while(!sc.hasNextInt()){
   System.out.print("Please enter a valid number
   for pages : ");
   sc.next();
  }
  this.num_pages = sc.nextInt();
  System.out.print("Enter Price : ");
  while(!sc.hasNextDouble()){
   System.out.print("Please enter a valid number
   for price : ");
   sc.next();
  }
  this.price = sc.nextDouble();
  sc.nextLine();

void getDetails(){
  System.out.println(this);
}
```

```
public String toString (){
    return "Name : " + name + "\n Author : " + author + "
\n Pages : " + num_pages + "\n Price : " + price;
  }
}

class BookDemo {
    public static void main (String [] args ) {
    Scanner sc = new Scanner (System.in);
    System.out.print ("Enter the number of books: ");
    while (!sc.hasNextInt ()) {
     system.out.print ("Please enter a valid number:");
    sc.next ();
}
int bookNum = sc.nextInt ();
 sc.nextLine ();

Book [] bookArray = new Book [bookNum];

for (int i =0; i< bookNum; i++) {
 bookArray [i] = new Book();
 bookArray [i].setDetails ();
 system.out.println(),
}

}

}
```

O/P ⟶

O/P

Enter the Number of books : 2
Enter Name : Atomic Habits
Enter Author : James Clear
Enter Pages : 450
Enter Price : 6.99

Enter Name : The 48 Powers of Law
Enter Author : Robert Greene
Enter Pages : 480
Enter Price : 7.99

Name : Atomic Habits
Author : James Clear
Pages : 450
Price : 6.99

Name : The 48 powers of Law
Author : Robert Greene
Pages : 480
Price : 7.99

Seen

Code :

```
import java.util.Scanner;

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

   void setDetails() {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter Name: ");
      this.name = sc.nextLine();
      System.out.print("Enter Author: ");
      this.author = sc.nextLine();
      System.out.print("Enter Pages: ");
      while (!sc.hasNextInt()) {
         System.out.print("Please enter a valid number for pages: ");
         sc.next();
      }
      this.num_pages = sc.nextInt();
      System.out.print("Enter Price: ");
      while (!sc.hasNextDouble()) {
         System.out.print("Please enter a valid number for price: ");
         sc.next();
      }
      this.price = sc.nextDouble();
      sc.nextLine();      }

   void getDetails() {
      System.out.println(this);
   }

   public String toString() {
      return "Name: " + name + "\nAuthor: " + author + "\nPages: " +
num_pages + "\nPrice: " + price;
   }
}

class BookDemo {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter the number of books: ");
      while (!sc.hasNextInt()) {
         System.out.print("Please enter a valid number: ");
         sc.next();
```

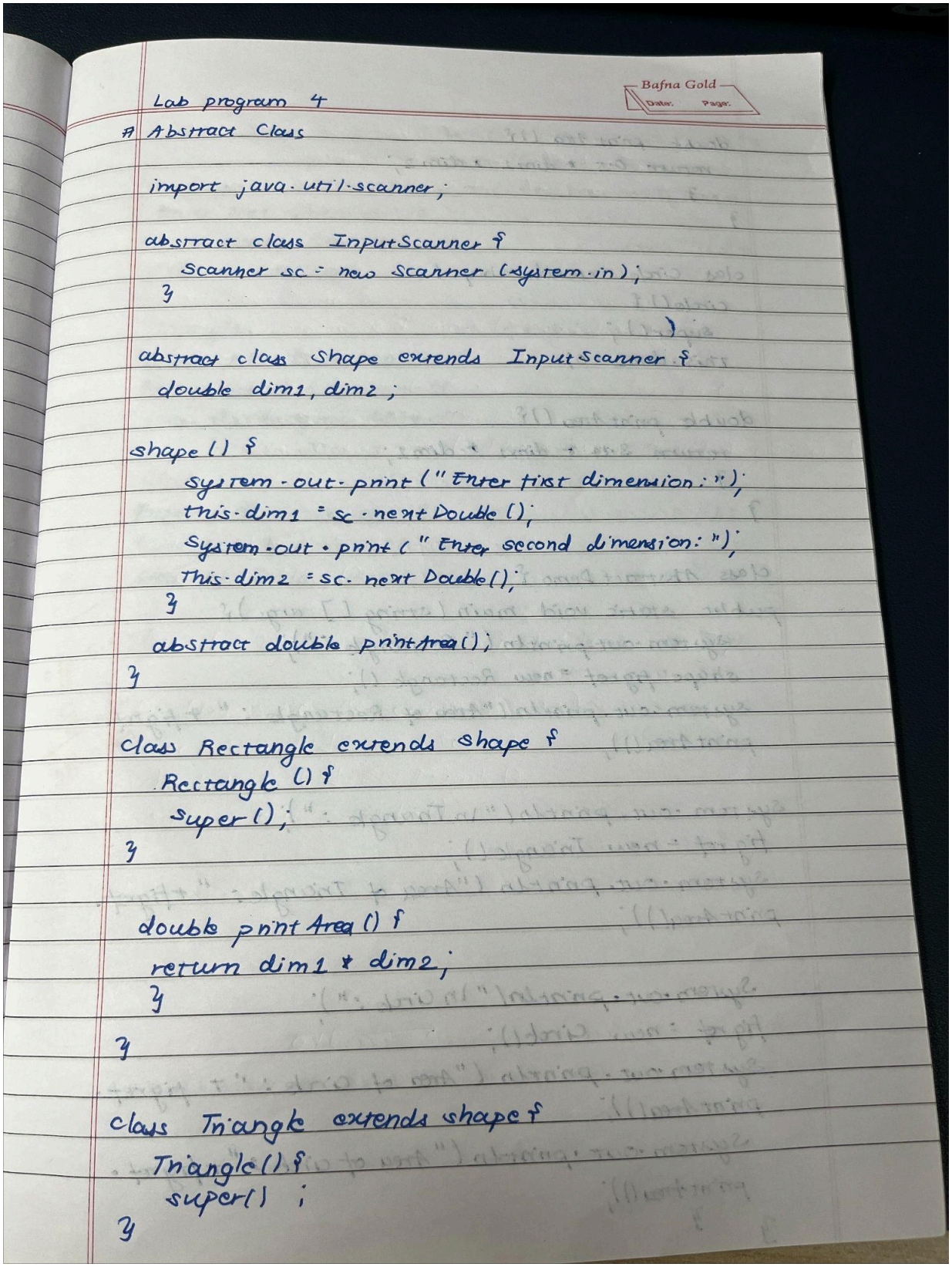
```
        }
        int bookNum = sc.nextInt();
        sc.nextLine();

        Book[] bookArray = new Book[bookNum];

        for (int i = 0; i < bookNum; i++) {
            bookArray[i] = new Book();
            bookArray[i].setDetails();
            System.out.println();
        }

        for (int i = 0; i < bookNum; i++) {
            bookArray[i].getDetails();
            System.out.println();
                System.out.println("AYAAN SHRESTHA, 1BM23CS056");
        }


    }
}
```

Output:

```
C:\Users\Admin\Desktop>java BookDemo
Enter the number of books: 2
Enter Name: Atomic Habits
Enter Author: James Clear
Enter Pages: 450
Enter Price: 699

Enter Name: The 48 Powers of Law
Enter Author: Robert Greene
Enter Pages: 480
Enter Price: 799

Name: Atomic Habits
Author: James Clear
Pages: 450
Price: 699.0

AYAAN SHRESTHA, 1BM23CS056
Name: The 48 Powers of Law
Author: Robert Greene
Pages: 480
Price: 799.0

AYAAN SHRESTHA, 1BM23CS056
```

Program 4 : Implement Abstract Class

Algorithm:



```
Lab program 4
// Abstract Class

import java.util.scanner;

abstract class InputScanner {
   Scanner sc = new Scanner (System.in);
}

abstract class Shape extends InputScanner {
  double dim1, dim2;

shape () {
    System.out.print ("Enter first dimension:");
    this.dim1 = sc.nextDouble ();
    System.out.print ("Enter second dimension:");
    This.dim2 = sc.next Double();
  }
  abstract double printArea();
}

class Rectangle extends Shape {
   Rectangle () {
    super();
}

 double printArea () {
  return dim1 * dim2;
  }
}

class Triangle extends Shape {
  Triangle() {
    super() ;
}
```

```
    double printArea (){
      return 0.5 * dim1 * dim2;
      }
}

class Circle extends Shape{
    Circle() {
      super();
      this.dim2 = 0;

double printArea (){
    return 3.14 * dim1 * dim1;
    }
}

class AbstractDemo {
public static void main (String [] args){
    System.out.println (" Rectangle :");
    Shape figref = new Rectangle ();
    System.out.println ("Area of Rectangle :" + figref.
    printArea());

System.out.println("\n Triangle :");
  figref = new Triangle();
  System.out.println ("Area of Triangle: " +figref.
  printArea());

