VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Akshat Basra (1BM23CS020)

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 **Akshat Basra (1BM23CS020),** 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.

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

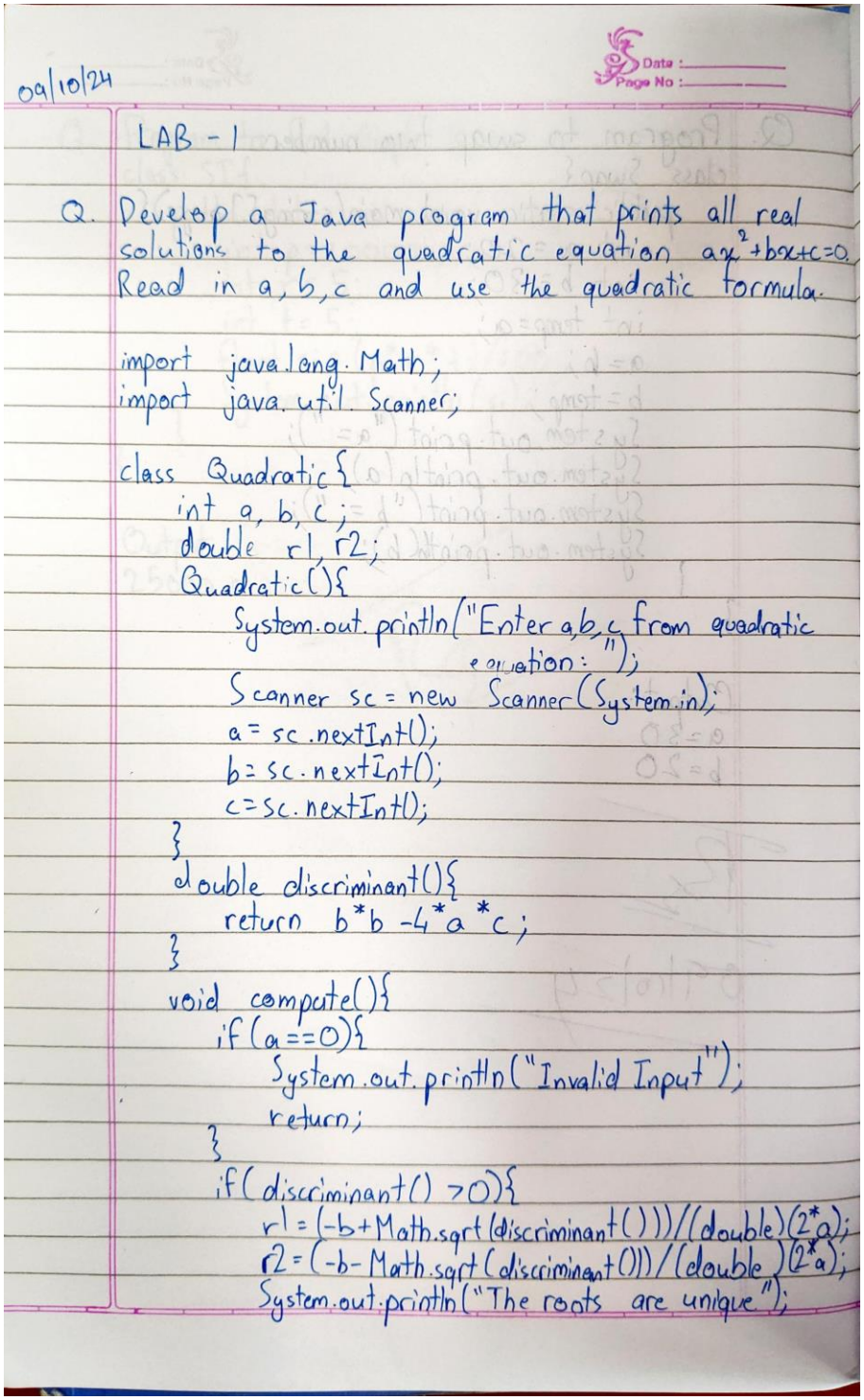
Index

Github Link:
https://github.com/AkshatBasra/JavaLab/tree/ace2e8eb9fb00a8ac4bad80eae33de2b82424c0a/Record

Program 1

Implement Quadratic Equation

Algorithm:



09/10/24

Date :
Page No :

LAB - 1

Q. 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.

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

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

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

class Run{
    public static void main(String[] args){
        Quadratic eq1 = new Quadratic();
        eq1.compute();
        Quadratic eq2 = new Quadratic();
        eq2.compute();
        Quadratic eq3 = new Quadratic();
        eq3.compute();
    }
}
```

Output:

Akshat Basra 1BM23CS020

Enter a, b and c from quadratic equation:

1

1

1

The roots are Imaginary.

First root: 0.0 + i 0.3660254037844386

Second root: 0.0 - i 0.3660254037844386

Enter a, b and c from quadratic equation:

1

-4

4

The roots are equal

The root is: 2.0

Enter a, b and c from quadratic equation:

1

-7

12

The roots are unique

First root: 4.0

Second root: 3.0

09/10/24

Code:

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

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


class Run{
```

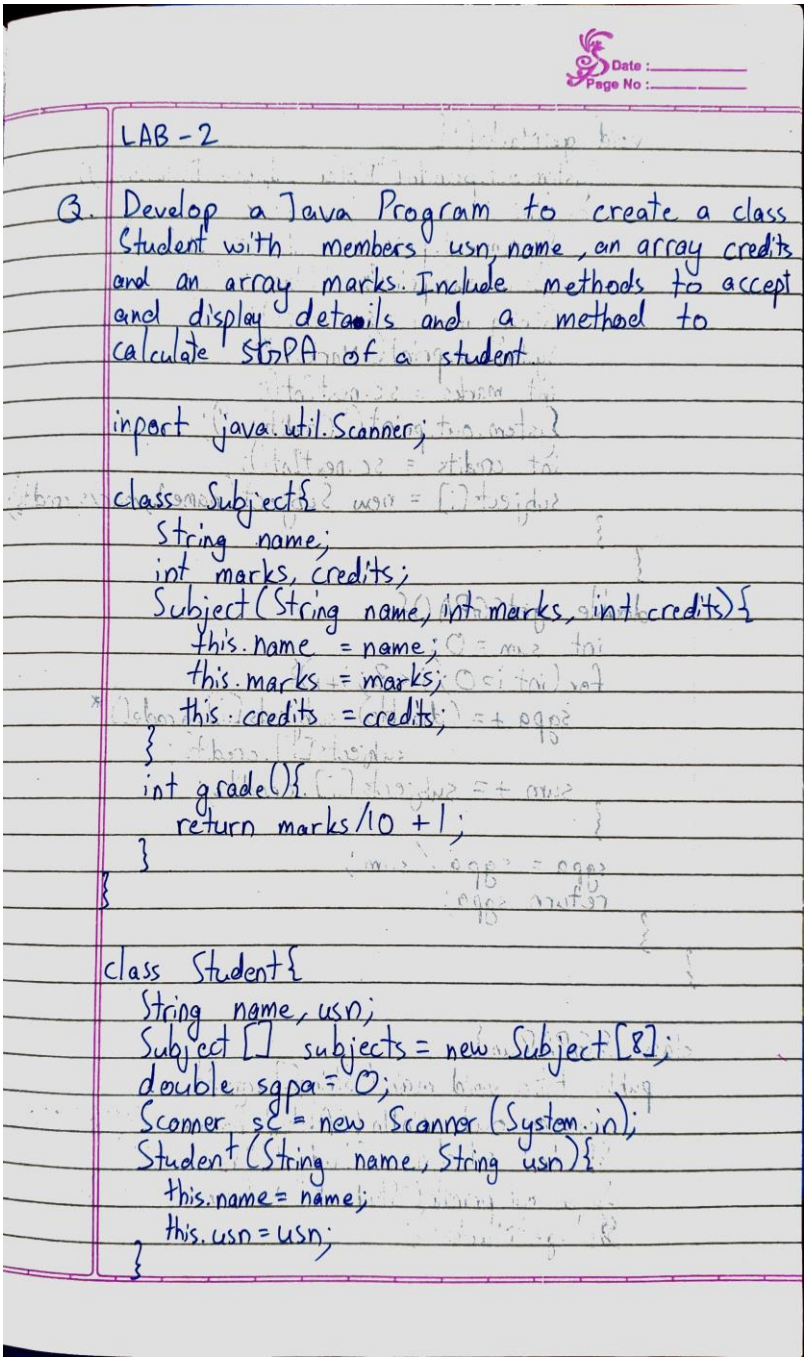
```
        public static void main(String[] args){
                System.out.println("Akshat Basra 1BM23CS020");
                Quadratic eq1 = new Quadratic();
                eq1.compute();
                Quadratic eq2 = new Quadratic();
                eq2.compute();
                Quadratic eq3 = new Quadratic();
                eq3.compute();
        }
}
```

```
D:\1BM23CS020>java Run
Akshat Basra 1BM23CS020
Enter a, b and c from quadratic equation:
1
1
1
The roots are Imaginary
First root: 0.0+i0.36602540378443864
Second root: 0.0-i0.36602540378443864
Enter a, b and c from quadratic equation:
1
-4
4
The roots are equal
The root is: 2.0
Enter a, b and c from quadratic equation:
1
-7
12
The roots are unique
First root: 4.0
Second root: 3.0
```

Program 2

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

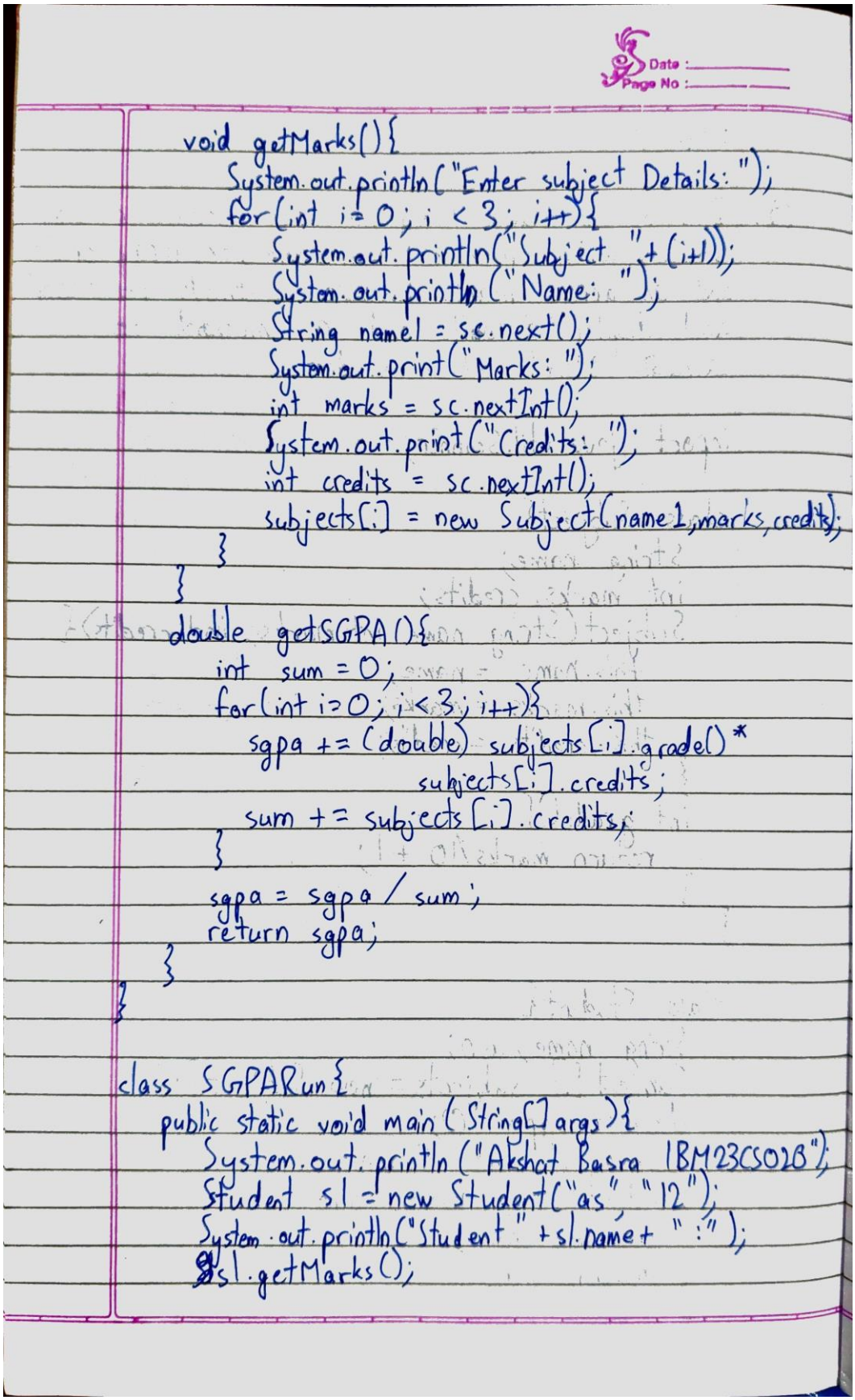
Algorithm:



