VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Alok (1BM23CS024)

in partial fulfilment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Sep-2024 to Jan-2025

B.M.S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "Object Oriented Java Programming (23CS3PCOOJ)" carried out by **Alok (1BM23CS024),** who is a bonafide student of **B.M.S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

Basavaraj Jakkali Associate Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

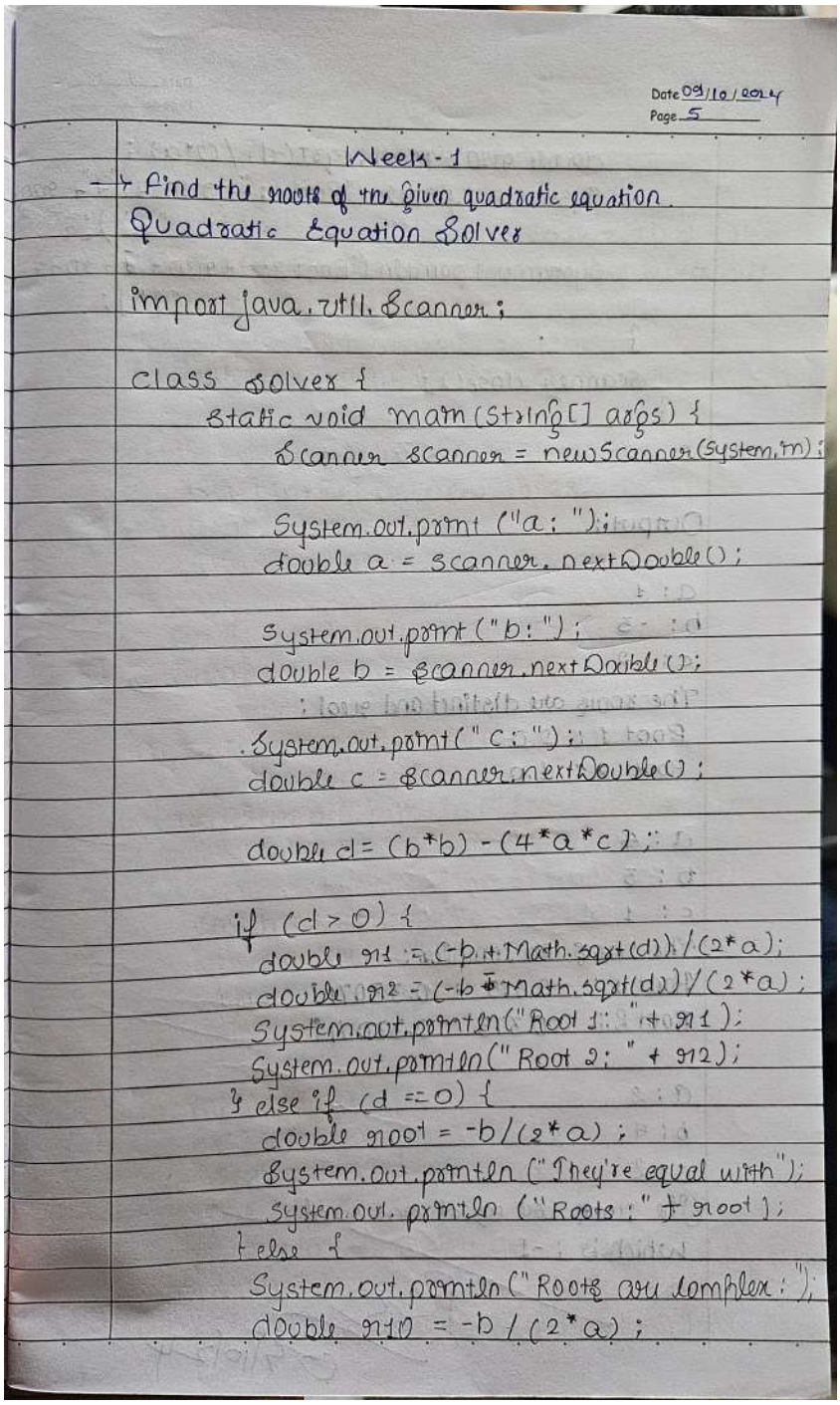
Index

Git-hub Link:
github.com/alok-aeri/BMSCE/tree/main/OOJP

Program 1

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

Observation Book



Date 09/10/2024
Page 5

Week-1

→ Find the roots of the given quadratic equation.

Quadratic Equation Solver

```
import java.util.Scanner;

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

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

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

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

        double d = (b*b) - (4*a*c);

        if (d > 0) {
            double r1 = (-b + Math.sqrt(d)) / (2*a);
            double r2 = (-b - Math.sqrt(d)) / (2*a);
            System.out.println("Root 1: " + r1);
            System.out.println("Root 2: " + r2);
        } else if (d == 0) {
            double root = -b/(2*a);
            System.out.println("They're equal with");
            System.out.println("Roots: " + root);
        } else {
            System.out.println("Roots are complex:");
            double r10 = -b / (2*a);
```

```
        double r20 = Math.sqrt(-d) / (2*a);
        System.out.println("Root 1: " + r10 + " + " + r20 + "i");
        System.out.println("Root 2: " + r10 + " - " + r20 + "i");
    }
    scanner.close();
}
```

Output:

a: 1
b: -3
c: 2
The roots are distinct and real:
Root 1: 2.0
Root 2: 1.0

a: 4
b: 3
c: 1
The roots are complex:
Root 1: -0.375 + 0.33071i
Root 2: -0.375 - 0.33071i

a: 2
b: 4
c: 2
The roots are equal and real:
Which is: -1

Record

Source Code

```
import java.util.Scanner;

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

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

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

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

      double d = (b * b)- (4 * a * c);

      if (d > 0) {
         double r1 = (-b + Math.sqrt(d)) / (2 * a);
         double r2 = (-b - Math.sqrt(d)) / (2 * a);
         System.out.println("The roots are distinct and real:");
         System.out.println("Root 1: " + r1);
         System.out.println("Root 2: " + r2);
      } else if (d == 0) {
         double root = -b / (2 * a);
         System.out.println("The roots are equal and real:");
         System.out.println("which is: " + root);
      } else {
         System.out.println("The roots are complex:");
         double r10 = -b / (2 * a);
         double r20 = Math.sqrt(-d) / (2 * a);
         System.out.println("Root 1: " + r10 + " + " + r20 + "i");
         System.out.println("Root 2: " + r10 + " - " + r20 + "i");
      }
      scanner.close();
   }
}
```

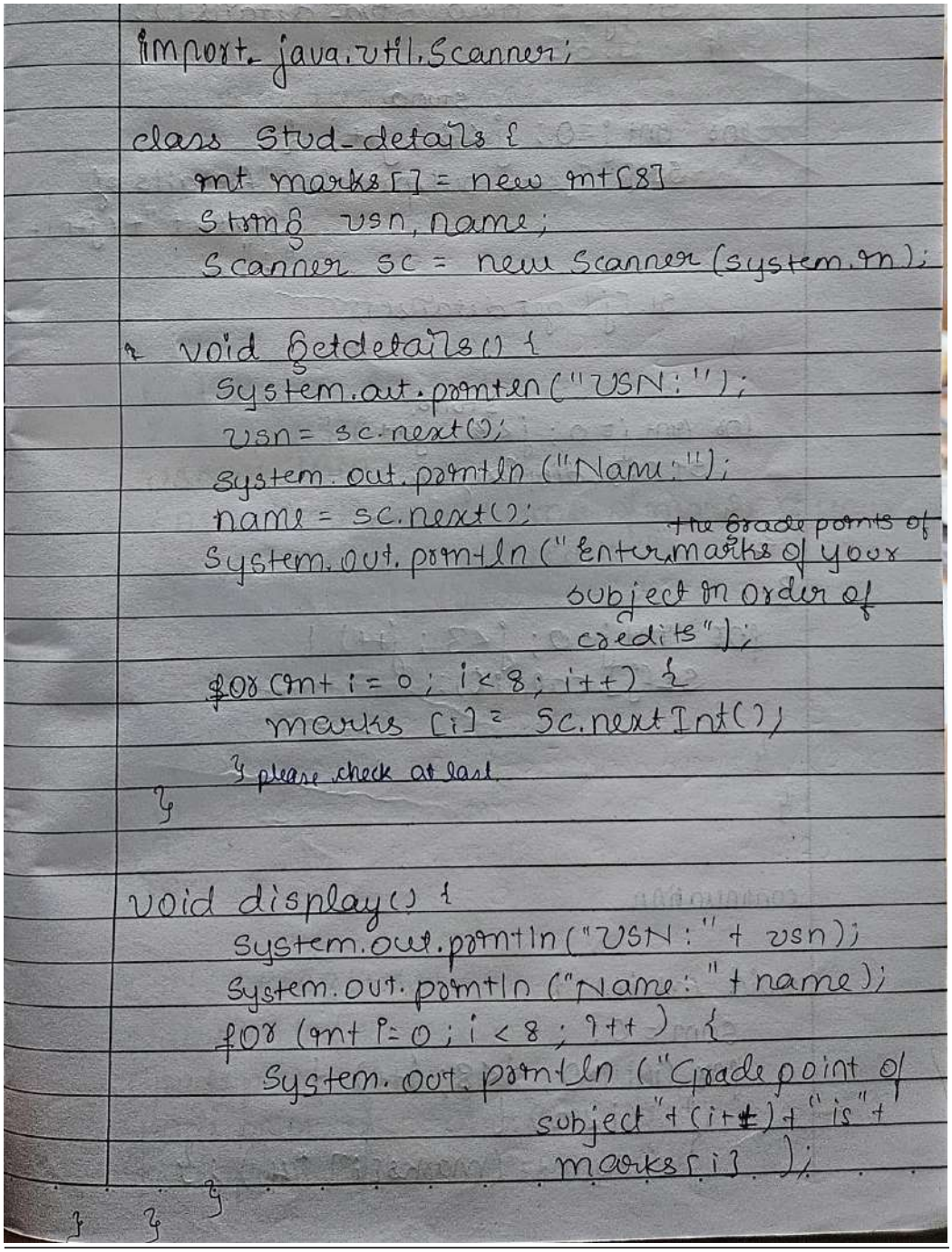
Output:

```
C:\Users\Admin\.vscode\23CS024\OOJP\Week1>javac EquationSolver.java

C:\Users\Admin\.vscode\23CS024\OOJP\Week1>java EquationSolver
a: 1
b: -3
c: 2
The roots are distinct and real:
Root 1: 2.0
Root 2: 1.0

C:\Users\Admin\.vscode\23CS024\OOJP\Week1>java EquationSolver
a: 4
b: 3
c: 1
The roots are complex:
Root 1: -0.375 + 0.330718913883073841
Root 2: -0.375 - 0.330718913883073841

C:\Users\Admin\.vscode\23CS024\OOJP\Week1>java EquationSolver
a: 2
b: 4
c: 2
The roots are equal and real:
which is: -1.0

Name : ALOK
USN  : 1BM23CS024
```