    System.out.println("\n Circle :");
    figref = new Circle();
    System.out.println ("Area of Circle :" + figref.
    printArea());
    System.out.println ("Area of circle :" +figref.
    printArea());
}   }
```

Bafna Gold
Date: Page:

O/p:

Rectangle :
Enter first dimension: 12
Enter second dimension : 16
Area of Rectangle : 192.0

Triangle:
Enter first dimension : 82
Enter second dimension : 62
Area of triangle : 2542.0

Circle:
Enter first dimension : 91
Enter second dimension : 0
Area of Circle : 26002.34

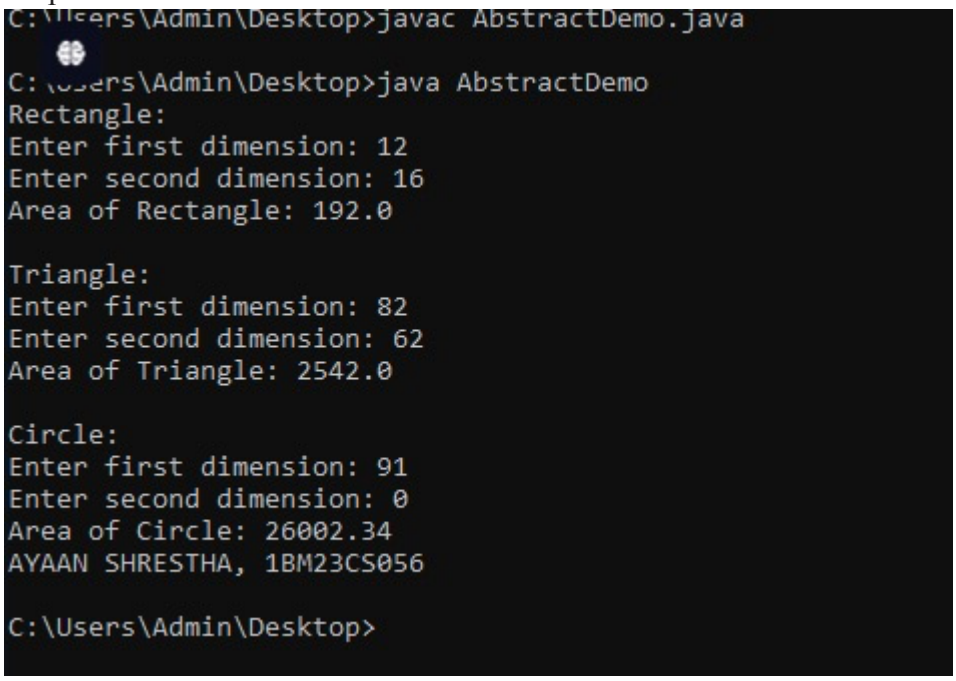
O/p seen
28/10/24

Code :

```
import java.util.Scanner;

abstract class InputScanner {
   Scanner sc = new Scanner(System.in);
}

abstract class Shape extends InputScanner {
   double dim1, dim2;

   Shape() {
      System.out.print("Enter first dimension: ");
      this.dim1 = sc.nextDouble();
      System.out.print("Enter second dimension: ");
      this.dim2 = sc.nextDouble();
   }

   abstract double printArea();
}

class Rectangle extends Shape {
   Rectangle() {
      super();
   }

   double printArea() {
      return dim1 * dim2;
   }
}

class Triangle extends Shape {
   Triangle() {
      super();
   }

   double printArea() {
      return 0.5 * dim1 * dim2;
   }
}

class Circle extends Shape {
   Circle() {
      super();
      this.dim2 = 0;      }
```

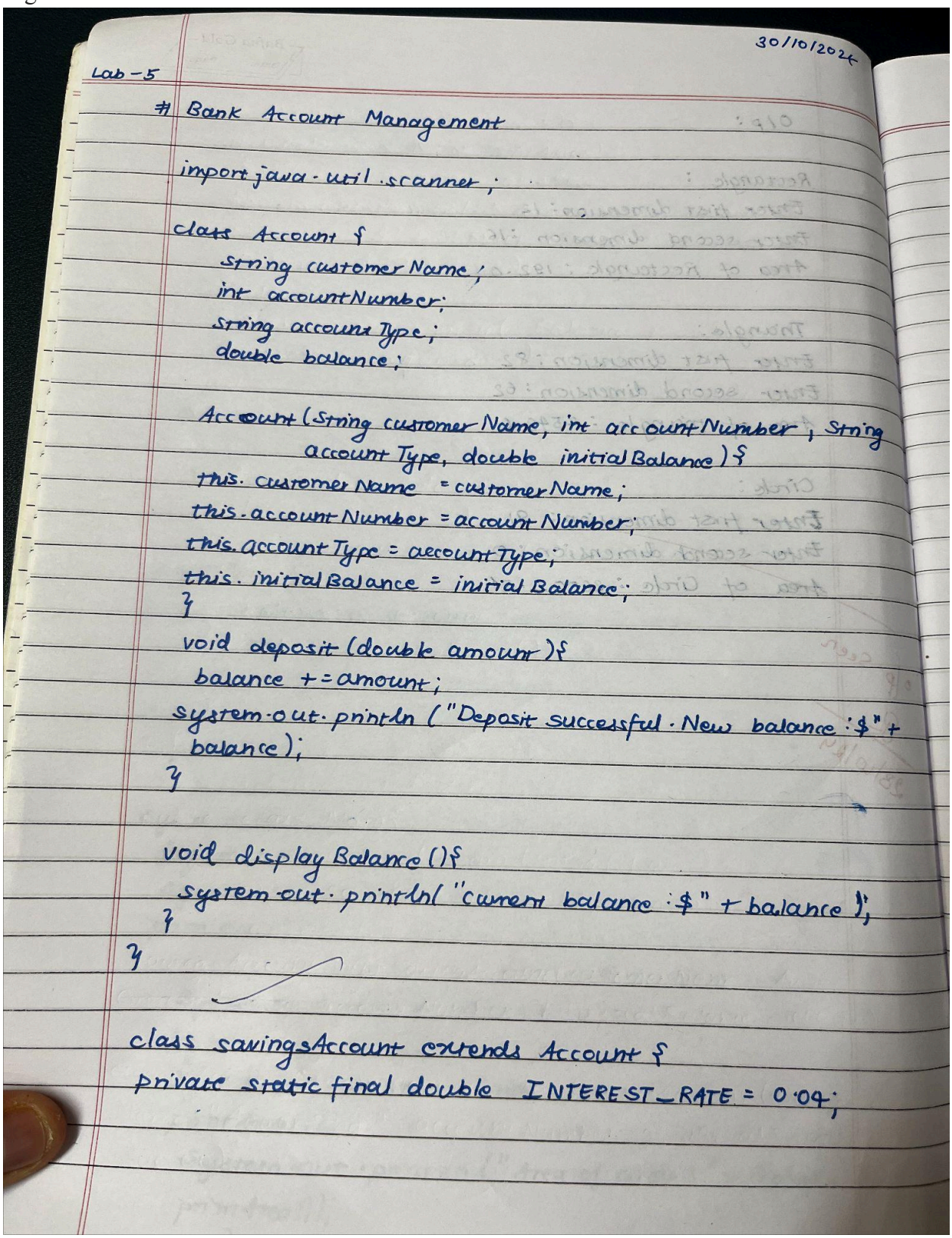
```
    double printArea() {
        return 3.14 * dim1 * dim1;
    }
}

class AbstractDemo {
    public static void main(String[] args) {
        System.out.println("Rectangle:");
        Shape figref = new Rectangle();
        System.out.println("Area of Rectangle: " + figref.printArea());

        System.out.println("\nTriangle:");
        figref = new Triangle();
        System.out.println("Area of Triangle: " + figref.printArea());

        System.out.println("\nCircle:");
        figref = new Circle();
        System.out.println("Area of Circle: " + figref.printArea());
            System.out.println("AYAAN SHRESTHA, 1BM23CS056");
    }
}
```

Output:



Program 5 : Bank Account Management

Algorithm :



Lab - 5

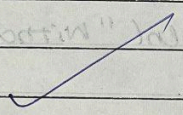
30/10/2024

Bank Account Management

```
import java.util.scanner;

class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;

    Account(String customerName, int accountNumber, String
            accountType, double initialBalance) {
    this.customerName = customerName;
    this.accountNumber = accountNumber;
    this.accountType = accountType;
    this.initialBalance = initialBalance;
    }
    void deposit(double amount){
     balance += amount;
    System.out.println("Deposit successful. New balance :$" +
     balance);
    }

    void displayBalance(){
     System.out.println("current balance :$" + balance);
    }
}

class savingsAccount extends Account {
private static final double INTEREST_RATE = 0.04;
```

```
SavingsAccount extends Account {
SavingsAccount (String customerName, int accountNumber,
  double initialBalance) {
  super (customerName, accountNumber, accountType:"
  Savings", initialBalance);
}

void computeInterest() {
   double interest = balance * INTEREST_RATE;
   balance += interest;
   System.out.println ("Interest added. New balance: $"
     +balance);
}

void withdraw (double amount) {
 if (amount > balance) {
   System.out.println ("Insufficient Balance");
 } else {
   balance -= amount;
   System.out.println("Withdrawal Successful. New Balance
   : $" +balance);
  }
 }
}

class CurrentAccount extends Account {
  private static final double MIN_BALANCE = 500.00;
  private static final double SERVICE_CHARGE = 25.00;

CurrentAccount(String customerName, int accountNumber, double
  initialBalance) {
super (customerName, accountNumber, accountType:"Current",
  initialBalance);
}
```

```
void checkMinimumBalance(){
  if (balance < MIN_BALANCE){
    balance -= SERVICE_CHARGE;
    System.out.println("Below minimum balance. Service
    charge of $" + SERVICE_CHARGE + " applied. New
    Balance :$" + balance);
  } else {
    system.out.println("Balance is above the
    minimum required.");
  }
}

void withdraw(double amount){
  if (amount > balance){
    system.out.println("Insufficient funds");
  } else {
    balance -= amount;
    checkMinimumBalance();
  }
 }
}
```



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

SavingsAccount = new SavingAccount(customerName: "Ayaan",
~~CurrentAccount~~ accountNumber: 1, initialBalance: 1000.00);

CurrentAccount currentaccount = new CurrentAccount(customer
Name: "AyaanStha", accountNumber: 2, initialBalance:
800.00);
```

```
while (true) {
    System.out.println ("\n --MENU-- ");
    System.out.println ("1. Deposit ");
    System.out.println("2. Withdraw ");
    System.out.println("3. Compute interest for Savings Account");
    System.out.println ("4. Display account details ");
    System.out.println(" 5. Exit ");
    System.out.println (" Enter your choice : ");
    int choice = sc.nextInt();

    Account selectedAccount = null;
    System.out.print("Enter the type of account (saving/current)
     : ");
    String accountType = sc.next();

    if (accountType.equalsIgnoreCase("saving")) {
     selected Account = savingsAccount;
    } else if (accountType.equalsIgnoreCase("current")) {
       selected Account = currentAccount;
    } else {
     System.out.println ("Invalid account type. Try again.");
     continue;
    }

  switch (choice) {
     case 1 :
      System.out.print("Enter the deposit amount : ");
      double depositAmount = sc.nextDouble();
      selectedAccount.deposit (depositAmount);
      break;
  case 2 :
     System.out.print("Enter the withdrawal amount : ");
```