LAB - 2

Q. Develop a Java Program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

```
import java.util.Scanner;
class Subject{
    String name;
    int marks, credits;
    Subject(String name, int marks, int credits){
        this.name = name;
        this.marks = marks;
        this.credits = credits;
    }
    int grade(){
        return marks/10 + 1;
    }
}

class Student{
    String name, usn;
    Subject[] subjects = new Subject[8];
    double sgpa = 0;
    Scanner s = new Scanner(System.in);
    Student(String name, String usn){
        this.name = name;
        this.usn = usn;
    }
```



```
    void getMarks(){
        System.out.println("Enter subject Details: ");
        for(int i=0; i<3; i++){
            System.out.println("Subject "+(i+1));
            System.out.println("Name: ");
            String name1 = sc.next();
            System.out.print("Marks: ");
            int marks = sc.nextInt();
            System.out.print("Credits: ");
            int credits = sc.nextInt();
            subjects[i] = new Subject(name1,marks,credits);
        }
    }
    double getSGPA(){
        int sum=0;
        for(int i=0;i<3;i++){
            sgpa += (double) subjects[i].grade() *
                            subjects[i].credits;
            sum += subjects[i].credits;
        }
        sgpa = sgpa / sum;
        return sgpa;
    }
}

class SGPARun{
    public static void main(String[] args){
        System.out.println("Akshat Basra 1BM23CS028");
        Student s1 = new Student("as", "12");
        System.out.println("Student "+s1.name+" :");
        s1.getMarks();
```

```
        Student s2 = new Student("df", "13");
        System.out.println("Student "+s2.name+" :");
        s2.getMarks();
        Student s3 = new Student("gh", "14");
        System.out.println("Student "+s3.name+" :");
        s3.getMarks();
        System.out.println(s1.name+" SGPA: "+s1.getSgpa());
        System.out.println(s2.name+" SGPA: "+s2.getSgpa());
        System.out.println(s3.name+" SGPA: "+s3.getSgpa());
    }
}
```

Output:

```
Akshat Basra 1BM23CS020
Student as :
Enter Subject Details:
Subject 1.
Name: Maths
Marks : 68
Credits: 4
Subject 2
Name : DBMS
Marks :99
Credits : 2
Subject 3
Name: USP
Marks: 86
Credits : 1
Student df:
Enter Subject Details:
Subject 1
Name: Maths
```

Code:

```
import java.util.Scanner;

class Subject{
   String name;
   int marks, credits;
   Subject(String name, int marks, int credits){
      this.name = name;
      this.marks = marks;
      this.credits = credits;
   }
   int grade(){
      return marks / 10 + 1;
   }
}

class Student{
   String name, usn;
   Subject[] subjects = new Subject[8];
   double sgpa = 0;
   Scanner sc = new Scanner(System.in);
   Student(String name, String usn){
      this.name = name;
      this.usn = usn;
   }
   void getMarks(){
      System.out.println("Enter Subject Details:");
      for(int i = 0; i < 3; i++){
         System.out.println("Subject "+ (i+1));
         System.out.print("Name: ");
         String name1 = sc.next();
         System.out.print("Marks: ");
         int marks = sc.nextInt();
         System.out.print("Credits: ");
         int credits = sc.nextInt();
         subjects[i] = new Subject(name1, marks, credits);
      }
   }
   double getSgpa(){
      int sum = 0;
      for(int i = 0; i < 3; i++) {
         sgpa += (double) subjects[i].grade() * subjects[i].credits;
         sum += subjects[i].credits;
      }
      sgpa = sgpa / sum;
      return sgpa;
   }
```

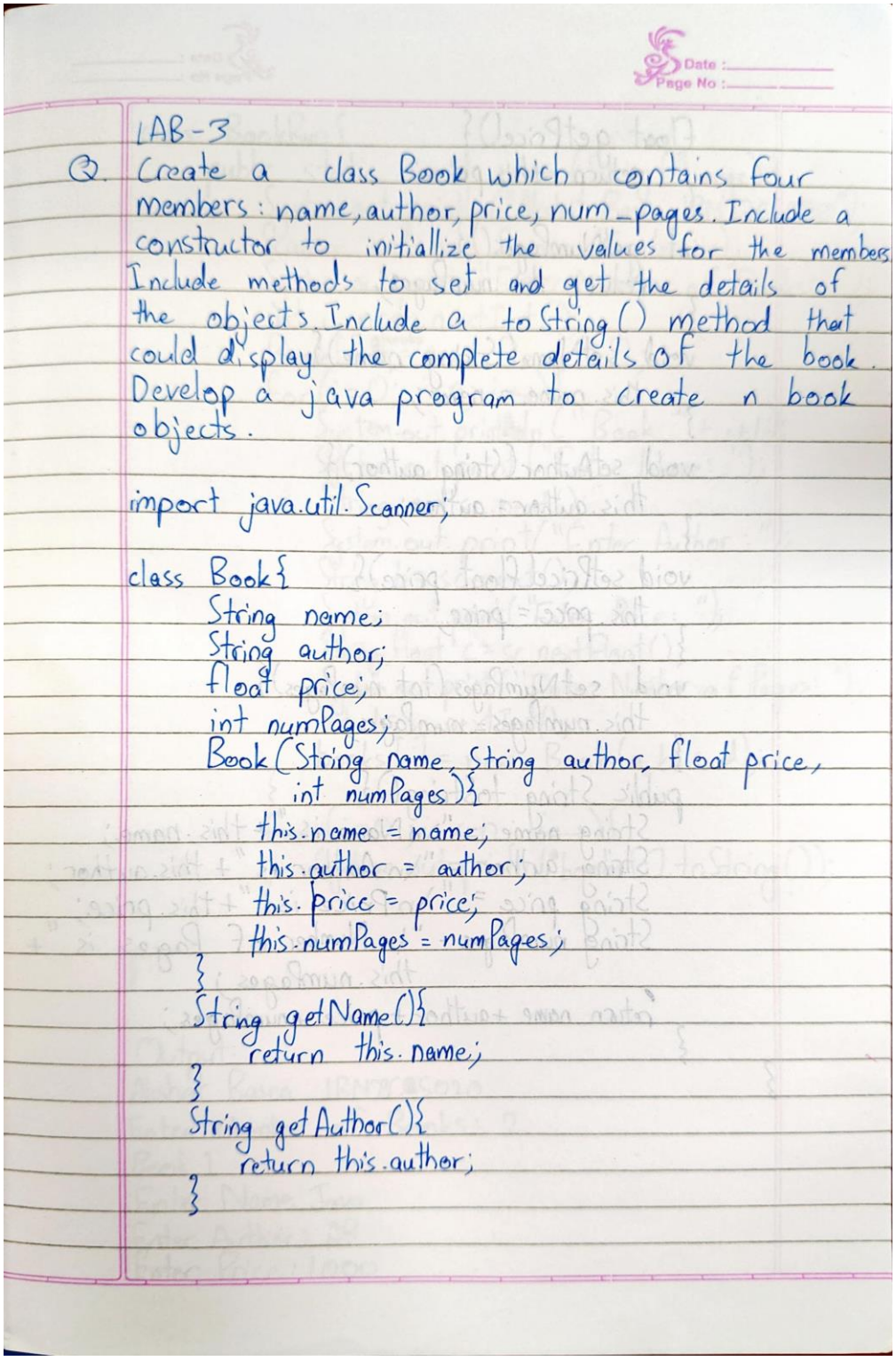
```
}

class SGPARun{
   public static void main(String[] args){
      System.out.println("Akshat Basra 1BM23CS020");
         Student s1 = new Student("as","12");
         System.out.println("Student " + s1.name + " :");
      s1.getMarks();
         Student s2 = new Student("df","13");
         System.out.println("Student " + s2.name + " :");
      s2.getMarks();
         Student s3 = new Student("gh","14");
         System.out.println("Student " + s3.name + " :");
      s3.getMarks();
      System.out.println(s1.name + " SGPA: "+ s1.getSgpa());
      System.out.println(s2.name + " SGPA: "+ s2.getSgpa());
      System.out.println(s3.name + " SGPA: "+ s3.getSgpa());
   }
}
```

```
PS D:\1BM23CS020> java SGPARun
Akshat Basra 1BM23CS020
Student as :
Enter Subject Details:
Subject 1
Name: Maths
Marks: 68
Credits: 4
Subject 2
Name: DBMS
Marks: 99
Credits: 2
Subject 3
Name: USP
Marks: 86
Credits: 1
Student df :
Enter Subject Details:
Subject 1
Name: Maths
Marks: 86
Credits: 4
Subject 2
Name: DBMS
Marks: 79
Credits: 2
Subject 3
Name: USP
Marks: 69
Credits: 1
Student gh :
Enter Subject Details:
Subject 1
Name: Maths
Marks: 70
Credits: 4
Subject 2
Name: DBMS
Marks: 78
Credits: 2
Subject 3
Name: USP
Marks: 79
Credits: 1
asSGPA: 8.1428571428571142
dfSGPA: 8.428571428571429
ghSGPA: 8.0
```

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.

