VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Amal Roy (1BM23CS025)

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 **Amal Roy (1BM23CS025),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

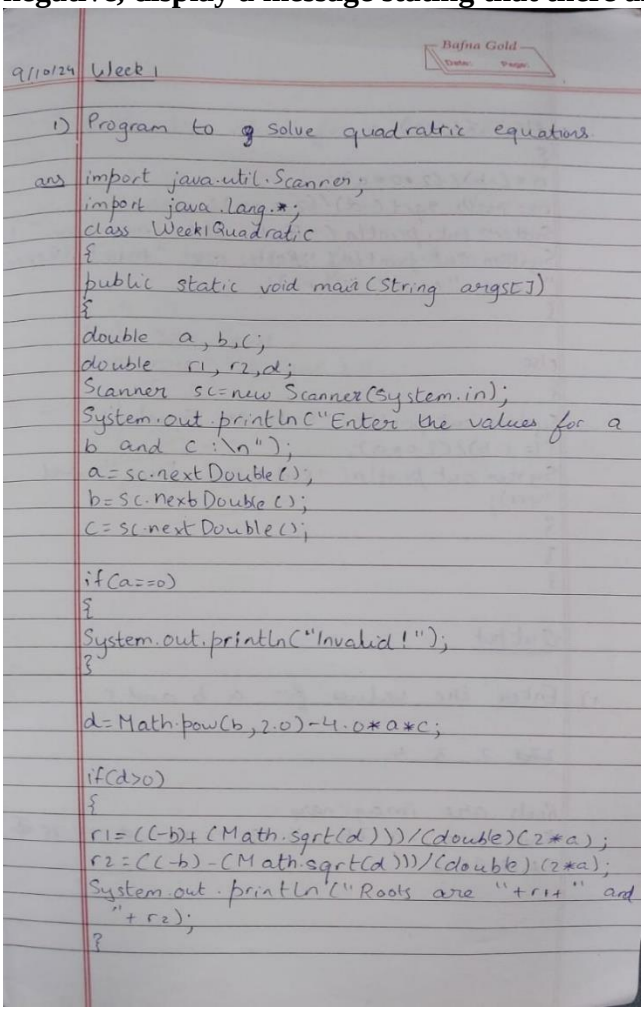
Lab faculty Incharge Mr. Basavaraj Jakalli Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Github Link:
https://github.com/amalrtms/OOJLABBMS

Program 1

Develop a Java program that prints all real solutions to the quadratic equation ax2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions.



9/10/24 Week 1

1) Program to solve quadratic equations

ans
```
import java.util.Scanner;
import java.lang.*;
class Week1Quadratic
{
public static void main(String args[])
{
double a,b,c;
double r1,r2,d;
Scanner sc=new Scanner(System.in);
System.out.println("Enter the values for a
b and c :\n");
a=sc.nextDouble();
b=sc.nextDouble();
c=sc.nextDouble();

if(a==0)
{
System.out.println("Invalid!");
}

d=Math.pow(b,2.0)-4.0*a*c;

if(d>0)
{
r1=((-b)+(Math.sqrt(d)))/(double)(2*a);
r2=((-b)-(Math.sqrt(d)))/(double)(2*a);
System.out.println("Roots are "+r1+" and
"+r2);
}
```

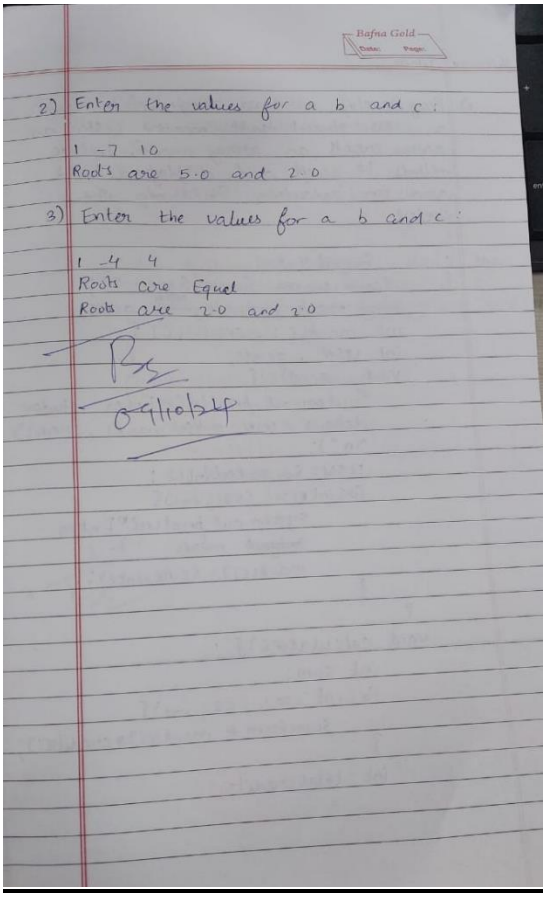
```
else if(d<0)
{
r1 = (-b)/(2.0*a);
r2 = Math.sqrt(-d)/(2.0*a);
System.out.println("Roots are imaginary.");
System.out.println("Roots are "+r1+"+i"+r2+
" and "+r1+"-i"+r2);
}

else
{
System.out.println("Roots are Equal.");
r1 = (-b)/(2.0*a);
System.out.println("Roots are "+r1+" and
"+r1);
}
}
}
```

Output

1) Enter the values for a b and c:

2 3 4

Roots are imaginary.
Roots are -0.75 + i1.1989578 and -0.75 - i1.1989578



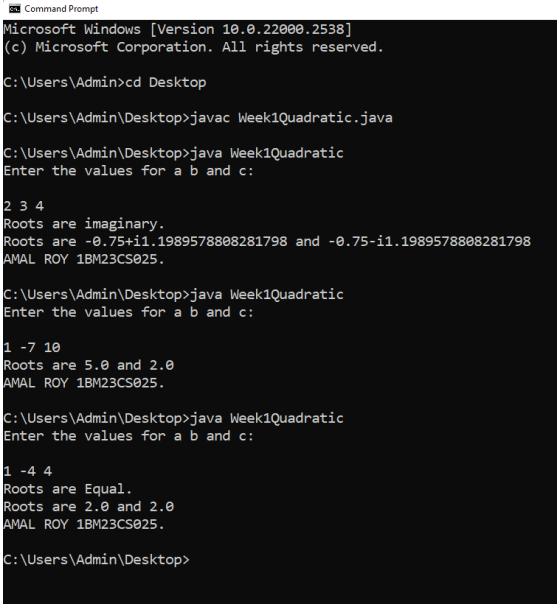
Observation Book:

Code:

```
import java.util.Scanner;
import java.lang.*;
class Week1Quadratic
{
public static void main(String args[])
{
double a,b,c;
double r1,r2,d;
Scanner sc=new Scanner(System.in);
System.out.println("Enter the values for a b and c:\n");
a=sc.nextDouble();
b=sc.nextDouble();
c=sc.nextDouble();

if(a==0)
{
System.out.println("Invalid!");
}
```

```
d=Math.pow(b,2.0)-4.0*a*c;

if(d>0)
{
r1=((-b)+(Math.sqrt(d)))/(double)(2*a);
r2=((-b)-(Math.sqrt(d)))/(double)(2*a);
System.out.println("Roots are "+r1+" and "+r2);
}

else if(d<0)
{
r1=(-b)/(2.0*a);
r2=Math.sqrt(-d)/(2.0*a);
System.out.println("Roots are imaginary.");
System.out.println("Roots are "+r1+"+i"+r2+" and "+r1+"-i"+r2);
}

else
{
System.out.println("Roots are Equal.");
r1=(-b)/(2.0*a);
System.out.println("Roots are "+r1+" and "+r1);
}
System.out.println("AMAL ROY 1BM23CS025.");
}
}
```

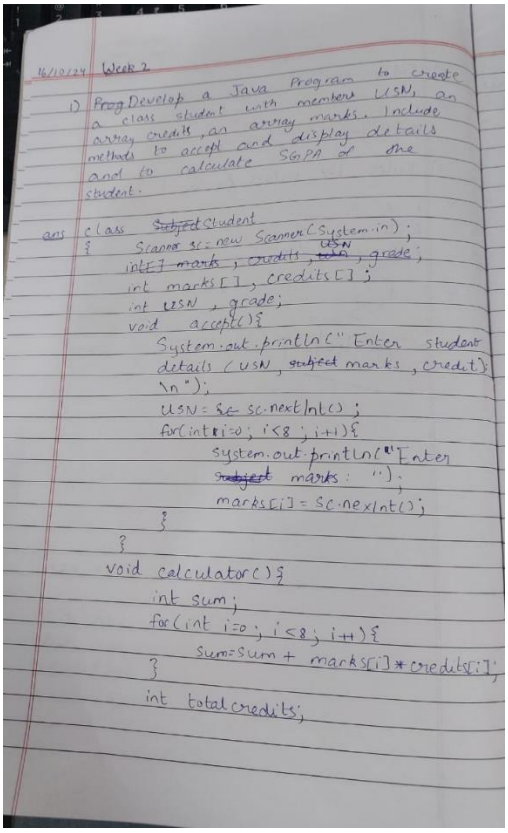
OUTPUT

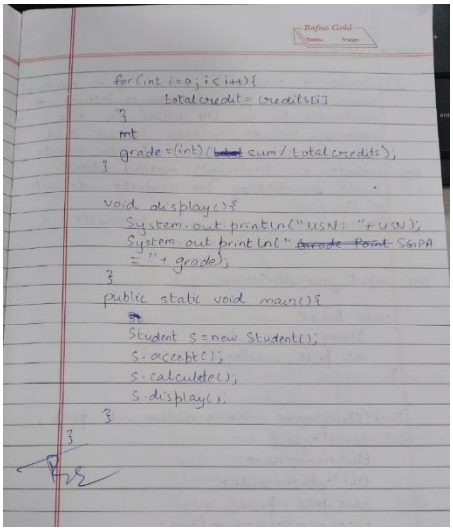


Program 2

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.