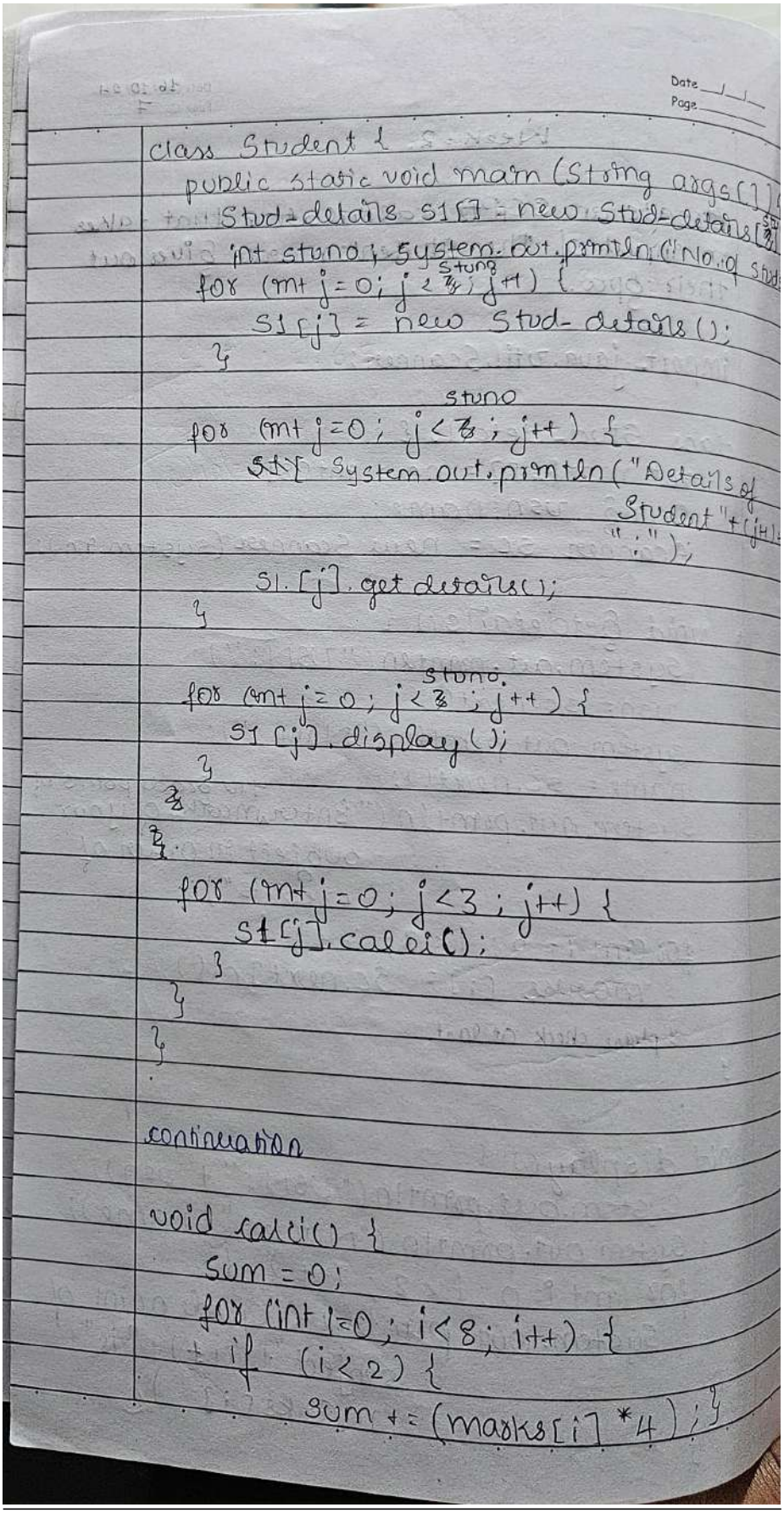
Program 2

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

Observation Book



```
import java.util.Scanner;

class Stud_details {
    int marks[] = new int[8]
    String usn, name;
    Scanner sc = new Scanner(System.in);

    void getdetails() {
        System.out.println("USN:");
        usn = sc.next();
        System.out.println("Name:");
        name = sc.next();
        System.out.println("Enter the Grade points of marks of your
                            subject in order of
                            credits");
        for(int i = 0; i<8; i++) {
            marks[i] = sc.nextInt();
        } please check at last.
    }

    void display() {
        System.out.println("USN:" + usn);
        System.out.println("Name:" + name);
        for(int i=0; i<8; i++) {
            System.out.println("Grade point of
                subject" + (i+1) + "is" +
                marks[i]);
        }
    }
}
```



```
class Student {
  public static void main (String args[]) {
    Stud_details S1[] = new Stud_details[stuno];
    int stuno; System.out.println("No. of stud
    for (int j=0; j<stuno; j++) {
      S1[j] = new Stud_details();
    }

    for (int j=0; j<stuno; j++) {
      S1[ System.out.println("Details of
                              Student "+(j+1)+
                              " :");
      S1.[j].get_details();
    }

    for (int j=0; j<stuno; j++) {
      S1[j].display();
    }
  }
}

for (int j=0; j<3; j++) {
  S1[j].calci();
}
}

continuation

void calci() {
  sum = 0;
  for (int i=0; i<8; i++) {
    if (i<2) {
      sum += (marks[i] *4);}
```

```
Scanner sc = new Scanner(System.in);
stuno = sc.next();
```

Date / /
Page

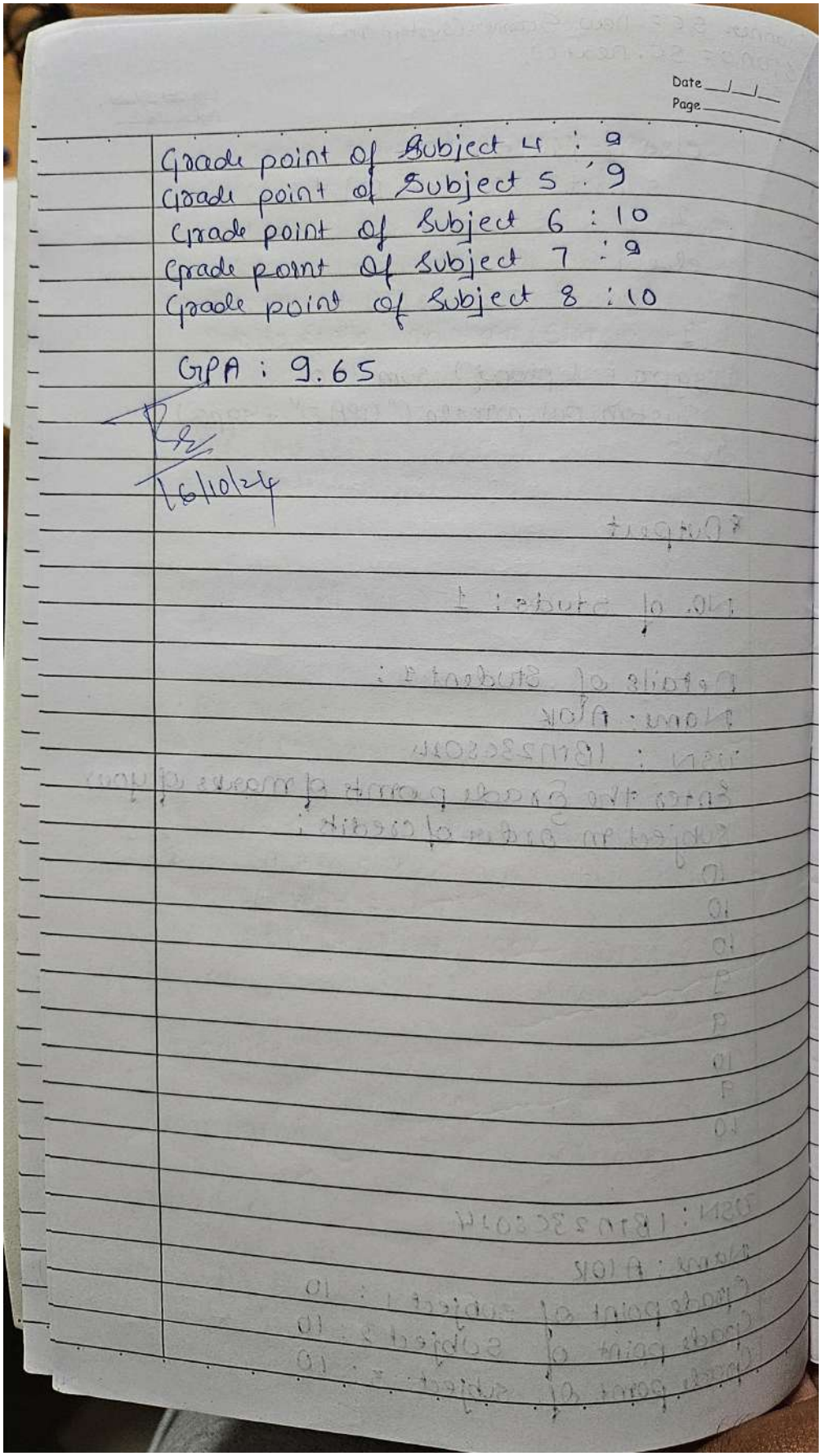
```
else if (i<5) {
    sum += (marks[i] * 3);
}
else {
    sum += marks(i);
}
sgpa = (float) sum/20;
System.out.println("GPA =" + sgpa);
}
```

*Output

NO. of studs: 1

Details of Student 1:
Name: Alok
USN : 1BM23CS024
Enter the Grade points of marks of your subject in order of credits:
10
10
10
9
9
10
9
10

USN : 1BM23CS014
Name : Alok
Grade point of subject 1 : 10
Grade point of subject 2 : 10
Grade point of subject 3 : 10


Date __/__/__
Page _____

Grade point of Subject 4 : 9
Grade point of Subject 5 : 9
Grade point of Subject 6 : 10
Grade point of Subject 7 : 9
Grade point of Subject 8 : 10

GPA : 9.65

16/10/24

Record

Source Code

```
import java.util.Scanner;

class Stud_details {
    int marks[] = new int[8];
    String usn, name;
    int sum = 0;
    float sgpa = 0;
    Scanner sc = new Scanner(System.in);

    void getdetails() {
        System.out.println("Enter the USN:");
        usn = sc.nextLine();
        System.out.println("Enter the name:");
        name = sc.nextLine();
        System.out.println("Enter the marks for 8 subjects:");
        for (int i = 0; i < 8; i++) {
            marks[i] = sc.nextInt();
        }
        sc.nextLine();
    }

    void calci() {
        sum = 0;
        for (int i = 0; i < 8; i++) {
            if (i < 2) {
                sum += (marks[i] * 4);
            } else if (i < 5) {
                sum += (marks[i] * 3);
            } else {
                sum += marks[i];
            }
        }
        sgpa = (float)sum / 20;
        System.out.println("SGPA = " + sgpa);
    }

    void display() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        for (int i = 0; i < 8; i++) {
            System.out.println("Marks for subject " + (i + 1) + ": " + marks[i]);
        }
    }
```

```
}

public class lab2 {
   public static void main(String args[]) {
      System.out.println("Alok 1BM23CS024");

      Stud_details s1[] = new Stud_details[3];

      for (int j = 0; j < 3; j++) {
         s1[j] = new Stud_details();
      }

      for (int j = 0; j < 3; j++) {
         System.out.println("Enter the details for student " + (j + 1) + ":");
         s1[j].getdetails();
      }

      for (int j = 0; j < 3; j++) {
         s1[j].calci();
      }

      for (int j = 0; j < 3; j++) {
         s1[j].display();
      }
   }
}
```

Output:

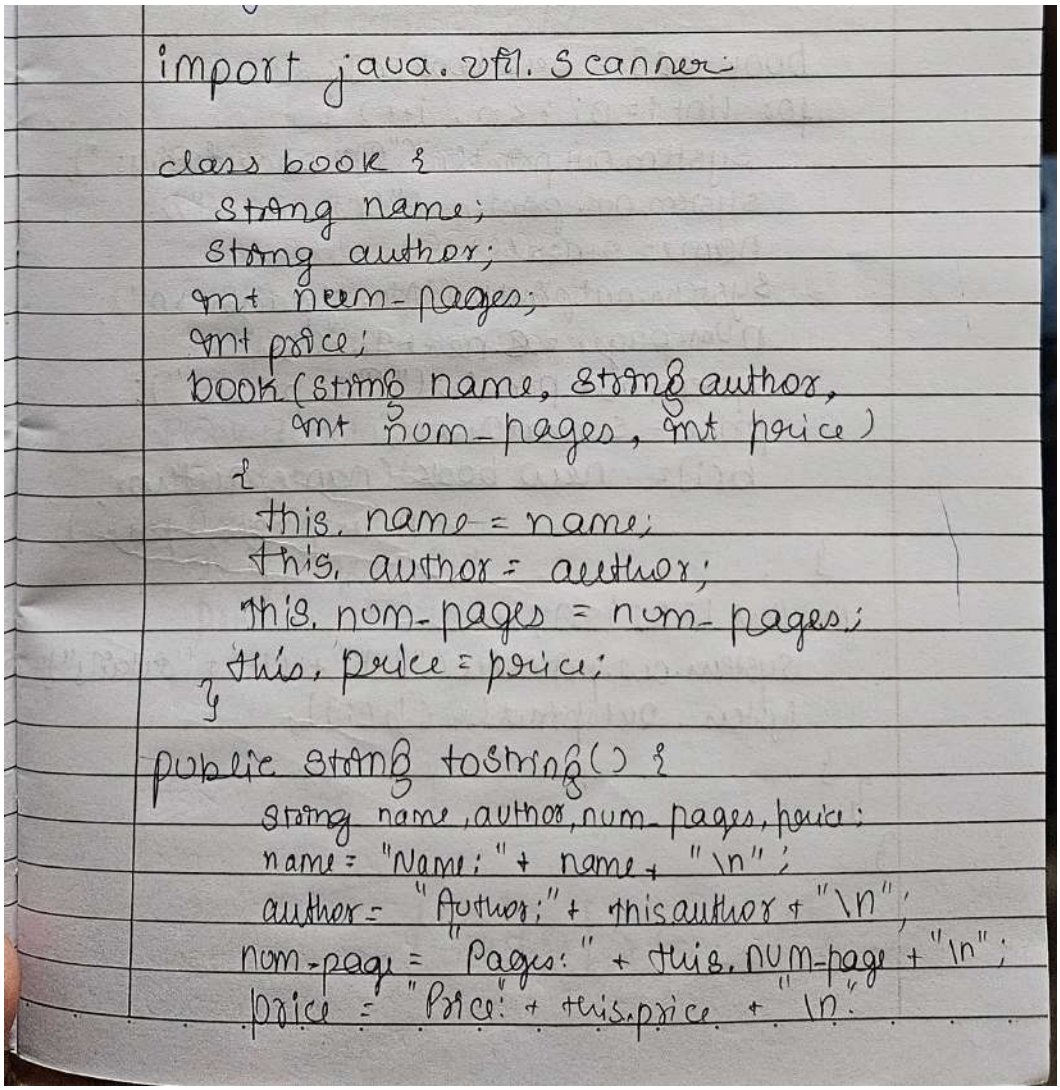
```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.4\lib\idea_rt.jar=59334:C:\Program Files\JetBrains\
Alok 1BM23CS024
Enter the details for student 1:
Enter the USN:
1BM23CS024
Enter the name:
Alok
Enter the marks for 8 subjects:
10
10
9
9
9
10
10
10
Enter the details for student 2:
Enter the USN:
1BM23CS023
Enter the name:
Akshay
Enter the marks for 8 subjects:
10
10
9
9
10
9
10
9
```

```
SGPA = 9.55
SGPA = 9.6
SGPA = 10.5
USN: 1BM23CS024
Name: Alok
Marks for subject 1: 10
Marks for subject 2: 10
Marks for subject 3: 9
Marks for subject 4: 9
Marks for subject 5: 9
Marks for subject 6: 10
Marks for subject 7: 10
Marks for subject 8: 10
USN: 1BM23CS023
Name: Akshay
Marks for subject 1: 10
Marks for subject 2: 10
Marks for subject 3: 9
Marks for subject 4: 9
Marks for subject 5: 10
Marks for subject 6: 9
Marks for subject 7: 10
Marks for subject 8: 9
USN: 1BM23CS968
Name: Bert
Marks for subject 1: 2
Marks for subject 2: 34
Marks for subject 3: 4
Marks for subject 4: 12
Marks for subject 5: 3
Marks for subject 6: 4
Marks for subject 7: 2
Marks for subject 8: 3
1
Process finished with exit code 0
```