Algorithm:

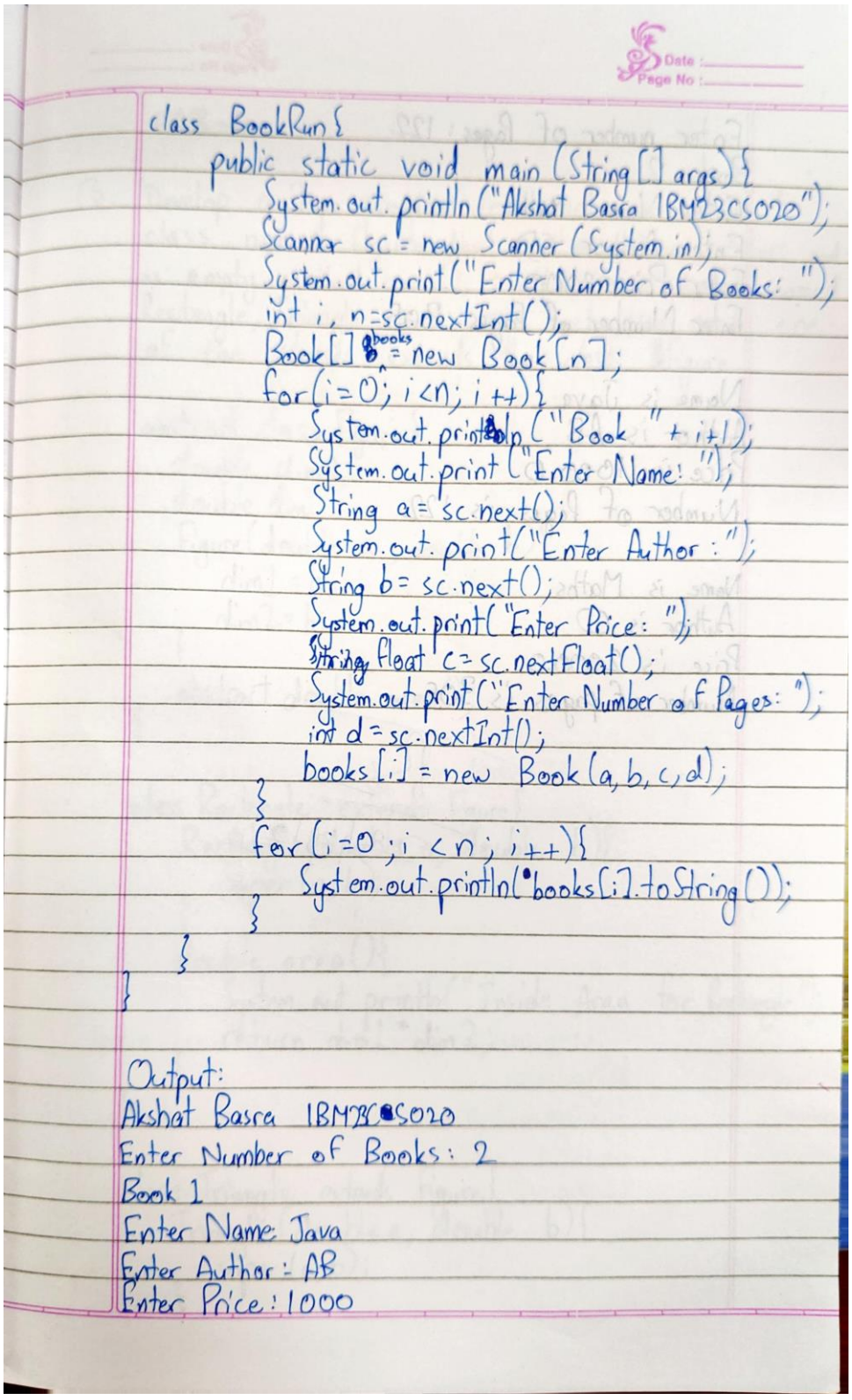


LAB-3

Q. Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to initiallize 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.

```
import java.util.Scanner;

class Book{
    String name;
    String author;
    float price;
    int numPages;
    Book(String name, String author, float price,
        int numPages){
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
    String getName(){
        return this.name;
    }
    String getAuthor(){
        return this.author;
    }
```

```
float getPrice(){
    return this.price;
}
int getNumPages(){
    return this.numPages;
}
void setName(String name){
    this.name = name;
}
void setAuthor(String author){
    this.author = author;
}
void setPrice(float price){
    this.price = price;
}
void setNumPages(int numPages){
    this.numPages = numPages;
}
public String toString(){
    String name = "\n Name is " + this.name;
    String author = "\n Author is " + this.author;
    String price = "\n Price is " + this.price;
    String numPages = "\n Number of Pages is " +
                      this.numPages;
    return name + author + price + numPages;
}
}
```



```
class BookRun{
    public static void main (String[] args){
        System.out.println("Akshat Basra 1BM23CS020");
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Number of Books: ");
        int i, n=sc.nextInt();
        Book[] books = new Book[n];
        for(i=0; i<n; i++){
            System.out.println("Book " + i+1);
            System.out.print("Enter Name: ");
            String a = sc.next();
            System.out.print("Enter Author :");
            String b = sc.next();
            System.out.print("Enter Price: ");
            float c = sc.nextFloat();
            System.out.print("Enter Number of Pages: ");
            int d = sc.nextInt();
            books[i] = new Book(a, b, c, d);
        }
        for(i=0 ; i <n ; i++){
            System.out.println(books[i].toString());
        }
    }
}
```

Output:

```
Akshat Basra 1BM23CS020
Enter Number of Books: 2
Book 1
Enter Name: Java
Enter Author: AB
Enter Price: 1000
```

Enter number of Pages: 122
Book 2
Enter Name: Maths
Enter Author: CD
Enter Price: 2000
Enter Number of Pages: 305

Name is Java
Author is AB
Price is 1000.0
Number of Pages is 122

Name is Maths
Author is CD
Price is 2000.0
Number of pages is 305

Code:

```
import java.util.Scanner;

class  Book{
	String name;
	String author;
	float price;
	int numPages;
	Book(String name, String author, float price, int numPages){
		this.name = name;
		this.author = author;
		this.price = price;
		this.numPages = numPages;
	}
	String getName(){
		return this.name;
	}
	String getAuthor(){
		return this.author;
	}
	float getPrice(){
		return this.price;
	}
	int getNumPages(){
		return this.numPages;
	}
	void setName(String name){
		this.name = name;
	}
	void setAuthor(String author){
		this.author = author;
	}
	void setPrice(float price){
		this.price = price;
	}
	void setNumPages(int numPages){
		this.numPages = numPages;
	}
	public String toString(){
		String name = "\nName is " + this.name;
		String author = "\nAuthor is " + this.author;
		String price = "\nPrice is " + this.price;
		String numPages = "\nNumber of pages is " + this.numPages;
		return name + author + price + numPages;
	}
}
```

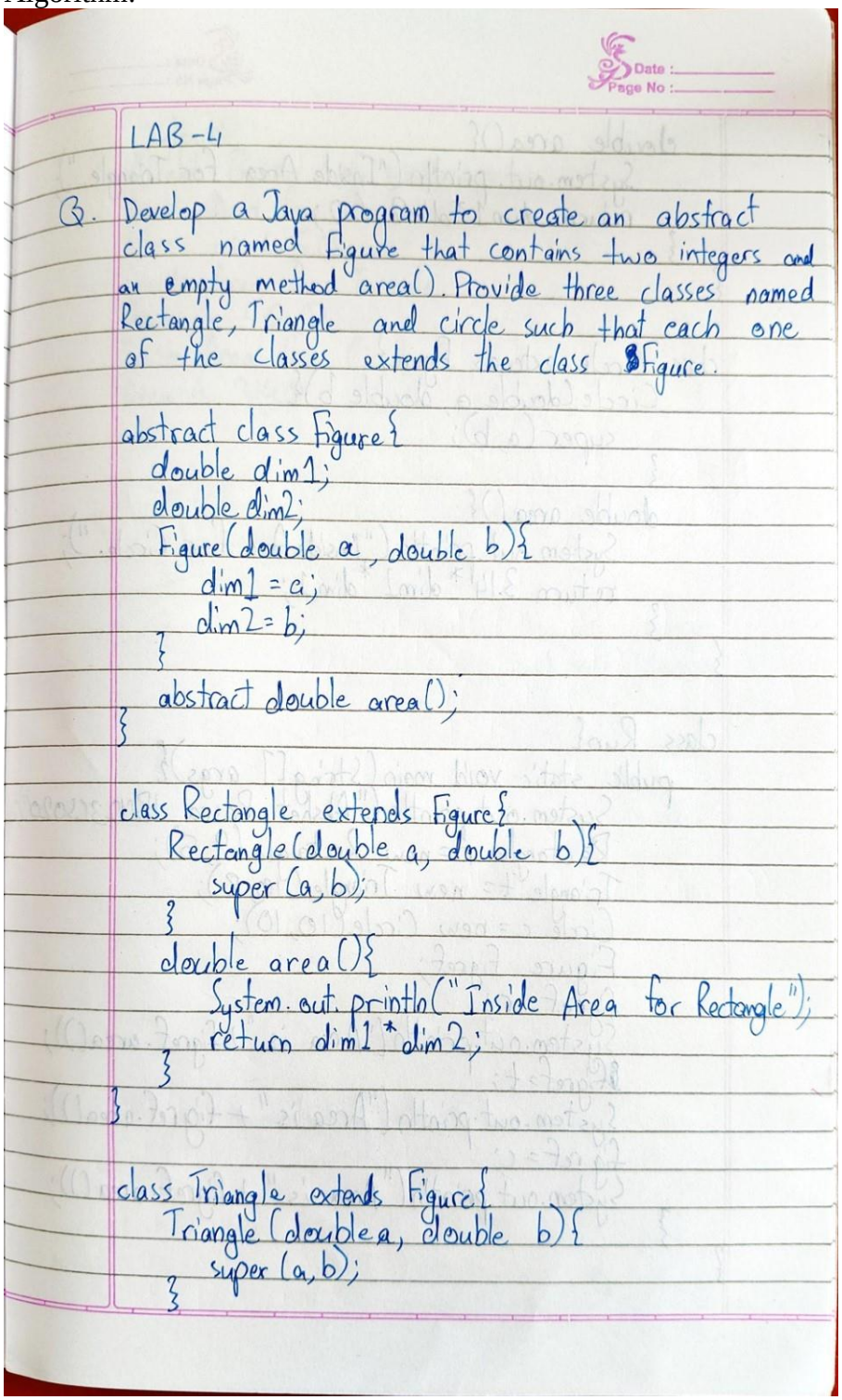
```
class BookRun{
	public static void main(String[] args){
		System.out.println("Akshat Basra 1BM23CS020");
		Scanner sc = new Scanner(System.in);
		System.out.print("Enter Number of Books: ");
		int i, n = sc.nextInt();
		Book[] books = new Book[n];
		for(i = 0; i < n; i++){
			System.out.println("Book " + (i+1));
			System.out.print("Enter Name: ");
			String a = sc.next();
			System.out.print("Enter Author: ");
			String b = sc.next();
			System.out.print("Enter Price: ");
			float c = sc.nextFloat();
			System.out.print("Enter Number of Pages: ");
			int d = sc.nextInt();
			books[i] = new Book(a, b, c, d);
		}
		for(i = 0; i < n; i++){
			System.out.println(books[i].toString());
		}
	}
}
```

```
D:\1BM23CS020>java BookRun
Akshat Basra 1BM23CS020
Enter Number of Books: 2
Book 1
Enter Name: Java
Enter Author: AB
Enter Price: 1000
Enter Number of Pages: 122
Book 2
Enter Name: Maths
Enter Author: CD
Enter Price: 2000
Enter Number of Pages: 305

Name is Java
Author is AB
Price is 1000.0
Number of pages is 122

Name is Maths
Author is CD
Price is 2000.0
Number of pages is 305
```

Program 4

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

Algorithm:



LAB-4

Q. Develop a Java program to create an abstract class named Figure that contains two integers and an empty method area(). Provide three classes named Rectangle, Triangle and circle such that each one of the classes extends the class Figure.

```
abstract class Figure{
  double dim1;
  double dim2;
  Figure(double a, double b){
     dim1 = a;
     dim2 = b;
  }
  abstract double area();
}

class Rectangle extends Figure{
   Rectangle(double a, double b){
      super(a,b);
   }
   double area(){
      System.out.println("Inside Area for Rectangle");
      return dim1 * dim2;
   }
}

class Triangle extends Figure{
   Triangle(double a, double b){
      super(a,b);
   }
```

```
    double area(){
        System.out.println("Inside Area for Triangle.");
        return dim1*dim2/2;
    }
}

class Circle extends Figure{
    Circle(double a, double b){
        super(a,b);
    }
    double area(){
        System.out.println("Inside Area for Circle.");
        return 3.14*dim1*dim1;
    }
}

class Run{
    public static void main(String[] args){
        System.out.println("Akshat Basra 1BM23CS020");
        Rectangle r=new Rectangle(9,5);
        Triangle t=new Triangle(10,8);
        Circle c=new Circle(10,10);
        Figure figref;
        figref=r;
        System.out.println("Area is "+figref.area());
        figref=t;
        System.out.println("Area is "+figref.area());
        figref=c;
        System.out.println("Area is "+figref.area());
    }
}
```

Output:

Akshat Basra 1BM22CS020

Inside Area for Rectangle.