Observation Book





Code:

```
import java.util.Scanner;
class SGPA{
Scanner sc=new Scanner(System.in);
int marks[]=new int[8];
int credits[]=new int[8];
int USN,sum,totalcredits;
float grade;

void accept()
{
System.out.println("\nEnter Student USN: ");
USN=sc.nextInt();
for(int i=0;i<8;i++)
{
System.out.println("Enter the subject marks: ");
marks[i]=sc.nextInt();
System.out.println("Enter the respective credits: ");
credits[i]=sc.nextInt();
}
}

void calculate()
{
for(int i=0;i<8;i++)
{
sum+=marks[i]*credits[i];
}
for(int i=0;i<8;i++)
{
totalcredits+=credits[i];
}
grade=(float)(sum/(totalcredits*10));
}

void display()
{
System.out.println("USN: "+USN);
System.out.println("SGPA="+grade);
}

public static void main(String args[])
{
SGPA s[]=new SGPA[3];
for(int i=0;i<3;i++)
{
s[i]=new SGPA();
```

```
}
for(int i=0;i<3;i++)
{
s[i].accept();
s[i].calculate();
s[i].display();
System.out.println("AMAL ROY 1BM23CS025");
}
}
}
```

OUTPUT

```
Enter Student USN:
1
Enter the subject marks:
90
Enter the respective credits:
2
Enter the subject marks:
99
Enter the respective credits:
4
Enter the subject marks:
100
Enter the respective credits:
4
Enter the subject marks:
92
Enter the respective credits:
1
Enter the subject marks:
98
Enter the respective credits:
4
Enter the subject marks:
89
Enter the respective credits:
2
Enter the subject marks:
97
Enter the respective credits:
4
Enter the subject marks:
99
Enter the respective credits:
3
USN: 1
SGPA=9.0
```

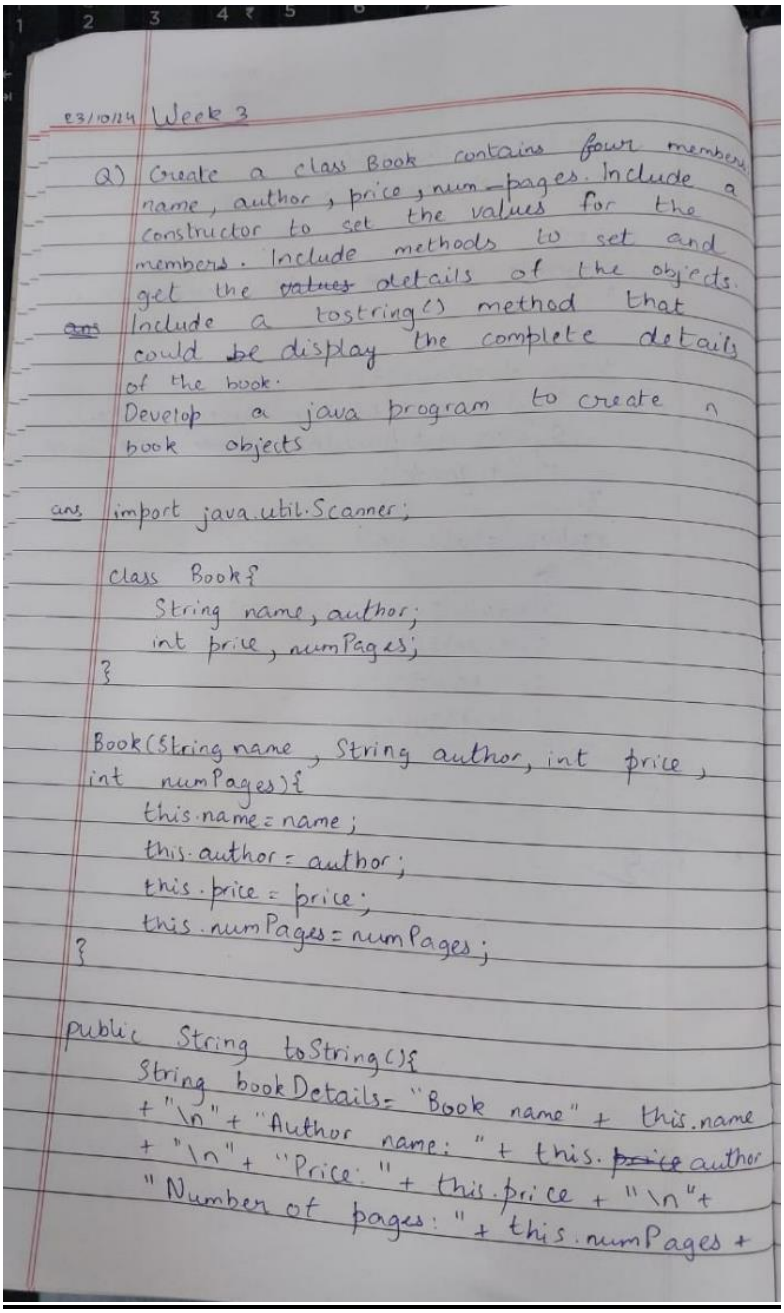
```
Enter Student USN:
45
Enter the subject marks:
42
Enter the respective credits:
1
Enter the subject marks:
24
Enter the respective credits:
4
Enter the subject marks:
7
Enter the respective credits:
5
Enter the subject marks:
7
Enter the respective credits:
67
Enter the subject marks:
8
Enter the respective credits:
1
Enter the subject marks:
99
Enter the respective credits:
4
Enter the subject marks:
99
Enter the respective credits:
4
Enter the subject marks:
99
Enter the respective credits:
4
USN: 45
SGPA=2.0
```

```
Enter Student USN:
2
Enter the subject marks:
34
Enter the respective credits:
2
Enter the subject marks:
67
Enter the respective credits:
2
Enter the subject marks:
46
Enter the respective credits:
4
Enter the subject marks:
22
Enter the respective credits:
9
Enter the subject marks:
34
Enter the respective credits:
7
Enter the subject marks:
34
Enter the respective credits:
6
Enter the subject marks:
45
Enter the respective credits:
1
Enter the subject marks:
34
Enter the respective credits:
1
USN: 2
SGPA=3.0
```

Program 3

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

Observation Book



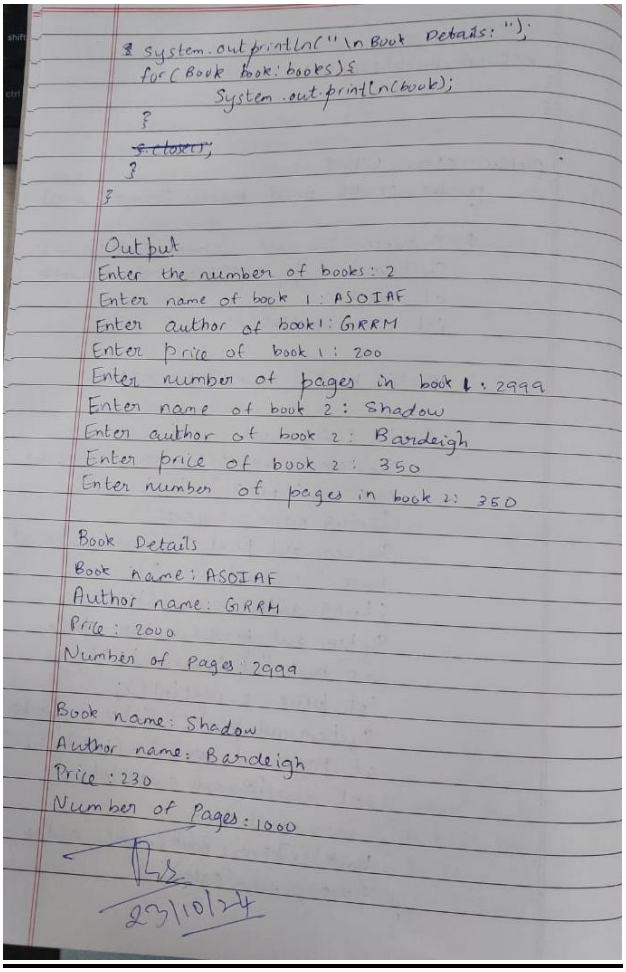
23/10/24 Week 3

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

ans

```
import java.util.Scanner;

class Book{
    String name, author;
    int price, numPages;
}

Book(String name, String author, int price,
int numPages){
    this.name = name;
    this.author = author;
    this.price = price;
    this.numPages = numPages;
}

public String toString(){
    String bookDetails= "Book name" + this.name
    + "\n" + "Author name: " + this.author
    + "\n" + "Price: " + this.price + "\n"+
    "Number of pages: " + this.numPages +
```

```
        "\n";
    return bookDetails;
}
}

public class Libr{
    public static void main(String[] args)
    {
    Scanner s = new Scanner(System.in);
    System.out.print("Enter the number
    of books: ");
    int n = s.nextInt();

    Book[] books = new Book[n];

    for(int i=0; i<n; i++){
        System.out.print("Enter name of
        book " + (i+1) + ": ");
        String name = s.next();
        System.out.print("Enter author of
        book " + (i+1) + ": ");
        String author=s.next();
        System.out.print("Enter price
        of book " + (i+1)+ ": ");
        int price= s.nextInt();
        System.out.print("Enter number
        of pages in book " + (i+1)+ ": ");
        int numPages = s.nextInt();

        books[i] = new Book(name, author,
        price, numPages);
    }
```



Code:

```
import java.util.Scanner;


class Book {

    String name;
    String author;
    int price;
    int numPages;


    Book(String name, String author, int price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
```

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


public class BookProgram{
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
            System.out.print("AMAL ROY 1BM23CS025\n");

        System.out.print("Enter the number of books: ");
        int n = s.nextInt();


        Book[] books = new Book[n];


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

            System.out.print("Enter name of book " + (i + 1) + ": ");
            String name = s.next();
                    s.nextLine();
            System.out.print("Enter author of book " + (i + 1) + ": ");
            String author = s.next();
                    s.nextLine();
            System.out.print("Enter price of book " + (i + 1) + ": ");
            int price = s.nextInt();
            System.out.print("Enter number of pages in book " + (i + 1) + ": ");
            int numPages = s.nextInt();


            books[i] = new Book(name, author, price, numPages);
        }


        System.out.println("\nBook Details:");
        for (Book book : books) {
            System.out.println(book);
        }
```

```
    }
}
```

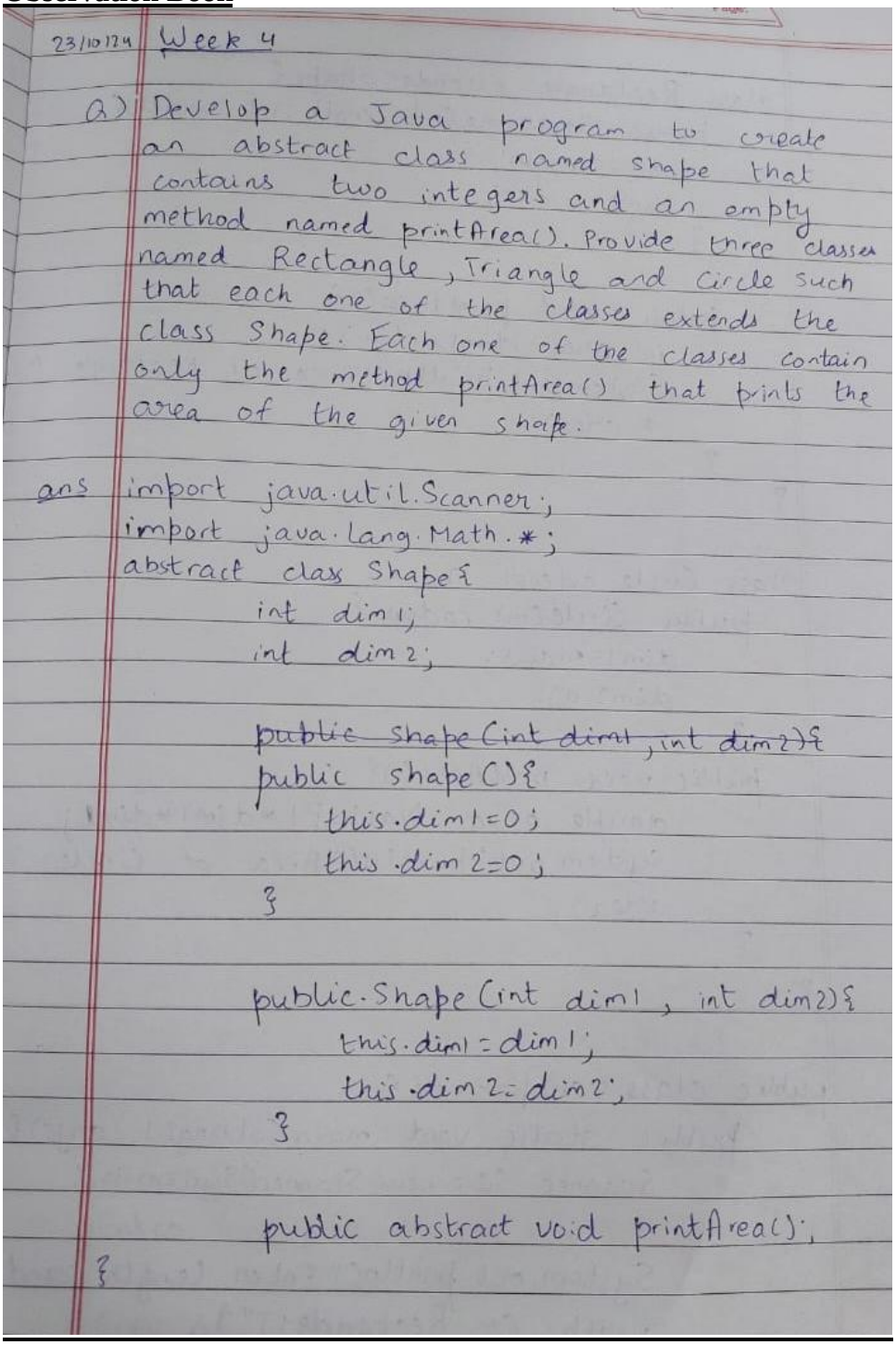
OUTPUT

```
AMAL ROY 1BM23CS025
Enter the number of books: 3
Enter name of book 1: ASOIAF
Enter author of book 1: GRRM
Enter price of book 1: 200
Enter number of pages in book 1: 2999
Enter name of book 2: Shadow and Bone
Enter author of book 2: Bardeigh
Enter price of book 2: 230
Enter number of pages in book 2: 1000
Enter name of book 3: The Alchemist
Enter author of book 3: Paulo Coehlo
Enter price of book 3: 260
Enter number of pages in book 3: 340