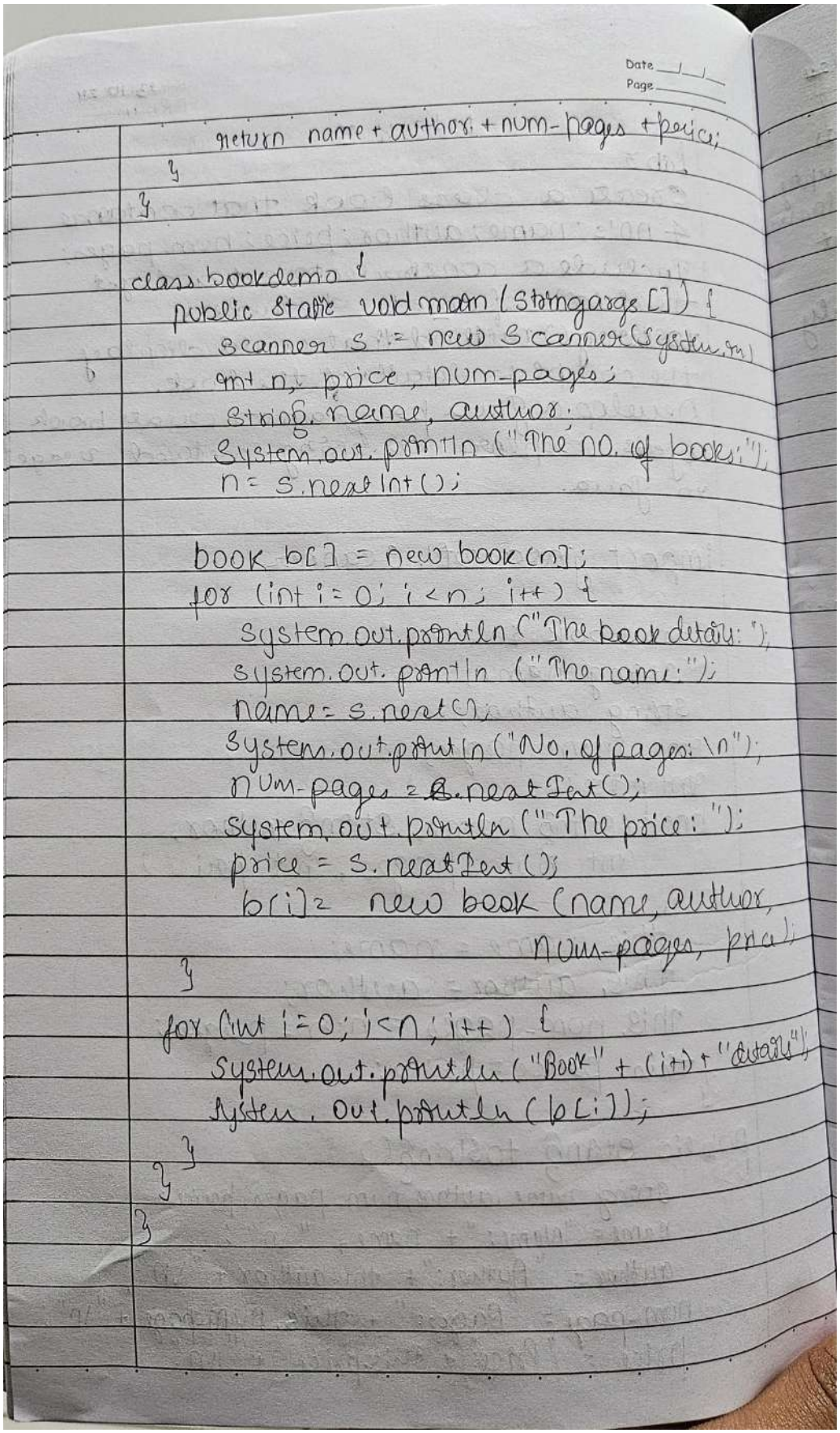
Program 3

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

Observation Book



```
import java.util.Scanner;

class book {
    string name;
    string author;
    int num_pages;
    int price;
    book(string name, string author,
        int num_pages, int price)
    {
        this.name = name;
        this.author = author;
        this.num_pages = num_pages;
        this.price = price;
    }

public string toString() {
        string name,author,num_pages,price;
        name = "Name:" + name + "\n";
        author = "Author:" + this.author + "\n";
        num_page = "Pages:" + this.num_page + "\n";
        price = "Price:" + this.price + "\n";
```



```
    return name + author + num_pages + price;
  }
}

class bookdemo {
  public static void main (String args[]) {
    Scanner s = new Scanner (System.in)
    int n, price, num_pages;
    String name, author;
    System.out.println ("The no. of books:");
    n = s.nextInt();

    book b[] = new book(n);
    for (int i = 0; i < n; i++) {
      System.out.println ("The book details:");
      System.out.println ("The name:");
      name = s.next();
      System.out.println ("No. of pages: \n");
      num_pages = s.nextInt();
      System.out.println ("The price:");
      price = s.nextInt();
      b[i] = new book (name, author,
                       num_pages, price);
    }
  for (int i=0; i<n; i++) {
    System.out.println ("Book" + (i+1) + "details");
    System.out.println (b[i]);
  }
}
```

Date __/__/__
Page

Output:

The no. of books: 2
The book details:
Name: Jufix
Author: Me
No. of pages: 50
Price: 100
The book details:
Name: FGP
Author: Ali
No. of page: 317
price: 548

Book 1 details:
Book name: Jufix
Author: Me
No. of pages: 50
Price: 100

Book 2 details:
Book name: FGP
Author: Ali
No. of pages: 317
price: 548.

13/10/24

Record

Source Code:

```
import java.util.Scanner;

class Publication {
   private String title;
   private String writer;
   private float cost;
   private int pageCount;

   Publication(String title, String writer, float cost, int pageCount) {
      this.title = title;
      this.writer = writer;
      this.cost = cost;
      this.pageCount = pageCount;
   }

   String getTitle() {
      return this.title;
   }

   String getWriter() {
      return this.writer;
   }

   float getCost() {
      return this.cost;
   }

   int getPageCount() {
      return this.pageCount;
   }

   void setTitle(String title) {
      this.title = title;
   }

   void setWriter(String writer) {
      this.writer = writer;
   }

   void setCost(float cost) {
      this.cost = cost;
   }

   void setPageCount(int pageCount) {
```

```
        this.pageCount = pageCount;
    }

    @Override
    public String toString() {
        return "\nTitle: " + this.title +
                "\nAuthor: " + this.writer +
                "\nPrice: " + this.cost +
                "\nPages: " + this.pageCount;
    }
}

class lab3 {
    public static void main(String[] args) {
        System.out.println("Alok 1BM23CS024");

        Scanner input = new Scanner(System.in);

        System.out.print("How many books would you like to enter? ");
        int numberOfBooks = input.nextInt();
        input.nextLine();  // Consume the leftover newline character
        Publication[] catalog = new Publication[numberOfBooks];

        for (int i = 0; i < numberOfBooks; i++) {
            System.out.println("\nEntering details for Book " + (i + 1));

            System.out.print("Enter book title: ");
            String bookTitle = input.nextLine();  // Changed to nextLine()

            System.out.print("Enter author name: ");
            String bookAuthor = input.nextLine();  // Changed to nextLine()

            System.out.print("Enter book price: ");
            float bookPrice = input.nextFloat();

            System.out.print("Enter number of pages: ");
            int bookPages = input.nextInt();
            input.nextLine();  // Consume the leftover newline character after reading an int

            catalog[i] = new Publication(bookTitle, bookAuthor, bookPrice, bookPages);
        }

        System.out.println("\n--- Book Catalog ---");
        for (int i = 0; i < numberOfBooks; i++) {
            System.out.println(catalog[i].toString());
        }

        input.close();
```

```
    }
}
```

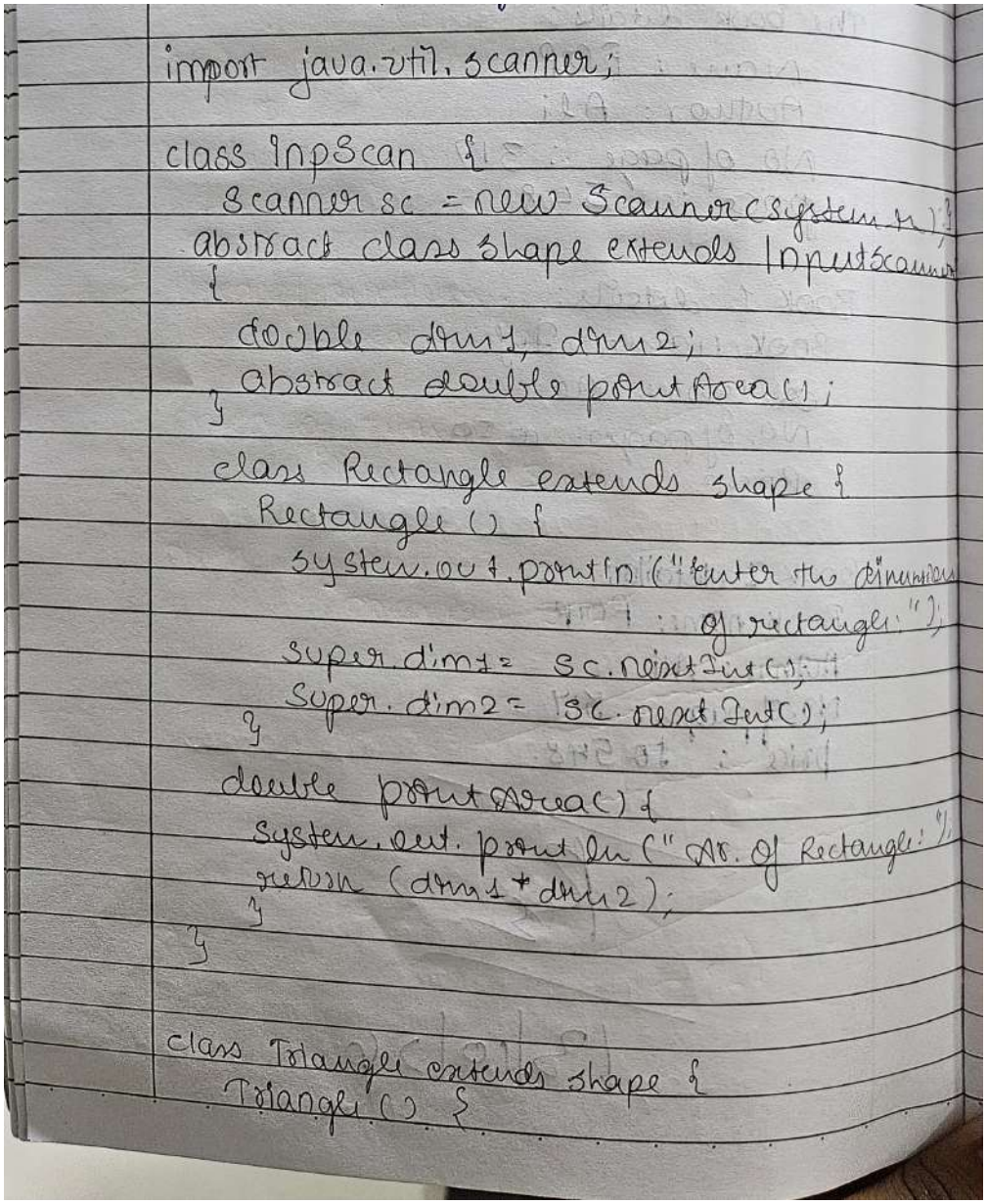
Output:


"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.4\lib\idea_rt.jar=50918:C:\Program Files\JetBrains\I
Alok 1BM23CS024
How many books would you like to enter? 2
Entering details for Book 1
Enter book title: Defender
Enter author name: Land Rover
Enter book price: 210
Enter number of pages: 260
Entering details for Book 2
Enter book title: Chimera
Enter author name: Koeniggseg
Enter book price: 510
Enter number of pages: 1050
--- Book Catalog ---
Title: Defender
Author: Land Rover
Price: 210.0
Pages: 260
Title: Chimera
Author: Koeniggseg
Price: 510.0
Pages: 1050
Process finished with exit code 0

Program 4

Develop a Java program to create an abstract class named Shape that contains 2 integers and an empty method named printArea(). Provide 3 classes names Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given shape.