Area is 45.0

Inside Area for Triangle.

Area is 40.0

Inside Area for Circle.

Area is 314.0

23/10/24

Code:

```
abstract class Figure {
	double dim1; double dim2;
	Figure(double a, double b){
		dim1 = a; dim2 = b;
	}
	abstract double area();
 }

class Rectangle extends Figure {
	Rectangle(double a, double b){
		super(a, b);
	}
	double area(){
		System.out.println("Inside Area for Rectangle.");
		return dim1 * dim2;
	}
}

class Triangle extends Figure {
	Triangle(double a, double b){
		super(a, b);
	}
	double area(){
		System.out.println("Inside Area for Triangle.");
		return dim1 * dim2 / 2;
	}
}

class Circle extends Figure{
	Circle(double a, double b){
		super(a, b);
	}
	double area(){
		System.out.println("Inside Area for Circle.");
		return 3.14*dim1*dim1;
	}
}

class AbstractAreas{
	public static void main(String args[]){
		System.out.println("Akshat Basra 1BM23CS020");
		// Figure f = new Figure(10, 10); // illegal now
		Rectangle r = new Rectangle(9, 5);
		Triangle t = new Triangle(10, 8);
		Circle c = new Circle(10, 10);
		Figure figref;
```

```
                figref = r;
                System.out.println("Area is " + figref.area());
                figref = t;
                System.out.println("Area is " + figref.area());
                figref = c;
                System.out.println("Area is " + figref.area());
        }
}
```

```
D:\1BM23CS020>java AbstractAreas
Akshat Basra 1BM23CS020
Inside Area for Rectangle.
Area is 45.0
Inside Area for Triangle.
Area is 40.0
Inside Area for Circle.
Area is 314.0
```

Program 5

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

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

Algorithm:

LAB-5

Q. Develop a java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and other current account. Include the necessary methods in order to achieve the following tasks.

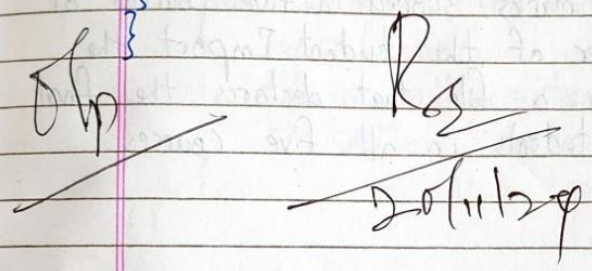
a) Accept deposit and update balance.
b) display the balance.
c) compute and deposit interest.
d) Permit withdrawal and update balance.
e) Check for minimum balance, impose penalty if necessary.

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

class Bank{
  int accountNo;
  double balance;
  Bank(int accountNo){
    this.accountNo = accountNo;
    this.balance = 0;
  }
  void deposit(double depAmount){
    this.balance += depAmount;
  }
  void withdraw(double withAmount){
    this.balance -= withAmount;
  }
  double interest(double rate, int time){
    System.out.println("Interest is not applicable
      in current account");
```

```
        return 0.0;
    }
}
class SavingsAccount extends Bank{
    SavingsAccount(int accountNo){
        super(accountNo);
    }
    double interest(double rate, int time){
        double interest = (balance * Math.pow((1+(rate
            /100)), time)) - balance;
        balance += interest;
        return interest;
    }
}

class CurrentAccount extends Bank {
    static double withdrawLimit = 1000;
    CurrentAccount(int accountNo){
        super(accountNo);
    }
    public void withdraw(double withAmount){
        super.balance -= withAmount;
        if(balance < withdrawLimit){
            System.out.println("Withdraw limit Reached -
                Deducting Service Charge");
            balance -= 100;
        }
    }
}
```

```
class Run {
  public static void main(String [] args){
    System.out.println("Akshat Basra 1BM23CS020");
    double amount;
    Scanner sc = new Scanner(System.in);
    System.out.print(" 1. Open savings Account\n
                      2. Open Current Account \n\n
                      Enter choice: ");
    int choice = sc.nextInt();
    Bank acc;
    if(choice == 1){
      acc = new SavingsAccount(101);
    }
    else{
      acc = new CurrentAccount(201);
    }
    System.out.println(" 1. Deposit\n
                         2. Withdraw \n
                         3. Show Balance \n
                         4. Compute Interest\n
                         5. Exit\n");
    while (true){
      System.out.print("Enter Choice: ");
      choice = sc.nextInt();
      switch(choice){
        case 1: System.out.print("Enter deposit amount:");
                amount = sc.nextDouble();
                acc.deposit(amount);
                break;
        case 2: System.out.print("Enter withdraw amount:");
                amount = sc.nextDouble();
                acc.withdraw(amount);
```

```
                break;
        case 3: System.out.println("The balance is" + acc.
                                                balance);
                break;
        case 4: System.out.println("The interest is" +
                                        acc.interest(5,1));
                break;
        default: System.exit(0);
    }
```



Code:

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

class Bank{
	int accountNo;
	double balance;
	Bank(int accountNo){
		this.accountNo = accountNo;
		this.balance = 0;
	}
	void deposit(double depAmount){
		this.balance += depAmount;
	}
	void withdraw(double withAmount){
		this.balance -= withAmount;
	}
	double interest(double rate, int time){
		System.out.println("Interest is not applicable in current account");
		return 0.0;
	}
}


class SavingsAccount extends Bank{
	SavingsAccount(int accountNo){
		super(accountNo);
	}
	double interest(double rate, int time){
		double interest = (balance * Math.pow((1 + (rate / 100)), time)) - balance;
		balance += interest;
		return interest;
	}
}

class CurrentAccount extends Bank{
	static double withdrawLimit = 1000;
	CurrentAccount(int accountNo){
		super(accountNo);
	}
	public void withdraw(double withAmount) {
		super.balance -= withAmount;
		if (balance < withdrawLimit) {
			System.out.println("Withdraw Limit Reached - Deducting Service Charge");
			balance -= 100;
		}
	}
```

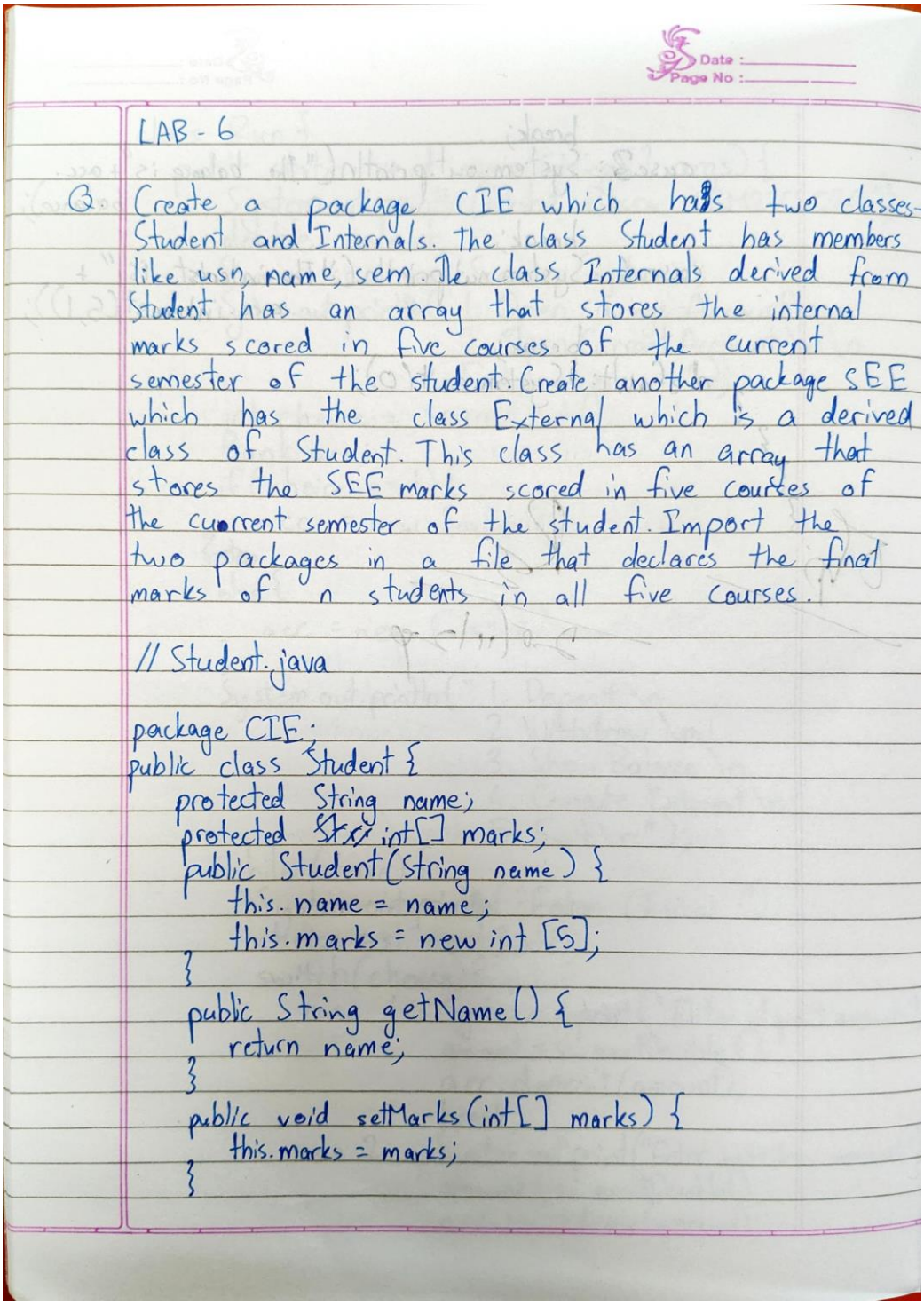
```
}


class Run {
	public static void main(String[] args) {
		System.out.println("Akshat Basra 1BM23CS020");
		double amount;
		Scanner sc = new Scanner(System.in);
		System.out.print("1. Open Savings Account\n2. Open Current Account\n\nEnter
Choice:");
		int choice = sc.nextInt();
		Bank acc;
		if(choice == 1) {
			acc = new SavingsAccount(101);
		}
		else {
			acc = new CurrentAccount(201);
		}
		System.out.println("1. Deposit\n2. Withdraw\n3. Show Balance\n4. Compute
Interest\n5. Exit\n");
		while(true){
			System.out.print("Enter Choice: ");
			choice = sc.nextInt();
			switch(choice){
				case 1: System.out.print("Enter deposit amount: ");
					amount = sc.nextDouble();
					acc.deposit(amount);
					break;
				case 2: System.out.print("Enter withdraw amount: ");
					amount = sc.nextDouble();
					acc.withdraw(amount);
					break;
				case 3: System.out.println("The balance is " + acc.balance);
					break;
				case 4: System.out.println("The interest is " + acc.interest(5, 1));
					break;
				default:System.exit(0);
			}
		}
	}
}
```

```
E:\1BM23CS020>java Run
Akshat Basra 1BM23CS020
1. Open Savings Account
2. Open Current Account

Enter Choice:1
1. Deposit
2. Withdraw
3. Show Balance
4. Compute Interest
5. Exit

Enter Choice: 1
Enter deposit amount: 10000
Enter Choice: 2
Enter withdraw amount: 5000
Enter Choice: 4
The interest is 250.0
Enter Choice: 3
The balance is 5250.0
Enter Choice: 5

E:\1BM23CS020>java Run
Akshat Basra 1BM23CS020
1. Open Savings Account
2. Open Current Account

Enter Choice:2
1. Deposit
2. Withdraw
3. Show Balance
4. Compute Interest
5. Exit

Enter Choice: 1
Enter deposit amount: 1000
Enter Choice: 2
Enter withdraw amount: 500
Withdraw Limit Reached - Deducting Service Charge
Enter Choice: 4
Interest is not applicable in current account
The interest is 0.0
Enter Choice: 3
The balance is 400.0
Enter Choice: 5
```