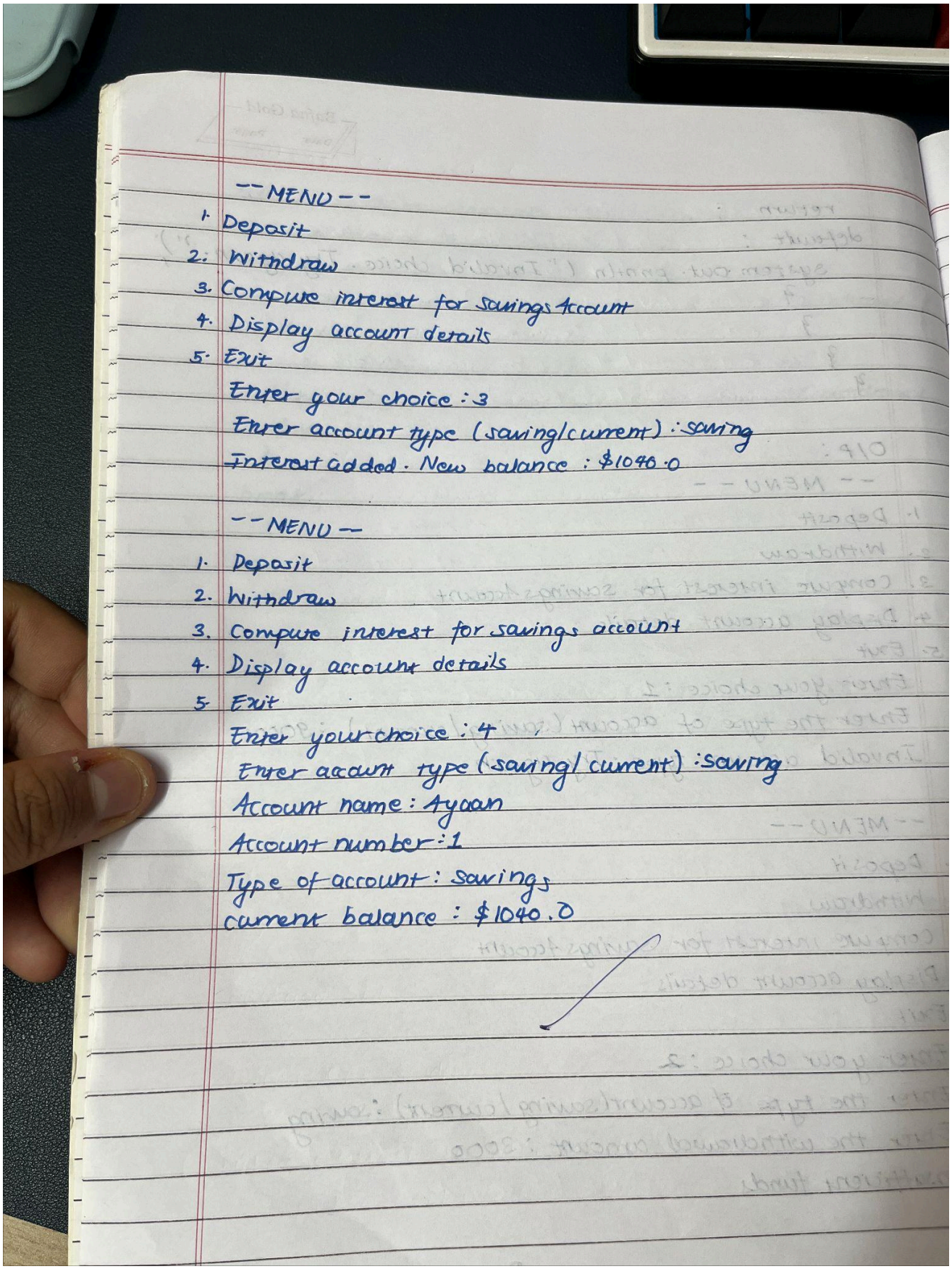
```
double withdrawalAmount = sc.nextDouble();
selectedAccount.deposit(depositAmount);
if (selectedAccount instanceof SavingsAccount){
  ((SavingsAccount) selectedAccount).withdraw(withdrawal
  Amount);
} else if (selectedAccount instance of CurrentAccount) {
((CurrentAccount) selectedAccount).withdrawal(withdrawal
Amount);
}
break;

case 3:
  if (selectedAccount instanceof SavingsAccount){
  ((SavingAccount) selectedAccount).computeInterest();
  } else {
  System.out.println("Interest computation is only for
  savings accounts.");
  }
  break;

Case 4:
  System.out.println("Account name:" + selectedAccount.
  customerName);
  System.out.println("Account number:" +selectedAccount.
  accountNumber);
  System.out.println("Type of account:" + selectedAccount.
  accountType);
selectedAccount.displayBalance();
break;

case 5:
System.out.println("Exiting program....");
sc.close();
```

```
            return ;
        default :
            system.out.println ("Invalid choice. Try again.");
        }
    }
}
```

O/P :

```
-- MENU --
1. Deposit
2. Withdraw
3. Compute interest for savings Account
4. Display account details
5. Exit
Enter your choice : 1
Enter the type of account (saving/current) : 90000
Invalid account type. Try again.

--MENU--
1. Deposit
2. Withdraw
3. Compute interest for savings Account
4. Display account details
5. Exit
Enter your choice : 2
Enter the type of account(saving/current) : saving
Enter the withdrawal amount : 3000
insufficient funds
```

P.T.O →



--MENU--
1. Deposit
2. Withdraw
3. Compute interest for savings Account
4. Display account details
5. Exit
Enter your choice: 3
Enter account type (saving/current): saving
Interest added. New balance: $1040.0

--MENU--
1. Deposit
2. Withdraw
3. Compute interest for savings account
4. Display account details
5. Exit
Enter your choice: 4
Enter accunt type (saving/current): saving
Account name: Ayaan
Account number: 1
Type of account: savings
current balance: $1040.0

Code :

```
import java.util.Scanner;

class Account {
   String customerName;
   int accountNumber;
   String accountType;
   double balance;

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

   void deposit(double amount) {
      balance += amount;
      System.out.println("Deposit successful. New balance: $" + balance);
   }

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

class SavingsAccount extends Account {
   private static final double INTEREST_RATE = 0.04;

   SavingsAccount(String customerName, int accountNumber, double
initialBalance) {
      super(customerName, accountNumber, "Savings", initialBalance);
   }

   void computeInterest() {
      double interest = balance * INTEREST_RATE;
      balance += interest;
      System.out.println("Interest added. New balance: $" + balance);
   }

   void withdraw(double amount) {
      if (amount > balance) {
         System.out.println("Insufficient funds.");
      } else {
         balance -= amount;
```

```
            System.out.println("Withdrawal successful. New balance: $" + balance);
        }
    }
}

class CurrentAccount extends Account {
    private static final double MIN_BALANCE = 500.00;
    private static final double SERVICE_CHARGE = 25.00;

    CurrentAccount(String customerName, int accountNumber, double
initialBalance) {
        super(customerName, accountNumber, "Current", initialBalance);
    }

    void checkMinimumBalance() {
        if (balance < MIN_BALANCE) {
            balance -= SERVICE_CHARGE;
            System.out.println("Below minimum balance. Service charge of $" +
SERVICE_CHARGE + " applied. New balance: $" + balance);
        } else {
            System.out.println("Balance is above the minimum required.");
        }
    }

    void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient funds.");
        } else {
            balance -= amount;
            checkMinimumBalance();
        }
    }
}

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


        SavingsAccount savingsAccount = new SavingsAccount("Ayaan", 1,
1000.00);
        CurrentAccount currentAccount = new CurrentAccount("AyaanStha", 2,
800.00);

        while (true) {
            System.out.println("\n---MENU---");
            System.out.println("1. Deposit");
```

```
            System.out.println("2. Withdraw");
            System.out.println("3. Compute interest for SavingsAccount");
            System.out.println("4. Display account details");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();

            Account selectedAccount = null;
            System.out.print("Enter the type of account (saving/current): ");
            String accountType = sc.next();

            if (accountType.equalsIgnoreCase("saving")) {
                selectedAccount = savingsAccount;
            } else if (accountType.equalsIgnoreCase("current")) {
                selectedAccount = currentAccount;
            } else {
                System.out.println("Invalid account type. Try again.");
                continue;
            }

            switch (choice) {
                case 1:
                    System.out.print("Enter the deposit amount: ");
                    double depositAmount = sc.nextDouble();
                    selectedAccount.deposit(depositAmount);
                    break;
                case 2:
                    System.out.print("Enter the withdrawal amount: ");
                    double withdrawalAmount = sc.nextDouble();
                    if (selectedAccount instanceof SavingsAccount) {
                        ((SavingsAccount)
selectedAccount).withdraw(withdrawalAmount);
                    } else if (selectedAccount instanceof CurrentAccount) {
                        ((CurrentAccount)
selectedAccount).withdraw(withdrawalAmount);
                    }
                    break;
                case 3:
                    if (selectedAccount instanceof SavingsAccount) {
                        ((SavingsAccount) selectedAccount).computeInterest();
                    } else {
                        System.out.println("Interest computation is only for savings
accounts.");
                    }
                    break;
                case 4:
                    System.out.println("Account name: " +
```

```
selectedAccount.customerName);
                System.out.println("Account number: " +
selectedAccount.accountNumber);
                System.out.println("Type of account: " +
selectedAccount.accountType);
                selectedAccount.displayBalance();
                break;
            case 5:
                System.out.println("Exiting program...");
                sc.close();
                return;
            default:
                System.out.println("Invalid choice. Try again.");
                System.out.println("Ayaan Shrestha 1BM23CS056");
        }
    }
  }
}
```