Observation Book



```
import java.util.Scanner;

class InpScan {
  Scanner sc = new Scanner(System.in);
abstract class Shape extends InputScanner
{
  double dim1, dim2;
  abstract double printArea();
}

class Rectangle extends Shape {
  Rectangle() {
    System.out.println("Enter the dimensions
                        of rectangle:");
    super.dim1 = sc.nextInt();
    super.dim2 = sc.nextInt();
  }

double printArea() {
  System.out.println("Ar. of Rectangle:");
  return (dim1 * dim2);
  }
}


class Triangle extends Shape {
  Triangle() {
```

Date __/__/__
Page _____

```
    system.out.println("Dimensions of Triangle: \n");
    super.dim1 = sc.nextInt();
    super.dim2 = sc.nextInt();
  }
  double printAr() {
    system.out.println("Ar. of Δle:");
    return 0.5 * dim1 * dim2;
  }
}

class circle extends shape {
  circle() {
    system.out.println("Enter dimension(rad)");
    super.dim1 = sc.nextInt();
  }
  double printArea() {
    system.out.println("Ar. of circle:");
    return 3.14 * dim1 * dim1;
  }
}
}

class Abstract Demo {
  public static void main(String a[]) {
    Rectangle r = new Rectangle();
    Triangle t = new Triangle();
    Circle c = new Circle();
    shape figref;
    figref = t;
    system.out.println("Area is:" + figref.printArea());
    figref = r;
    system.out.println("Area is:" + figref.printArea() + "\n");
    figref = c;
```

Date / /
Page

```
        System.out.println("Area is: " + figref.printArea() +
                           "\n");
    }
}
```

Output:

Dimensions of Rectangle: 40 60

Dimensions of Triangle:
10 5

Dimensions of (rad):
1

Area of rectangle:
Area is: 2400

Area of Triangle:
Area is: 25

Area of circle:
Area is: 3.14

23/10/24

Record

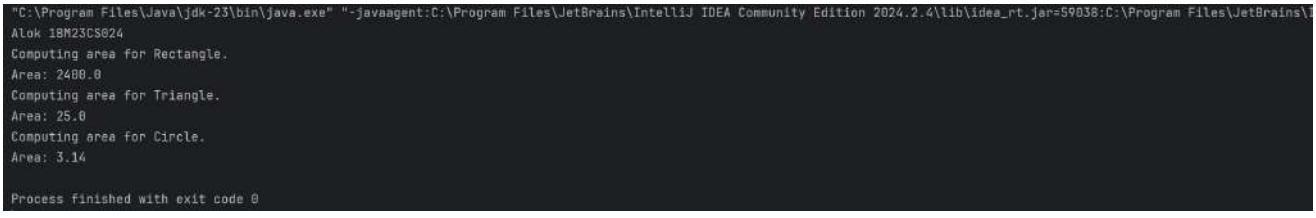
Source Code

```
abstract class Shape {
    double dimension1, dimension2;

    Shape(double x, double y) {
        dimension1 = x;
        dimension2 = y;
    }

    abstract double computeArea();
}

class RectangleShape extends Shape {
    RectangleShape(double x, double y) {
        super(x, y);
    }

    double computeArea() {
        System.out.println("Computing area for Rectangle.");
        return dimension1 * dimension2;
    }
}
```

```
class TriangleShape extends Shape {
    TriangleShape(double x, double y) {
        super(x, y);
    }

    double computeArea() {
        System.out.println("Computing area for Triangle.");
        return dimension1 * dimension2 / 2;
    }
}

class CircleShape extends Shape {
    CircleShape(double x, double y) {
        super(x, y);
    }

    double computeArea() {
        System.out.println("Computing area for Circle.");
        return 3.14 * dimension1 * dimension1;
    }
}

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

```
        System.out.println("Alok 1BM23CS024");


        RectangleShape rect = new RectangleShape(40, 60);
        TriangleShape tri = new TriangleShape(10, 5);
        CircleShape circ = new CircleShape(1, 1);


        Shape shapeRef;


        shapeRef = rect;
        System.out.println("Area: " + shapeRef.computeArea());


        shapeRef = tri;
        System.out.println("Area: " + shapeRef.computeArea());


        shapeRef = circ;
        System.out.println("Area: " + shapeRef.computeArea());
    }
}
```

Output

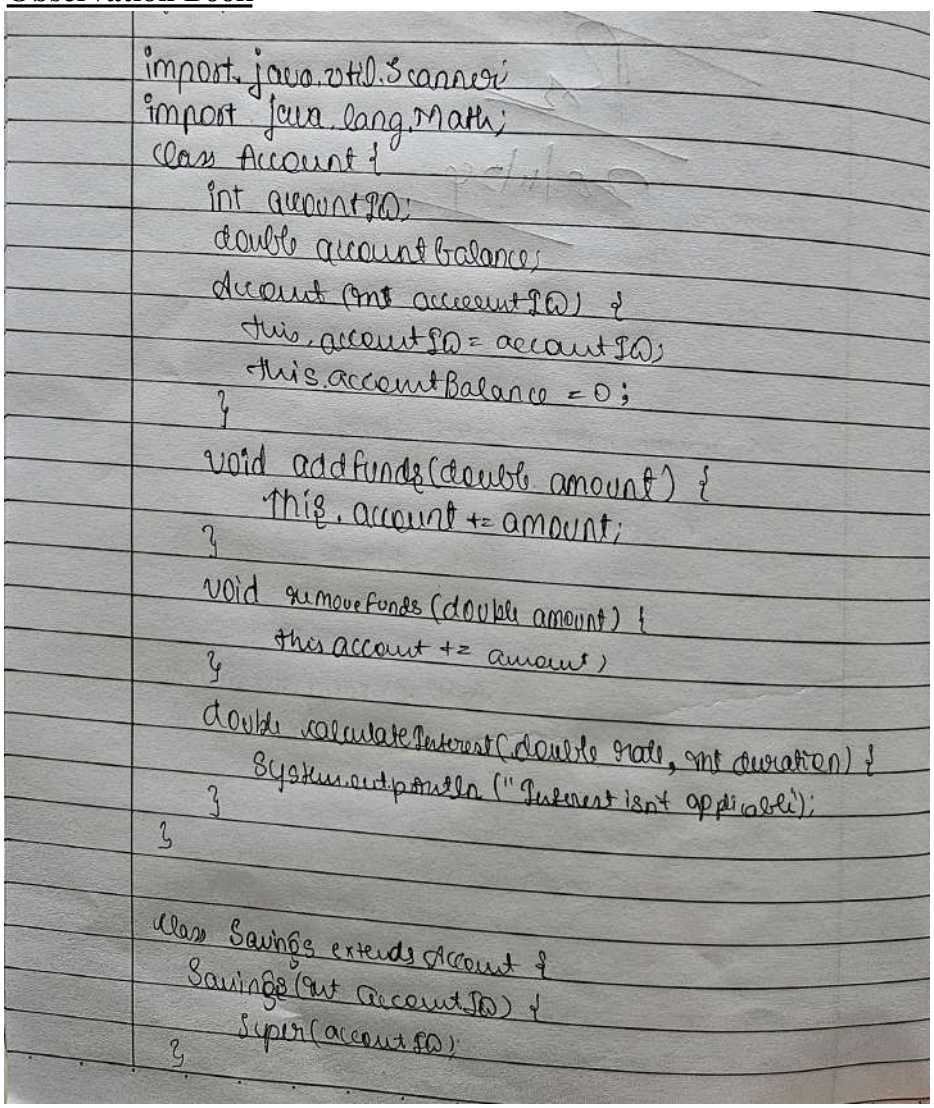


Program 5

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

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

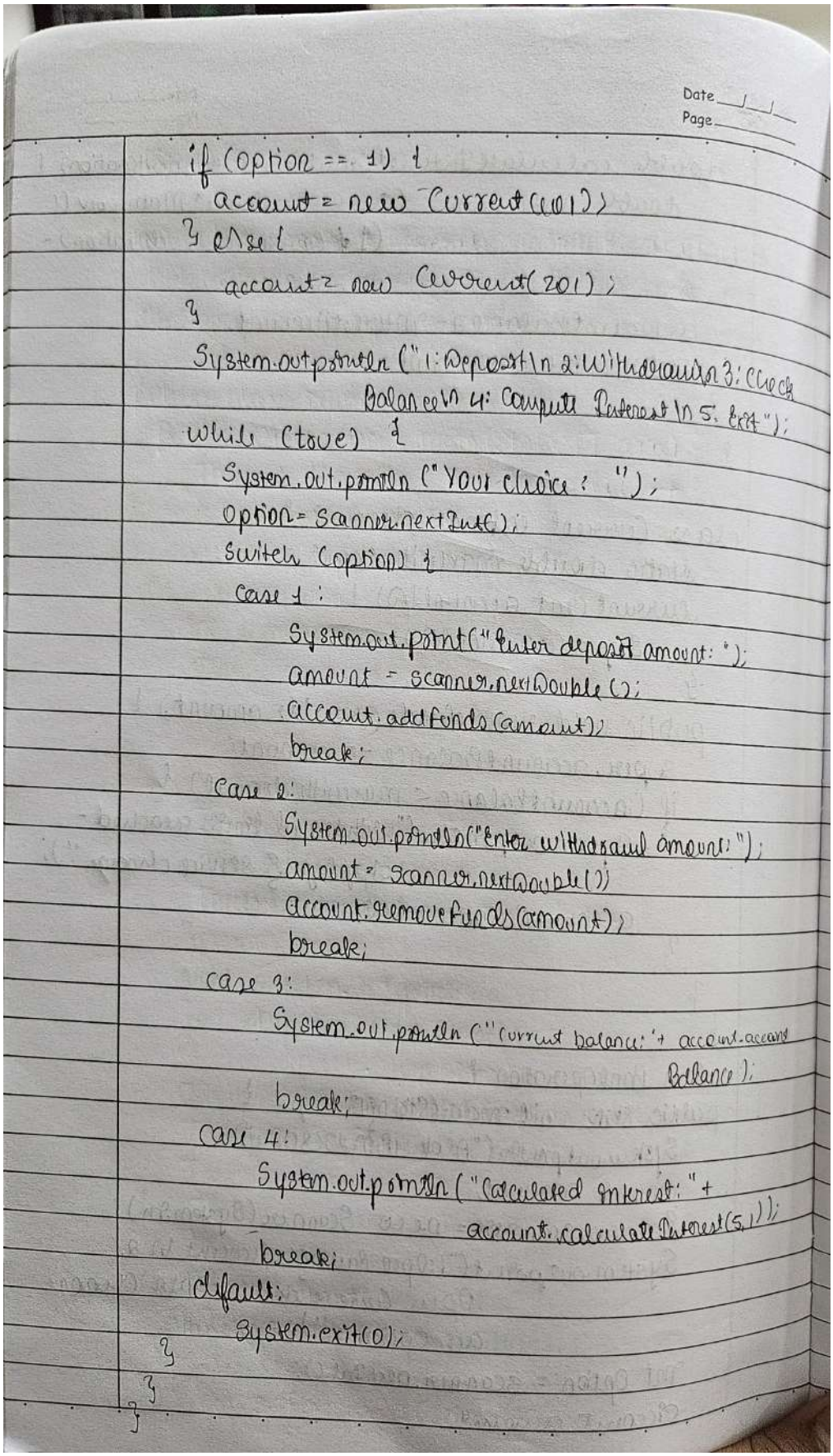
Observation Book



```
import java.util.Scanner;
import java.lang.Math;
class Account {
    int accountID;
    double accountBalance;
    Account(int accountID) {
        this.accountID = accountID;
        this.accountBalance = 0;
    }
    void addFunds(double amount) {
        this.account += amount;
    }
    void removeFunds(double amount) {
        this.account += amount;
    }
    double calculateInterest(double rate, int duration) {
        System.out.println("Interest isn't applicable");
    }
}

class Savings extends Account {
    Savings(int accountID) {
        super(accountID);
    }
```

Date __/__/__
Page _____