Program 6

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

Algorithm:



LAB - 6

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

// Student.java

```
package CIE;
public class Student {
    protected String name;
    protected int[] marks;
    public Student(String name) {
        this.name = name;
        this.marks = new int [5];
    }
    public String getName() {
        return name;
    }
    public void setMarks(int[] marks) {
        this.marks = marks;
    }
```

```
    public int[] getMarks(){
        return marks;
    }
}

// Internal.java

package CIE;
public class Internal extends Student {
    int[] internalMarks;
    public Internal(String name, int[] internalMarks){
        super(name);
        this.internalMarks = internalMarks;
        this.setMarks(internalMarks);
    }
}

// External.java
package SEE;
import CIE.Student;
public class External extends Student {
    int[] externalMarks;
    public External(String name, int[] externalMarks){
        super(name);
        this.externalMarks = externalMarks;
        this.setMarks(externalMarks);
    }
}
```

```
//Main.java
import CIE.Internal;
import SEE.External;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        System.out.println("Akshat Basra 1BM23CS020");
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = sc.nextInt();
        Internal[] internalStudents = new Internal[n];
        External[] externalStudents = new External[n];
        for(int i=0; i<n; i++){
            System.out.print("Enter the name of student" + (i+1)
                                                + ": ");
            String name = sc.nextLine();
            System.out.println("Enter internal Marks (5 courses) for"
                                        + name + ": ");
            int[] internalMarks = new int[5];
            for(int j=0; j<5; j++) {
                System.out.print("Enter Marks: ");
                internalMarks[j] = sc.nextInt();
            }
            int
            System.out.println("Enter External Marks (5 courses) for"
                                        + name + ": ");
            int externalMarks[] = new int[5];
            for(int j=0; j<5; j++){
                System.out.print("Enter Marks: ");
                externalMarks[j] = sc.nextInt();
            }
```

```
            internalStudents[i] = new Internal(name, internalMarks);
            externalStudents[i] = new External(name, externalMarks);
        }
        System.out.println("\nFinal Marks for all students:");
        for(i=0; i<n; i++){
            int[] internalMarks = internalStudents[i].getMarks();
            int[] externalMarks = externalStudents[i].getMarks();
            System.out.println("Student: " + internalStudents[i].getName());
            System.out.print("Internal Marks: ");
            for(int mark : internalMarks){
                System.out.print(mark + " ");
            }
            System.out.print("External Marks: ");
            for(int mark: externalMarks){
                System.out.print(mark + " ");
            }
            System.out.print("Final Marks: ");
            for(int j=0; j<5; j++) {
                int finalMark = internalMarks[i] + externalMarks[i];
                System.out.print(finalMark+" ");
            }
            sc.close();
        }
        sc.close();
    }
}
```

Output:

```
Enter the number of students: 1
Enter the name of student 1: A
Enter internal marks (5 courses) for A:
40
48
46
25
44
Enter external marks(5 courses) for A:
42
46
44
49
44

Final Marks for all students:

Student: A
Internal Marks: 40 48 46 25 44
External Marks: 42 46 44 49 44
Final Marks: 82 94 90 74 88
```

20/11/24

Code:

```
//Student.java
package CIE;
public class Student {
    protected String name;
    protected int[] marks;
    public Student(String name) {
        this.name = name;
        this.marks = new int[5];
    }
    public String getName() {
        return name;
    }
    public void setMarks(int[] marks) {
        this.marks = marks;
    }
    public int[] getMarks() {
        return marks;
    }
}

//Internal.java
package CIE;
public class Internal extends Student {
    int[] internalMarks;
    public Internal(String name, int[] internalMarks) {
        super(name);
        this.internalMarks = internalMarks;
        this.setMarks(internalMarks);
    }
}

//External.java
package SEE;
import CIE.Student;
public class External extends Student {
    int[] externalMarks;
    public External(String name, int[] externalMarks) {
        super(name);
        this.externalMarks = externalMarks;
        this.setMarks(externalMarks);
    }
}

//Main.java
```

```
import CIE.Internal;
import SEE.External;
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        System.out.println("Akshat Basra 1BM23CS020");
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = sc.nextInt();
        sc.nextLine();
        Internal[] internalStudents = new Internal[n];
        External[] externalStudents = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.print("Enter the Name of student " + (i + 1) + ": ");
            String name = sc.nextLine();
            System.out.println("Enter Internal Marks (5 courses) for " + name + ": ");
            int[] internalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                internalMarks[j] = sc.nextInt();
            }
            sc.nextLine();
            System.out.println("Enter External Marks (5 courses) for " + name + ": ");
            int[] externalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                System.out.print("Enter Marks: ");
                externalMarks[j] = sc.nextInt();
            }
            sc.nextLine();
            internalStudents[i] = new Internal(name, internalMarks);
            externalStudents[i] = new External(name, externalMarks);
        }
        System.out.println("\nFinal Marks for all students:");
        for (int i = 0; i < n; i++)
        {
            int[] internalMarks = internalStudents[i].getMarks();
            int[] externalMarks = externalStudents[i].getMarks();
            System.out.println("\nStudent: " + internalStudents[i].getName());
            System.out.print("Internal Marks: ");
            for (int mark : internalMarks) {
                System.out.print(mark + " ");
            }
            System.out.print("\nExternal Marks: ");
            for (int mark : externalMarks) {
                System.out.print(mark + " ");
            }
            System.out.print("\nFinal Marks: ");
            for (int j = 0; j < 5; j++) {
```

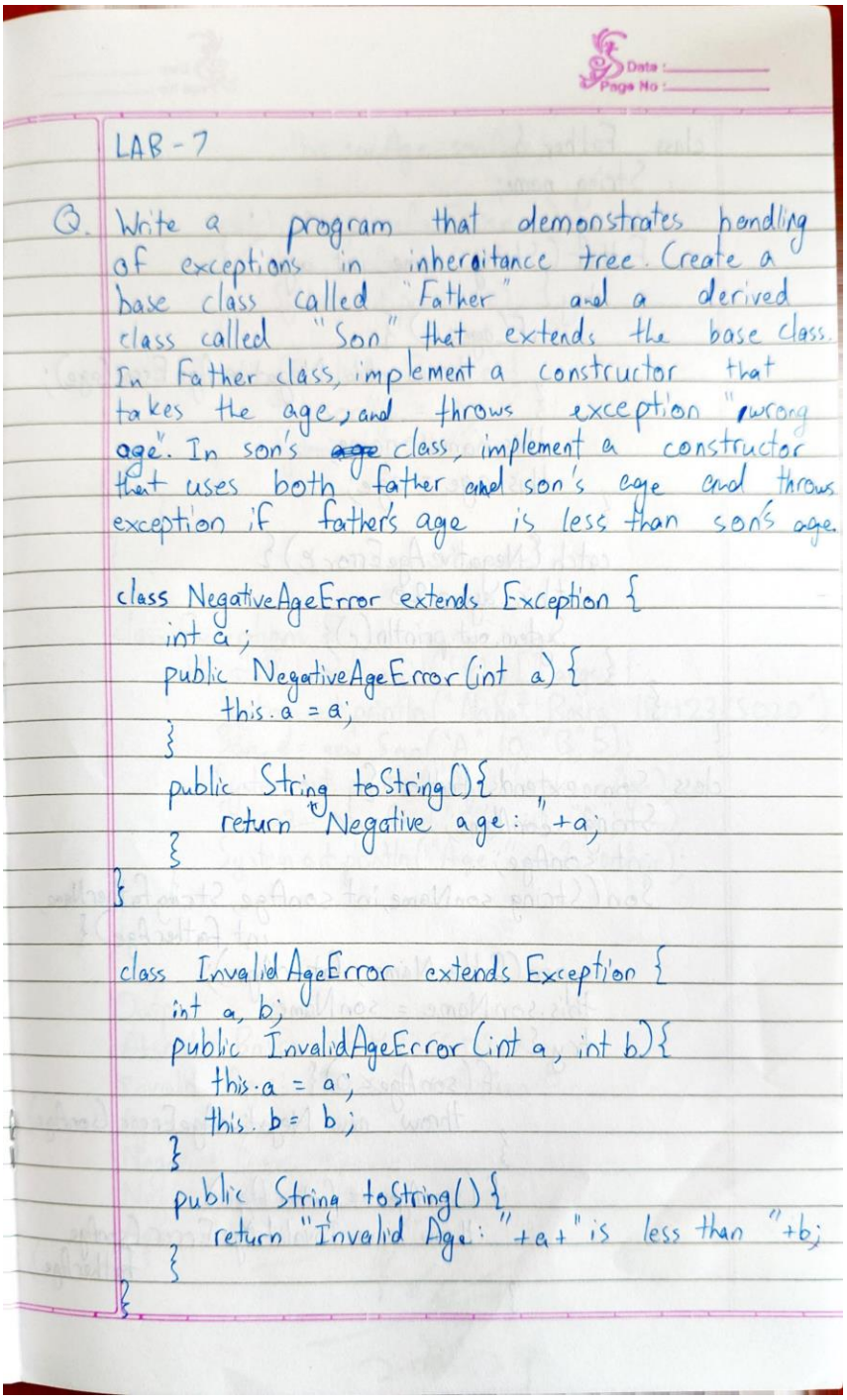
```
            int finalMark = internalMarks[j] + externalMarks[j];
            System.out.print(finalMark + " ");
        }
        System.out.println();
    }
    sc.close();
  }
}
```

```
E:\1BM23CS020\New folder>java Main
Enter the number of students: 1
Enter the name of student 1: A
Enter internal marks (5 courses) for A:
40
48
46
25
44
Enter external marks (5 courses) for A:
42
46
44
49
44

Final Marks for all students:

Student: A
Internal Marks: 40 48 46 25 44
External Marks: 42 46 44 49 44
Final Marks: 82 94 90 74 88
```

Program 7

Write a program that demonstrates handling of exceptions using inheritance tree. Create a base class called “Father” and a derived class called “Son” that extends the base class. In Father class, implement a constructor that takes the age and throws exception “wrong age”. In Son class, implement a constructor that takes both father and son’s age and throws exception if father’s age is less than son’s age.