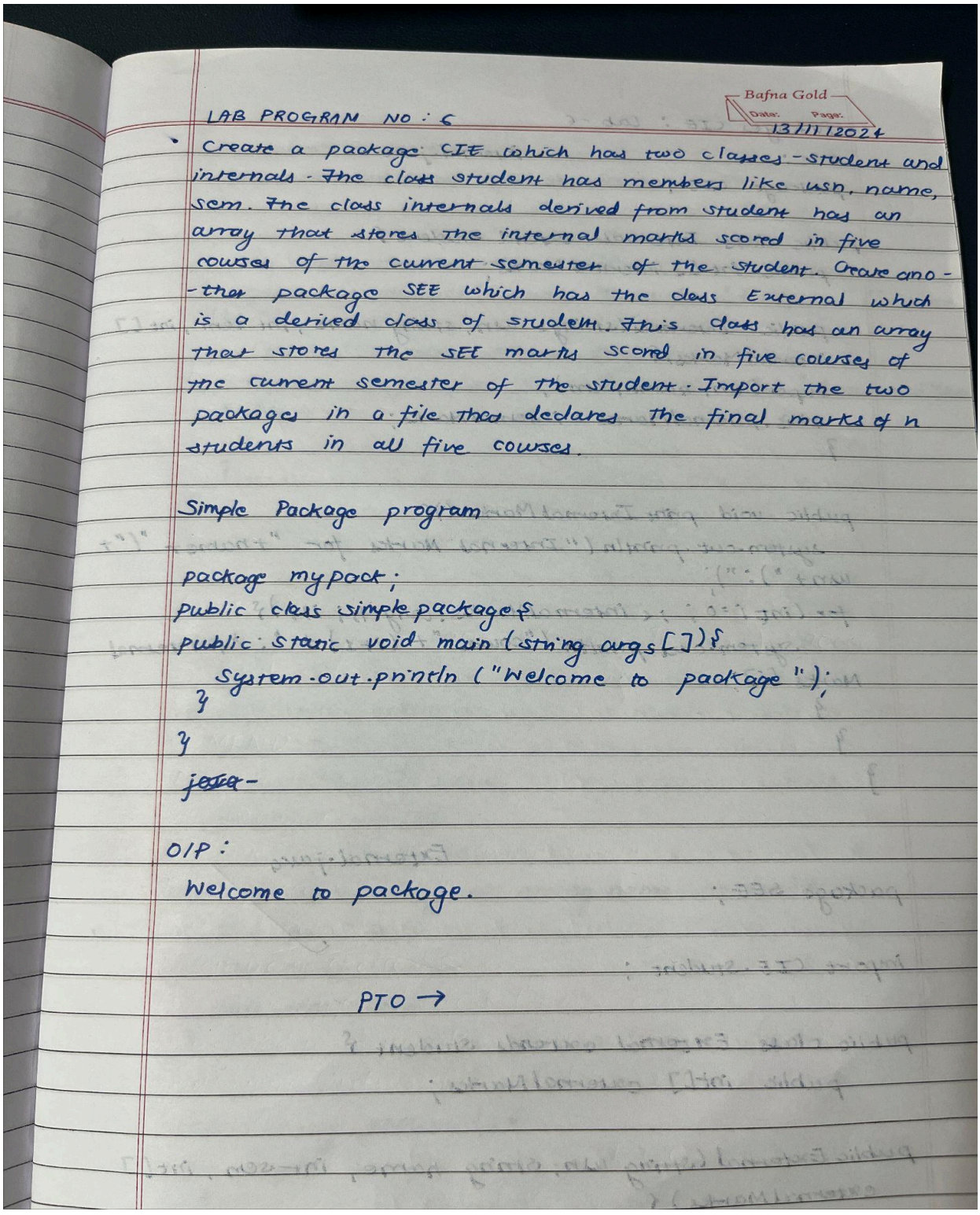
Output:

```
Ayaan Shrestha 1BM23CS056
---MENU---
1. Deposit
2. Withdraw
3. Compute interest for SavingsAccount
4. Display account details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): 90000
Invalid account type. Try again.

Ayaan Shrestha 1BM23CS056
---MENU---
1. Deposit
2. Withdraw
3. Compute interest for SavingsAccount
4. Display account details
5. Exit
Enter your choice: 2
Enter the type of account (saving/current): saving
Enter the withdrawal amount: 3000
Insufficient funds.

Ayaan Shrestha 1BM23CS056
---MENU---
1. Deposit
2. Withdraw
3. Compute interest for SavingsAccount
4. Display account details
5. Exit
Enter your choice: 3
Enter the type of account (saving/current): saving
Interest added. New balance: $1040.0
```

```
Ayaan Shrestha 1BM23CS056
---MENU---
1. Deposit
2. Withdraw
3. Compute interest for SavingsAccount
4. Display account details
5. Exit
Enter your choice: 4
Enter the type of account (saving/current): saving
Account name: Ayaan
Account number: 1
Type of account: Savings
Current balance: $1040.0
```

Program No: 6
Implement Packages

Algorithm :



LAB PROGRAM NO : 6

Bafna Gold
Date: Page:
13/11/2024

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

Simple Package program

```
package mypack;
public class simplepackage{
public static void main(string args[]){
   System.out.println("Welcome to package");
}
}
```

~~javac~~-

O/P :

Welcome to package.

PTO →

Package: CIE : Lab - 6

code:

Internals.java

```
package CIE;

public class Internals extend student {
    public int [] internalMarks;

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

    public void printInternalMarks () {
      system.out.println ("Internal Marks for "+name+ "("+ usn + "):");
    for (int i=0; i< internalMarks.length; i++) {
        System.out.println ("course "+ (i+1) + ": "+ internal Marks [i];
      }
    }
}
```

External.java

```
package SEE;

import CIE.Student;

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

public External (string usn; string name, int sem, int[]
  externalMarks) {
```

```
super(usn, name, sem);
this.externalMarks = externalMarks;
}

public void printExternalMarks(){
  System.out.println("External Marks for " + name + " (" + usn +
"): ");
for(int i=0; i<externalMarks.length; i++){
  System.out.println("Course " + (i+1) + ": " + externalMarks
[i]);
  }
 }
}
```

Student.java

```
package CIE;

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

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

Public String getUsn() {
  return usn;
}
```

```
}
public String getName(){
    return name;
}
public int getSem(){
    return sem;
}
}
```

Main.java

```
import CIE.Internals;
import SEE.Externals;

import java.util.Scanner;

public class Main{
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter number of students :");
        int n = scanner.nextInt();
        scanner.nextLine();

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

        System.out.println("\n Enter details for student "+
        (i+1) + ":" );

        System.out.print("Enter USN :");
        String usn = scanner.nextLine();

        System.out.print("Enter Name :");
        String name = scanner.nextLine();
```

```
System.out.print("Enter Semester: ");
int sem = scanner.nextInt();

int[] internalMarks = new int[5];
System.out.println("Enter Internal Marks for 5 courses:");
for(int j = 0; j < 5; j++){
 System.out.print("course " + (j+1) + " :");
 internalMarks[j] = scanner.nextInt();
}

int[] externalMarks = new int[5];
System.out.println("Enter External Marks for 5 courses :");
for(int j = 0; j<5; j++){
 System.out.print("Course" + (j+1) + " : ");
 externalMarks[j] = scanner.nextInt();
}
scanner.nextLine();

                                    internalMarks);
Internals internalStudent = new Internals(usn, name, sem,
External externalStudent = new External(usn, name, sem, externalMarks);

internalStudent.printInternalMarks();
external.printExternalMarks();

printFinalMarks(internalStudent, externalStudent);
}

scanner.close();
}

public static void printFinalMarks(Internals internal, External
external){
```

```
int[] internalMarks = internal.internalMarks;
int[] externalMarks = external.externalMarks;
int totalMarks;

System.out.println("Final Marks for " + internal.getName() + "
("+ internal.getUsn() + "):");
for(int i=0; i< internalMarks.length ; i++){
totalMarks = internalMarks[i]+externalMarks[i];
  System.out.println("Course" + (i+1) + ": " + totalMarks);
    }
  }
}
```

O/P

Enter the number of students : 1

Enter details for Student 1 :

USN: 1BM23CS056

Name : Ayaan Shrestha

Sem: 3

Enter internal marks for 5 courses:

20 21 23 18 25

Enter SEE marks for 5 courses:

72 73 70 69 60

Final Marks of Students :

USN : 1BM23CS056

Name : Ayaan Shrestha

sem : 3

P.T.O →

Name() + "

Marks};

Bafna Gold
Date: Page:

Final Marks:
course 1 : 92
course 2 : 94
course 3 : 93
course 4 : 87
course 5 : 85 //

P.T.O →

Code :

```
package SEE;

import CIE.Student;

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


    public External(String usn, String name, int sem, int[] externalMarks) {
        super(usn, name, sem);
        this.externalMarks = externalMarks;
    }


    public void printExternalMarks() {
        System.out.println("External Marks for " + name + " (" + usn + "):");
        for (int i = 0; i < externalMarks.length; i++) {
            System.out.println("Course " + (i + 1) + ": " + externalMarks[i]);
        }
    }
}
```

```
package CIE;

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


    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }


    public void printInternalMarks() {
        System.out.println("Internal Marks for " + name + " (" + usn + "):");
        for (int i = 0; i < internalMarks.length; i++) {
            System.out.println("Course " + (i + 1) + ": " + internalMarks[i]);
        }
    }
}
```

```
import CIE.Internals;
import SEE.External;

import java.util.Scanner;

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


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

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

            System.out.println("\nEnter details for Student " + (i + 1) + ":");

            System.out.print("Enter USN: ");
            String usn = scanner.nextLine();

            System.out.print("Enter Name: ");
            String name = scanner.nextLine();

            System.out.print("Enter Semester: ");
            int sem = scanner.nextInt();


            int[] internalMarks = new int[5];
            System.out.println("Enter Internal Marks for 5 Courses:");
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                internalMarks[j] = scanner.nextInt();
            }


            int[] externalMarks = new int[5];
            System.out.println("Enter External Marks for 5 Courses:");
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                externalMarks[j] = scanner.nextInt();
            }
            scanner.nextLine();


            Internals internalStudent = new Internals(usn, name, sem, internalMarks);
            External externalStudent = new External(usn, name, sem, externalMarks);
```

```
            internalStudent.printInternalMarks();
            externalStudent.printExternalMarks();


            printFinalMarks(internalStudent, externalStudent);
        }

        scanner.close();
    }


    public static void printFinalMarks(Internals internal, External external) {
        int[] internalMarks = internal.internalMarks;
        int[] externalMarks = external.externalMarks;
        int totalMarks;

        System.out.println("Final Marks for " + internal.getName() + " (" + internal.getUsn() + "):");
        for (int i = 0; i < internalMarks.length; i++) {
            totalMarks = internalMarks[i] + externalMarks[i];
            System.out.println("Course " + (i + 1) + ": " + totalMarks);
        }
    }
}
```