Book Details:
Book name: ASOIAF
Author name: GRRM
Price: 200
Number of pages: 2999

Book name: Shadow
Author name: Bardeigh
Price: 230
Number of pages: 1000

Book name: The
Author name: Paulo
Price: 260
Number of pages: 340
```

Program 4

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

Observation Book

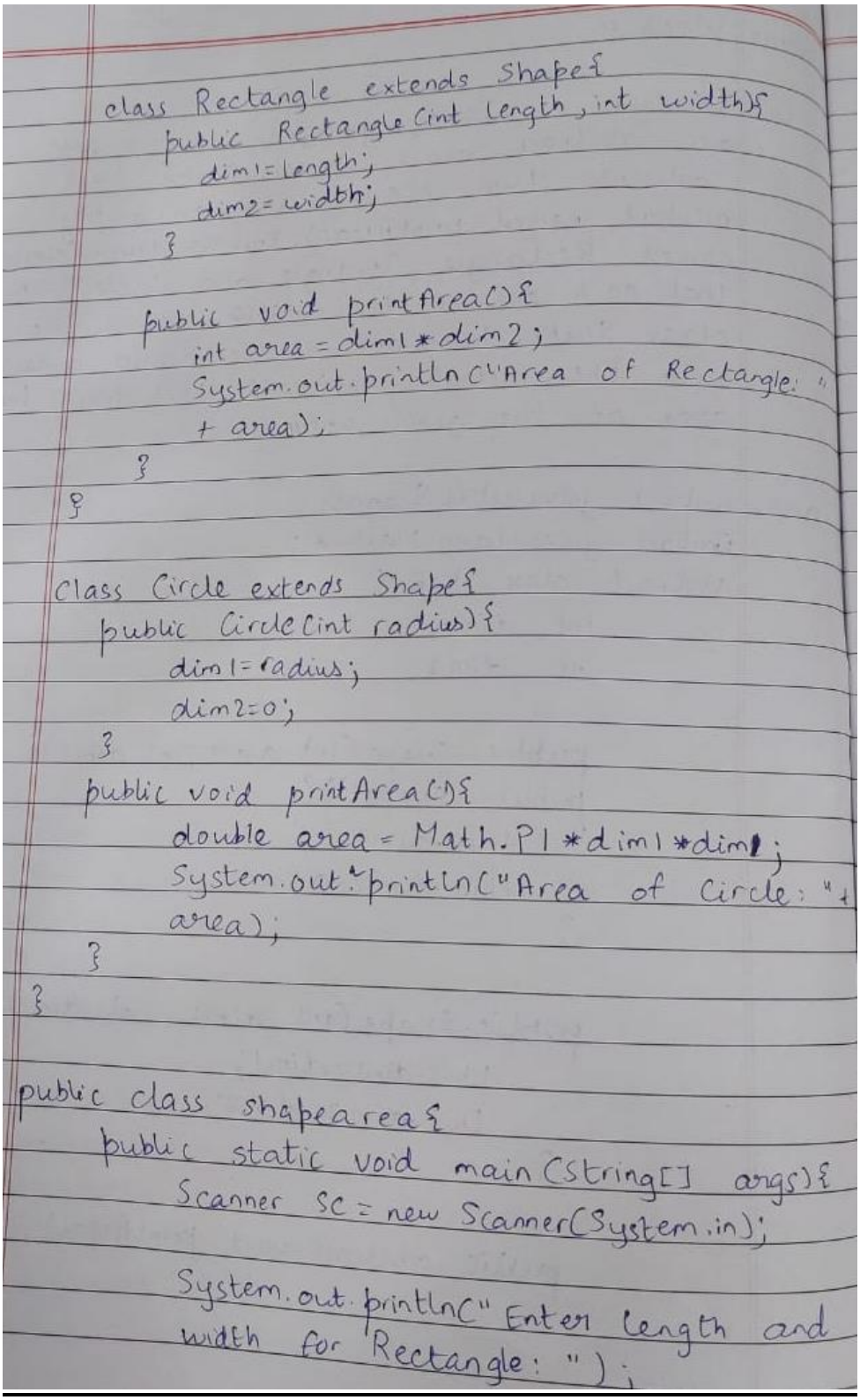


23/10/24 Week 4

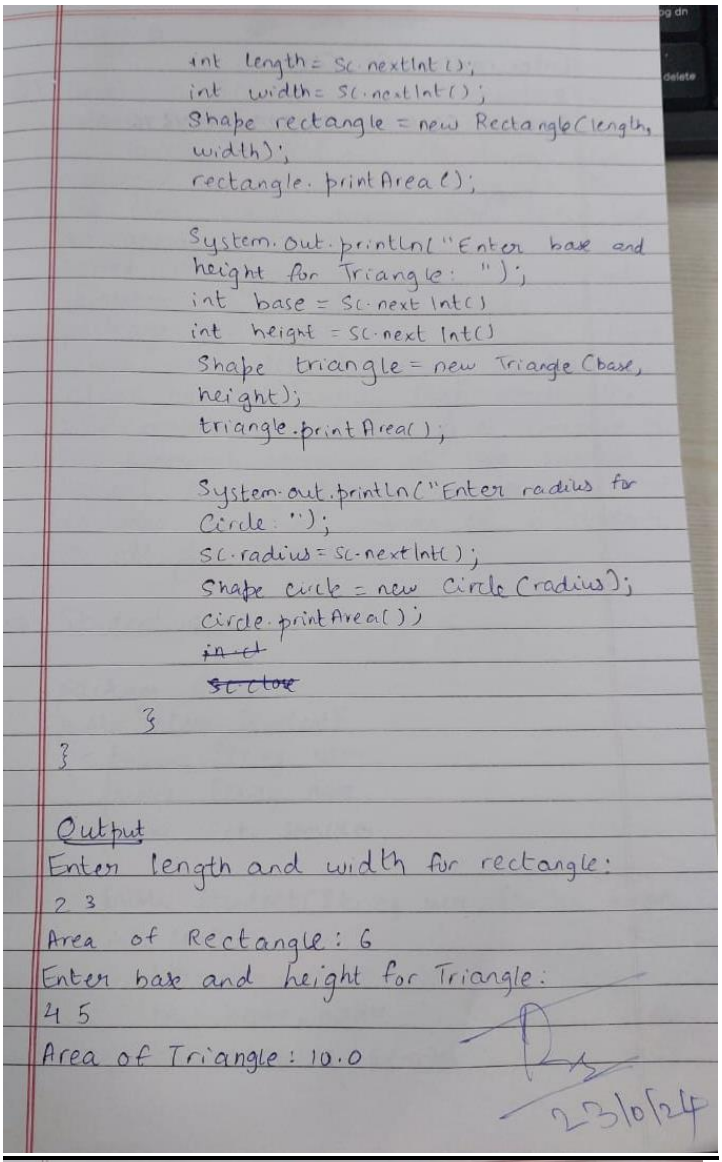
Q) Develop a Java program to create an abstract class named shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea()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

ans

```
import java.util.Scanner;
import java.lang.Math.*;
abstract class Shape{
        int dim1;
        int dim2;

        public Shape(){
            this.dim1=0;
            this.dim2=0;
        }

        public Shape(int dim1, int dim2){
            this.dim1=dim1;
            this.dim2=dim2;
        }

        public abstract void printArea();
}
```



```
class Rectangle extends Shape{
    public Rectangle(int length, int width){
        dim1=length;
        dim2=width;
    }

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

class Circle extends Shape{
    public Circle(int radius){
        dim1=radius;
        dim2=0;
    }
    public void printArea(){
        double area = Math.PI*dim1*dim1;
        System.out.println("Area of Circle: "+
        area);
    }
}

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

        System.out.println("Enter length and
        width for Rectangle: ");
```



```
int length = sc.nextInt();
int width = sc.nextInt();
Shape rectangle = new Rectangle(length,
width);
rectangle.printArea();

System.out.println("Enter base and
height for Triangle: ");
int base = sc.nextInt()
int height = sc.nextInt()
Shape triangle = new Triangle(base,
height);
triangle.printArea();

System.out.println("Enter radius for
Circle: ");
sc.radius = sc.nextInt();
Shape circle = new Circle(radius);
circle.printArea());
```

~~in t~~

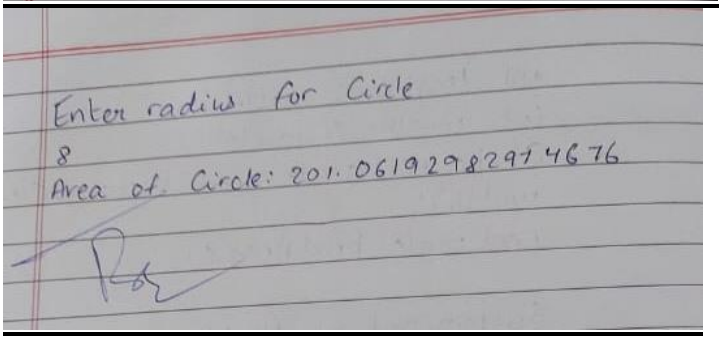
~~sc.clos~~

```
}
}
```

Output

```
Enter length and width for rectangle:
2 3
Area of Rectangle: 6
Enter base and height for Triangle:
4 5
Area of Triangle: 10.0
```

23/10/24



```
Enter radius for Circle
8
Area of Circle: 201.06192982974676
```

Code:

```
import java.util.Scanner;
abstract class Shape {
    int dim1;
    int dim2;
```

```
    public Shape() {
        this.dim1 = 0;
        this.dim2 = 0;

    }

    public Shape(int dim1, int dim2) {
        this.dim1 = dim1;
        this.dim2 = dim2;
    }

    public abstract void printArea();

}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        dim1 = length;
        dim2 = width;
    }

    public void printArea() {

        int area = dim1 * dim2;
        System.out.println("Area of Rectangle: " + area);

    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        dim1 = base;
        dim2 = height;
    }

    public void printArea() {

        double area = 0.5 * dim1 * dim2;
        System.out.println("Area of Triangle: " + area);

    }
}

class Circle extends Shape {
```

```
    public Circle(int radius) {

        dim1 = radius;
        dim2 = 0;
    }

    public void printArea() {

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

public class shapearea{

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
            System.out.println("AMAL ROY 1BM23CS025\n\n");
        System.out.println("Enter length and width for Rectangle:");

        int length = sc.nextInt();
        int width = sc.nextInt();
        Shape rectangle = new Rectangle(length, width);
        rectangle.printArea();