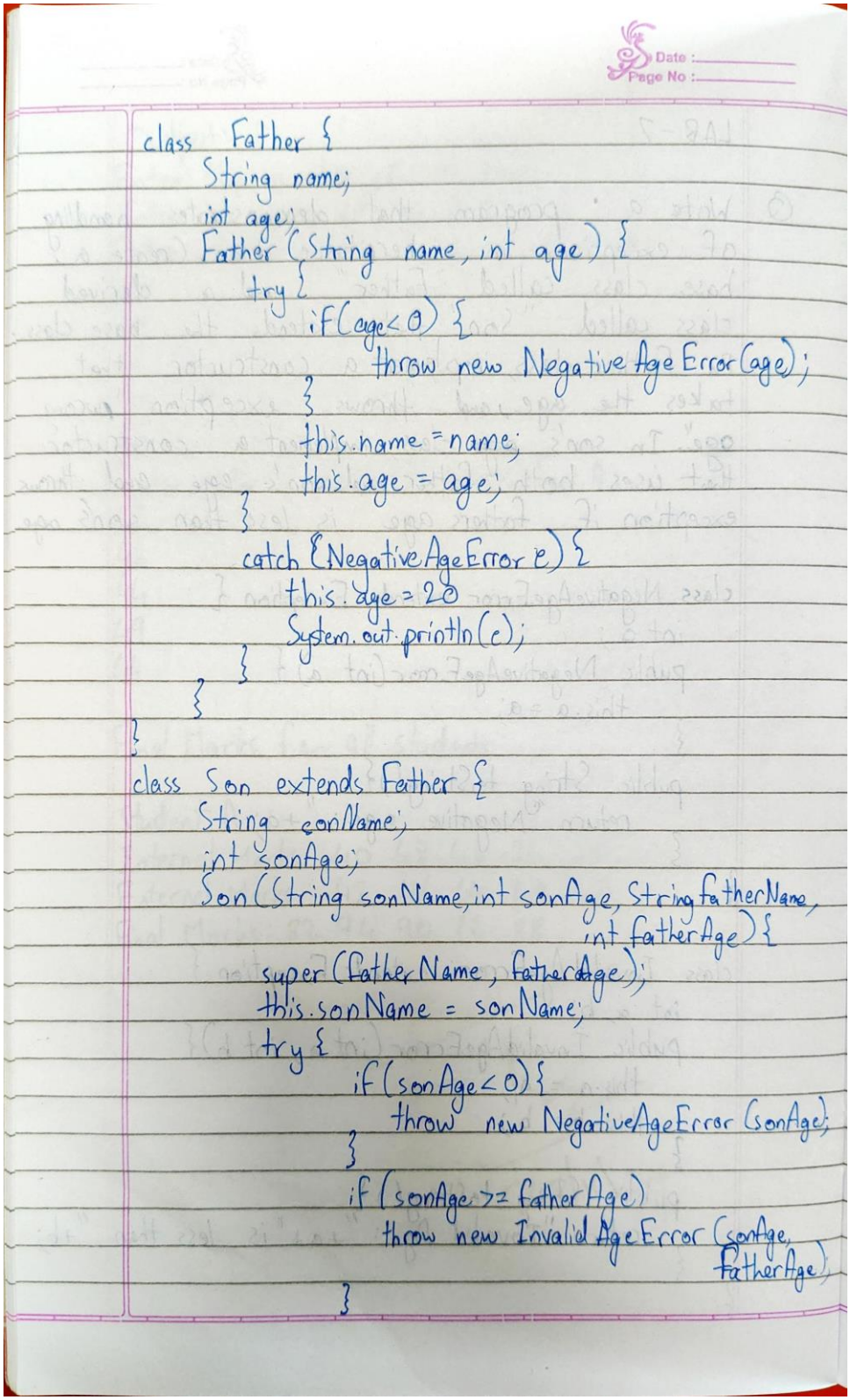
Algorithm:



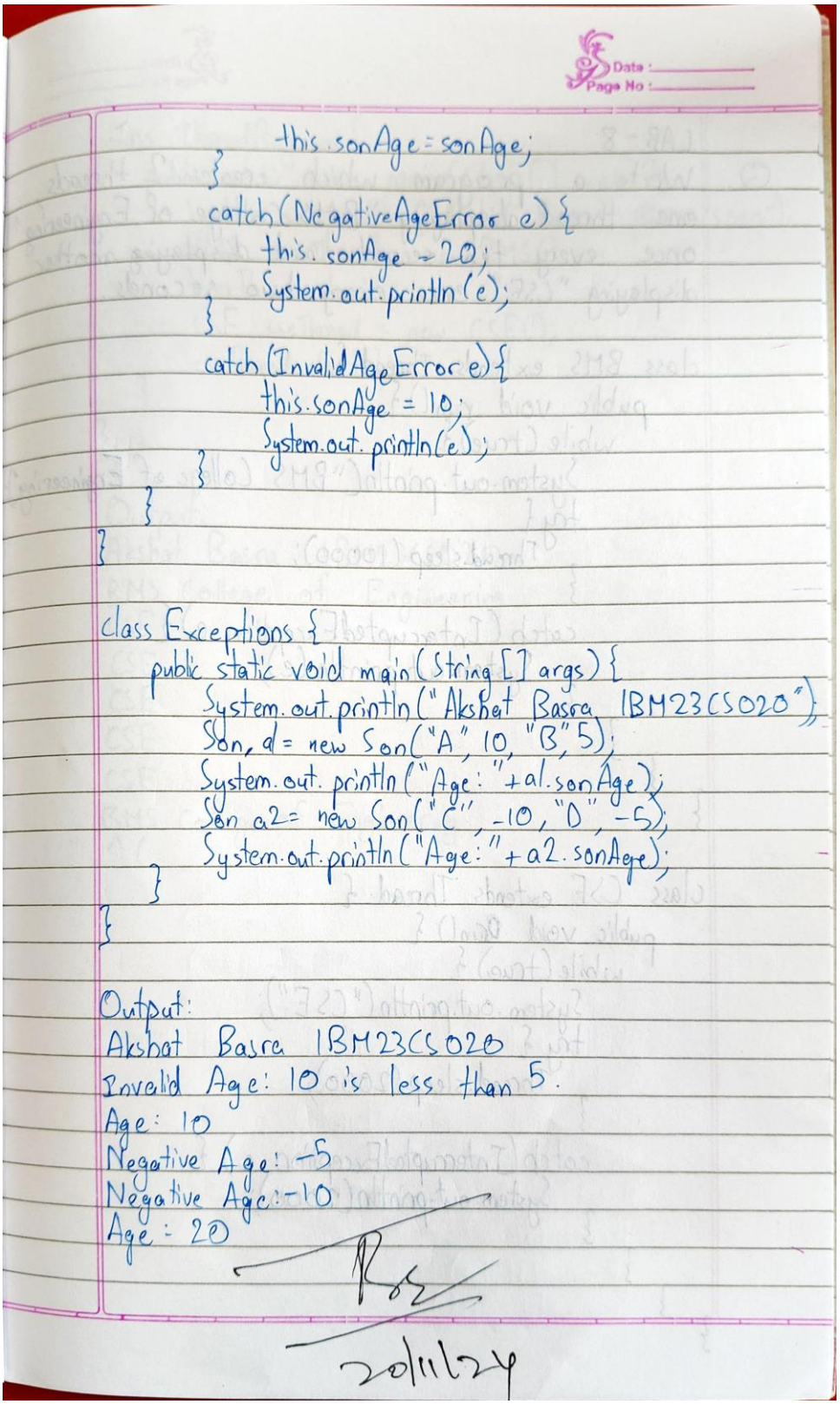
LAB - 7

Q. Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and a derived class called "Son" that extends the base class. In Father class, implement a constructor that takes the age, and throws exception "wrong age". In son's ~~age~~ class, implement a constructor that uses both father and son's age and throws exception if father's age is less than son's age.

```
class NegativeAgeError extends Exception {
    int a;
    public NegativeAgeError(int a) {
        this.a = a;
    }
    public String toString(){
        return "Negative age: "+a;
    }
}

class InvalidAgeError extends Exception {
    int a, b;
    public InvalidAgeError(int a, int b){
        this.a = a;
        this.b = b;
    }
    public String toString() {
        return "Invalid Age: "+a+" is less than "+b;
    }
}
```



```
class Father {
    String name;
    int age;
    Father (String name, int age) {
        try {
            if(age<0) {
                throw new NegativeAgeError(age);
            }
            this.name = name;
            this.age = age;
        }
        catch (NegativeAgeError 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 NegativeAgeError(sonAge);
            }
            if(sonAge >= fatherAge)
                throw new InvalidAgeError(sonAge,
                                          fatherAge);
            }
```



```
            this.sonAge = sonAge;
        }
        catch(NegativeAgeError e) {
            this.sonAge = 20;
            System.out.println(e);
        }
        catch (InvalidAgeError e){
            this.sonAge = 10;
            System.out.println(e);
        }
    }
}

class Exceptions {
    public static void main(String[] args){
        System.out.println("Akshat Basra 1BM23CS020");
        Son a = new Son("A", 10, "B", 5);
        System.out.println("Age: "+a.sonAge);
        Son a2 = new Son("C", -10, "D", -5);
        System.out.println("Age: "+a2.sonAge);
    }
}
```

Output:

```
Akshat Basra 1BM23CS020
Invalid Age: 10 is less than 5.
Age: 10
Negative Age: -5
Negative Age: -10
Age: 20
```

20/11/24

Code:

```
class NegativeAgeError extends Exception {
	int a;
	public NegativeAgeError(int a) {
		this.a = a;
	}
	public String toString() {
		return "Negative Age: " + a;
	}
}

class InvalidAgeError extends Exception {
	int a, b;
	public InvalidAgeError(int a, int b) {
		this.a = a;
		this.b = b;
	}
	public String toString() {
		return "Invalid Age: " + a + " is less than " + b;
	}
}

class Father {
	String name;
	int age;
	Father(String name, int age) {
		try {
			if(age < 0) {
				throw new NegativeAgeError(age);
			}
			this.name = name;
			this.age = age;
		}
		catch(NegativeAgeError 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;
```

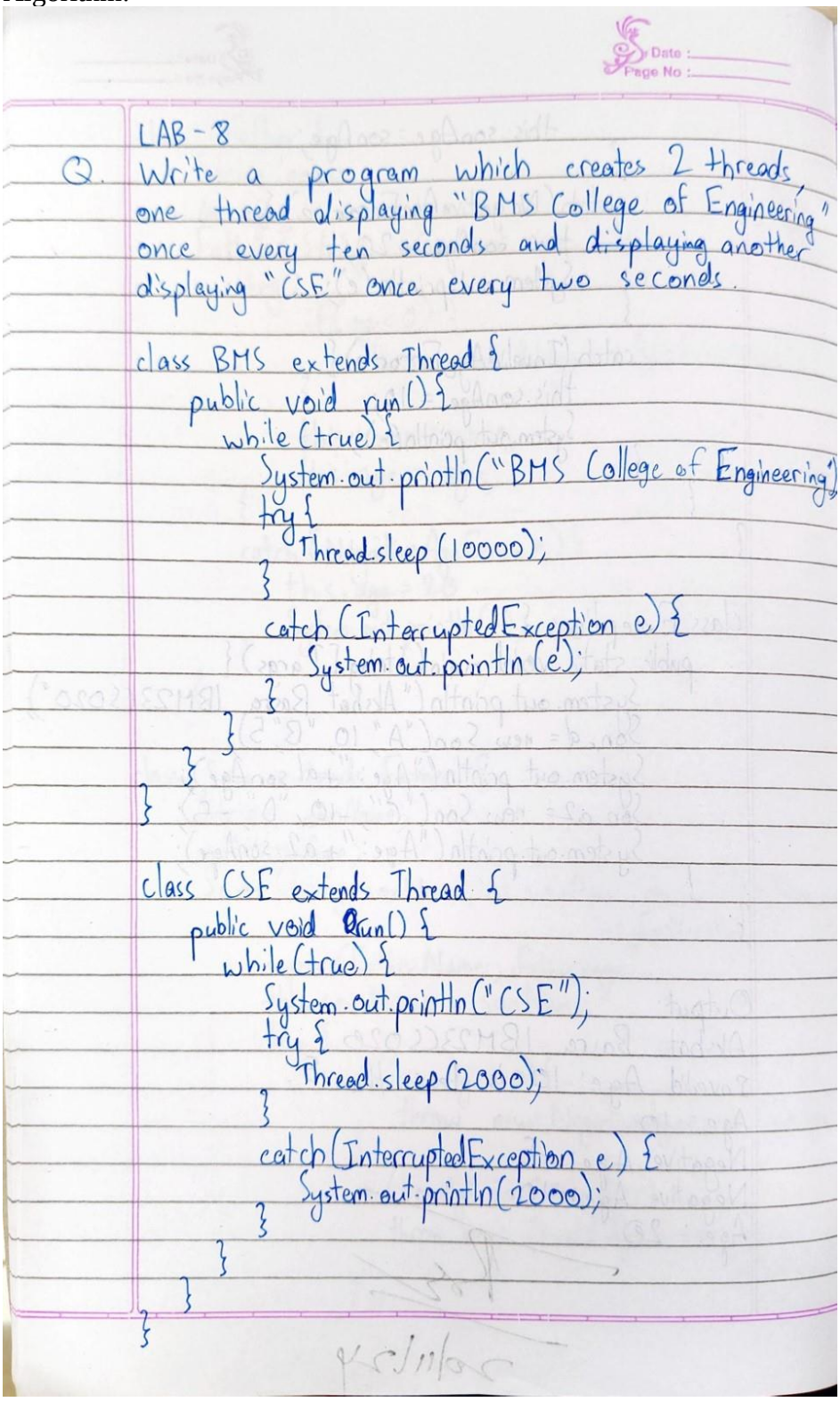
```
                try {
                        if(sonAge < 0) {
                                throw new NegativeAgeError(sonAge);
                        }
                        if(sonAge >= fatherAge){
                                throw new InvalidAgeError(sonAge, fatherAge);
                        }
                        this.sonAge = sonAge;
                }
                catch(NegativeAgeError e) {
                        this.sonAge = 20;
                        System.out.println(e);
                }
                catch(InvalidAgeError e) {
                        this.sonAge = 10;
                        System.out.println(e);
                }
        }
}

class Exceptions {
        public static void main(String[] args){
                System.out.println("Akshat Basra 1BM23CS020");
                Son a1 = new Son("A", 10, "B", 5);
                System.out.println("Age: " + a1.sonAge);
                Son a2 = new Son("C", -10, "D", -5);
                System.out.println("Age: " + a2.sonAge);
        }
}
```

```
E:\1BM23CS020>java Exceptions
Akshat Basra 1BM23CS020
Invalid Age: 10 is less than 5
Age: 10
Negative Age: -5
Negative Age: -10
Age: 20
```