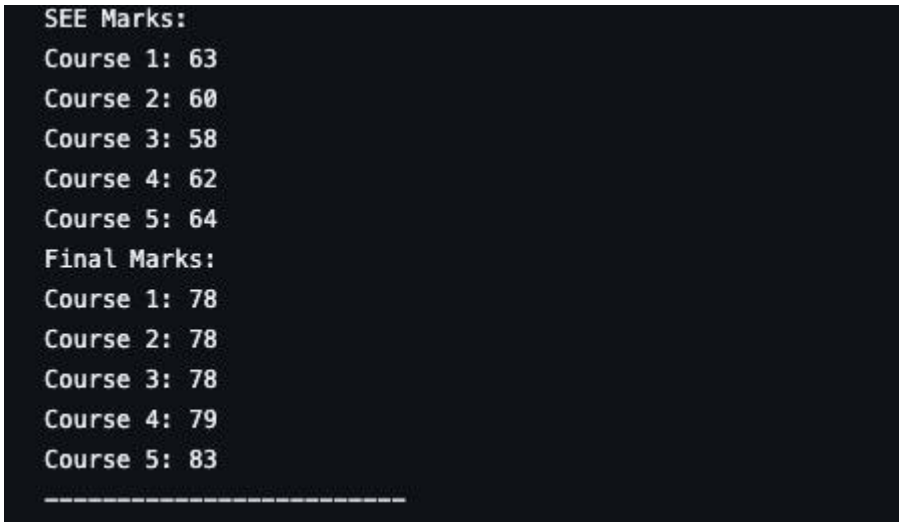
```
package CIE;

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


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

    // Getter methods
    public String getUsn() {
        return usn;
    }

    public String getName() {
        return name;
```

```
    }

    public int getSem() {
        return sem;
    }
}
```

Output :

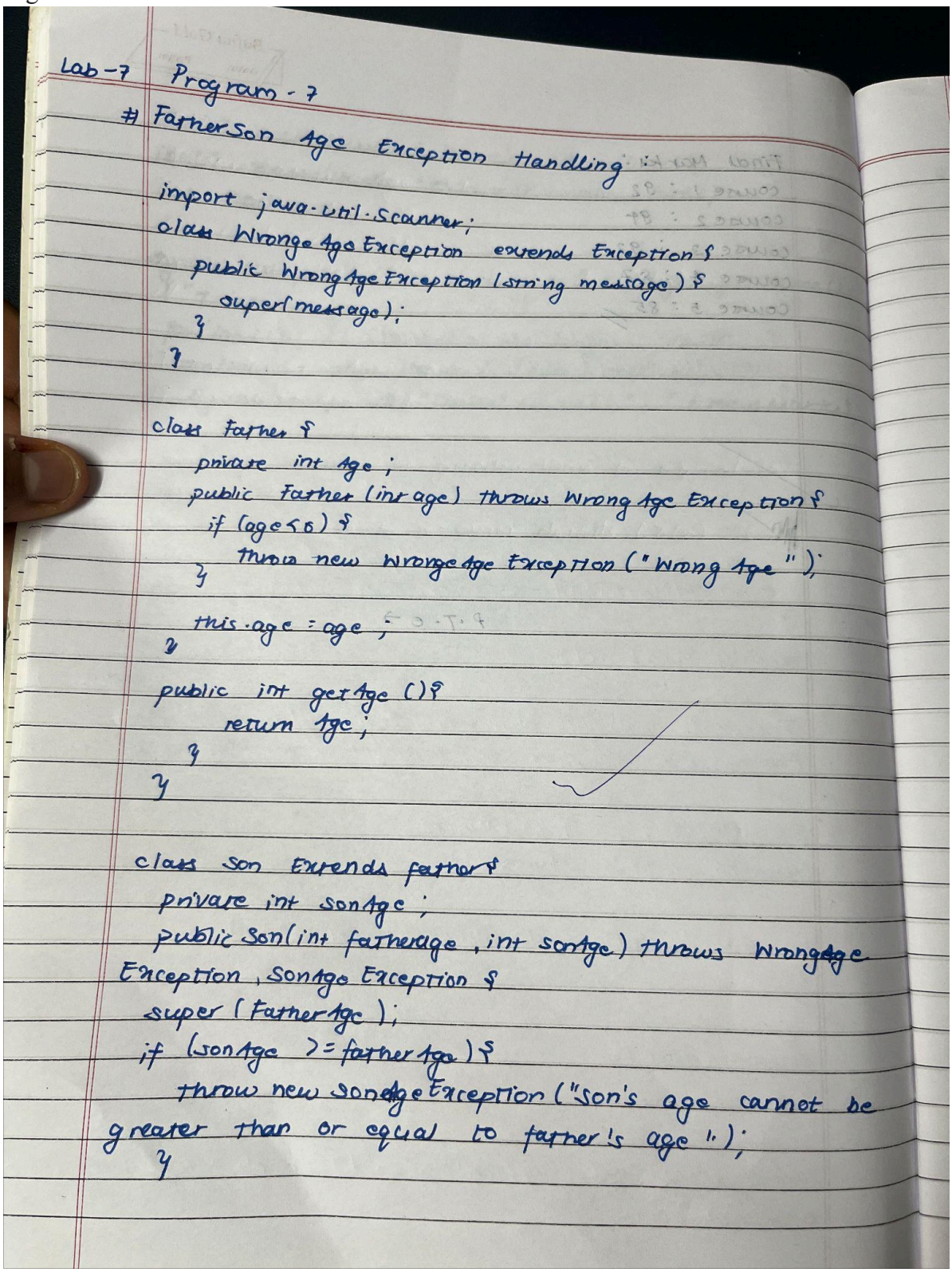
```
Enter the number of students: 2

Enter details for student 1:
USN: 1BM23CS056
Name: Ayaan Shrestha
Semester: 3
Enter internal marks for 5 courses:
18 20 17 19 16
Enter SEE marks for 5 courses:
60 58 62 55 59

Enter details for student 2:
USN: 1BM23CS058
Name: Ayush Ranjan
Semester: 3
Enter internal marks for 5 courses:
15 18 20 17 19
Enter SEE marks for 5 courses:
63 60 58 62 64
```

```
Final Marks of Students:

USN: 1BM23CS056
Name: Ayaan Shrestha
Semester: 3
Internal Marks:
Course 1: 18
Course 2: 20
Course 3: 17
Course 4: 19
Course 5: 16
SEE Marks:
Course 1: 60
Course 2: 58
Course 3: 62
Course 4: 55
Course 5: 59
Final Marks:
Course 1: 78
Course 2: 78
Course 3: 79
Course 4: 74
Course 5: 75
------------------------

USN: 1BM23CS058
Name: Ayush Ranjan
Semester: 3
Internal Marks:
Course 1: 15
Course 2: 18
Course 3: 20
Course 4: 17
Course 5: 19
SEE Marks:
Course 1: 63
Course 2: 60
Course 3: 58
Course 4: 62
Course 5: 64
```


SEE Marks:
Course 1: 63
Course 2: 60
Course 3: 58
Course 4: 62
Course 5: 64
Final Marks:
Course 1: 78
Course 2: 78
Course 3: 78
Course 4: 79
Course 5: 83

Program No : 7
Implement Exception Handling

Algorithm :



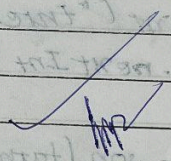
```
Lab-7  Program - 7
# FatherSon Age Exception Handling

import java.util.Scanner;
class WrongAgeException extends Exception {
  public WrongAgeException(String message) {
    super(message);
  }
}

class Father {
  private int age;
  public Father(int age) throws WrongAgeException {
    if (age<0) {
      throw new WrongAgeException("Wrong Age");
    }
    this.age = age;
  }
  public int getAge() {
    return age;
  }
}

class Son Extends father {
  private int sonAge;
  public Son(int fatherage, int sonAge) throws WrongAge
Exception, SonAge Exception {
  super(FatherAge);
  if (sonAge >= fatherAge) {
    throw new SonAgeException("son's age cannot be
greater than or equal to father's age.");
  }
```

```
        this.sonAge = sonAge;
    }
    public int getSonAge() {
        return sonAge;
    }
}

public class FatherSon {
    public static void main(String[] args) {
        while(true) {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter Father's Age : ");
            int fatherAge = sc.nextInt();
            System.out.print("Enter Son's Age:");
            int sonAge = sc.nextInt();
            try {
                Son son = new Son(fatherAge, sonAge);
                System.out.println("Age Accepted Successfully");
                break;
            }
            catch (WrongAgeException e) {
                System.out.println(e.getMessage());
            }
            catch (SonAgeException e) {
                System.out.println(e.getMessage());
            }
            System.out.println("Would you like to re-enter
            details (Y/n)");
            String input = sc.next();
            if (input.equalsIgnoreCase("n")) {
                break;
            }
        }
    }
}
```

O/P

Enter Father's Age : 51
Enter son's Age : 19
Accepted Successfully
Would you like to re-enter details (Y/n)
y

Enter Father's Age : 20
Enter son's Age : 21
~~Accepted Successfully~~
Son's age cannot be greater than or equal to father's age
Would you like to re-enter details (Y/n)
n