        System.out.println("Enter base and height for Triangle:");

        int base = sc.nextInt();
        int height = sc.nextInt();
        Shape triangle = new Triangle(base, height);
        triangle.printArea();

        System.out.println("Enter radius for Circle:");

        int radius = sc.nextInt();
        Shape circle = new Circle(radius);
        circle.printArea();


    }
}
```

OUTPUT

```
AMAL ROY 1BM23CS025


Enter length and width for Rectangle:
2 3
Area of Rectangle: 6
Enter base and height for Triangle:
4 5
Area of Triangle: 10.0
Enter radius for Circle:
8
Area of Circle: 201.06192982974676
```

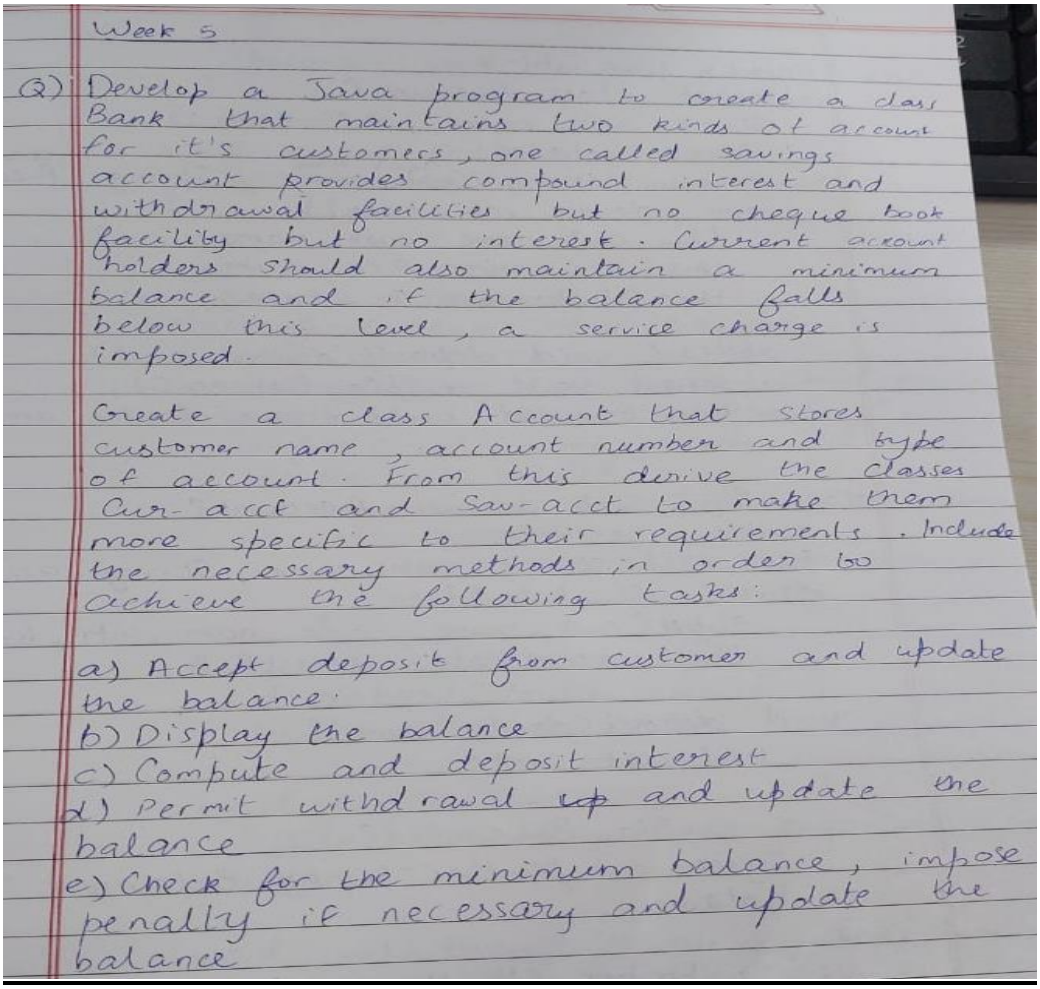
Program 5

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

a) Accept deposit from customer and update the balance
b) Display the balance.
c) Compute and deposit interest
d) Permit withdrawal and update the balance

Check for the minimum balance, impose penalty if necessary and update the balance.

Observation Book



Week 5

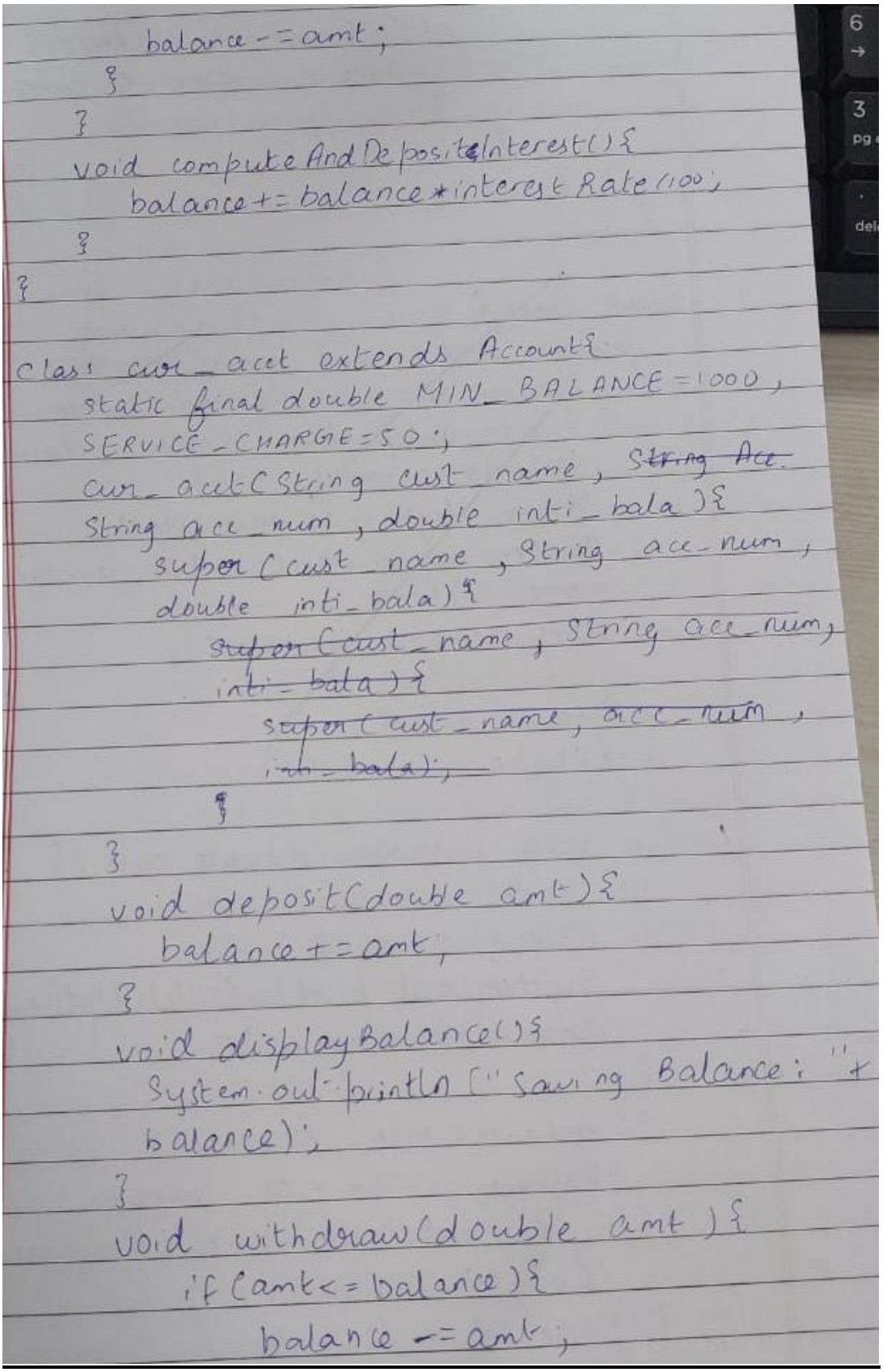
Q) Develop a Java program to create a class Bank that maintains two kinds of account for it's customers, one called savings account provides compound interest and withdrawal facilities but no cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

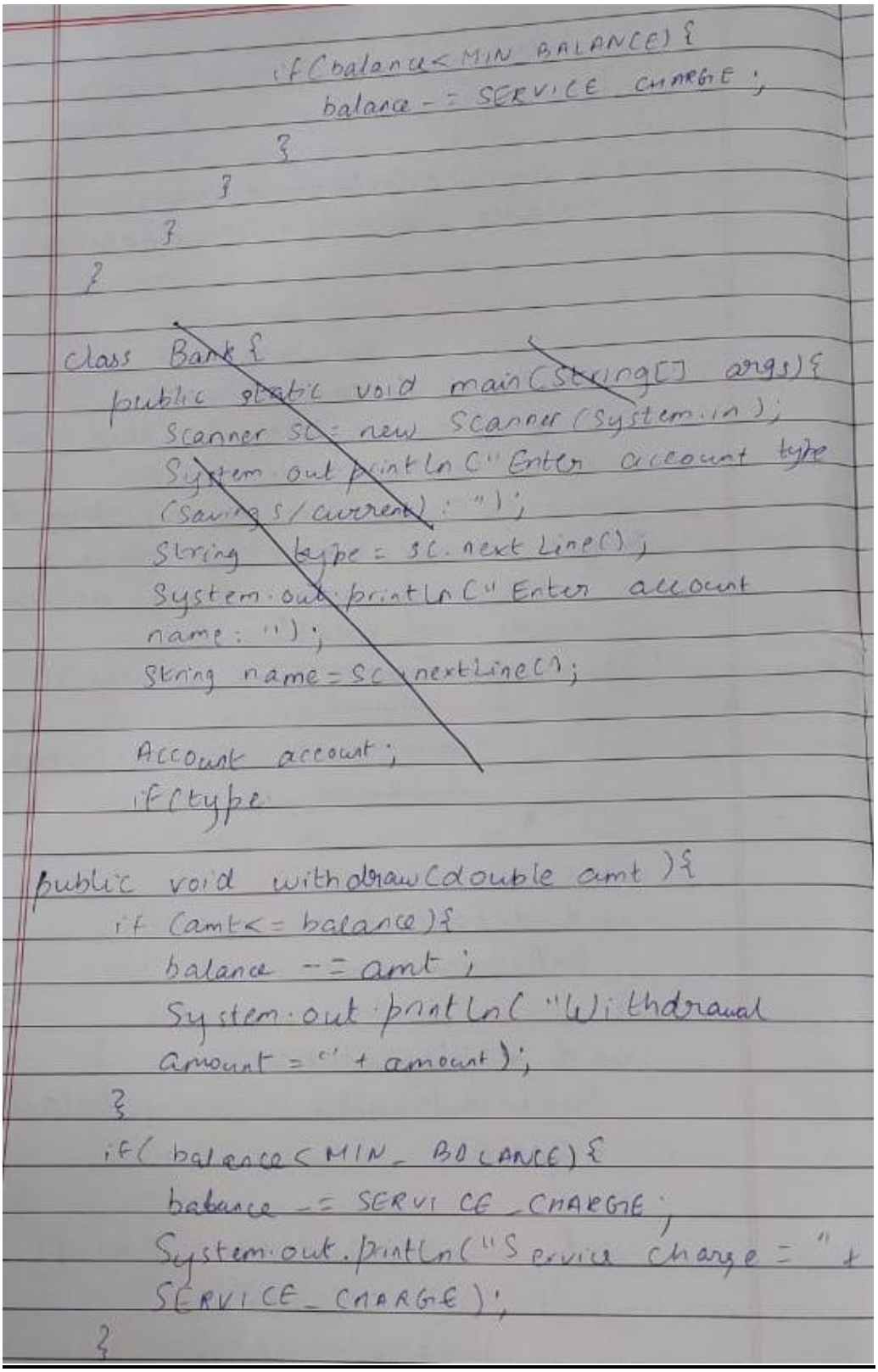
a) Accept deposit from customer and update the balance.
b) Display the balance
c) Compute and deposit interest
d) Permit withdrawal ~~up~~ and update the balance
e) Check for the minimum balance, impose penalty if necessary and update the balance

Ans