Program 8

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

Algorithm:

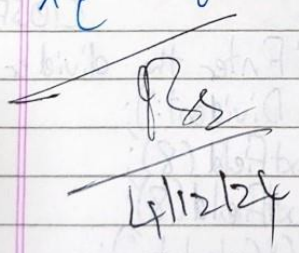


```
LAB - 8
Q. Write a program which creates 2 threads,
one thread displaying "BMS College of Engineering"
once every ten seconds and displaying another
displaying "CSE" once every two seconds.

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

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

```
class ThreadProgram {
    public static void main(String[] args) {
        System.out.println("Akshat Basra 1BM23CS020");
        BMS bmsThread = new BMS();
        bmsThread.start();
        CSE cseThread = new CSE();
        cseThread.start();
    }
}
```

Output:

```
Akshat Basra 1BM23CS020
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
^C
```



4/12/24

Code:

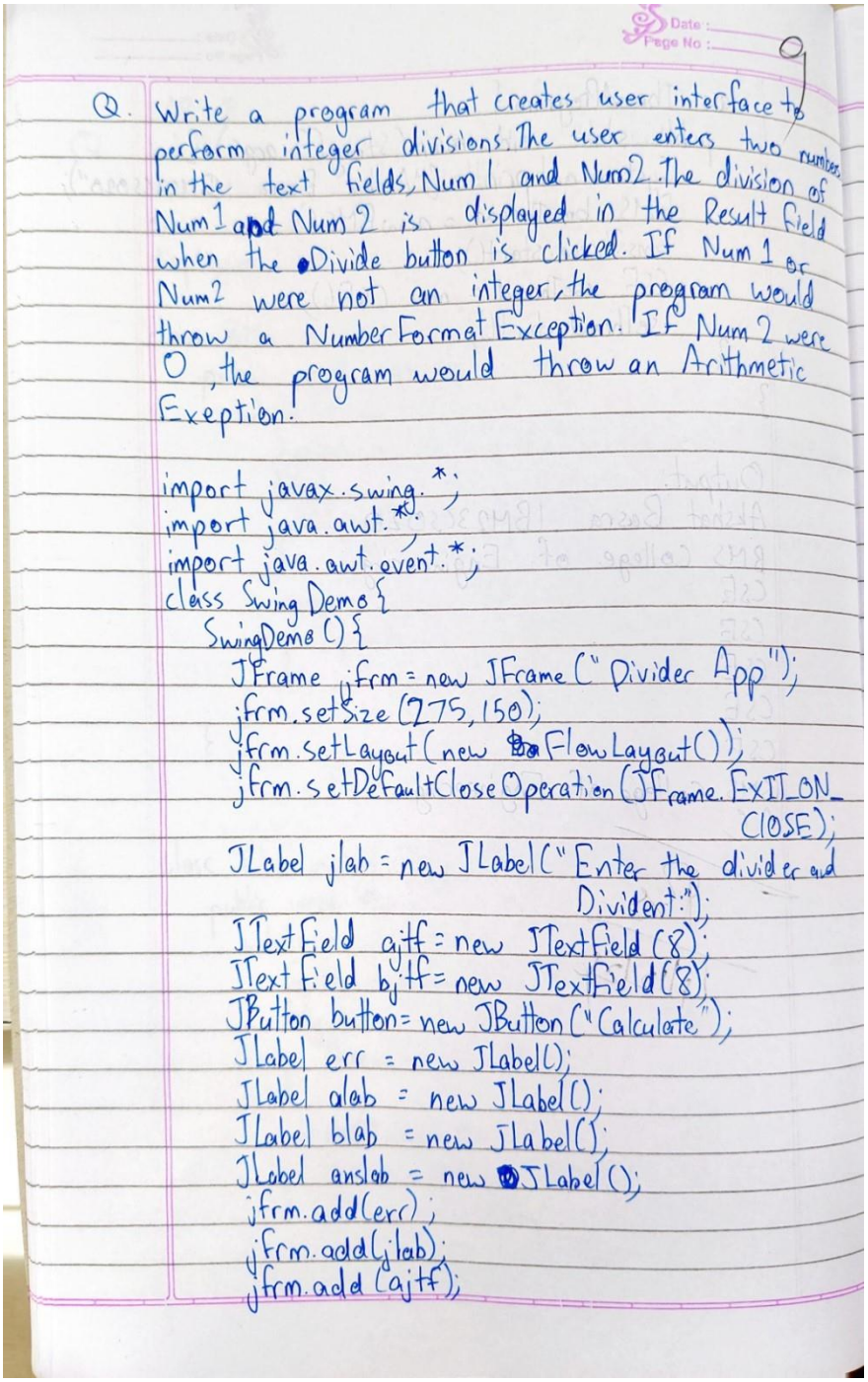
```
class BMS extends Thread {
   public void run() {
      while (true) {
         System.out.println("BMS College of Engineering");
         try {
            Thread.sleep(10000);
         } catch (InterruptedException e) {
            System.out.println(e);
         }
      }
   }
}

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

class ThreadProgram {
   public static void main(String[] args) {
         System.out.println("Akshat Basra 1BM23CS020");
         BMS bmsThread = new BMS();
         bmsThread.start();
         CSE cseThread = new CSE();
         cseThread.start();
   }
}
```

```
D:\1BM23CS020>java ThreadProgram
Akshat Basra 1BM23CS020
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
^C
D:\1BM23CS020>
```

Program 9

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

Algorithm:



Q. Write a program that creates 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 0, the program would throw an Arithmetic Exception.

```
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
                                Divident:");
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);
      JButton button = new JButton("Calculate");
      JLabel err = new JLabel();
      JLabel alab = new JLabel();
      JLabel blab = new JLabel();
      JLabel anslab = new JLabel();
      jfrm.add(err);
      jfrm.add(jlab);
      jfrm.add(ajtf);
```

```
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab)
jfrm.add(blab);
jfrm.add(anslab);
ActionListener l = new ActionListener() {
  public void actionPerformed(ActionEvent evt)
    System.out.println("Action Event from a text
                                        field");
  }
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent evt){
    try{
      int a = Integer.parseInt(ajtf.getText());
      int b = Integer.parseInt(bjtf.getText());
      int ans = a / b;
      alab.setText("\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("");
```

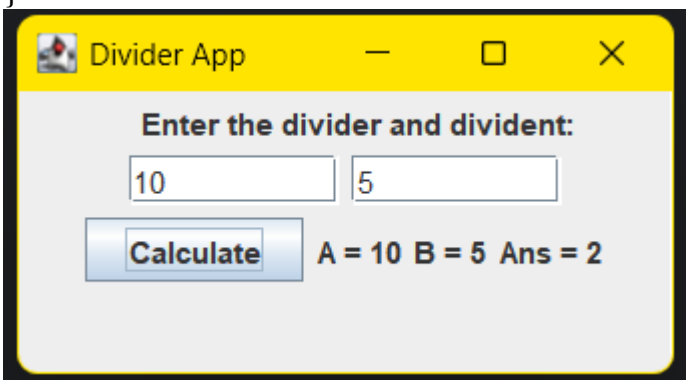
```
            ansLab.setText(" ");
            errLab.setText("B should be Non zero");
        }
    }
});
frm.setVisible(true);
}
public static void main(String[]args){
    SwingUtilities.invokeLater(new Runnable() {
        public void run(){
            new SwingDemo();
        }
    });
}
}
```

4/12/24

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 divident:");
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);
      JButton button = new JButton("Calculate");
      JLabel err = new JLabel();
      JLabel alab = new JLabel();
      JLabel blab = new JLabel();
      JLabel anslab = new JLabel();
      jfrm.add(err);
      jfrm.add(jlab);
      jfrm.add(ajtf);
      jfrm.add(bjtf);
      jfrm.add(button);
      jfrm.add(alab);
      jfrm.add(blab);
      jfrm.add(anslab);
      ActionListener l = new ActionListener() {
         public void actionPerformed(ActionEvent evt) {
            System.out.println("Action event from a text field");
         }
      };
      ajtf.addActionListener(l);
      bjtf.addActionListener(l);
      button.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent evt) {
            try{
               int a = Integer.parseInt(ajtf.getText());
               int b = Integer.parseInt(bjtf.getText());
               int ans = a/b;
               alab.setText("\nA = " + a);
               blab.setText("\nB = " + b);
               anslab.setText("\nAns = "+ ans);
            }
            catch(NumberFormatException e){
               alab.setText("");
               blab.setText("");
               anslab.setText("");
```