```
double calculateInterest(double rate, int duration) {
    double interestAmount = (accountBalance * Math.pow((
                            (1 + (rate/100)), duration) -
                            accountBalance;
    accountBalance += interestAmount;
    return interestAmount;
  }
}


class Current extends Account {
  static double maxWithdrawal = 1000;
  Current(int accountID) {
    super(accountID);
  }
  public void removeFunds(double amount) {
    super.accountBalance -= amount;
    if (accountBalance < maxWithdrawal) {
      System.out.println("Withdrawal limit reached -
                          Applying service charge.");
      accountBalance -= 100;
    }
  }
}
class BankOperation {
public static void main(String[] args) {
  System.out.println("ACOK IBM 23CS024");
  double amount;
  Scanner scanner = new Scanner(System.in);
  System.out.print("1: Open Savings Account \n 2.
                    Open Current Account \n\n Choose
                    an option: ");
  int option = scanner.nextInt();
  Account account;
```



```
if (option == 1) {
    account = new Current(101);
} else {
    account = new Current(201);
}
System.out.println("1: Deposit\n2: Withdrawal\n3: Check
                    Balance\n4: Compute Interest\n5: Exit");
while (true) {
    System.out.println("Your choice: ");
    option = scanner.nextInt();
    switch (option) {
    case 1:
        System.out.print("Enter deposit amount: ");
        amount = scanner.nextDouble();
        account.addFunds(amount);
        break;
    case 2:
        System.out.println("Enter withdrawal amount: ");
        amount = scanner.nextDouble();
        account.removeFunds(amount);
        break;
    case 3:
        System.out.println("Current balance: " + account.accountBalance);
        break;
    case 4:
        System.out.println("Calculated interest: " +
                           account.calculateInterest(5.1));
        break;
    default:
        System.exit(0);
    }
}
}
```

Date ___/___/___
Page _____

Output:

1: Open Savings Account
2: Open Current Account

Choose an option: 1
1. Deposit
2. Withdraw
3. Check Balance
4. Compute Interest
5. Exit
Enter your choice: 1
Enter deposit amount: 10000
Enter your choice: 3
Current balance: 10000
Enter your choice: 4
Calculated interest: 500
Enter your choice: 5

13/12/24

Record

```
import java.util.Scanner;
import java.lang.Math;
class Account {
   int accountID;
   double accountBalance;
   Account(int accountID) {
      this.accountID = accountID;
      this.accountBalance = 0;
   }
   void addFunds(double amount) {
      this.accountBalance += amount;
   }
   void removeFunds(double amount) {
      this.accountBalance -= amount;
   }
   double calculateInterest(double rate, int duration) {
      System.out.println("Interest is not applicable for this account type.");
      return 0.0;
   }
}
class Savings extends Account {
   Savings(int accountID) {
      super(accountID);
   }
   double calculateInterest(double rate, int duration) {
      double interestAmount = (accountBalance * Math.pow((1 + (rate / 100)), duration)) -
accountBalance;
      accountBalance += interestAmount;
      return interestAmount;
   }
}
class Current extends Account {
   static double maxWithdrawalLimit = 1000;
   Current(int accountID) {
      super(accountID);
   }
   public void removeFunds(double amount) {
      super.accountBalance -= amount;
      if (accountBalance < maxWithdrawalLimit) {
         System.out.println("Withdrawal limit reached - Applying service charge.");
         accountBalance -= 100;
      }
   }
}
class BankOperations {
   public static void main(String[] args) {
```

```
        System.out.println("Alok 1BM23CS024");
        double amount;
        Scanner scanner = new Scanner(System.in);
        System.out.print("1. Open Savings Account\n2. Open Current Account\n\nChoose an option: ");
        int option = scanner.nextInt();
        Account account;
        if (option == 1) {
            account = new Savings(101);
        } else {
            account = new Current(201);
        }
        System.out.println("1. Deposit\n2. Withdraw\n3. Check Balance\n4. Compute Interest\n5. Exit");
        while (true) {
            System.out.print("Enter your choice: ");
            option = scanner.nextInt();
            switch (option) {
                case 1:
                    System.out.print("Enter deposit amount: ");
                    amount = scanner.nextDouble();
                    account.addFunds(amount);
                    break;
                case 2:
                    System.out.print("Enter withdrawal amount: ");
                    amount = scanner.nextDouble();
                    account.removeFunds(amount);
                    break;
                case 3:
                    System.out.println("Current balance: " + account.accountBalance);
                    break;
                case 4:
                    System.out.println("Calculated interest: " + account.calculateInterest(5, 1));
                    break;
                default:
                    System.exit(0);
            }
        }
    }
}
```

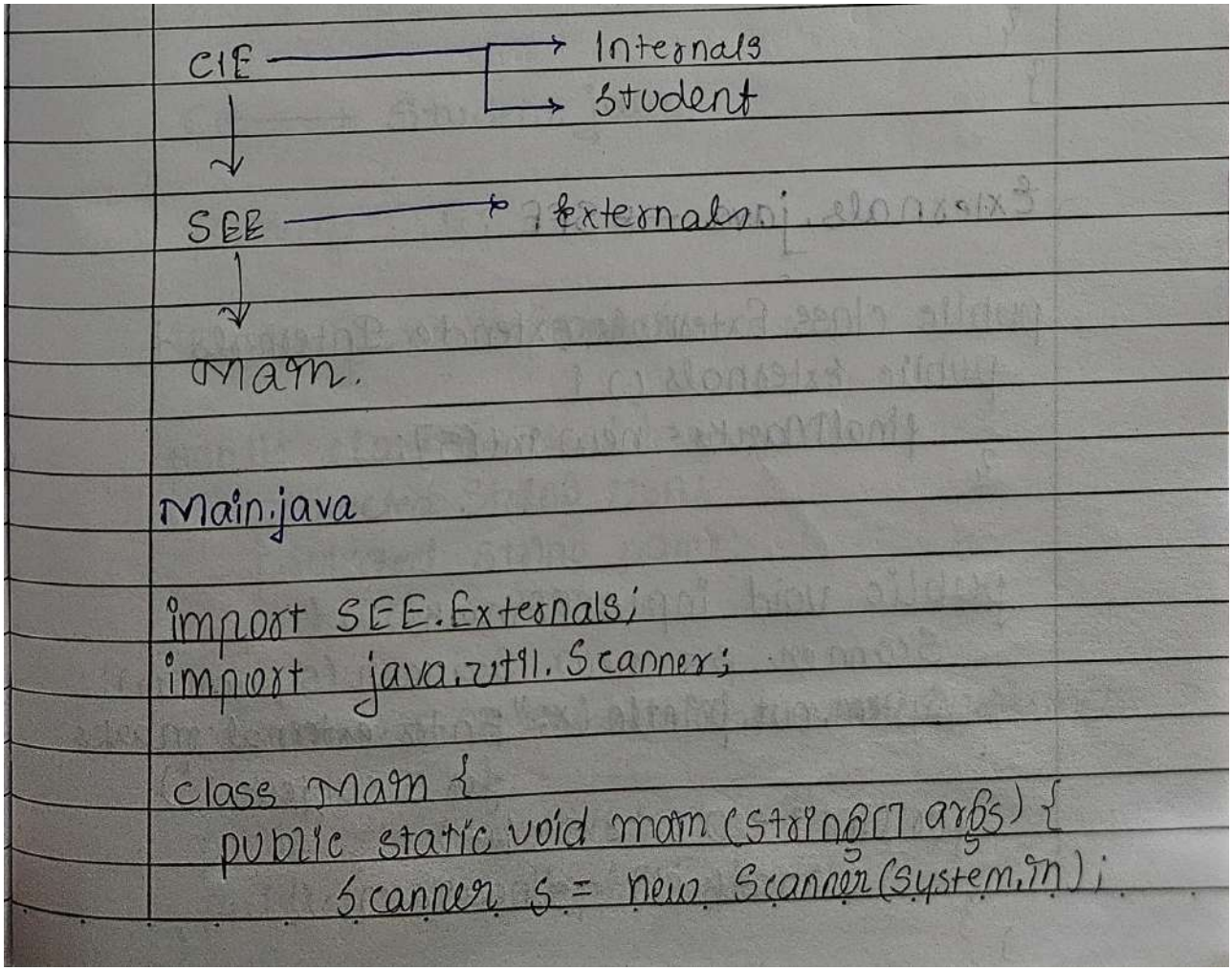
Output

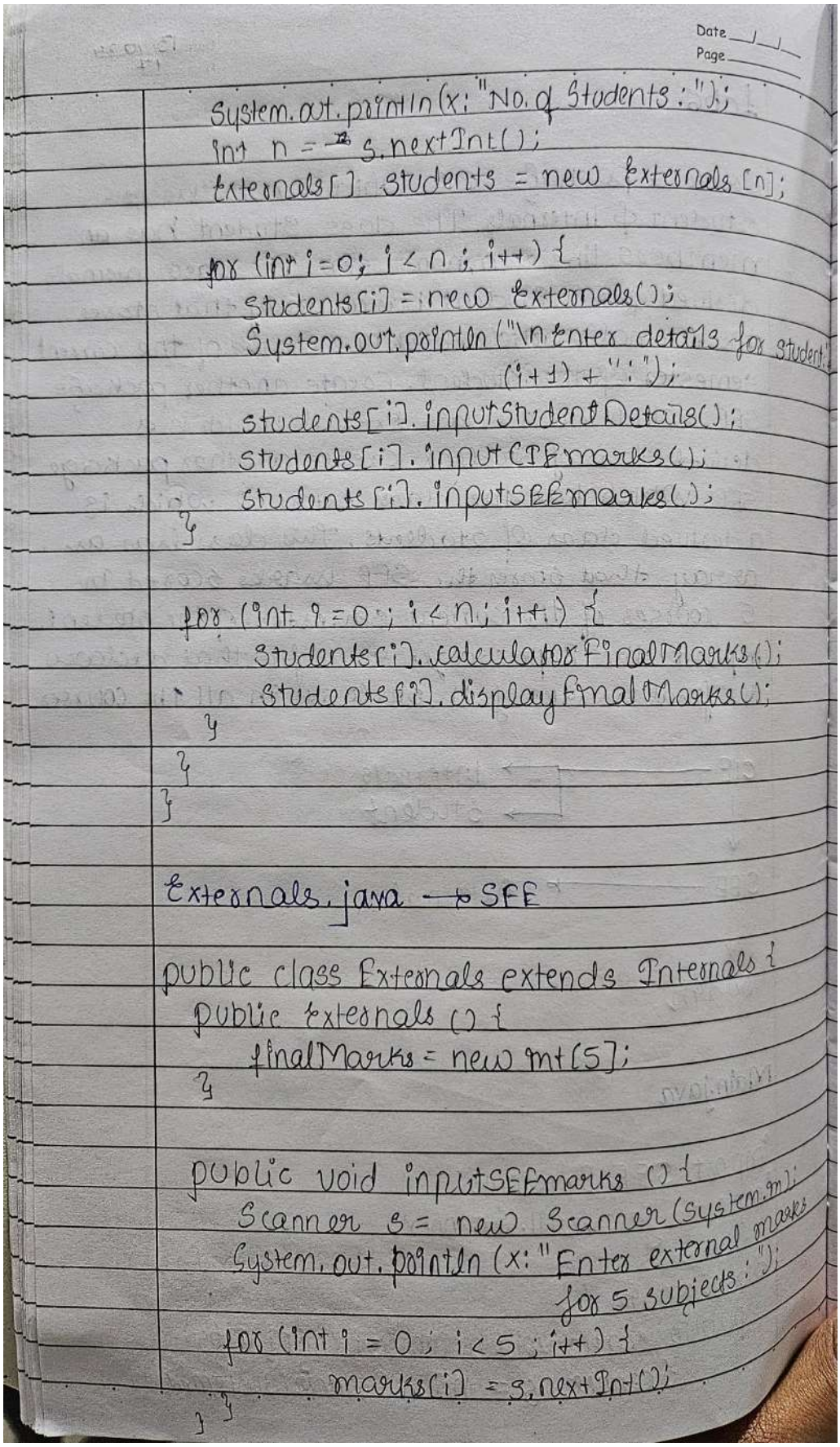

"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.4\lib\idea_rt.jar=59103:C:\Program Files\JetBrains\I
Alok 18M23CS024
1. Open Savings Account
2. Open Current Account
Choose an option: 1
1. Deposit
2. Withdraw
3. Check Balance
4. Compute Interest
5. Exit
Enter your choice: 1
Enter deposit amount: 10000
Enter your choice: 3
Current balance: 10000.0
Enter your choice: 4
Calculated interest: 500.0
Enter your choice: 5
Process finished with exit code 0

Program 6

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

Observation Book





```
System.out.println(x: "No. of Students: ");
int n = s.nextInt();
Externals[] students = new Externals[n];

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

for (int i=0; i<n; i++) {
    students[i].calculateFinalMarks();
    students[i].displayFinalMarks();
}
}
}
```

Externals.java → SEE

```
public class Externals extends Internals {
  public Externals() {
      finalMarks = new int[5];
  }

  public void inputSEEmarks() {
      Scanner s = new Scanner(System.in);
      System.out.println(x: "Enter external marks
                         for 5 subjects: ");
      for (int i = 0; i<5; i++) {
          marks[i] = s.nextInt();
  } }
```

```
public void calculateFinalMarks(){
    for (int i = 0; i < 5; i++) {
    finalMarks[i] = marks[i] + this.marks[i];
  }
}

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

CIE ⟶ Student.java

```
package CIE;

import java.util.Scanner;

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

    Scanner sc = new Scanner(System.in);

    public void inputDetails();
    public void displayDetails();
}
```

CIE → Internals.java

```
package CIE;

import java.util.Scanner;
public class Internals extends Student {
public void inputCIEmarks();
public void displayCIEmarks();
};
```

Output:

```
No. of Students: 1
USN: 1BM23CS024
Name: Alok
Semester: 2
Enter Internal Marks for 5 subjects:
49
46
46
44
47
Enter External Marks for 5 subjects:
49
50
46
39
34
Final Marks:
Subject 1: 98
Subject 2: 100
Subject 3: 92
```

Date ___/___/___
Page _____

Subject 4 : 78
Subject 5 : 68.