```
import java.util.*;
abstract class Account{
    String cust_name, acc_num;
    double balance;
    Account(String cust_name, String Acc_
    num, double inti_bala){
        this.cust_name = cust_name;
        this.acc_num = Acc_num;
        this.balance = inti_bala;
    }
    abstract void deposit(double amt);
    abstract void displayBalance();
    abstract void withdraw(double amt);
}

class Sav_acct extends Account{
    double interestRate;
    Sav_Acct(String cust_name, String acc_num,
    double inti_bala){
        super(cust_name, acc_num, inti_bala);
        this.interestRate = interestRate;
    }
    void deposit(double amt){
        balance+=amt;
    }
    void displayBalance(){
        System.out.println("Saving Balance: " +
        balance);
    }
    void withdraw(double amt){
        if(amt<=balance){
```

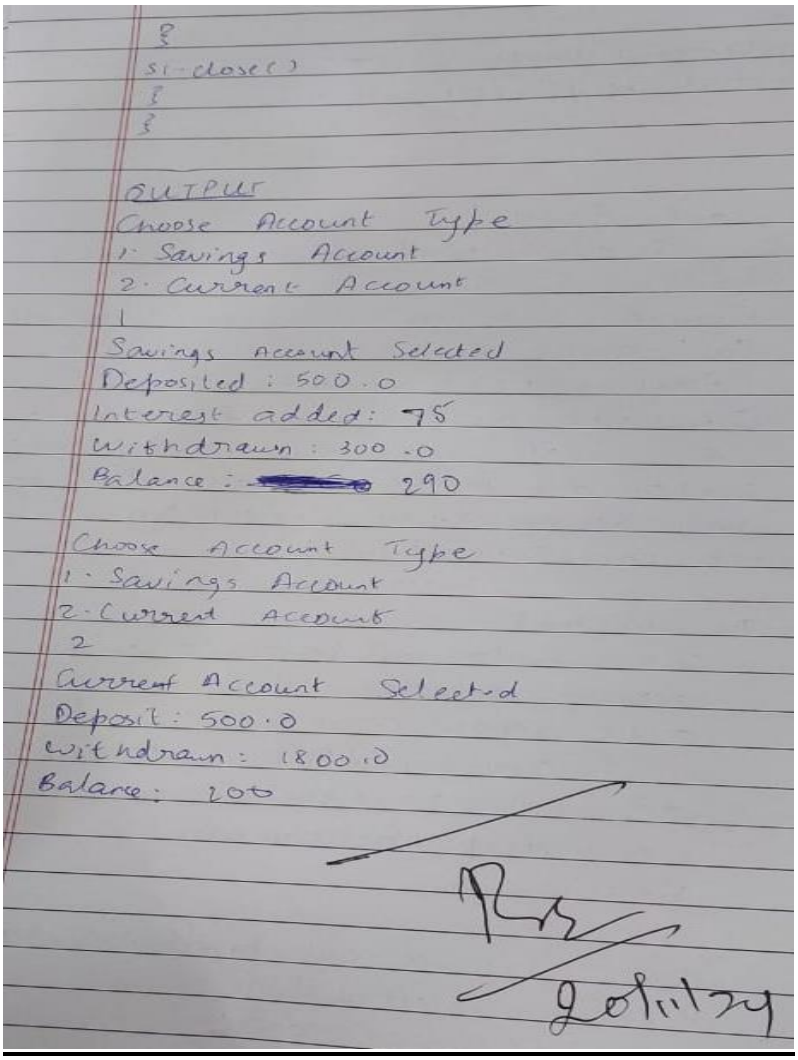


```
        balance -= amt;
    }
}
void computeAndDepositInterest(){
    balance += balance * interestRate/100;
}
}

class cur_acct extends Account{
    static final double MIN_BALANCE = 1000,
    SERVICE_CHARGE = 50;
    cur_acct(String cust_name, String Acc.
    String acc_num, double inti_bala){
        super(cust_name, String acc_num,
        double inti_bala){
            super(cust_name, String acc_num,
            inti_bala){
                super(cust_name, acc_num,
                inti_bala);
            }
    }
    void deposit(double amt){
        balance += amt;
    }
    void displayBalance(){
        System.out.println("Saving Balance: "+
        balance);
    }
    void withdraw(double amt){
        if(amt <= balance){
            balance -= amt;
```



```
        if(balance< MIN_BALANCE){
            balance -= SERVICE_CHARGE;
        }
    }
}

class Bank{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter account type
        (Savings/current): ");
        String type = sc.nextLine();
        System.out.println("Enter account
        name: ");
        String name = sc.nextLine();

        Account account;
        if(type

public void withdraw(double amt){
    if (amt<= balance){
        balance -= amt;
        System.out.println("Withdrawal
        amount = " + amount);
    }
    if( balance< MIN_BALANCE){
        balance -= SERVICE_CHARGE;
        System.out.println("Service charge = " +
        SERVICE_CHARGE);
    }
```

```
    else {
        System.out.println("Service Charge
        deduct Insufficient balance.");
    }
}

public class Bank {
    public static void main(String[]
    args){
        Scanner sc = new Scanner(System.in);
        Sav_acct acc1 = new Sav_acct(
        "AMAL", 123, 12345, 15);
        Cur_acct acc2 = new Cur_acct(
        "A", 1234, 100000, 5000, 500);
        System.out.println("Choose Acc type
        \n1. Savings \n2. Current \n");
        int ch = sc.nextInt();

        switch(ch){
            case 1: System.out.println("Savings
            account selected.");
            acc1.deposit(500);
            acc1.computeAndDepositInterest();
            acc1.withdraw(300);
            acc1.displayBalance();
            break;

            case 2: System.out.println("Current
            Account Selected");
            acc2.deposit(500);
            acc2.withdraw(1000);
            acc2.displayBalance();
            break;
```



Code:

```
import java.util.*;
abstract class Account {
   String Cust_name ,Acc_num;
   double balance;
   Account(String Cust_name,String Acc_num, double inti_bala){
         this.Cust_name=Cust_name;
         this.Acc_num=Acc_num;
         this.balance=inti_bala;
  }
  abstract void deposit(double amt);
  abstract void displayBalance();
  abstract void withdraw(double amt);
}

class Sav_Acct extends Account{
    double intrestRate;
```

```
    Sav_Acct(String Cust_name,String Acc_num, double inti_bala){
                super(Cust_name,Acc_num,inti_bala);
                this.intrestRate=intrestRate;
    }
    void deposit(double amt){
      balance+=amt;
    }
    void displayBalance(){
        System.out.println("Saving Balance:"+balance);
    }
    void withdraw(double amt){
     if(amt<=balance){
        balance-=amt;
     }
    }
    void computeAndDepositeIntrest(){
         balance+=balance*intrestRate/100;
    }
}

class Cur_Acct extends Account{
   static final double MIN_BALANCE=1000, SEVICE_CHARGE=50;
   Cur_Acct(String Cust_name,String Acc_num, double inti_bala){
              super(Cust_name,Acc_num,inti_bala);
   }
   void deposit(double amt){
      balance+=amt;
   }
   void displayBalance(){
        System.out.println("Saving Balance:"+balance);
   }
   void withdraw(double amt){
     if(amt<=balance){
        balance-=amt;
        if(balance<MIN_BALANCE){
          balance-=SEVICE_CHARGE;
        }
     }
   }
}

class Bank{

    public static void main(String[] args){
        Scanner scn=new Scanner(System.in);
        System.out.println("Enter account type(savings/current):");
```

```
        String type=scn.nextLine();

        System.out.println("Enter account name:");
        String name=scn.nextLine();

        System.out.println("Enter account number:");
        String number=scn.nextLine();

       Account account;
       if(type.equals("Savings")){
           System.out.println("Initial balance and interest rate:");
           account=new Sav_Acct(name,number,scn.nextDouble());
       }
       else{
          System.out.println("Intial balance :");
          account =new Cur_Acct(name,number,scn.nextDouble());
       }
       while(true){
           System.out.println("1.Deposit  2.Display Balance    3.Withdraw   4.Interest   5.Exit");
           int choice=scn.nextInt();
           switch(choice){
             case 1:account.deposit(scn.nextDouble());
                    break;
             case 2: account.displayBalance();
                    break;
             case 3: account.withdraw(scn.nextDouble());
                    break;
             case 4: if(account instanceof Sav_Acct){
                    ((Sav_Acct)account).computeAndDepositeIntrest();
                    }
                    break;
             case 5 : return;
            }
       }
   }
}
```

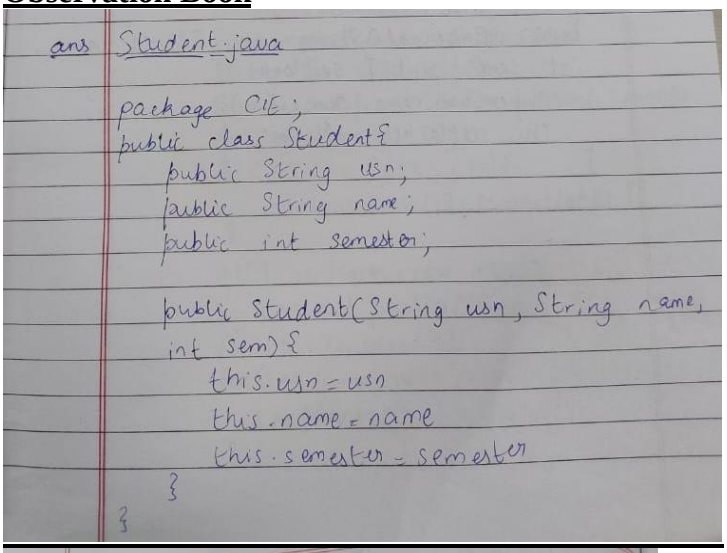
OUTPUT

```
AMAL ROY 1BM23CS025
Choose Account Type:
1. Savings Account
2. Current Account
1
Savings Account Selected
Deposited: 300.0
Interest added: 65.0
Withdrawn: 100.0
Balance: 1265.0
```

Program 6

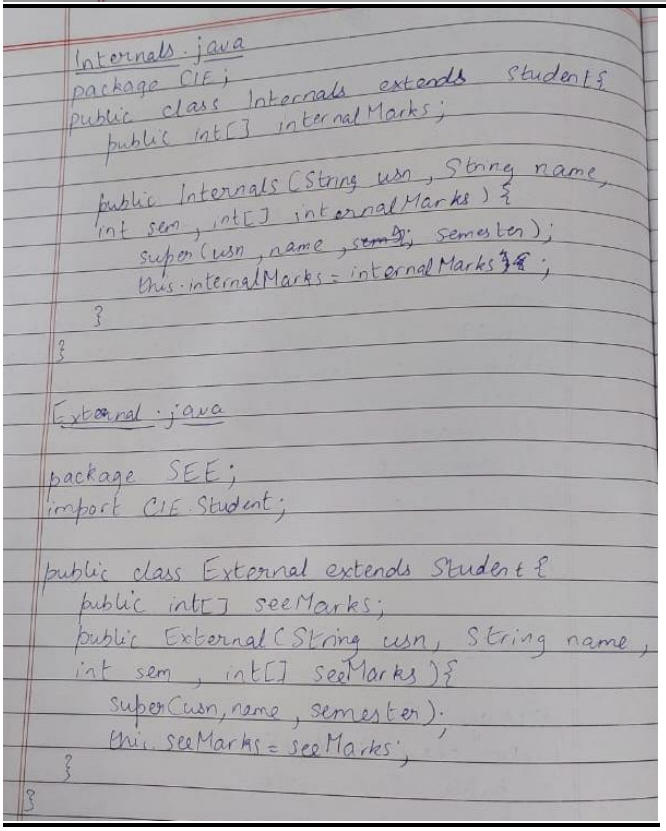
Create a package CIE which has two classes - Personal and Internals. The class Personal has members like usn, name, sem. The class Internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External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

Observation Book



ans Student.java

```
package CIE;
public class Student{
    public String usn;
    public String name;
    public int semester;

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



Internals.java

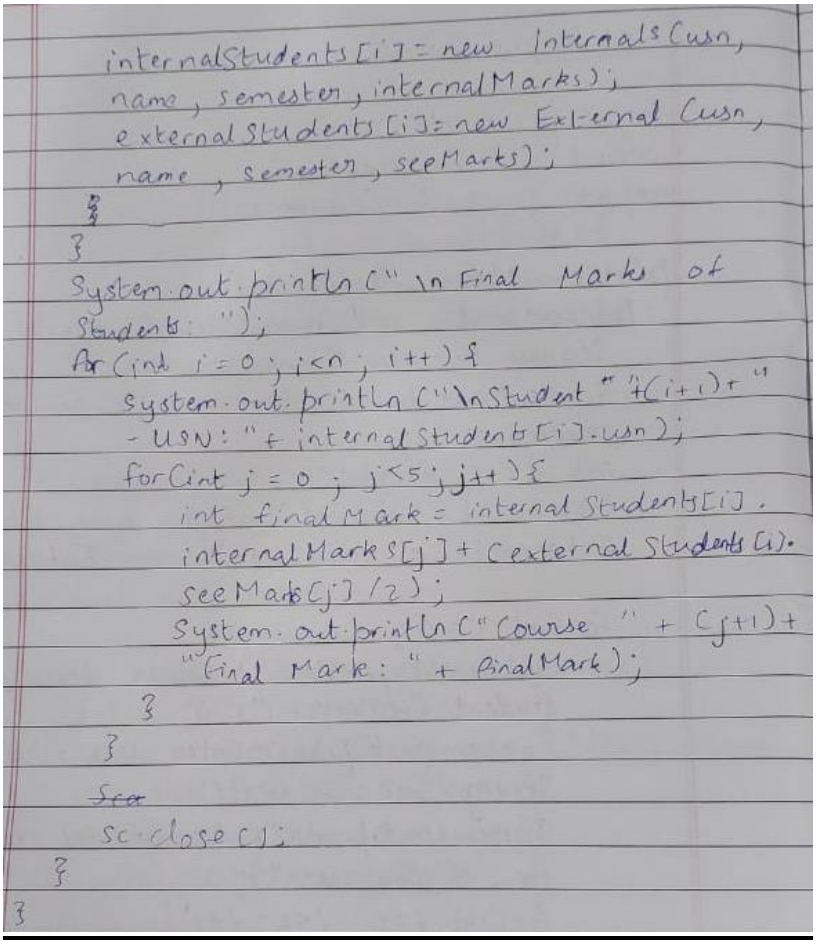
```
package CIE;
public class Internals extends Student{
    public int[] internalMarks;

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

External.java

```
package SEE;
import CIE.Student;

public class External extends Student{
    public int[] seeMarks;
    public External(String usn, String name,
    int sem, int[] seeMarks){
        super(usn, name, semester);
        this.seeMarks = seeMarks;
    }
}
```

MainProgram.Java

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

public class FinalMarks{
  public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter number of students: ");
    int n = sc.nextInt();
    Internals[] internalStudents = new Internals[n];
    External[] externalStudents = new External[n];

    for(int i=0; i<n; i++){
      System.out.println("\nEnter details for student "+(i+1)+":");
      System.out.print("Enter USN: ");
      String usn = sc.next();
      System.out.print("Enter internal marks for 5 courses: ");
      for(int j=0; j<5; j++){
        internalMarks[j] = sc.nextInt()
      }
      int[] seeMarks = new int[5];
      System.out.println("Enter SEE marks for 5 courses: ");
      for(int j=0; j<5; j++){
        seeMarks[j] = sc.nextInt();
      }
```



Code:

Internals.java

```
package CIE;

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

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

Student.java

```
package CIE;

public class Student {
   public String usn;
   public String name;
   public int semester;
```

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

External.java

```
package SEE;
import CIE.Student;

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

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

MainProgram.java

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

public class MainProgram{
    public static void main(String[] args) {
            System.out.println("AMALROY 1BM23CS025");
        Scanner scanner = new Scanner(System.in);

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

        Internals[] internalStudents = new Internals[n];
        External[] externalStudents = new External[n];

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for student " + (i + 1) + ":");

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

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

            System.out.print("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++) {
            internalMarks[j] = scanner.nextInt();
        }

        int[] seeMarks = new int[5];
        System.out.println("Enter SEE marks for 5 courses:");
        for (int j = 0; j < 5; j++) {
            seeMarks[j] = scanner.nextInt();
        }

        internalStudents[i] = new Internals(usn, name, sem, internalMarks);
        externalStudents[i] = new External(usn, name, sem, seeMarks);
    }

    System.out.println("\nFinal Marks of Students:");
    for (int i = 0; i < n; i++) {
        System.out.println("\nStudent " + (i + 1) + " - USN: " + internalStudents[i].usn);
        for (int j = 0; j < 5; j++) {
            int finalMark = internalStudents[i].internalMarks[j] + (externalStudents[i].seeMarks[j]
/ 2);
            System.out.println("Course " + (j + 1) + " Final Mark: " + finalMark);
        }
    }

    scanner.close();
        System.out.println("AMALROY 1BM23CS025");
  }
}
```

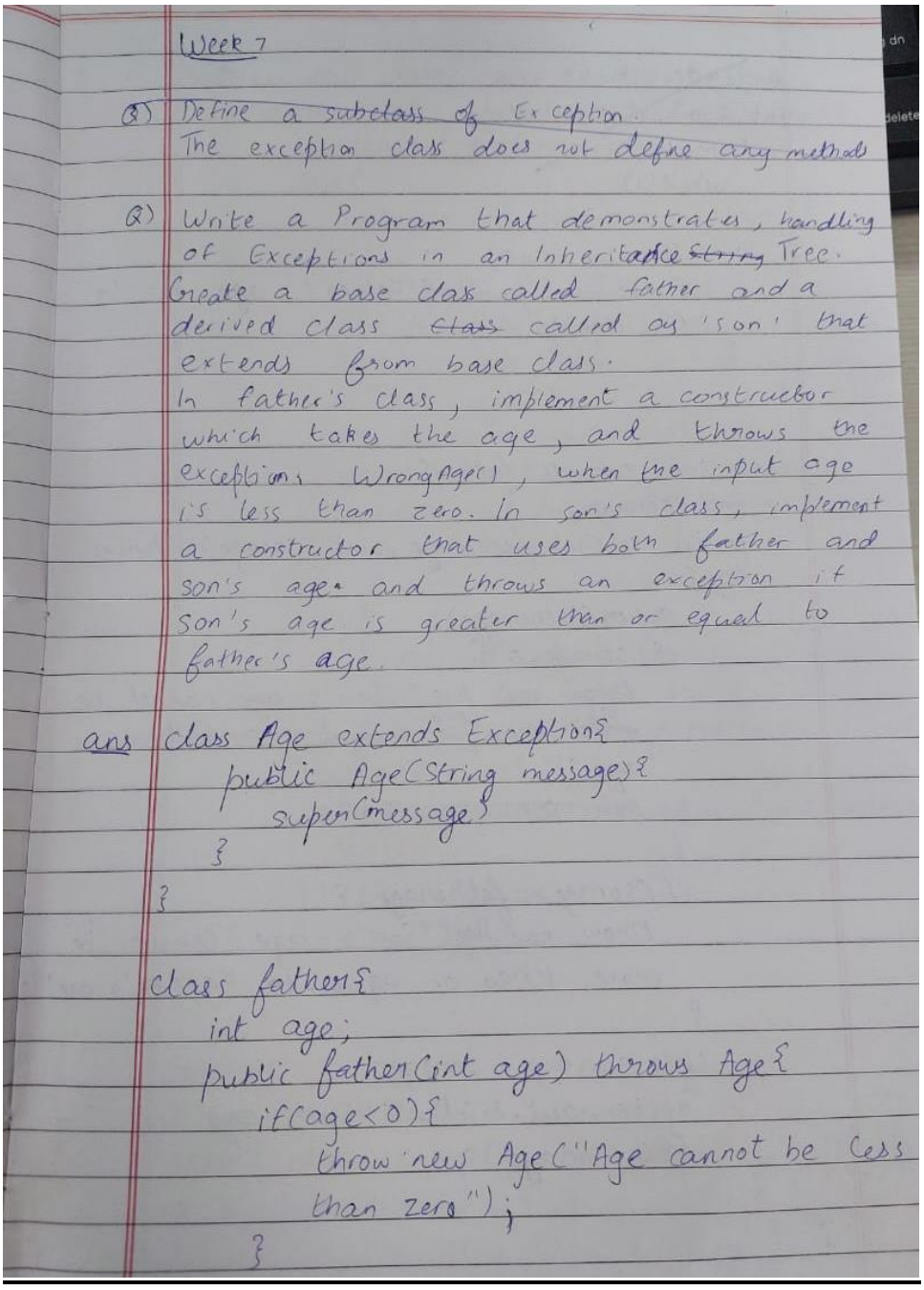
OUTPUT

```
AMALROY 1BM23CS025
Enter number of students: 2

Enter details for student 1:
USN: 1
Name: A
Semester: 3
Enter internal marks for 5 courses:
45 49 48 47 46
Enter SEE marks for 5 courses:
90 98 97 94 93

Enter details for student 2:
USN: 2
Name: B
Semester: 2
Enter internal marks for 5 courses:
34 35 2 23 49
Enter SEE marks for 5 courses:
90 98 45 3 46

Final Marks of Students:

Student 1 - USN: 1
Course 1 Final Mark: 90
Course 2 Final Mark: 98
Course 3 Final Mark: 96
Course 4 Final Mark: 94
Course 5 Final Mark: 92

Student 2 - USN: 2
Course 1 Final Mark: 79
Course 2 Final Mark: 84
Course 3 Final Mark: 24
Course 4 Final Mark: 24
Course 5 Final Mark: 72
```

Program 7

Write a program that demonstrates handling of exceptions in inheritance tree. Create a case class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that uses both father and son's age and throws an exception if son's age is >=father's age.

Observation Book



Week 7

Q) Define a subclass of Exception.
The exception class does not define any methods

Q) Write a Program that demonstrates, handling of Exceptions in an Inheritance ~~String~~ Tree.
Create a base class called father and a derived class ~~class~~ called as 'son' that extends from base class.
In father's class, implement a constructor which takes the age, and throws the exceptions WrongAge(), when the input age is less than zero. In son's class, implement a constructor that uses both father and son's age and throws an exception if son's age is greater than or equal to father's age.

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

class father{
    int age;
    public father(int age) throws Age{
        if(age<0){
            throw new Age("Age cannot be less
            than zero");
        }
```

```
        this.age = age;
        System.out.println("Father's age is= " +
        this.age);
    }
}

class son extends father{
    int sonage;
    public son(int fatherage, int sonage) throws
    Age{
        super(fatherage);
        if(sonage<0){
            throw new Age("Son's age cannot be
            more(less) than zero(or) equal to Father's
            age");
            son son1 = new son(6
        }
        if(sonage >= fatherage){
            throw new Age("Son's age cannot be
            more than or equal to Father's age");
        }
        this.sonage = sonage;
        System.out.println("Son's age is= " +
        this.sonage);
    }
}
```

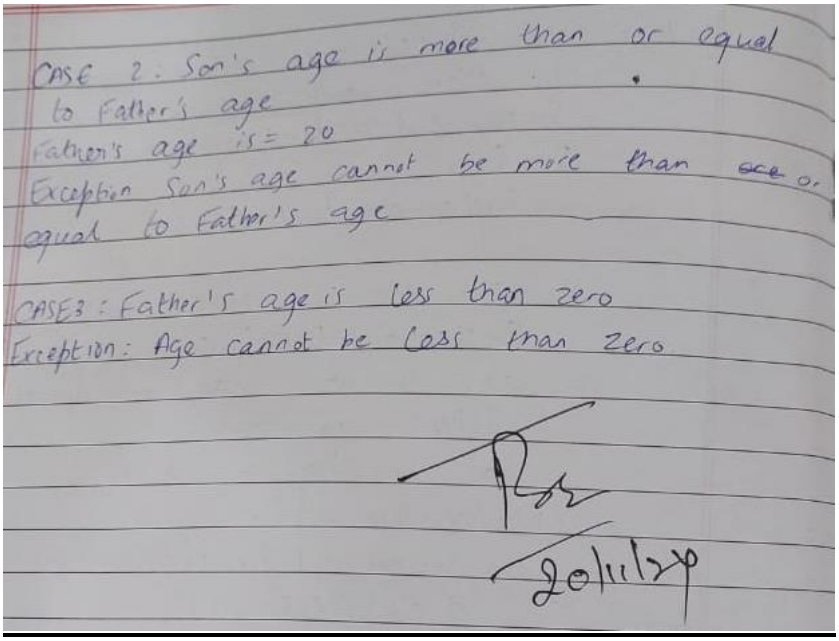
```
public class InheritanceTree {
    public static void main(String[] args) {
        System.out.println("AHA
        try {
            System.out.println("CASE 1: Valid case
            ");
            Son son1 = new Son(60, 20);
            System.out.println("\nCASE 2: Son's
            age is more than or equal to
            Father's age");
            Son son2 = new Son(20,30);
        } catch(Age a){
            System.out.println("Exception: " +
            a.getMessage());
        }
        try{
            System.out.println("\nCASE 3: Father's
            age is less than zero");
            father father1 = new father(-20);
        } catch(Age a){
            System.out.println("Exception: " + a.getMessage
            ());
        }
    }
}
```

OUTPUT

CASE 1: Valid case

Father's age is = 60

Son's age is = 20



Code

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

class father{
    int age;
    public father(int age) throws Age {
        if (age < 0) {
            throw new Age("Age cannot be less than zero");
        }
        this.age = age;
        System.out.println("Father's age is= " + this.age);
    }
}

class son extends father{
    int sonage;
    public son(int fatherage, int sonage) throws Age {
        super(fatherage);
        if (sonage < 0) {
            throw new Age("Son's age cannot be less than zero");
        }
        if (sonage >= fatherage) {
            throw new Age("Son's age cannot be more than or equal to Father's age");
```

```
        }
        this.sonage = sonage;
        System.out.println("Son's age is= " + this.sonage);
    }
}

public class InheritanceTree{
    public static void main(String[] args) {
            System.out.println("AMALROY 1BM23CS025");
        try{
            System.out.println("CASE 1: Valid case");
            son son1 = new son(60, 20);
            System.out.println("\nCASE 2: Son's age is more than or equal to Father's age");
            son son2 = new son(20, 30);
        } catch(Age a){

            System.out.println("Exception: " + a.getMessage());
        }
        try{
            System.out.println("\nCASE 3: Father's age is less than zero");
            father father1 = new father(-20);
        }
            catch (Age a){
            System.out.println("Exception: " + a.getMessage());
        }
    }
}
```

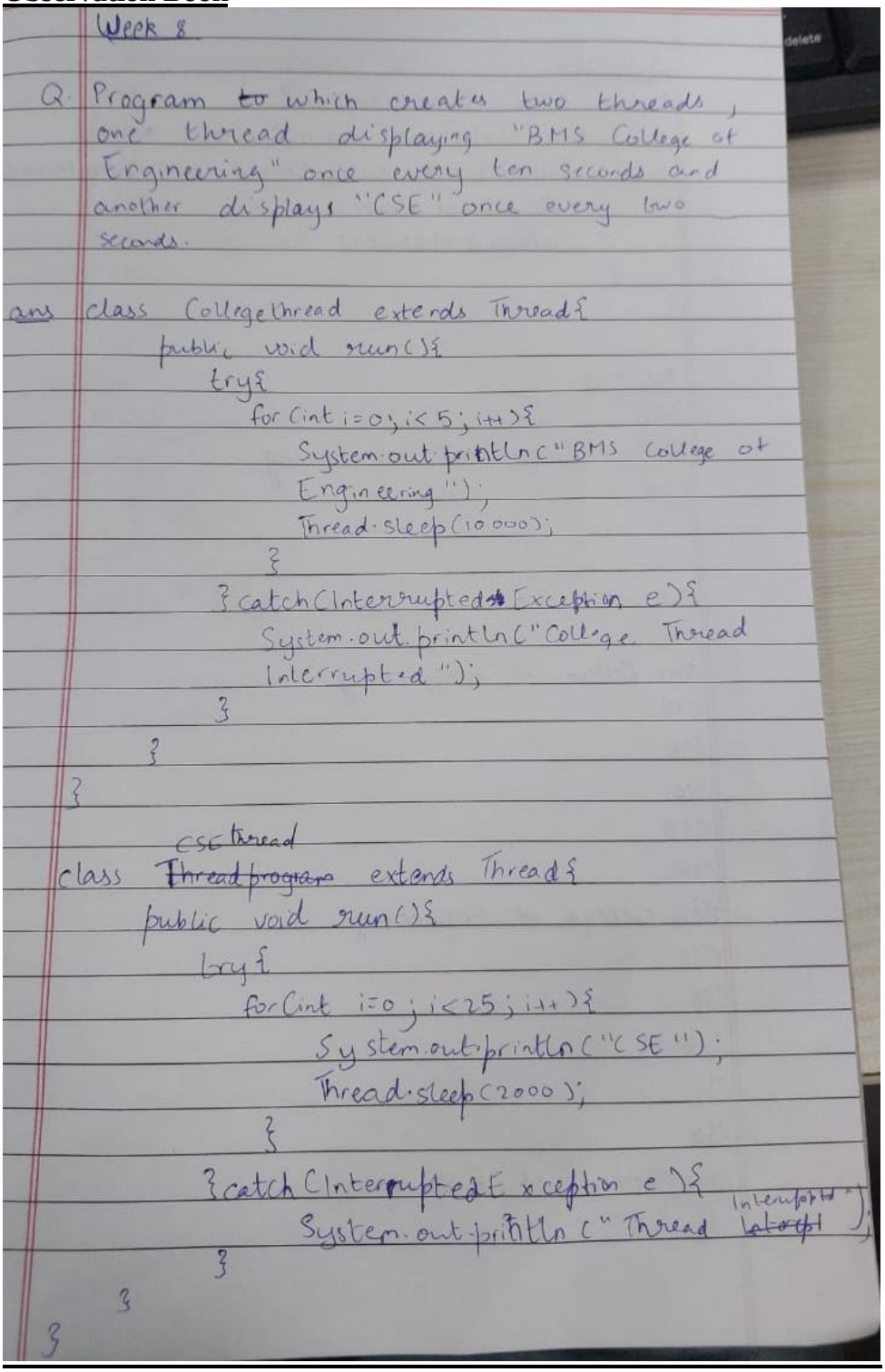
OUTPUT

```
AMALROY 1BM23CS025
CASE 1: Valid case
Father's age is= 60
Son's age is= 20

CASE 2: Son's age is more than or equal to Father's age
Father's age is= 20
Exception: Son's age cannot be more than or equal to Father's age

CASE 3: Father's age is less than zero
Exception: Age cannot be less than zero
```

Program 8

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

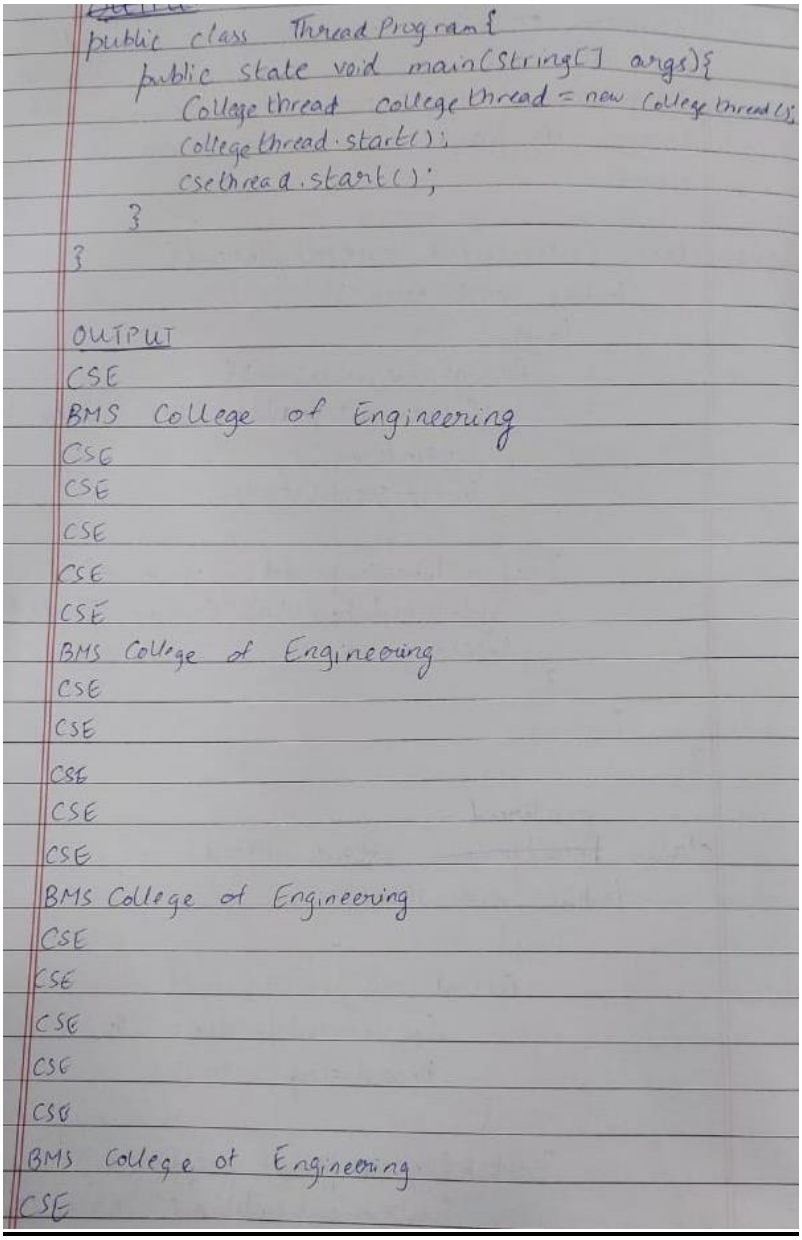
Observation Book



Week 8

Q. Program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displays "CSE" once every two seconds.

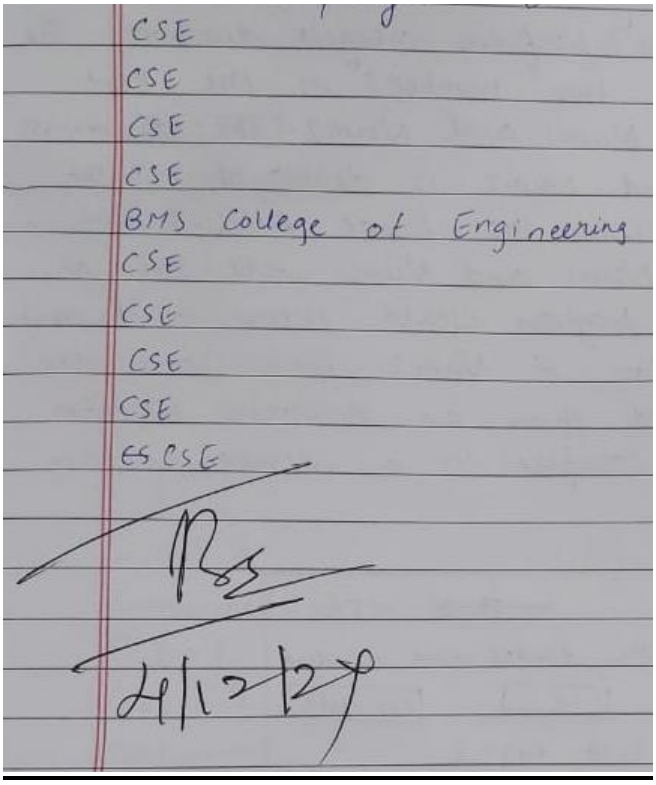
ans
```
class CollegeThread extends Thread{
    public void run(){
        try{
            for(int i=0;i<5;i++){
                System.out.println("BMS College of
                Engineering");
                Thread.sleep(10000);
            }
        }catch(InterruptedException e){
            System.out.println("College Thread
            Interrupted");
        }
    }
}

CSE Thread
class Threadprogram extends Thread{
    public void run(){
        try{
            for(int i=0;i<25;i++){
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        }catch(InterruptedException e){
            System.out.println("Thread Interrupted");
        }
    }
}
```



```
public class ThreadProgram{
    public state void main(String[] args){
        CollegeThread collegeThread = new CollegeThread();
        collegeThread.start();
        cseThread.start();
    }
}
```

OUTPUT

```
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
```



Code

```
class Collegethread extends Thread{
    public void run(){
        try{
            for(int i=0;i<5;i++){
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            }
        }
        catch(InterruptedException e){
            System.out.println("College Thread interrupted.");
        }
    }
}

class CSEthread extends Thread{
    public void run(){
        try{
            for (int i = 0; i < 25; i++){
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        }
        catch (InterruptedException e){
```

```
            System.out.println("CSE Thread interrupted.");
        }
    }
}

public class ThreadProgram{
    public static void main(String[] args){
        System.out.println("AMAL ROY 1BM23CS025");
        Collegethread collegethread=new Collegethread();
        CSEthread csethread=new CSEthread();
        collegethread.start();
        csethread.start();
    }
}
```

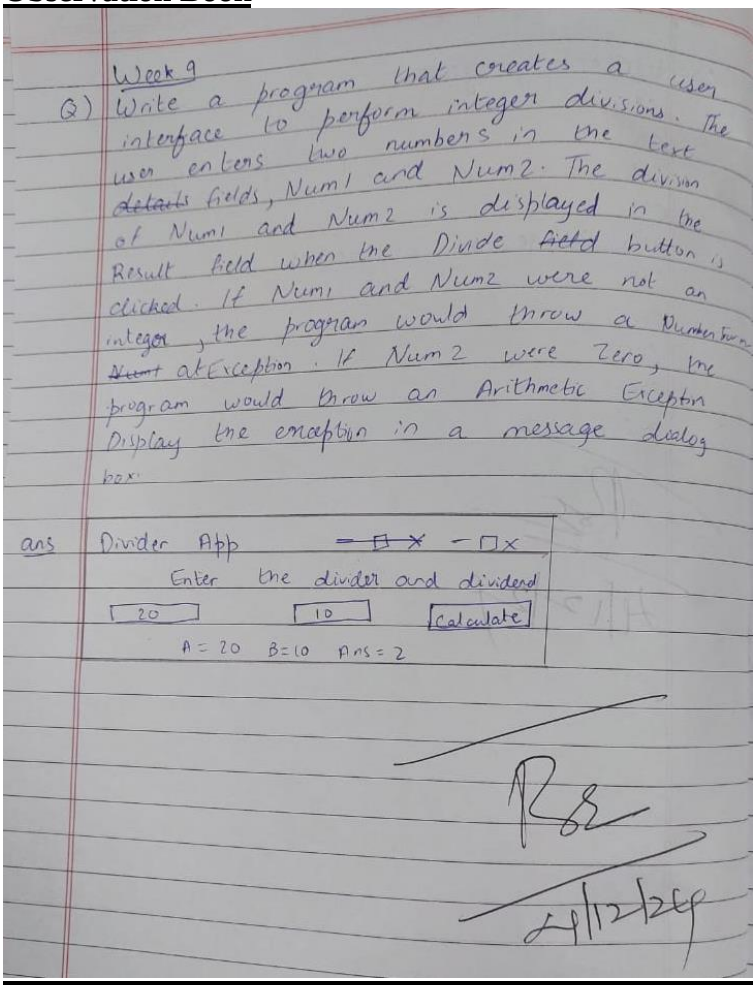
OUTPUT

```
AMAL ROY 1BM23CS025
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
```

Week 9

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

Observation Book



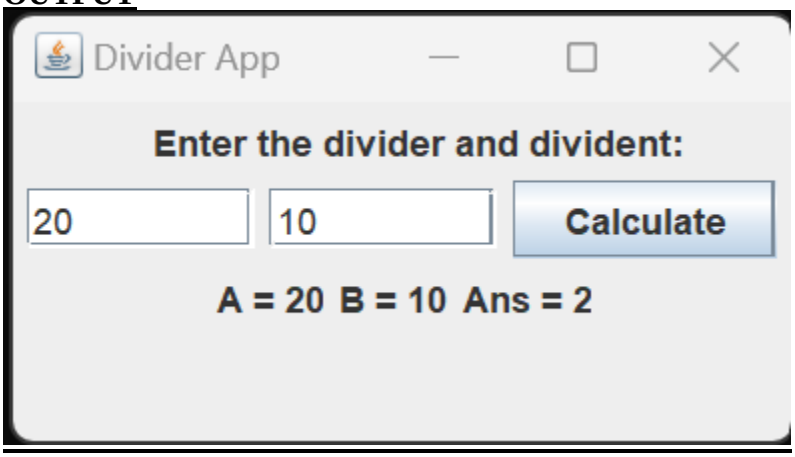
Code

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

class SwingDemo {

   SwingDemo() {
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 150);
```

```
jfrm.setLayout(new FlowLayout());

jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JLabel jlab = new JLabel("Enter the divider and 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); // to display error messages
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

ActionListener l = new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
      System.out.println("Action event from a text field");
   }
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
      try {
         int a = Integer.parseInt(ajtf.getText());
         int b = Integer.parseInt(bjtf.getText());
         int ans = a / b;
         alab.setText("\nA = " + a);
         blab.setText("\nB = " + b);
         anslab.setText("\nAns = " + ans);
      } catch (NumberFormatException e) {
         alab.setText("");
         blab.setText("");
         anslab.setText("");
         err.setText("Enter Only Integers!");
      } catch (ArithmeticException e) {
```

```
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("B should be NON zero!");
            }
        }
    });
    jfrm.setVisible(true);
  }

  public static void main(String args[]) {
    // create frame on event dispatching thread
    SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        new SwingDemo();
      }
    });
  }
}
```

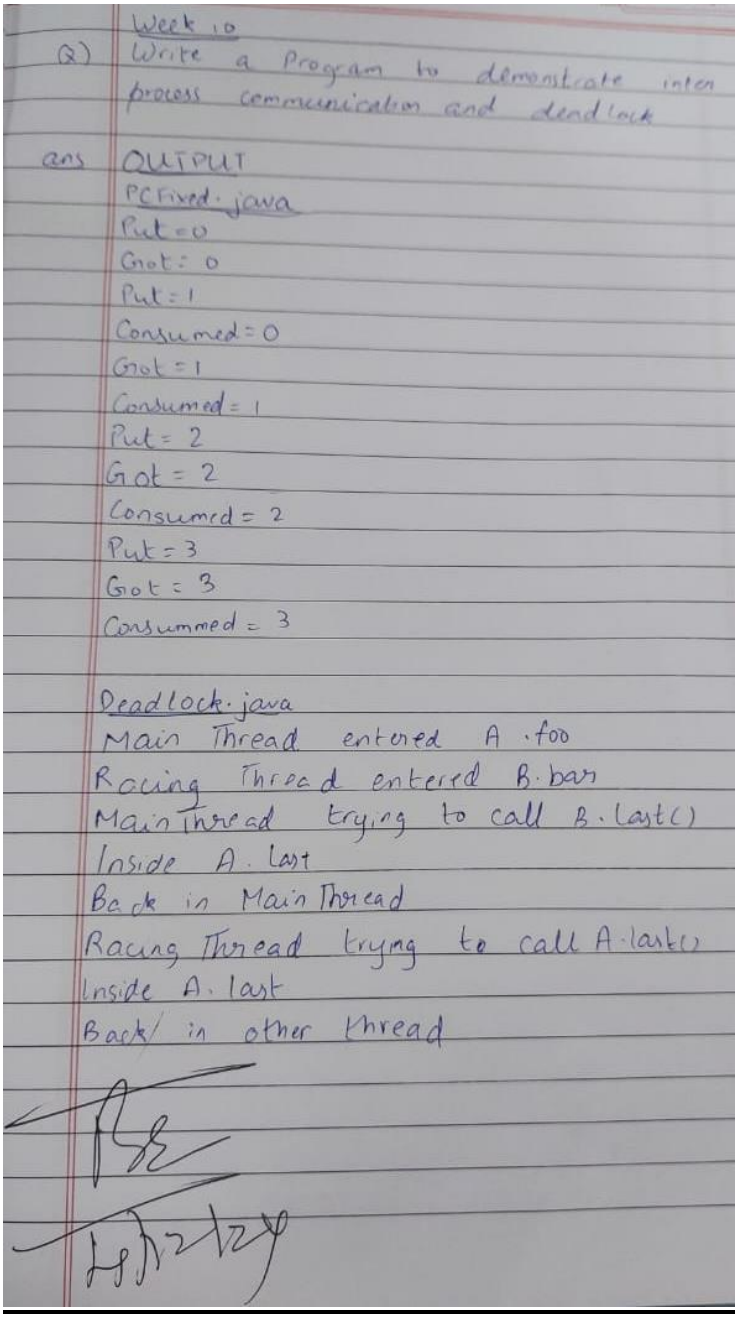
OUTPUT



Week 10

Demonstrate Inter process Communication and deadlock

Observation Book


Week 10

(2) Write a Program to demonstrate inter process communication and deadlock

ans OUTPUT

PCFixed.java

Put=0
Got: 0
Put: 1
Consumed=0
Got=1
Consumed=1
Put= 2
Got = 2
Consumed = 2
Put=3
Got = 3
Consummed = 3

Deadlock.java

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

Code

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

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

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

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

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

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

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

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

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

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

        a.foo(b);
```

```
        System.out.println("Back in main thread");
    }

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

    public static void main(String args[]) {
            System.out.println("AMAL ROY 1BM23CS025");
        new Deadlock();
    }
}
```

OUTPUT

```
AMAL ROY 1BM23CS025
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
MainThread trying to call B.last()
```