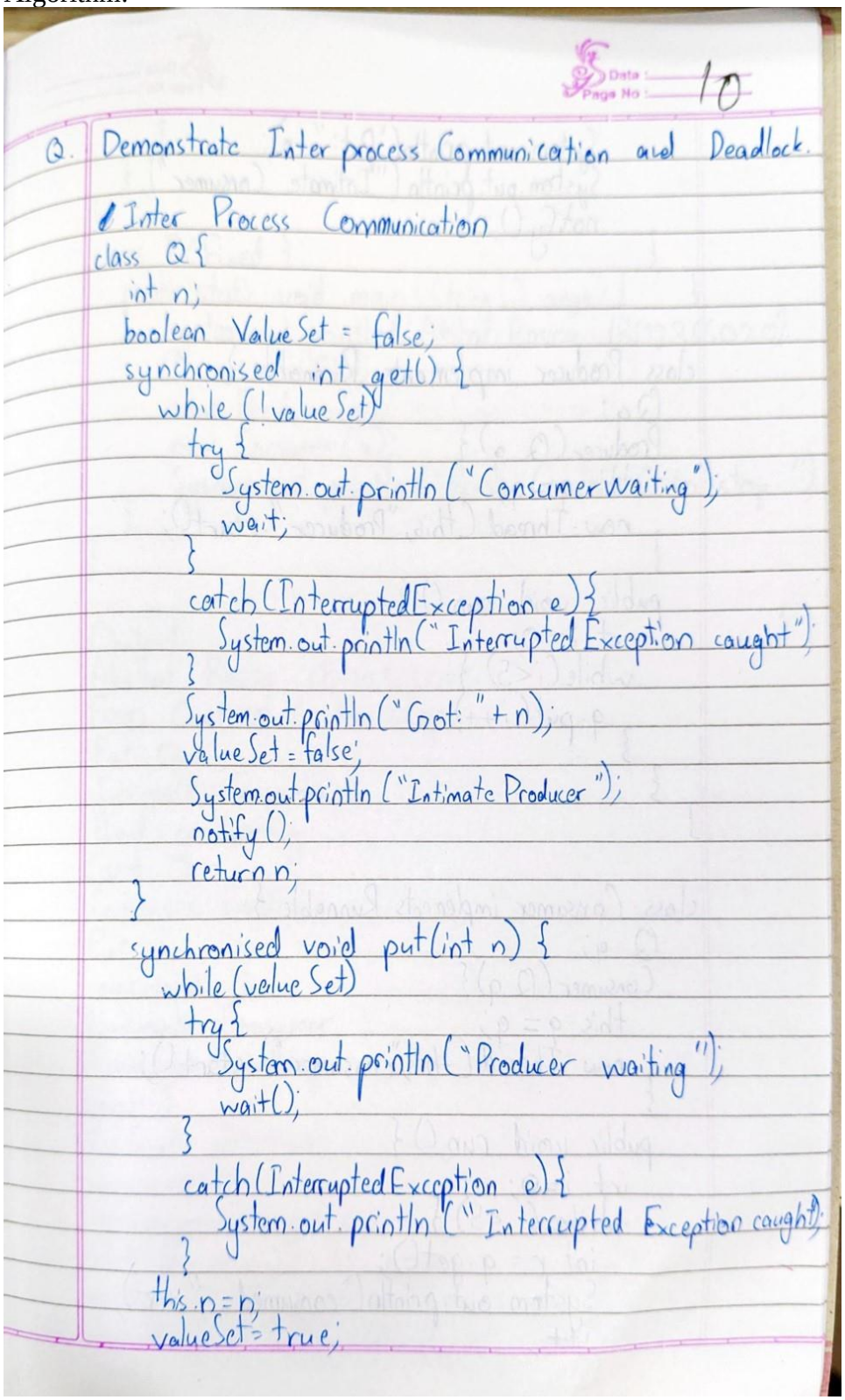
```
                err.setText("Enter Only Integers!");
            }
            catch(ArithmeticException e){
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("B should be NON zero!");
            }
        }
    });
    jfrm.setVisible(true);
  }
  public static void main(String[] args){
    SwingUtilities.invokeLater(new Runnable(){
      public void run(){
        new SwingDemo();
      }
    });
  }
}
```



Program 10

Demonstrate Inter process Communication and deadlock

Algorithm:



Q. Demonstrate Inter process Communication and Deadlock.

```
// Inter Process Communication
class Q {
  int n;
  boolean ValueSet = false;
  synchronised int get() {
    while (!valueSet)
      try {
        System.out.println("Consumer waiting");
        wait;
      }
      catch (InterruptedException e){
        System.out.println("Interrupted Exception caught");
      }
      System.out.println("Got: " + n);
      valueSet = false;
      System.out.println("Intimate Producer");
      notify();
      return n;
  }
  synchronised void put(int n) {
    while (valueSet)
      try {
        System.out.println("Producer waiting");
        wait();
      }
      catch (InterruptedException e){
        System.out.println("Interrupted Exception caught");
      }
    this.n = n;
    valueSet = true;
```

Page No: 10

Q. Demonstrate Inter process Communication and Deadlock.

```
// Inter Process Communication
class Q {
  int n;
  boolean ValueSet = false;
  synchronised int get() {
    while (!valueSet)
      try {
        System.out.println("Consumer waiting");
        wait;
      }
      catch(InterruptedException e){
        System.out.println("Interrupted Exception caught");
      }
      System.out.println("Got: " + n);
      valueSet = false;
      System.out.println("Intimate Producer");
      notify();
      return n;
  }
  synchronised void put(int n) {
    while(valueSet)
      try{
        System.out.println("Producer waiting");
        wait();
      }
      catch(InterruptedException e){
        System.out.println("Interrupted Exception caught");
      }
    this.n = n;
    valueSet = true;
```

```
    }
  }
}
class PCFixed {
  public static void main(String[] args) {
    System.out.println("Akshat Basra 1BM23CS020");
    Q q = new Q();
    new Producer(q);
    new Consumer(q);
    System.out.println("Press Control-C to stop.");
  }
}
```

Output:

```
Akshat Basra 1BM23CS020
Press Control-C to stop.
Put: 0
Intimate Consumer
Producer waiting
Got: 0
Intimate Producer
Put: 1
consumed: 0
Intimate Consumer
Producer Waiting
Got: 1
Intimate Producer
consumed: 1
consumer waiting
Put: 2
:
: // Till consumed: 4
```

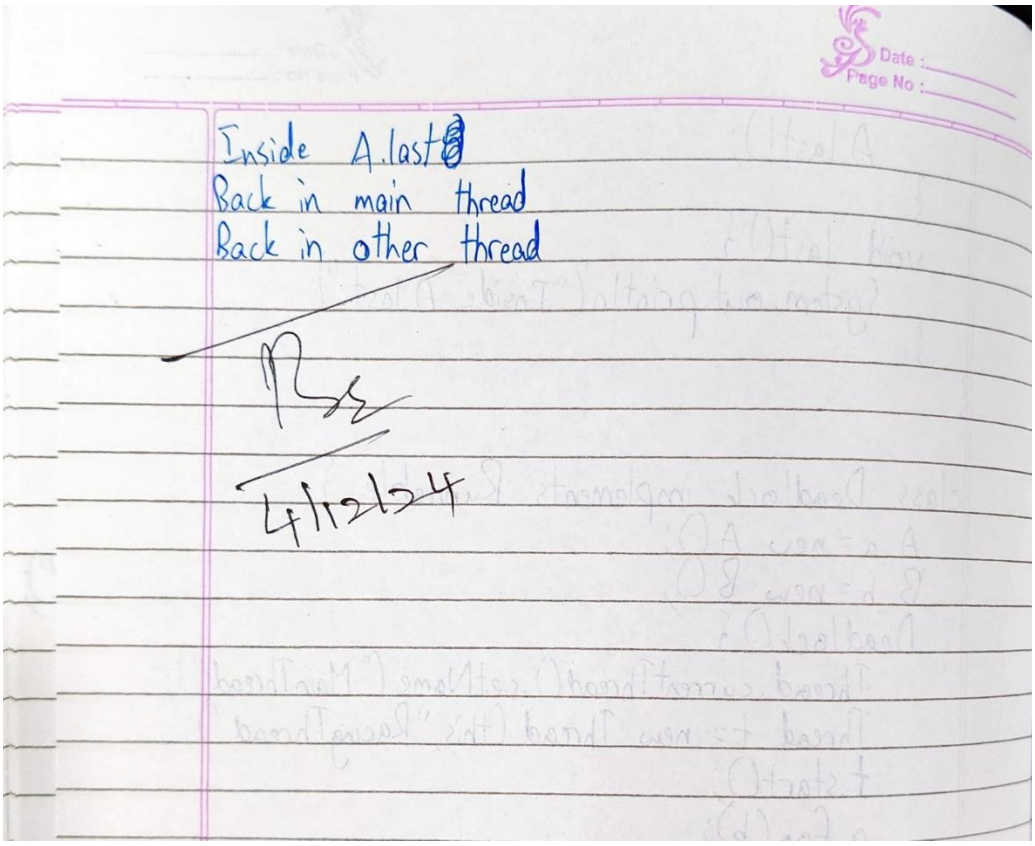
DeadLock:

```
class A{
  synchronised 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("Inside A.t name + " trying to call
                        B.last()");
    b.last();
  }
  void last(){
    System.out.println("Inside A.last");
  }
}

class B {
  synchronised 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();
  }
  void last(){
    System.out.println("Inside A.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("Akshat Basra 1BM23CS020");
      new Deadlock();
    }
}
```

Output:

```
Akshat Basra 1BM23CS020
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
Inside A.last
Mainthread trying to call B.last()
```



Code:

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get() {
        while(!valueSet)
            try {
                System.out.println("Consumer waiting");
                wait();
            } catch(InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("Intimate Producer");
        notify();
        return n;
    }
    synchronized void put(int n) {
        while(valueSet)
            try {
                System.out.println("Producer waiting");
                wait();
            } catch(InterruptedException e) {
                System.out.println("InterruptedException caught");
```

```
            }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("Intimate Consumer");
        notify();
    }
}

class Producer implements Runnable {
    Q q;
    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run() {
        int i = 0;
        while(i<15) {
            q.put(i++);
        }
    }
}
class Consumer implements Runnable {
    Q q;
    Consumer(Q q) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void run() {
        int i=0;
        while(i<15) {
            int r=q.get();
            System.out.println("consumed:"+r);
            i++;
        }
    }
}

class PCFixed {
    public static void main(String[] args) {
        System.out.println("Akshat Basra 1BM23CS020");
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
    }
}
```

```
Akshat Basra 1BM23CS020
Press Control-C to stop.
Put: 0
Intimate Consumer
Producer waiting
Got: 0
Intimate Producer
Put: 1
consumed:0
Intimate Consumer
Producer waiting
Got: 1
Intimate Producer
consumed:1
Consumer waiting
Put: 2
Intimate Consumer
Producer waiting
Got: 2
Intimate Producer
consumed:2
Consumer waiting
Put: 3
Intimate Consumer
Producer waiting
Got: 3
Intimate Producer
consumed:3
Consumer waiting
Put: 4
Intimate Consumer
```

```
Put: 5
Intimate Consumer
Producer waiting
Got: 5
Intimate Producer
consumed:5
Put: 6
Intimate Consumer
Producer waiting
Got: 6
Intimate Producer
consumed:6
Put: 7
Intimate Consumer
Producer waiting
Got: 7
Intimate Producer
consumed:7
Put: 8
Intimate Consumer
Producer waiting
Got: 8
Intimate Producer
consumed:8
Put: 9
Intimate Consumer
Producer waiting
Got: 9
Intimate Producer
consumed:9
```

```
Put: 10
Intimate Consumer
Producer waiting
Got: 10
Intimate Producer
consumed:10
Put: 11
Intimate Consumer
Producer waiting
Got: 11
Intimate Producer
consumed:11
Put: 12
Intimate Consumer
Producer waiting
Got: 12
Intimate Producer
consumed:12
Put: 13
Intimate Consumer
Producer waiting
Got: 13
Intimate Producer
consumed:13
Put: 14
Intimate Consumer
Got: 14
Intimate Producer
consumed:14
```

```
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();
   }
   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();
   }
   void last() {
      System.out.println("Inside A.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("Akshat Basra 1BM23CS020");
        new Deadlock();
    }
}
```

```
Akshat Basra 1BM23CS020
RacingThread entered B.bar
MainThread entered A.foo
RacingThread trying to call A.last()
Inside A.last
MainThread trying to call B.last()
Inside A.last
Back in main thread
Back in other thread
```