23/11/24

Record

Source Code

(i) CIE/Internals.java:

```
package CIE;

import java.util.Scanner;

public class Internals extends Student {
   protected int marks[] = new int[5];  // Array for internal marks


   public void inputCIEmarks() {
      Scanner s = new Scanner(System.in);
      System.out.println("Enter internal marks for 5 subjects:");
      for (int i = 0; i < 5; i++) {
         marks[i] = s.nextInt();
      }
   }


   public void displayCIEmarks() {
      System.out.println("Internal Marks:");
      for (int i = 0; i < 5; i++) {
         System.out.println("Subject " + (i + 1) + ": " + marks[i]);
      }
   }
}
```

(ii) CIE/Student.java:

```
package CIE;

import java.util.Scanner;

public class Student {

   protected String usn;
   protected String name;
   protected int sem;

   Scanner sc = new Scanner(System.in);

   public void inputStudentDetails() {
      System.out.println("Enter the USN:");
```

```
        usn = sc.next();
        System.out.println("Enter the name:");
        name = sc.next();
        System.out.println("Enter the semester:");
        sem = sc.nextInt();
    }

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

(iii) SEE/Externals.java:

```
package SEE;

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

public class Externals extends Internals {

    protected int marks[] = new int[5];
    protected int finalMarks[] = new int[5];

    // Constructor to initialize marks
    public Externals() {
        marks = new int[5];
        finalMarks = new int[5];
    }

    public void inputSEEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter external marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            marks[i] = s.nextInt();
        }
    }

    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = marks[i] + this.marks[i];
        }
    }

    public void displayFinalMarks() {
        displayStudentDetails();
```

```
        displayCIEmarks();
        System.out.println("Final Marks (Internal + External):");
        for (int i = 0; i < 5; i++) {
            System.out.println("Subject " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}
```

(iv) Main:

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

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

        System.out.println("Enter number of students:");
        int n = s.nextInt();
        Externals[] students = new Externals[n];

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

        for (int i = 0; i < n; i++) {
            students[i].calculateFinalMarks();
            students[i].displayFinalMarks();
        }
    }
}
```

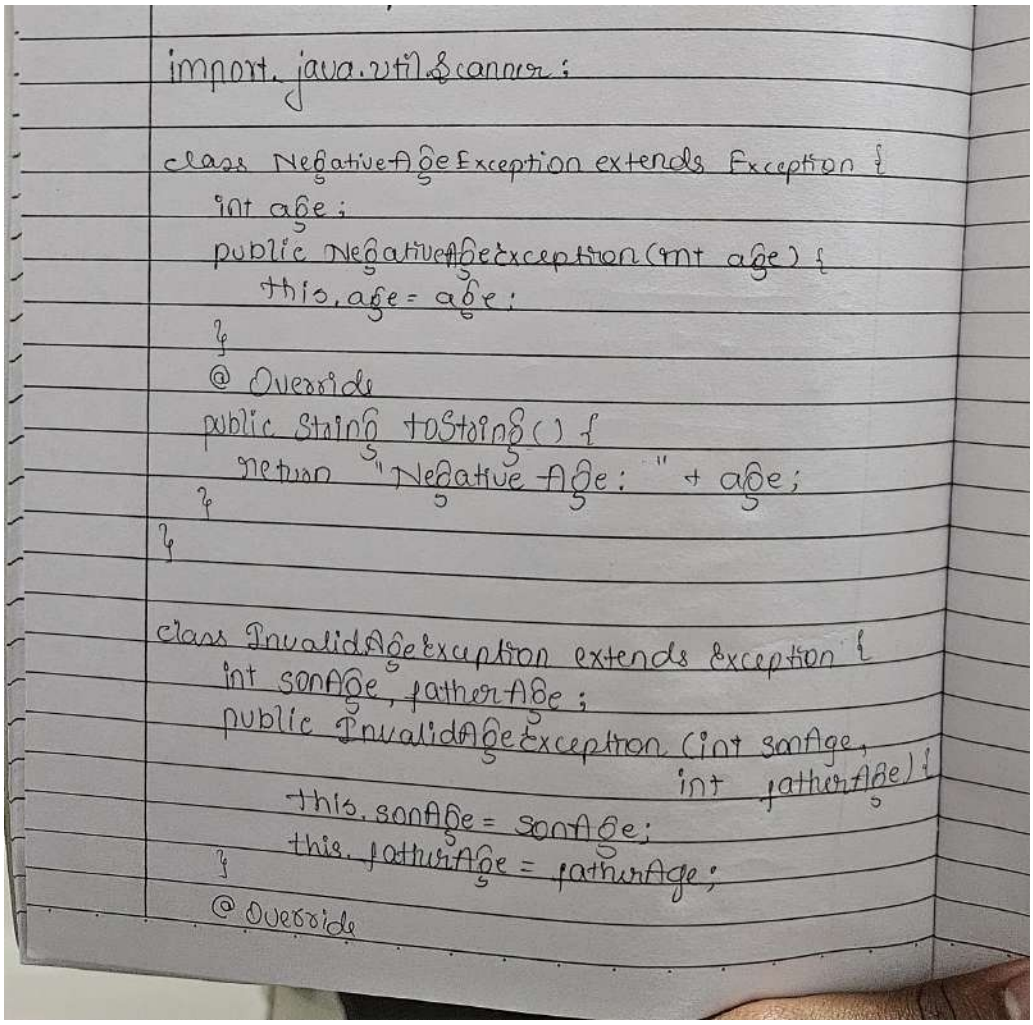
Output:

```
C:\Users\Admin\.vscode\CS024\Week6>java Main
Enter number of students:
1

Enter details for student 1:
Enter the USN:
1BM23CS024
Enter the name:
Alok
Enter the semester:
2
Enter internal marks for 5 subjects:
49
46
46
44
47
Enter external marks for 5 subjects:
49
50
46
39
34
USN: 1BM23CS024
Name: Alok
Semester: 2
Internal Marks:
Subject 1: 49
Subject 2: 46
Subject 3: 46
Subject 4: 44
Subject 5: 47
Final Marks (Internal + External):
Subject 1: 98
Subject 2: 100
Subject 3: 92
Subject 4: 78
Subject 5: 68
```

Program 7

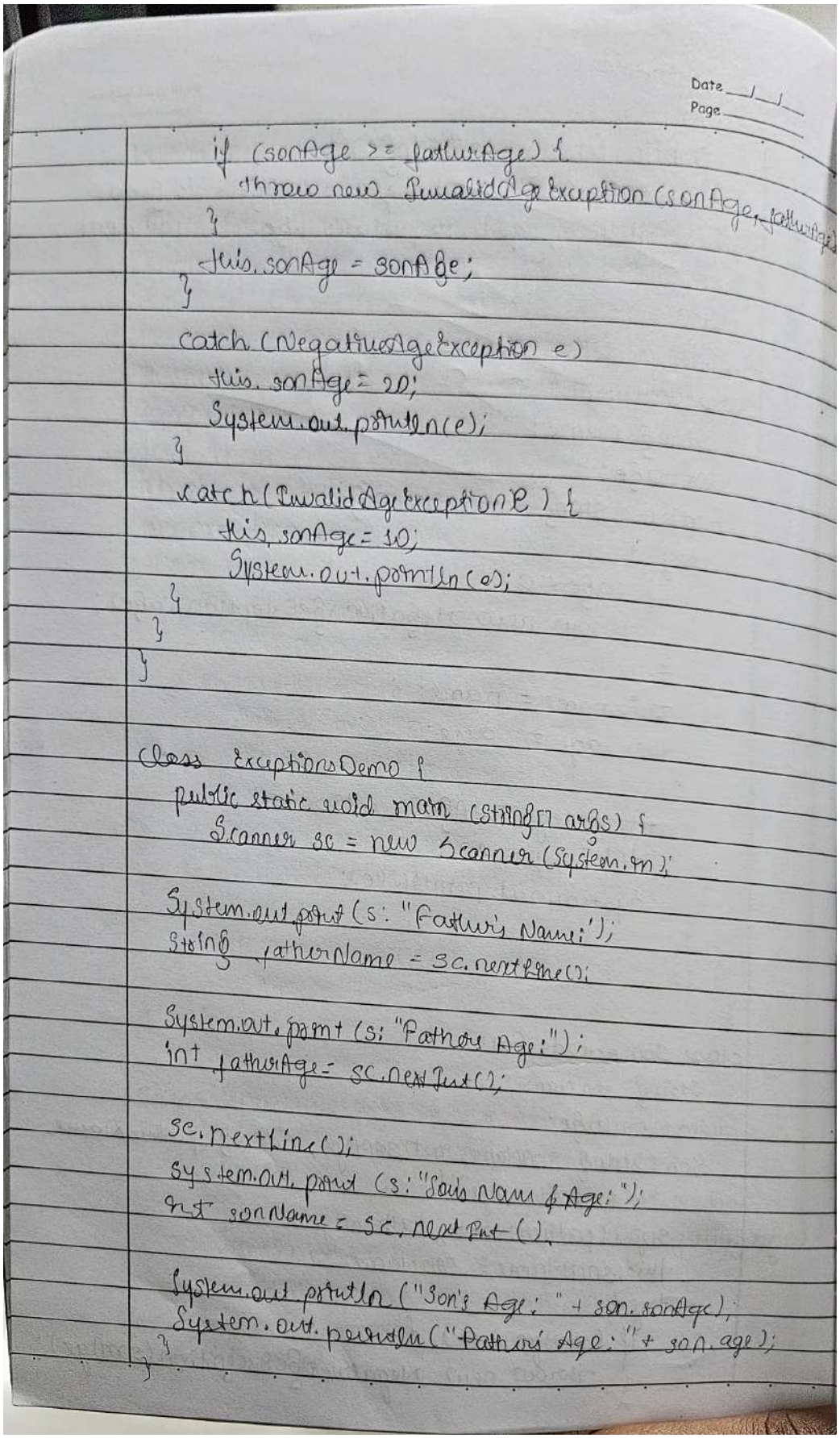
Write a program that demonstrates handling of exceptions in inheritance stream. Create a base class called “Father” and a derived class called “Son” which extends the base class. In class Father, implement a constructor which takes the age and throws an exception WrongAge() when the input age is less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

Observation Book

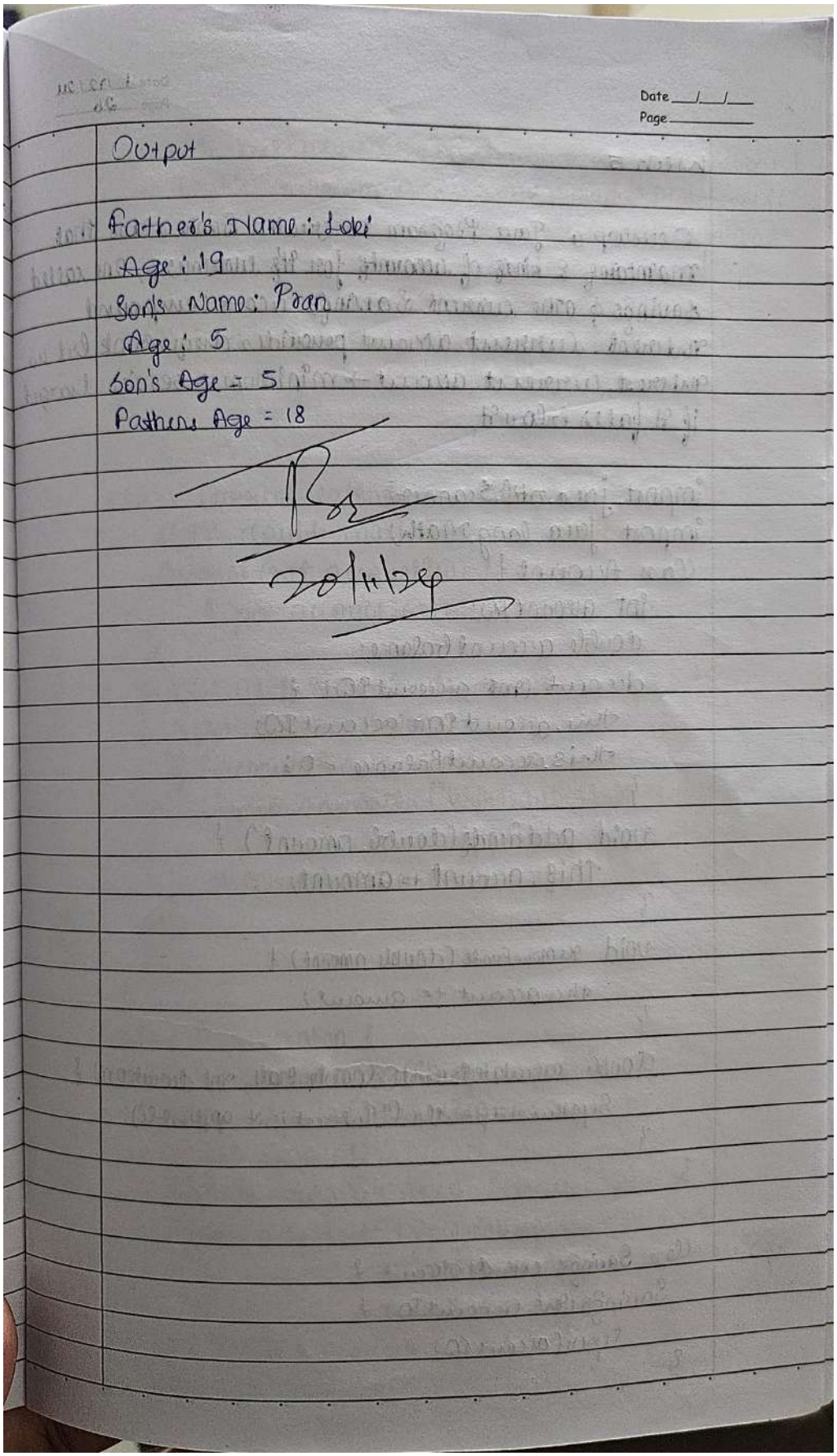


```
import java.util.Scanner;

class NegativeAgeException extends Exception {
    int age;
    public NegativeAgeException(int age) {
        this.age = age;
    }
    @Override
    public String toString() {
        return "Negative Age: " + age;
    }
}

class InvalidAgeException extends Exception {
    int sonAge, fatherAge;
    public InvalidAgeException(int sonAge,
                               int fatherAge) {
        this.sonAge = sonAge;
        this.fatherAge = fatherAge;
    }
    @Override
```

```
    public String toString() {
        return "Invalid Age: " + sonAge + " is greater
        than or equal to father's age: " + fatherAge;
    }
}

class Father {
    String name;
    int age;
    Father(String name, int age) {
        try {
            if (age < 0) {
                throw new NegativeAgeException(age);
            }
            this.name = name;
            this.age = age;
        }
        catch (NegativeAgeException e) {
            this.age = 20;
            System.out.println(e);
        }
    }
}

class Son extends Father {
    String sonName;
    int sonAge;
    Son(String sonName, int sonAge, String fatherName,
        int fatherAge) {
        super(fatherName, fatherAge);
        this.sonName = sonName;
        try {
            if (sonAge < 0) {
                throw new NegativeAgeException(sonAge);
            }
```



```
        if (sonAge >= fatherAge) {
            throw new InvalidAgeException(sonAge, fatherAge);
        }
        this.sonAge = sonAge;
    }

    catch (NegativeAgeException e)
        this.sonAge = 20;
        System.out.println(e);
    }
    catch (InvalidAgeException e) {
        this.sonAge = 10;
        System.out.println(e);
    }
}
}

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

    System.out.print (s: "Father's Name:");
    String fatherName = sc.nextLine();

    System.out.print (s: "Father's Age:");
    int fatherAge = sc.nextInt();

    sc.nextLine();
    System.out.print (s: "Son's Name & Age:");
    int sonName = sc.nextInt();

    System.out.println ("Son's Age: " + son.sonAge);
    System.out.println ("Father's Age: " + son.age);
    }
}
```



Date __/__/__
Page _____

Output

Father's Name : Loki
Age : 19
Son's Name : Pran
Age : 5
Son's Age = 5
Fathers Age = 18

20/11/24

Record

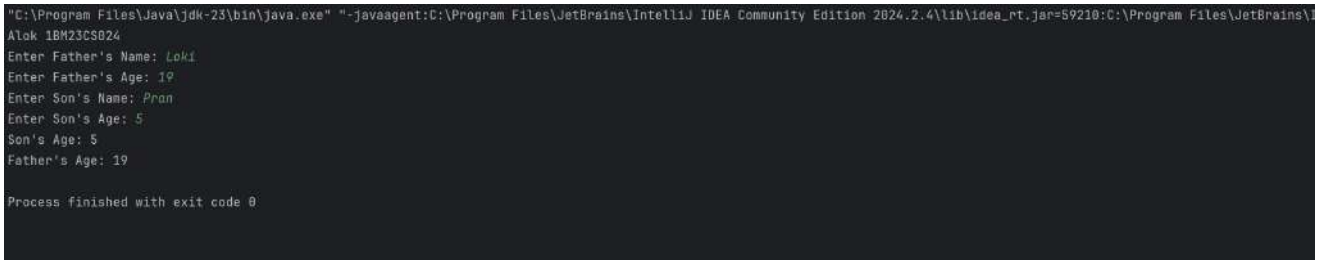
Source Code

```
import java.util.Scanner;

class NegativeAgeException extends Exception {
   int age;
   public NegativeAgeException(int age) {
      this.age = age;
   }
   @Override
   public String toString() {
      return "Negative Age: " + age;
   }
}

class InvalidAgeException extends Exception {
   int sonAge, fatherAge;
   public InvalidAgeException(int sonAge, int fatherAge) {
      this.sonAge = sonAge;
      this.fatherAge = fatherAge;
   }
   @Override
   public String toString() {
      return "Invalid Age: " + sonAge + " is greater than or equal to father's age: " + fatherAge;
   }
}

class Father {
   String name;
   int age;
   Father(String name, int age) {
      try {
         if(age < 0) {
            throw new NegativeAgeException(age);
         }
         this.name = name;
         this.age = age;
      }
      catch(NegativeAgeException e) {
         this.age = 20;
         System.out.println(e);
      }
   }
}
```

```
class Son extends Father {
    String sonName;
    int sonAge;
    Son(String sonName, int sonAge, String fatherName, int fatherAge) {
        super(fatherName, fatherAge);
        this.sonName = sonName;
        try {
            if(sonAge < 0) {
                throw new NegativeAgeException(sonAge);
            }
            if(sonAge >= fatherAge) {
                throw new InvalidAgeException(sonAge, fatherAge);
            }
            this.sonAge = sonAge;
        }
        catch(NegativeAgeException e) {
            this.sonAge = 20;
            System.out.println(e);
        }
        catch(InvalidAgeException e) {
            this.sonAge = 10;
            System.out.println(e);
        }
    }
}

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

        System.out.println("Alok 1BM23CS024");

        System.out.print("Enter Father's Name: ");
        String fatherName = sc.nextLine();

        System.out.print("Enter Father's Age: ");
        int fatherAge = sc.nextInt();

        sc.nextLine();
        System.out.print("Enter Son's Name: ");
        String sonName = sc.nextLine();

        System.out.print("Enter Son's Age: ");
        int sonAge = sc.nextInt();

        Son son = new Son(sonName, sonAge, fatherName, fatherAge);

        System.out.println("Son's Age: " + son.sonAge);
```

```
        System.out.println("Father's Age: " + son.age);
    }
}
```

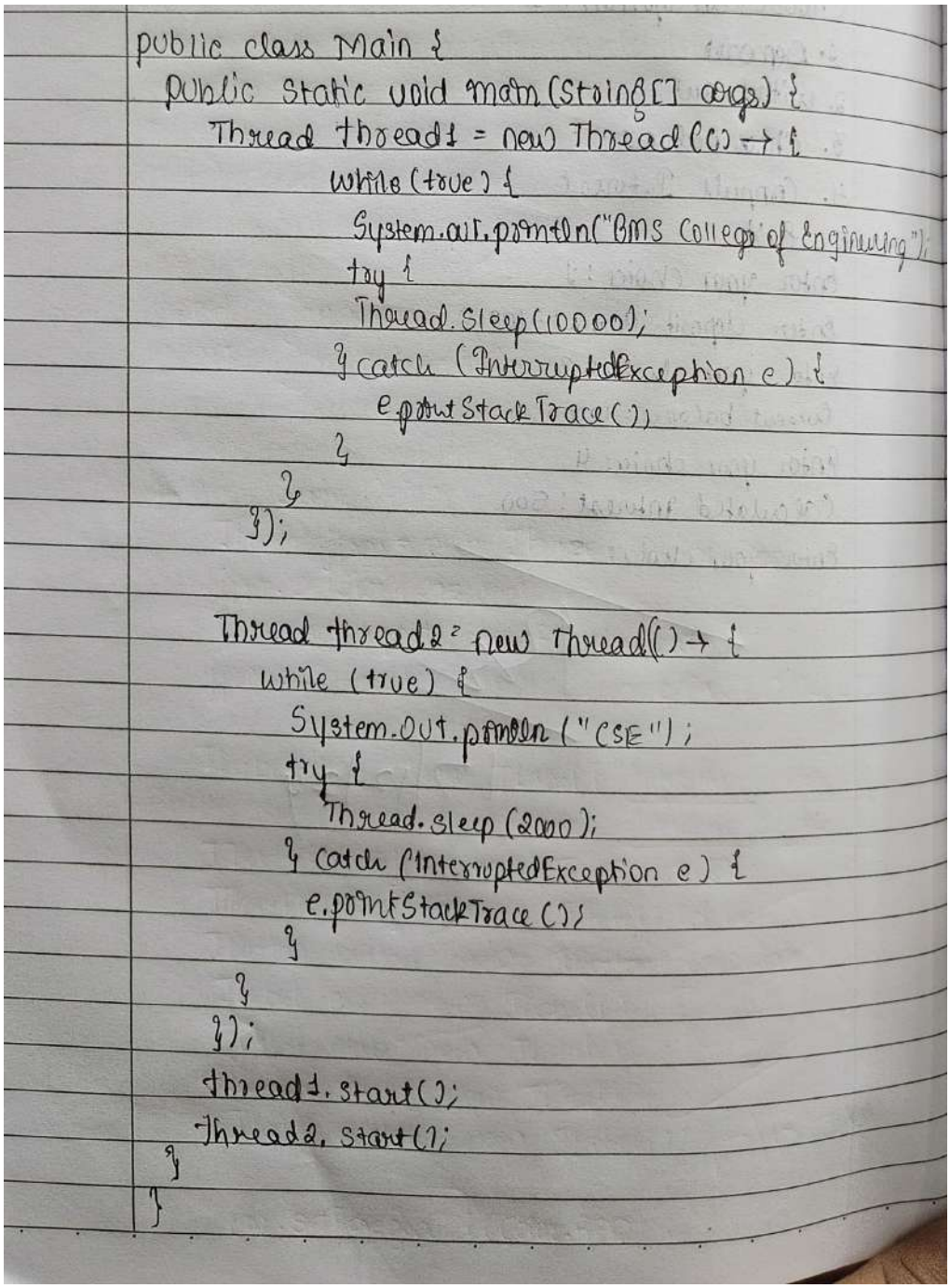
Output:



Program 8

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

Observation Book



```
public class Main {
  public static void main(String[] args) {
    Thread thread1 = new Thread(() -> {
        while (true) {
          System.out.println("BMS College of Engineering");
          try {
            Thread.sleep(10000);
          } catch (InterruptedException e) {
            e.printStackTrace();
          }
        }
    });

    Thread thread2 = new Thread(() -> {
      while (true) {
        System.out.println("CSE");
        try {
          Thread.sleep(2000);
        } catch (InterruptedException e) {
          e.printStackTrace();
        }
      }
    });
    thread1.start();
    thread2.start();
  }
}
```

Date __/__/__
Page _____

Output:

BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
:
:
:

17/12/24

Record

Source Code

```
public class Main {

   public static void main(String[] args) {
      Thread thread1 = new Thread(() -> {
         while (true) {
            System.out.println("BMS College of Engineering");
            try {
               Thread.sleep(10000);
            } catch (InterruptedException e) {
               e.printStackTrace();
            }
         }
      });

      Thread thread2 = new Thread(() -> {
         while (true) {
            System.out.println("CSE");
            try {
               Thread.sleep(2000);
            } catch (InterruptedException e) {
               e.printStackTrace();
            }
         }
      });

      thread1.start();
      thread2.start();
   }
}
```

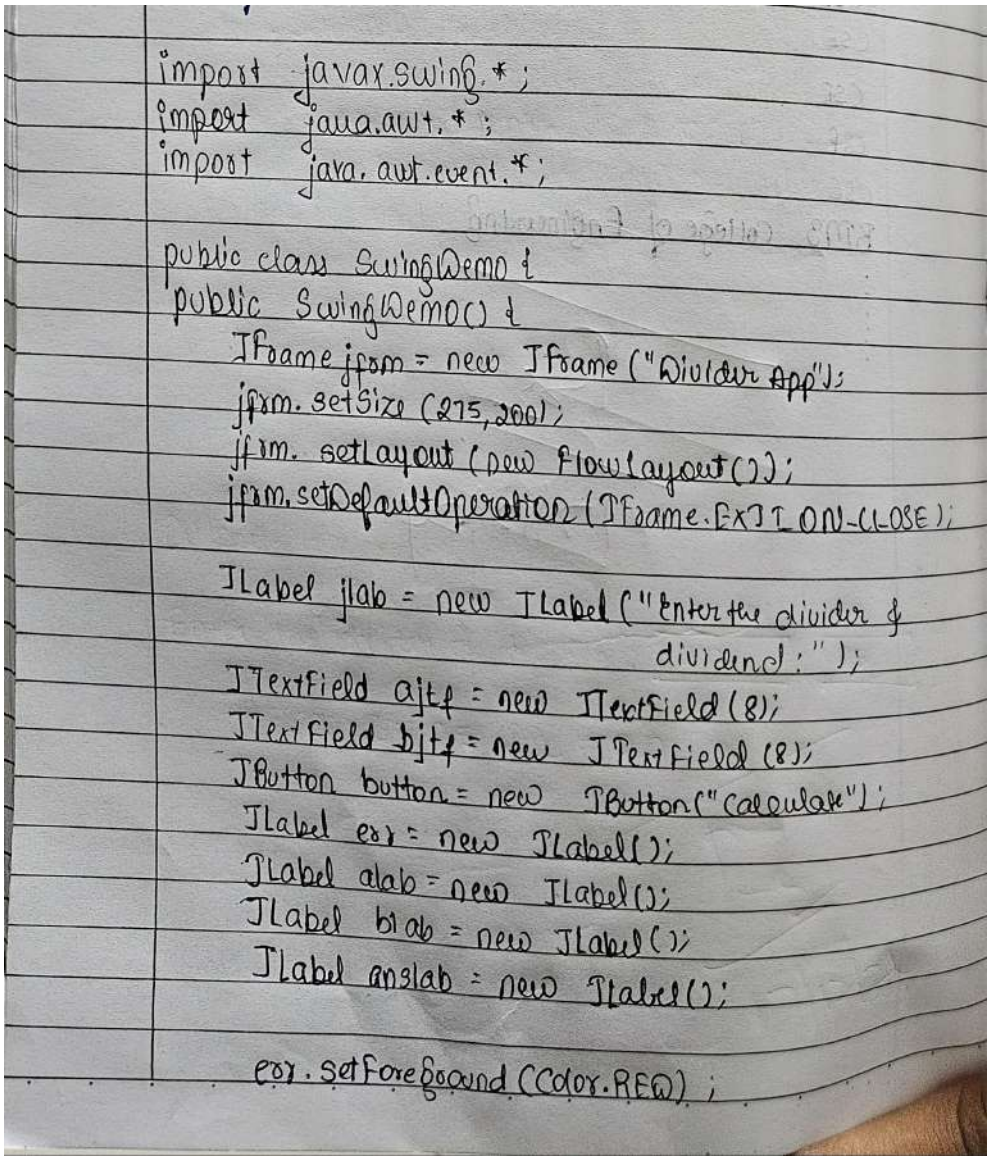
Output

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

Program 9

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

Observation Book



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

public class SwingDemo {
public SwingDemo() {
    JFrame jfrm = new JFrame("Divider App");
    jfrm.setSize(275,200);
    jfrm.setLayout(new FlowLayout());
    jfrm.setDefaultOperation(JFrame.EXIT_ON_CLOSE);

    JLabel jlab = new JLabel("Enter the divider &
                              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();

    err.setForeground(Color.RED);
```

```
err.setFont(new Font("Arial", Font.BOLD, 12));

jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);
jfrm.add(err);

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;

  dlab.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 Integral");
  JOptionPane.showMessageDialog(jfrm, "Invalid
  Input! Please enter integers only.", "Input Error",
  JOptionPane.ERROR_MESSAGE);
}
catch (ArithmeticException e) {
  alab.setText("");
```

```
        blab.setText("");
        anslab.setText("");
        err.setText("B should be NON zero!");
        JOptionPane.showMessageDialog(jfrm, "Cannot
        divide by zero!", "Math Error", JOptionPane.
        ERROR_MESSAGE);
      }
    }
  });
  System.out.println("Error, IBM23CS004");
  jfrm.setVisible(true);
}
public static void main(String args[]) {
  SwingUtilities.invokeLater(new Runnable() {
    public void run() {
      new SwingDemo();
    }
  });
}
}
```

Output.

Enter the divider & dividen : 500 25 Calculate

A=500 B=25 Ans = 20

17/12/24

Record

Source Code

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

public class SwingDemo {

   public SwingDemo() {
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 200);
      jfrm.setLayout(new FlowLayout());
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      JLabel jlab = new JLabel("Enter the divider 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();

      err.setForeground(Color.RED);
      err.setFont(new Font("Arial", Font.BOLD, 12));

      jfrm.add(jlab);
      jfrm.add(ajtf);
      jfrm.add(bjtf);
      jfrm.add(button);
      jfrm.add(alab);
      jfrm.add(blab);
      jfrm.add(anslab);
      jfrm.add(err);

      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);
```

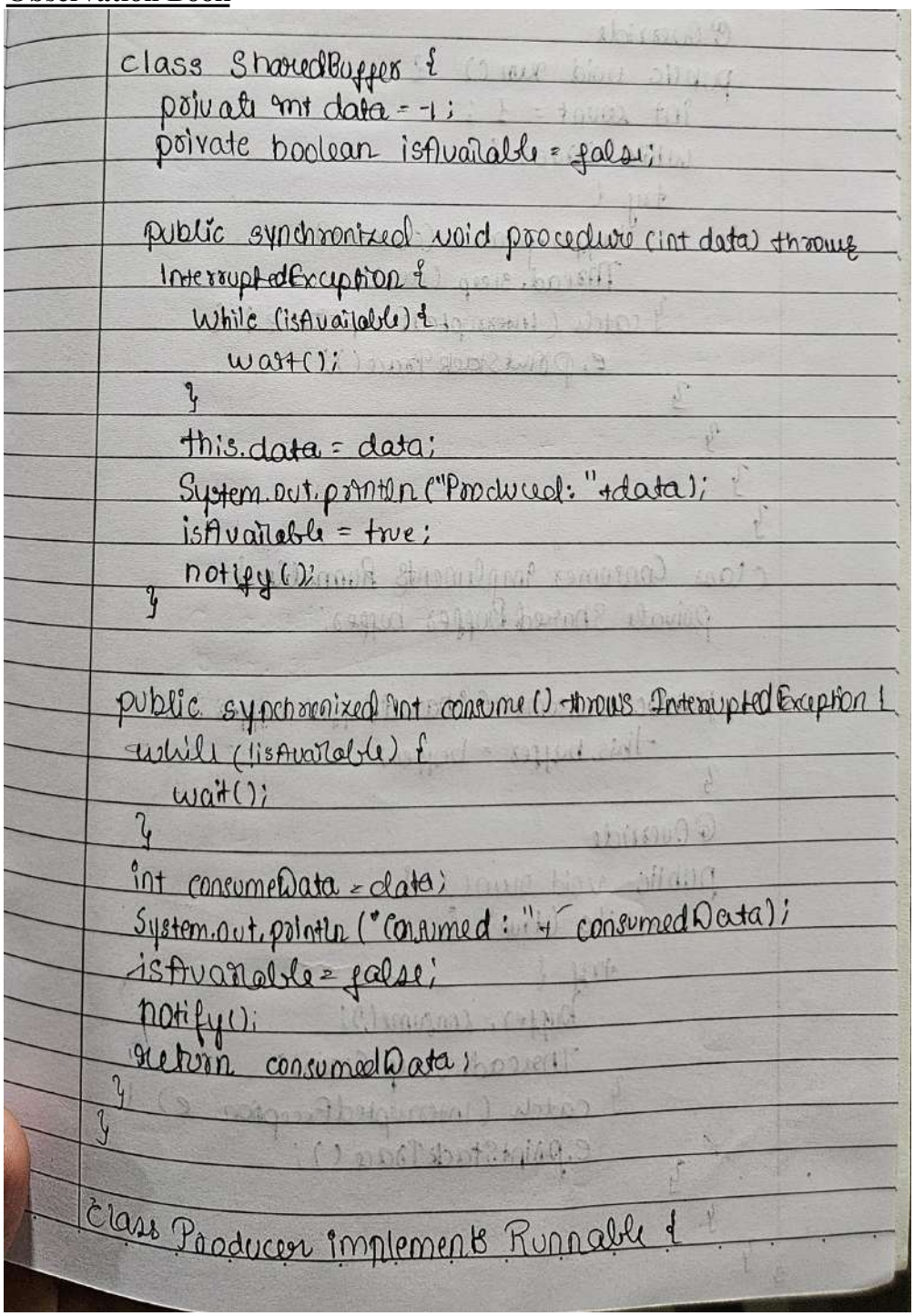
```
                err.setText("");

            } catch (NumberFormatException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("Enter Only Integers!");
                JOptionPane.showMessageDialog(jfrm, "Invalid input! Please enter integers only.",
"Input Error", JOptionPane.ERROR_MESSAGE);

            } catch (ArithmeticException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("B should be NON zero!");
                JOptionPane.showMessageDialog(jfrm, "Cannot divide by zero!", "Math Error",
JOptionPane.ERROR_MESSAGE);
            }
        }
    });

    System.out.println("Alok, 1BM23CS024");
    jfrm.setVisible(true);
  }

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

Output

Program 10

Demonstrate Inter Process Communication (IPC) and Deadlock.

Observation Book



```
class SharedBuffer {
  private int data = -1;
  private boolean isAvailable = false;

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

public synchronized int consume() throws InterruptedException {
  while (!isAvailable) {
    wait();
  }
  int consumedData = data;
  System.out.println("Consumed: "+ consumedData);
  isAvailable = false;
  notify();
  return consumedData;
  }
}

class Producer implements Runnable {
```

```
private SharedBuffer buffer;
public Producer(SharedBuffer buffer) {
    this.buffer = buffer;
}
@Override
public void run() {
    int count = 1;
    while (true) {
        try {
            buffer.produce(count++);
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
class Consumer implements Runnable {
    private SharedBuffer buffer;

public Consumer(SharedBuffer buffer) {
    this.buffer = buffer;
}
@Override
public void run() {
    while (true) {
        try {
            buffer.consume();
            Thread.sleep(1500);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

Date __/__/__
Page _____

```
public class IPCExample {
  public static void main (String[] args) {
    SharedBuffer buffer = new SharedBuffer();

    Thread producerThread = new Thread(new
                                Producer(buffer));
    Thread consumerThread = new Thread (new
                                Consumer(buffer));

    producerThread.start();
    consumerThread.start();
  }
}
```

Output

```
Produced : 1
Consumed : 1
Produced : 2
Consumed : 2
Produced : 3
Consumed : 3
Produced : 4
Consumed : 4
Produced : 5
Consumed : 5
```

17/12/24

Record

Source Code

```
class SharedBuffer {
   private int data = -1;
   private boolean isAvailable = false;

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

   public synchronized int consume() throws InterruptedException {
      while (!isAvailable) {
         wait();
      }
      int consumedData = data;
      System.out.println("Consumed: " + consumedData);
      isAvailable = false;
      notify();
      return consumedData;
   }
}

class Producer implements Runnable {
   private SharedBuffer buffer;

   public Producer(SharedBuffer buffer) {
      this.buffer = buffer;
   }

   @Override
   public void run() {
      int count = 1;
      while (true) {
         try {
            buffer.produce(count++);
            Thread.sleep(1000);
         } catch (InterruptedException e) {
            e.printStackTrace();
         }
```

```
        }
    }
}

class Consumer implements Runnable {
    private SharedBuffer buffer;

    public Consumer(SharedBuffer buffer) {
        this.buffer = buffer;
    }

    @Override
    public void run() {
        while (true) {
            try {
                buffer.consume();
                Thread.sleep(1500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

public class IPCExample {
    public static void main(String[] args) {
        System.out.println("Alok, 1BM23CS024");

        SharedBuffer buffer = new SharedBuffer();

        Thread producerThread = new Thread(new Producer(buffer));
        Thread consumerThread = new Thread(new Consumer(buffer));

        producerThread.start();
        consumerThread.start();
    }
}
```

Output

```
Alok, 1BM23CS024
Produced: 1
Consumed: 1
Produced: 2
Consumed: 2
Produced: 3
Consumed: 3
Produced: 4
Consumed: 4
Produced: 5
Consumed: 5
Produced: 6
Consumed: 6
Produced: 7
Consumed: 7
Produced: 8
Consumed: 8
Produced: 9
Consumed: 9
Produced: 10
Consumed: 10
Produced: 11
Consumed: 11
Produced: 12
Consumed: 12
Produced: 13
Consumed: 13
Produced: 14
Consumed: 14
Produced: 15
```