Code :

```
import java.util.Scanner;
class WrongAgeException extends Exception {
   public WrongAgeException(String message) {
      super(message);
   }
}

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

class Father {
   private int age;
   public Father(int age) throws WrongAgeException {
      if (age < 0) {
         throw new WrongAgeException("Wrong age");
      }
      this.age = age;
   }
   public int getAge() {
      return age;
   }
}

class Son extends Father {
   private int sonAge;
   public Son(int fatherAge, int sonAge) throws WrongAgeException,
SonAgeException {
      super(fatherAge);
      if (sonAge >= fatherAge) {
         throw new SonAgeException("Son's age cannot be greater than or equal
to father's age");
      }
      this.sonAge = sonAge;
   }
   public int getSonAge() {
      return sonAge;
   }
}
public class FatherSon{
   public static void main(String[] args) {
      while(true){
         Scanner sc = new Scanner(System.in);
```

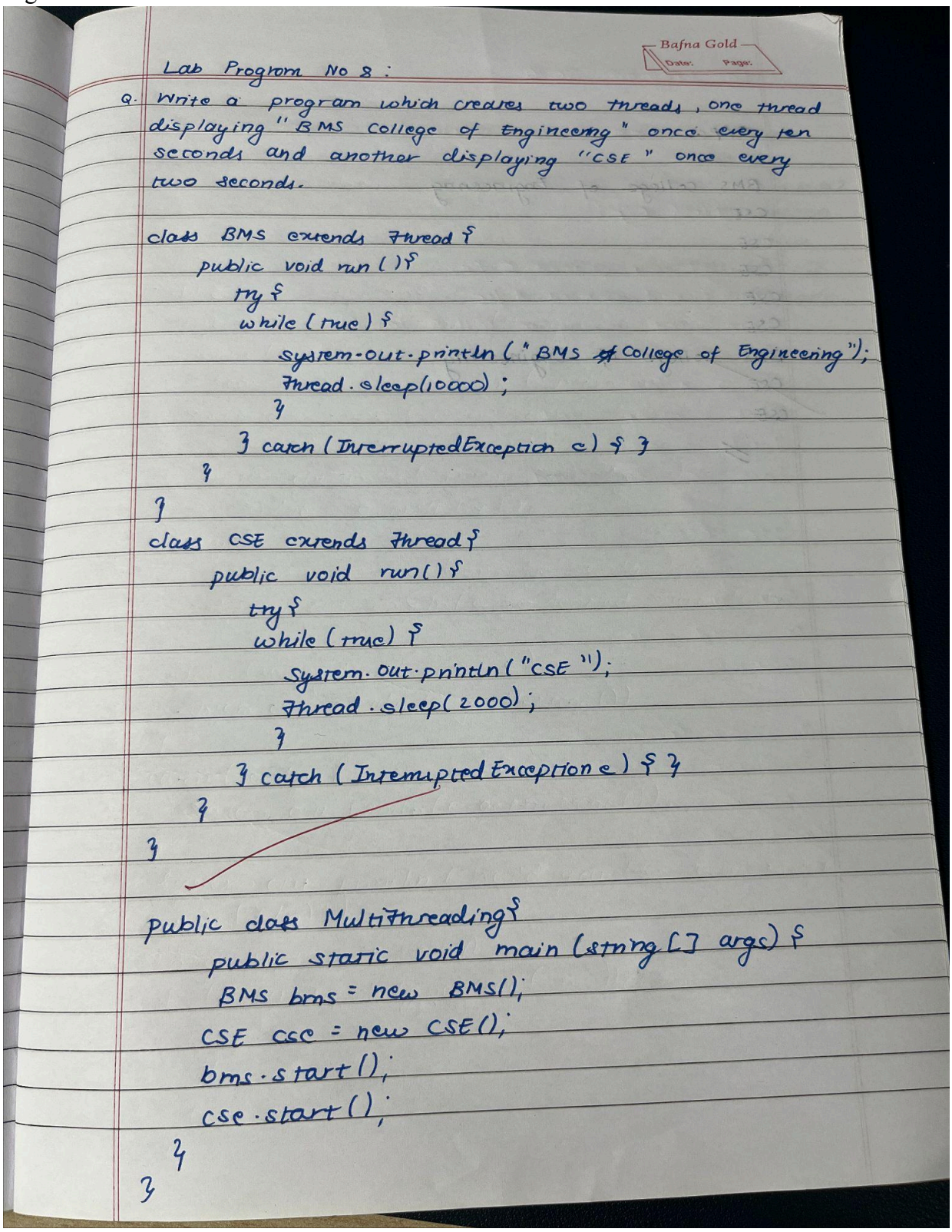
```
            System.out.print("Enter Father's Age: ");
            int fatherAge = sc.nextInt();
            System.out.print("Enter Son's Age: ");
            int sonAge = sc.nextInt();
            try {
                Son son = new Son(fatherAge, sonAge);
                System.out.println("Accepted Succesfully");
            }
            catch (WrongAgeException e) {
                System.out.println(e.getMessage());
            }
            catch (SonAgeException e) {
                System.out.println(e.getMessage());
            }
            System.out.println("Would you like to re-enter details (Y/n)");
            String input = sc.next();
            if (input.equalsIgnoreCase("n")) {
                break;
            }
        }
    }
}
```

Output:

```
C:\Users\Admin\Desktop>javac FatherSon.java

C:\Users\Admin\Desktop>java FatherSon
Enter Father's Age: 51
Enter Son's Age: 19
Accepted Succesfully
Would you like to re-enter details (Y/n)
Y
Enter Father's Age: 24
Enter Son's Age: 25
Son's age cannot be greater than or equal to father's age
Would you like to re-enter details (Y/n)
n
```

Program No : 8
Multithreading, Creating Threads in Java

Algorithm :



Lab Program No 8 :

Q. 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.

```
class BMS extends Thread {
    public void run() {
        try {
        while (true) {
            System.out.println("BMS College of Engineering");
            Thread.sleep(10000);
            }
        } catch (InterruptedException e) { }
    }
}
class CSE extends Thread {
    public void run() {
        try {
        while (true) {
            System.out.println("CSE");
            Thread.sleep(2000);
            }
        } catch (InterruptedException e) { }
    }
}

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

O/P :

BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE

//

Code :

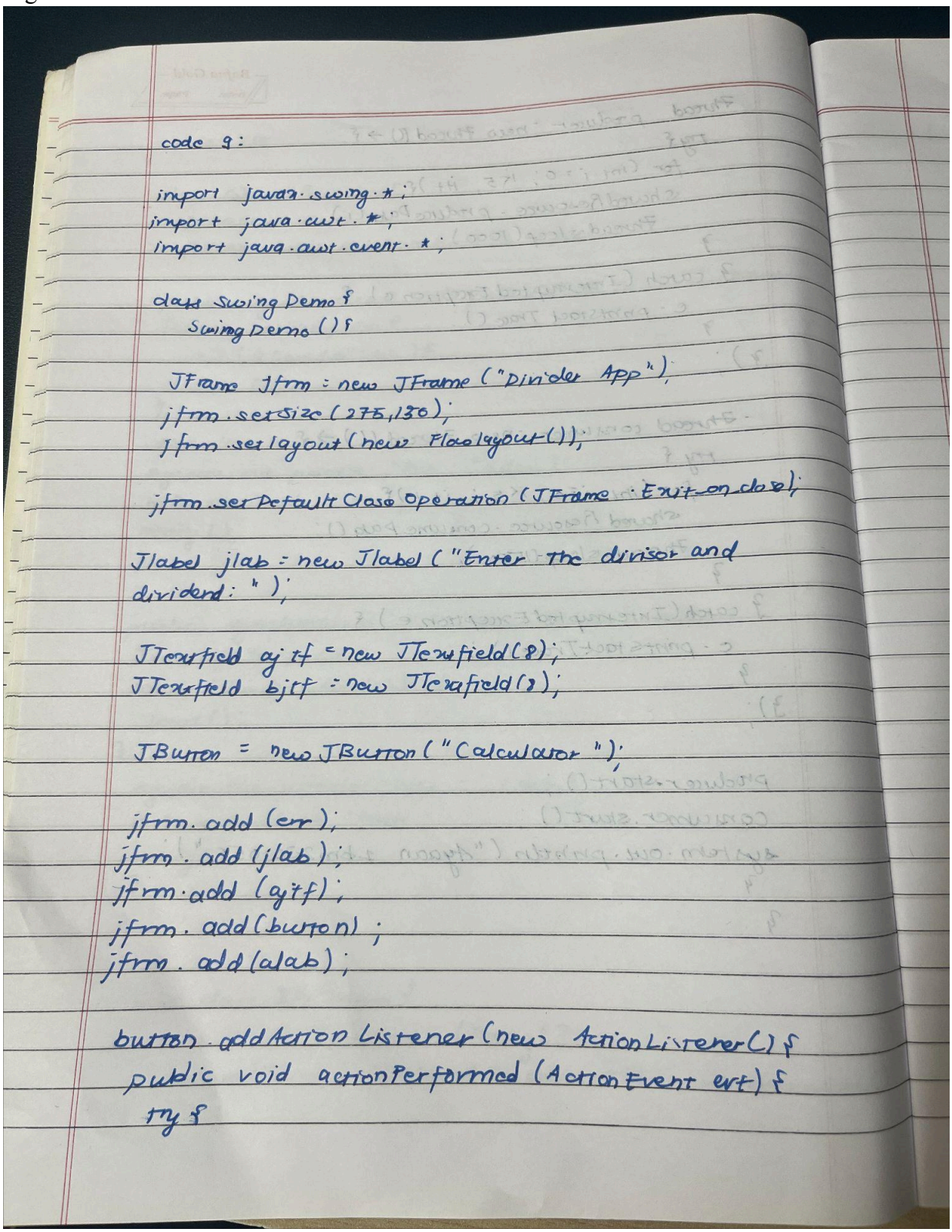
```
class BMS extends Thread {
   public void run() {
      try {
         while (true) {
            System.out.println("BMS College of Engineering");
            Thread.sleep(10000); // Sleep for 10 seconds
         }
      }catch (InterruptedException e) {}
   }
}
class CSE extends Thread {
   public void run() {
      try {
         while (true) {
            System.out.println("CSE");
            Thread.sleep(2000); // Sleep for 2 seconds
         }
      }catch (InterruptedException e) {}
   }
}

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

Output:

```
C:\Users\Admin\Desktop>java Multithreading
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
^C
C:\Users\Admin\Desktop>
```

Program No : 9
Interface to Perform Integer Division

Algorithm :



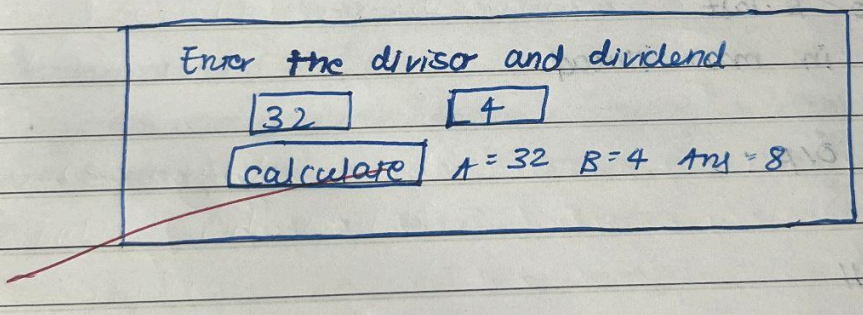
```
code 9:

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

class SwingDemo {
   SwingDemo() {

    JFrame jfrm = new JFrame("Divider App");
    jfrm.setSize(275,130);
    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 = new JButton("Calculator");

jfrm.add(err);
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(button);
jfrm.add(alab);

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("B should be Non-zero!");
    }
  }
});

    jfrm.setVisible(true);
}

public static void main(String args[]) {

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

Lab Program No: 9

Q. WAP that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num 1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num 1 or Num 2 were not an integer, The program would throw a NumberFormatException. If Num 2 were 0, The program would throw an arithmetic Exception

O/P

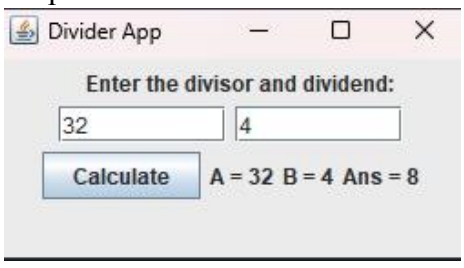


Code :

```
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);


      button.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent evt) {
            try {

               int a = Integer.parseInt(ajtf.getText());
```

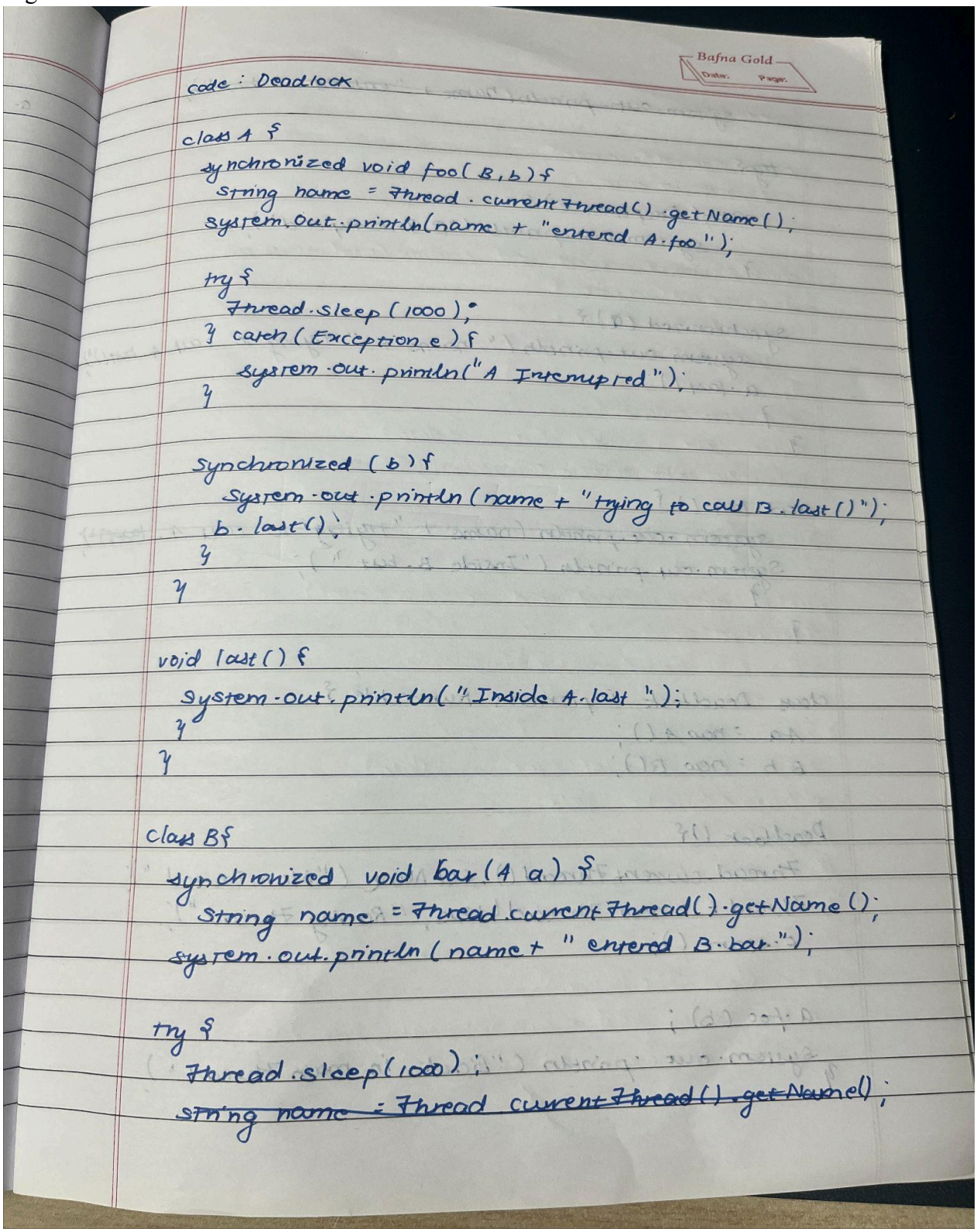
```
                int b = Integer.parseInt(bjtf.getText());


                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[]) {

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

Output :



Program 10.1
Implement Deadlock

Algorithm :



code : Deadlock

```
class A {
  synchronized void foo(B, b){
   String name = Thread.currentThread().getName();
   System.Out.println(name + "entered A.foo");

   try{
    Thread.sleep(1000);
   } catch(Exception e){
     System.out.println("A Interupted");
   }

   Synchronized (b){
     System.out.println(name + "trying to call B.last()");
     b.last();
   }
  }

  void last(){
    System.out.println("Inside A.last");
  }
}

class B{
  synchronized void bar(A a){
    String name = Thread.current Thread().getName();
   system.out.println(name + " entered B.bar");

  try{
    Thread.sleep(1000);
    String name = Thread current Thread().getName();
```

```
    system.out.println(name + "entered B.bar");
    try{
    }
    catch(Exception e){
        system.out.println("B Interrupted");
    }

    synchronized(A){
        system.out.println(" name + " trying to call A.last()");
        a.last();
    }
}

void last(){
    system.out.println(name + "trying to call A.last()");
    System.out.println("Inside B.bar");
}
}

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();
}
}
```

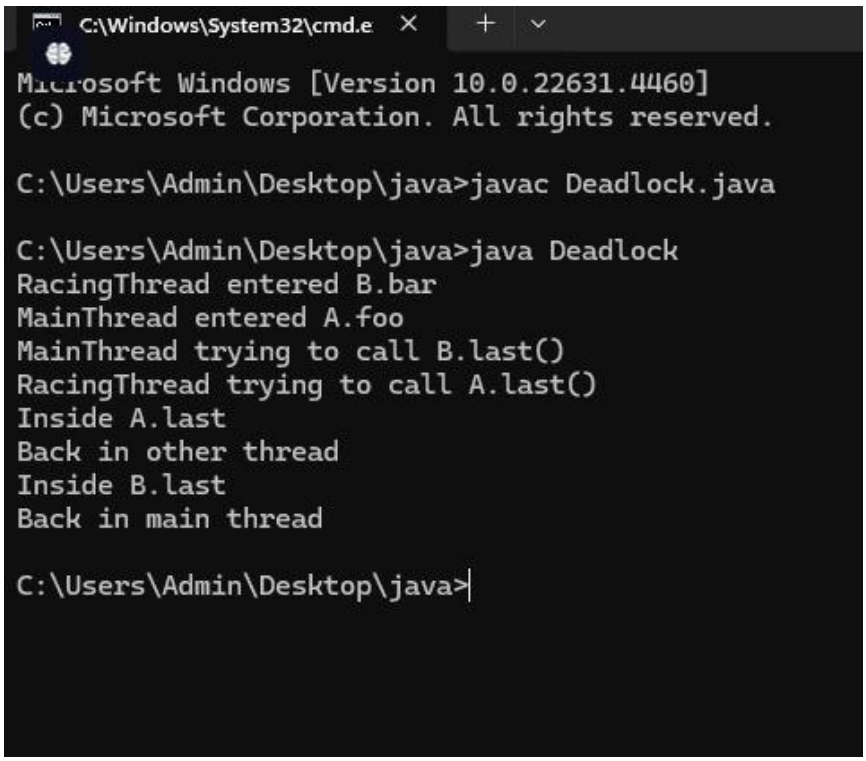
~~code~~

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");
      }


      synchronized (b) {
         System.out.println(name + " trying to call B.last()");
         b.last();
      }
   }

   void last() {
      System.out.println("Inside A.last");
   }
}

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

      try {
         Thread.sleep(1000);
      } catch (Exception e) {
         System.out.println("B Interrupted");
      }


      synchronized (a) {
         System.out.println(name + " trying to call A.last()");
         a.last();
      }
   }

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

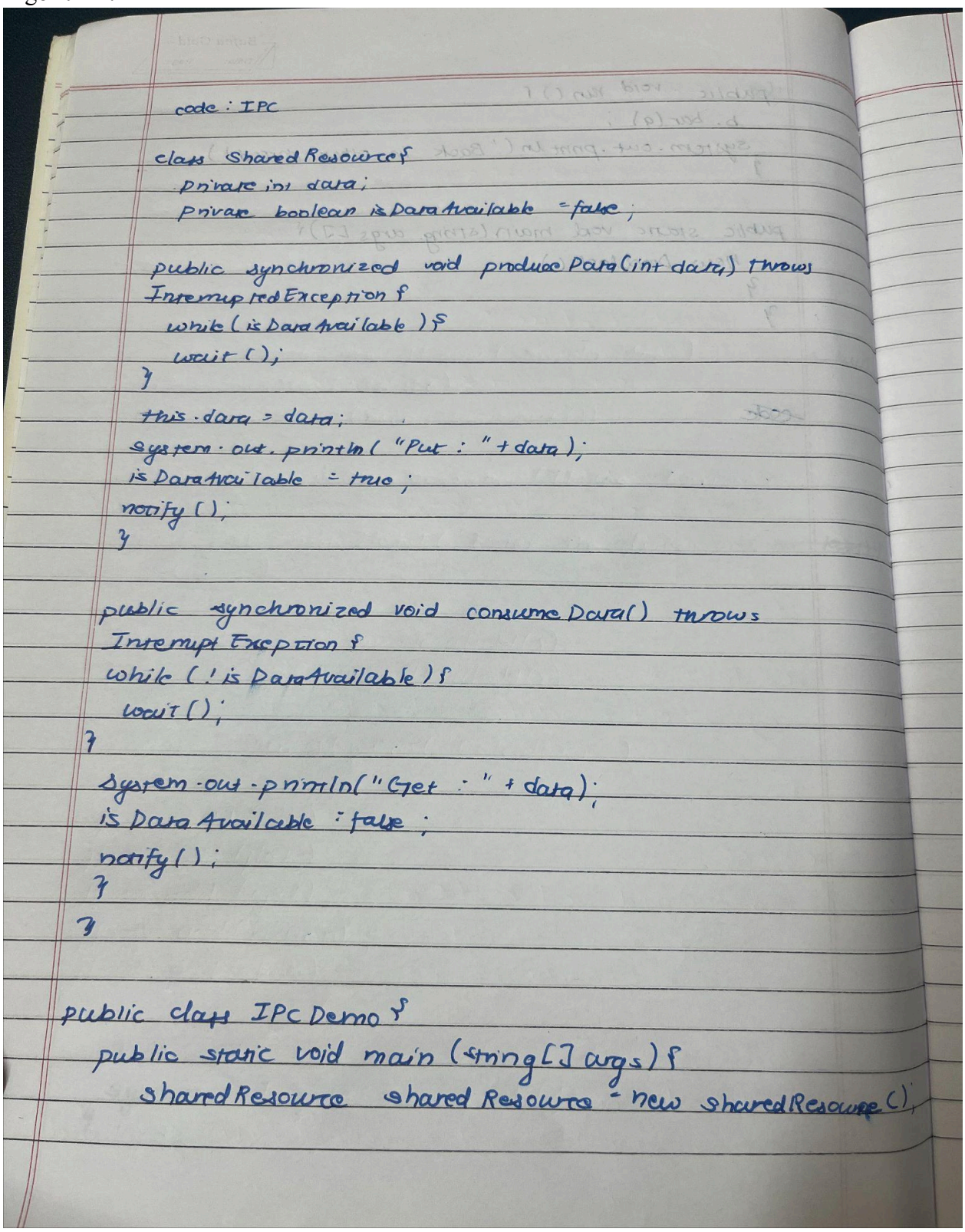
```
class Deadlock implements Runnable {
   A a = new A();
   B b = new B();

   Deadlock() {
      Thread.currentThread().setName("MainThread");
      Thread t = new Thread(this, "RacingThread");
      t.start();

      a.foo(b);
      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:

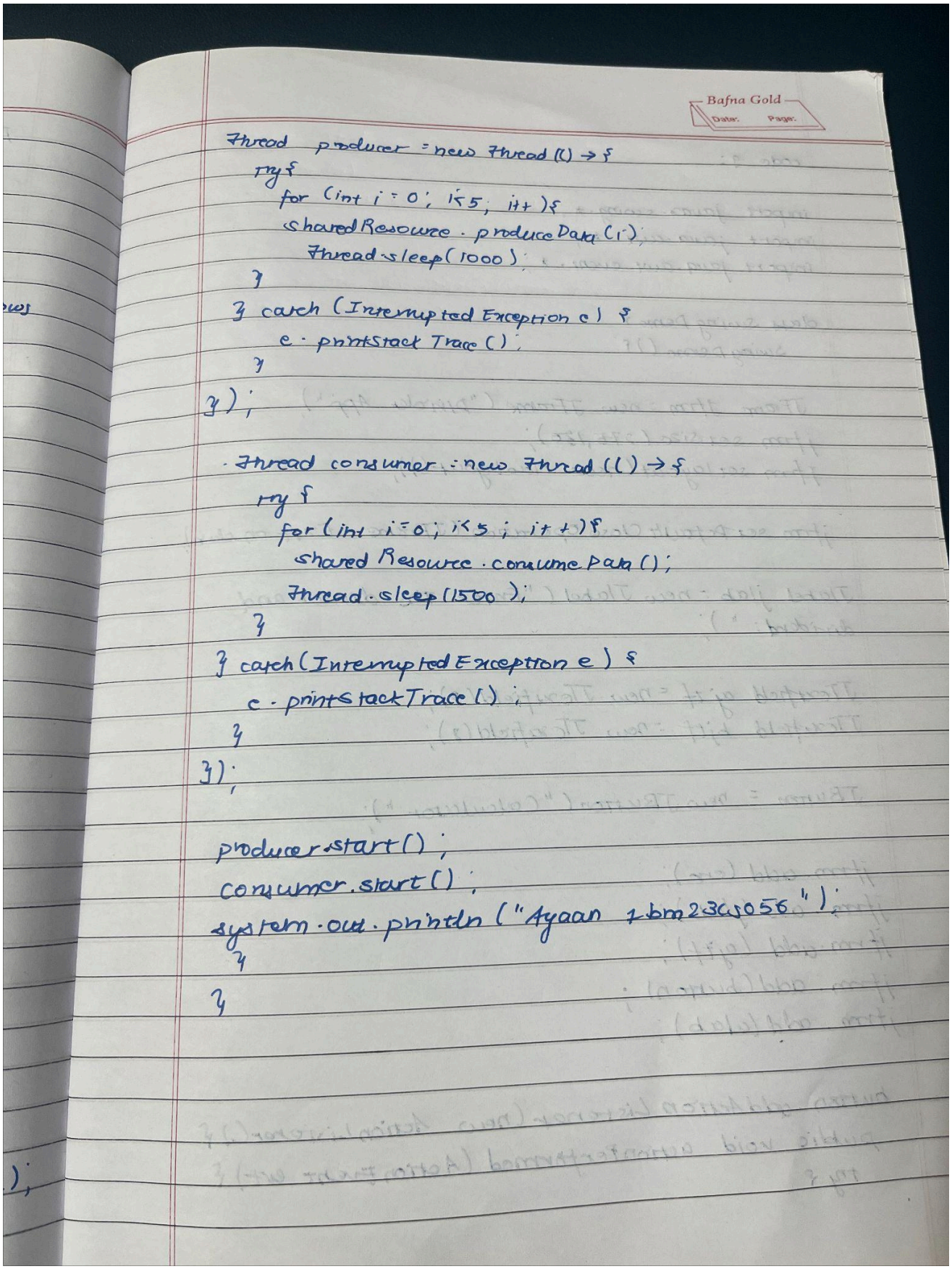


Program 10.2 :
Implement Inter-process Communication

Algorithm :



```
code : IPC

class SharedResource{
  private int data;
  private boolean isDataAvailable = false;

  public synchronized void produceData(int data) throws
  InterruptedException{
    while(isDataAvailable){
    wait();
  }
  this.data = data;
  System.out.println("Put : " + data);
  isDataAvailable = true;
  notify();
  }

  public synchronized void consumeData() throws
  InterruptException{
  while(!isDataAvailable){
    wait();
  }
  System.out.println("Get : " + data);
  isDataAvailable : false;
  notify();
  }
}

public class IPCDemo{
  public static void main(String[] args){
    SharedResource sharedResource = new SharedResource();
```



```
Thread producer = new Thread(() -> {
    try {
        for (int i = 0; i < 5; i++) {
            sharedResource.produceData(i);
            Thread.sleep(1000);
        }
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
});

Thread consumer = new Thread(() -> {
    try {
        for (int i = 0; i < 5; i++) {
            sharedResource.consumeData();
            Thread.sleep(1500);
        }
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
});

producer.start();
consumer.start();
System.out.println("Ayaan 1bm23cs056");
    }
}
```

Lab Program No 10

- Demonstrate Inter process Communication and deadlock

Deadlock O/p:

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

IPC O/P:

Got : 11
Consumed : 11
Put : 12

Intimate Consumer
Producer waiting
Got : 12
Consumed : 12
Put : 13

9/11/24

//

Code :

```
class SharedResource {
   private int data;
   private boolean isDataAvailable = false;


   public synchronized void produceData(int data) throws InterruptedException {
      while (isDataAvailable) {
         wait();
      }
      this.data = data;
      System.out.println("Put: " + data);
      isDataAvailable = true;
      notify();
   }


   public synchronized void consumeData() throws InterruptedException {
      while (!isDataAvailable) {
         wait();
      }
      System.out.println("Get: " + data);
      isDataAvailable = false;
      notify();
   }
}

public class IPCDemo {
   public static void main(String[] args) {
      SharedResource sharedResource = new SharedResource();


      Thread producer = new Thread(() -> {
         try {
            for (int i = 0; i < 5; i++) {
               sharedResource.produceData(i);
               Thread.sleep(1000);
            }
         } catch (InterruptedException e) {
            e.printStackTrace();
         }
      });


      Thread consumer = new Thread(() -> {
         try {
```

```
            for (int i = 0; i < 5; i++) {
                sharedResource.consumeData();
                Thread.sleep(1500);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    });

    producer.start();
    consumer.start();
    System.out.println("Ayaan 1bm23cs056");
  }
}
```

OutPut: