SHLOK SHIVARAM IYER

1BM22CS260 3RD SEM SEC:E

INDEX

NAME: Shlok S. Iyer STD.: 3rd sem SEC.: E ROLL NO.: _____ SUB.: OOP in Java 23CS3PCOOJ

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
1	~~13~~/12/23	~~Quadratic~~ Sample Progs	~~10~~	15-12
2	12/12/23	~~Rectangle~~ Quadratic equation	10	12-12
3.	19/12/23	Student & GPA	10	19/12
4.	26/12/23	Books name	10	26/12
5	2/1/24	Abstract Area Main	10	2/1/2024
6.	9/1/24	Bank class with Savings/Current	10	16/1/2024
6.	23/1/24	Packages for final marks	10	
7	30/1/24	Father Son Age Validation	10	30/1/2024
8.	6/2/24	Multithreading	10	6/2/2024
9.	20/2/24	User-Interface for Int Division		20/2/2024
10.	13/2/24	Interprocess comm & Deadlock	10	15/2/24

LAB1: INPUT

Batch II
SHLOK IYER
1BM22CS260

4-12-23

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

```
import java.util.Scanner;
class Quadratic
{
    int a, b, c;
    double r1, r2, d;
    void getd()
    {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter the coefficients of a,b,c");
        a = s.nextInt();
        b = s.nextInt();
        c = s.nextInt();
    }
    void compute()
    {
        while (a == 0)
        {
            System.out.println("Not a quadratic eqn");
            System.out.println("Enter a non zero number");
            Scanner s = new Scanner(System.in);
        }
        d = b*b - 4*a*c;
        if (d == 0)
        {
            r1 = (-b)/(2*a);
            System.out.println("Roots are equal");
            System.out.println("Root 1 = Root 2 = " + r1);
```

```
        }
        else if(d>0)
        {
            r1 = ((-b) + (Math.sqrt(d)) / (double)(2*a);
            r2 = ((-b) - (Math.sqrt(d)) / (double(2*a);
            System.out.println("Roots are real and distinct
            System.out.println("Root1 = " + r1 + "Root2 = " + r2
        }

        else if(d<0)
        {
            System.out.println("Roots are imaginary");
            r1 = (-b) / (2*a);
            r2 = Math.sqrt(-d) / (2*a);
            System.out.println("Root1 = " + r1 + " + i" + r2);
            System.out.println("Root2 = " + r1 + " - i" + r2);
        }
    }
}

class Quadratic main
{
    public static void main(String args[])
    {
        Quadratic q = new Quadratic();
        q.getd();
        q.compute();
    }
}
```

LAB1 OUTPUT

Output:

Enter the coefficient of a, b, c

3

1

2

The roots are imaginary

Root1 = 0.0 + i0.7993052538854531

Root1 = 0.0 - i0.7993052538854531

Enter the coefficient of a, b, c

1

4

1

The roots are real and distinct

Root1 = -2.86794919243112 Root 2 = -5.732050807568877

Enter the coefficients of a, b, c

1

2

1

The roots are equal

Root1 = Root2 = -1.0

13/10/2023

LAB2

SHLOK
1BM22CS260

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 the details, and a method to calculate SGPA of a student

$$SGPA = \frac{\sum (\text{course credits} * \text{grade Points})}{\sum (\text{course credits})}$$

```
import java.util.*Scanner;
class Subject
{
    int subjectMarks;
    int credits;
    int grade;
}
class Student
{
    String name;
    String usn;
    Subject subject[];
    double SgPa;
    Scanner s;

    Student()
    {
        int i;
        subject = new Subject[9];
        for(i=0; i<9; i++)
            subject[i] = new Subject();
        s = new Scanner(System.in);
    }
```

```
void getStudentDetails()
{
    System.out.println("Enter the USN : ");
    usn = s.nextLine();
    System.out.println("Enter the name : ");
    name = s.nextLine();
}
void getmarks()
{
    for(int i=0; i<9; i++)
    {
        System.out.println("Enter the subject marks:
        subject[i].subjectMarks = s.nextInt();
        System.out.println("Enter the credits : ");
        subject[i].credits = s.nextInt();
        subject[i].grade = (subject[i].subjectMarks/10);
        if(subject[i].grade > 10){
            subject[i].grade = 10;
        if (subject[i].grade <=4)
            subject[i].grade = 0;
    }
}
void computeSGPA()
{
    int effectivescore = 0;
    int totalcred = 0;
    for(int i=0; i<9; i++)
    {
        effectivescore += (subject[i].grade * subject[i].credits
        totalcred += subject[i].credits;
    }
    sgpa = (double) effectivescore/(double) totalcred;
}
```

```
}
Public class Main
{
    public static void main(String args[])
    {
        Student s1 = new Student();
        s1.getStudentDetails();
        s1.getmarks();
        s1.computeSGPA();
        System.out.println("Name : " + s1.name);
        System.out.println("USN : " + s1.usn);
        System.out.println("SGPA : " + s1.sgpa);
    }
}
```

Output:

Enter the USN:

260

Enter the name:

Shivik

Enter the subject marks:

90

Enter the credits:

4

Enter the subject marks:

71

Enter the credits:

4

Enter the subject marks:

84

Enter the credits:

3

Enter the Subjectmarks:
94
Enter the credits:
3
Enter the subject marks:
88
Enter the credits:
3
Enter the subject marks:
91
Enter the credits:
1
Enter the subject marks:
90
Enter the credits:
1
Enter the subject marks:
84
Enter the credits:
1
Enter the subject marks:
88
Enter the credits:
1
Name: SNok
USN : 260
SGPA: 9.238095238095237

19/12/2023

LAB3

SHLOK IYER
1BM22CS260

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

```
import java.util.Scanner;
class Books.
{
    String name;
    String author;
    int price;
    int numPages;
    Books(String name, String author, int prince, int numPages)
    {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
    public String toString()
    {
        String name, author, price, numPages;
        name = "Book name: " + this.name + "\n";
        author = "Author name: " + this.author + "\n";
        price = "Price: " + this.price + "\n";
        numPages = "Number of pages :" + this.numPages + "\n";
        return name + author + Price + numPages;
    }
}
```

```
class BooksMain
{
    public static void main(String args[])
    {
        Scanner s = new Scanner(System.in);
        int n;
        String name;
        String author;
        int price;
        int numPages;
        String ans;
        System.out.println("Enter the amount of bo
        n = s.nextInt();
        Books b[];
        b = new Books[n];
        for(int i=0; i<n; i++)
        {
            System.out.println("Enter the name of the bo
            name = s.next();
            System.out.println("Enter the author of
            author = s.next();
            System.out.println("Enter the amount of pag
            numPages = s.nextInt();
            b[i] = new Books(name, author, price, num
        }
        for(int i=0; i<n; i++)
        {
            ans = b[i].toString();
            System.out.println(ans);
        }
    }
}
```

Output :

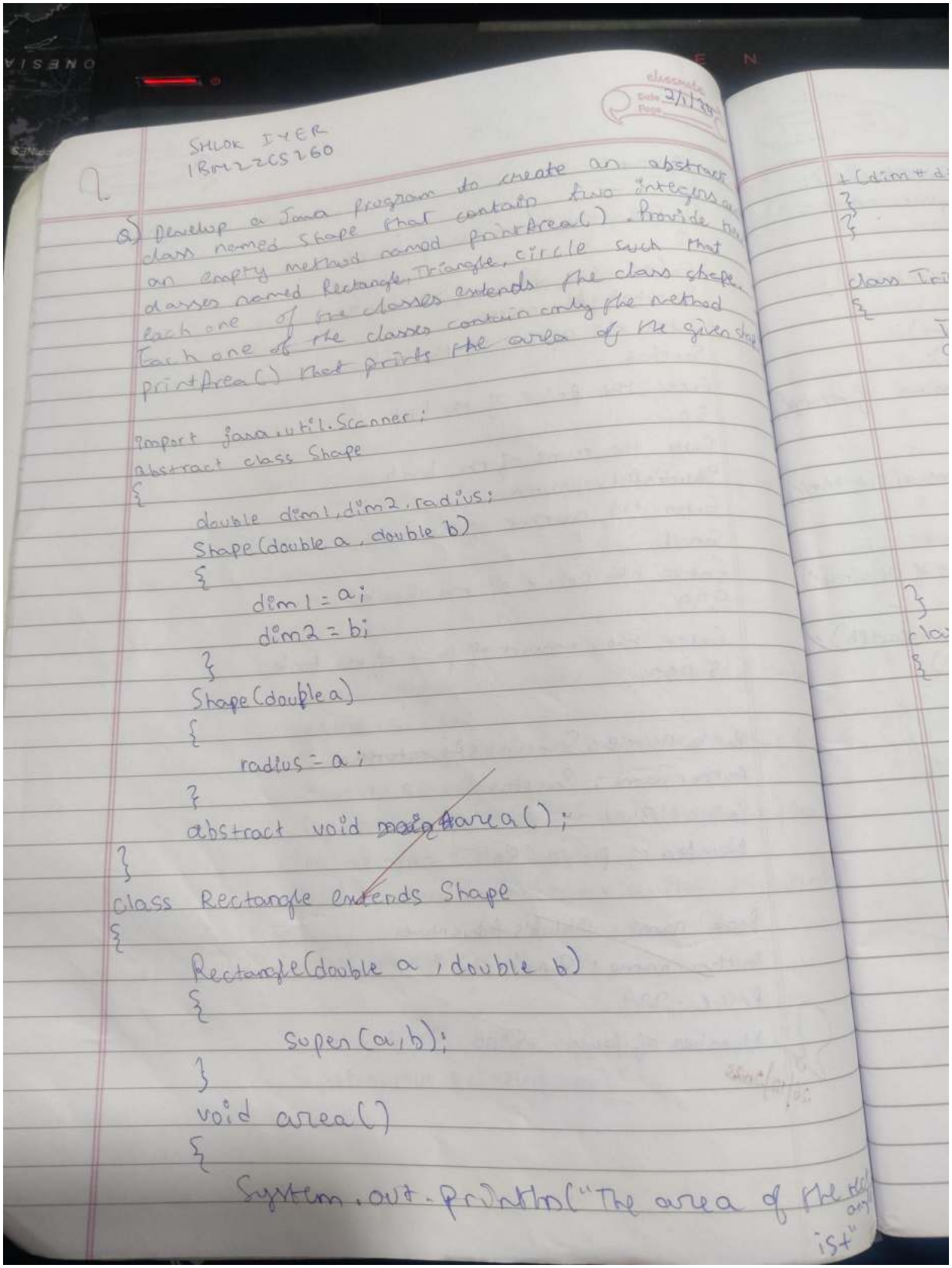
Enter the amount of books
2
Enter the name of the book
Sarthak's Adventures
Enter the author of the book
Sarthak
Enter the price of the book
500
Enter the name of the book
Shivi's Adventures
Enter the author of the book
Sanvi
Enter the price of the book
999
Enter the amount of page of the book
15000

Book name : Sarthak's Adventures
Author name : Sarthak
Price : 300
Number of pages : 500

Book name : Shivi's Adventures
Author name : Sanvi
Price : 999
Number of pages : 15000

26/12/2023

LAB4



SHLOK IYER
1BM22CS260

Q) Develop a Java program to create an abstract class named Shape that contain two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle, 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.

```
import java.util.Scanner;
abstract class Shape
{
    double dim1, dim2, radius;
    Shape(double a, double b)
    {
        dim1 = a;
        dim2 = b;
    }
    Shape(double a)
    {
        radius = a;
    }
    abstract void area();
}
class Rectangle extends Shape
{
    Rectangle(double a, double b)
    {
        super(a,b);
    }
    void area()
    {
        System.out.println("The area of the
ist"
```

```
+(dim1 * dim2) + " units");
}
}

class Triangle extends Shape
{
    Triangle(double a, double b)
    {
        super(a,b);
    }
    void area()
    {
        System.out.println("Area of triangle:"+(dim1*dim2)/2
+units);
    }
}
class Circle extends Shape
{
    Circle(double a)
    {
        super(a);
    }
    void area()
    {
        System.out.println("Area of circle is:"+(3.14*radius
*radius);
    }
}

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

```
Scanner sc = new Scanner(System.in);
System.out.println("Enter your name:");
String name = sc.nextLine();

System.out.println("Enter your USN:");
String usn = sc.nextLine();
System.out.println("NAME:" + name);
System.out.println("USN:" + usn);
System.out.println("Enter the dimensions of the
double length = sc.nextInt();
double breadth = sc.nextInt();
System.out.println("Enter the dimensions of the
double base = sc.nextInt();
double height = sc.nextInt();
System.out.println("Enter the radius of the
double r = sc.nextInt();
Rectangle rect = new Rectangle(length, breadth
Triangle tri = new Triangle(base, height);
Circle cir = new Circle(r);
Shape shaperef;
shaperef = rect;
shaperef.area();
shaperef = tri;
shaperef.area();
shaperef = cir;
shaperef.area();

}

}
```

Output:

Enter your name:
Shlok
Enter your USN:
1BM22CS260
Enter the dimensions of rectangle:
2
4
Enter the dimensions of triangle
6
8
Enter the radius of the circle
3
NAME: Shlok
USN: 1BM22CS260
The area of the rectangle is : 8.0 units
The area of the triangle is : 24.0 units
The area of the triangle is : 28.259999999998 units

2/1/2024

LAB-5

SHLOK IYER
1BM22CS260

Date 9-1-24

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 facility 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

```
import java.util.*;
class account
{
    String name;
    int accnum;
    String typeacc;
    double balance;
    account(String name, int accnum, String typeacc, double
            balance)
    {
        this.name = name;
        this.accnum = accnum;
        this.typeacc = typeacc;
        this.balance = balance;
    }

    public void deposit(double amount)
    {
        balance += amount;
        System.out.println("Deposit Successful, Updated balance
        :"+balance);
    }
    public void withdraw(double amount)
```

```
        }
        if(balance-amount >=0)
        {
            balance -=amount;
            System.out.println("The balance after
            withdrawal is :" + balance);
        }

        else
        {
            System.out.println(" Insufficient Funds ");
        }
    }
    public void display()
    {
        System.out.println(" Name :" + name +" Account name:"
                +accnum +" Type :" + typeacc +" Balance:
                + balance);

    }
}
class savacct extends account
{
    private double rate=10;
    savacct(String name, int accnum, String typeacc, double balance)
    {
        super(name,accnum, typeacc, balance);
    }
    public void interest()
    {
        balance += balance * (rate/100);
        System.out.println("The balance after applying an
        interest is :" + balance);
    }
```

```
}
class curracct extends account
{
    private double minbal=500;
    private double service=10;
    curracct(String name, int accnum, String typeacc, double
            balance)
    {
        super(name, accnum, typeacc, balance);
    }
    public void check(double balance)
    {
        if(balance<minbal)
        {
            balance -= service;
            System.out.println("Service charge is impose
            since balance is below min bal of 500");
            System.out.println("balance after service charge:"
                                +balance);
        }
        else
        {
            System.out.println("No service charge imposed");
        }
    }
}
class bank
{
    public static void main(String args[])
    {
        Scanner s= new Scanner(System.in);
        System.out.println("Enter your name");
```

```
String name = s.next();
System.out.println("Enter your account number: ");
int accnum = s.nextInt();
System.out.println("Enter the type of account (Savings/Current
                                                    ");
String typeacc = s.next();
System.out.println("Enter the current balance value");
double balance = s.nextDouble();
int ch;
double depamt, withamt;
account acc = new account(name, accnum, typeacc, balance);
Savacct sa = new savacct(name, accnum, typeacc, balance);
current cur = new current(name, accnum, typeacc, balance);
while (true)
{
   if (acc.typeacc.equals("Savings"))
   {
      System.out.println("\n Menu\n 1. deposit 2.
      withdraw 3. compute interest 4. display 5. End loop");
      System.out.println("Enter the choice");
      ch = s.nextInt();
      switch(ch)
      {
         case 1:
               { System.out.println("enter the amount:");
               depamt = s.nextInt();
               sa.deposit(depamt);
               break;}

         case 2:
               { System.out.println("enter the amount:");
               withamt = s.nextInt();
               sa.withdraw(withamt);
```

```
                case 4:
                        {sa.display();
                         break;}
                case 5:
                        { System.exit(0); }

                default:
                        { System.out.println("invalid input");
                          break;}
                }
            }
        else
        {
            System.out.println("\n Menu bol. Check penalty2.en
                                dwop");
            System.out.println("Enter the choice");
            ch = s.nextInt();
            switch(ch)
            {
                case 1:
                        { cur.check(balance);
                          break;}
                case 2:
                        { System.exit(0); }
                default:
                        { System.out.println("invalid input");
                          break;}
                }
            }
        }
    }
}
```

Output:

Enter your name
Shwk
Enter your account number:
123
Enter the type of account (Savings/Current)
Savings
Enter the current balance value
2000

Menu
1. deposit 2. withdraw 3. compute interest 4. display 5. end
Enter the choice
1
Enter the amount:
100
Deposit successful. Update balance: 2100.0

Menu
1. deposit 2. withdraw 3. compute interest 4. display 5. end loop
Enter the choice
2
Enter the amount:
3000
Insufficient funds
Menu
1. deposit 2. withdraw 3. compute interest 4. display 5. end loop
Enter choice
3
The balance after applying an interest: 2310.0

23-01-24

SHLOK IYER
1BM22CS260

LAB-6

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

//Inside cie Student.java

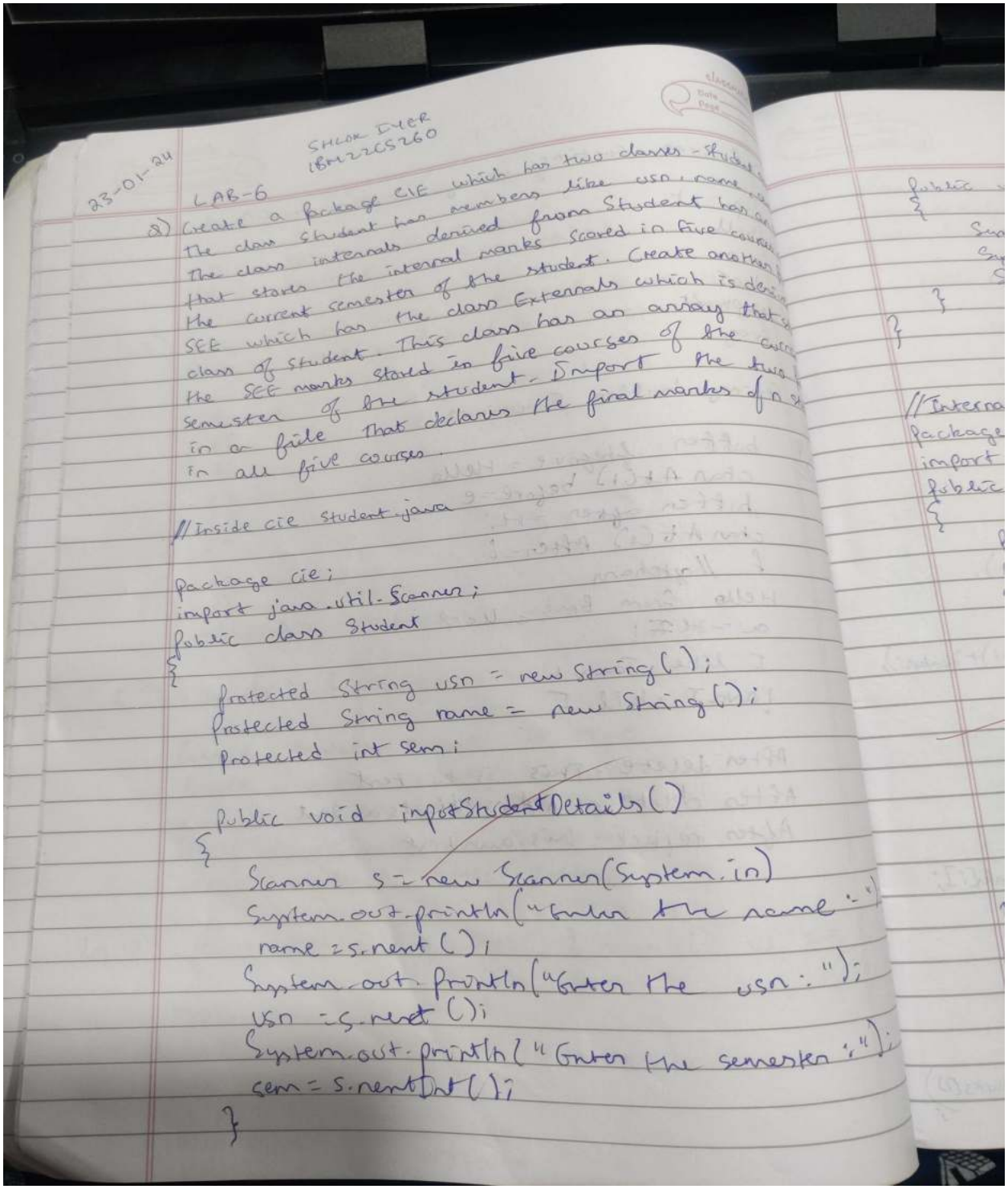
```
package cie;
import java.util.Scanner;
public class Student
{
    protected String usn = new String();
    protected String name = new String();
    protected int sem;

    public void importStudentDetails()
    {
        Scanner s = new Scanner(System.in)
        System.out.println("Enter the name:")
        name = s.next();
        System.out.println("Enter the usn:");
        usn = s.next();
        System.out.println("Enter the semester:");
        sem = s.nextInt();
    }
```

For Current account

Enter your name
Shlok
Enter your account number:
123
Enter the type of account (Savings / Current)
Current
Enter the current balance value
200
Menu
1. Check penalty 2. end loop
Enter the choice
1
Service charge is imposed since your balance is below minimum balance of 500
balance after service charge : 190.0

16/1/2024

LAB6



23-01-24

SHLOK IYER
1BM22CS260

LAB-6

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

//Inside cie Student.java

```
package cie;
import java.util.Scanner;
public class Student
{
    protected String usn = new String();
    protected String name = new String();
    protected int sem;

    public void importStudentDetails()
    {
        Scanner s = new Scanner(System.in)
        System.out.println("Enter the name:")
        name = s.next();
        System.out.println("Enter the usn:");
        usn = s.next();
        System.out.println("Enter the semester:");
        sem = s.nextInt();
    }
```

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


//Internals.java inside cie
package cie;
import java.util.Scanner;
public class Internals extends Student
{
    protected int marks[] = new int[5];
    public void inputCIEmarks()
    {
        Scanner s = new Scanner(System.in);
        for(int i=0; i<5; i++)
        {
            System.out.println("Enter the "+(i+1)+"st/th subject marks: ");

            marks[i] = s.nextInt();
        }

    }
}
```

```
//Externals.java inside see
package see;
import cie.Internals;
import java.util.Scanner;
public class Externals extends Internals
{
    protected int marks[],
    protected int finalMarks[];
    public Externals()
    {
        marks = new int[5];
        finalMarks = new int[5];
    }
public void inputSEEmarks()
{
    Scanner s=new Scanner(System.in);
    for(int i =0; i<5;i++)
    {
        System.out.println("Subject"+(i+1)+"
        marks[i]=s.nextInt();
    }
}
public void calculateFinalMarks()
{
    for(int i=0;i<5;i++)
        finalMarks[i]=marks[i]/2 +super.mark
}
public void displayFinalMarks()
{
    displayStudentDetails();
    for(int i=0;i<5;i++)
        System.out.println("Subject "+(i+1)+": "+finalMa
}
}
```

```
//mainfinals.java
import see.Externals;
public class mainFinals
{
    public static void main(String args[])
    {
        int numOfStudents = 2;
        Externals finalMarks[] = new Externals[numOfStudents];
        for(int i=0; i<numofStudents; i++)
        {
            finalMarks[i] = new Externals();
            finalMarks[i].inputStudentDetails();
            System.out.println("Enter CIE marks:");
            finalMarks[i].inputCIEmarks();
            System.out.println("Enter SEE marks");
            finalMarks[i].inputSEEmarks();
        }
        System.out.println("Display the final marks:\n");
        for(int i=0; i<numOfStudents; i++)
        {
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

Output:

```
Enter the name:
Shlok
Enter the USN:
1BM22CS260
```

Enter the semester:
3
Enter CIE marks:
Enter the 1st/th subject mark: 45
Enter the 2 st/th subject mark: 40
Enter the 3st/th subject mark: 35
Enter the 4th/th subject marks: 44
Enter the 5 st/th subject marks: 23

Enter SEE marks
Subject 1 marks: 100
Subject 2 marks: 88
Subject 3 marks: 89
Subject 4 marks: 75
Subject 5 marks: 88

Enter the name:
Simon
Enter the USN:
1BM22CS277
Enter the semester:
2
Enter CIE marks
Enter the 1st/th subject mark: 12
Enter the 2st/th subject mark: 45
Enter the 3st/th subject marks: 23
Enter the 4st/th subject marks: 5
Enter the 5st/th subject marks: 33
Enter the SEE marks
Subject 1 marks: 100
Subject 2 marks: 12
Subject 3 marks: 23

Subject 4 marks : 56
Subject 5 marks : 100

Displaying the final marks :

Name : Simon
usn : 1BM22CS260
sem : 3
Subject 1 : 95
Subject 2 : 84
Subject 3 : 79
Subject 4 : 81
Subject 5 : 67

Name : Simon
usn : 1BM22CS277
Sem : 3
Subject 1 : 62
Subject 2 : 51
Subject 3 : 34
Subject 4 : 33
Subject 5 : 83

20/11/24

LAB7

LAB 7 – EXCEPTION HANDLING

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

```
import java.util.*;
class WrongAge extends Exception
{
    public WrongAge(String s)
    {
        super(s);
    }
}

class father
{
    private int age;
    public father(int age) throws WrongAge
    {
        this.age = age;
        if(age < 0)
        {
            throw new WrongAge("Age of father cannot be
                                negative");
        }
    }
}
```

```
class Son extends father
{
    Private int SonAge;
    public Son(int fatherAge, int sonAge) throws WrongAge
    {
        super(fatherAge);
        if(SonAge <0)
        {
            throw new WrongAge("Son's age cannot be negative");
        }
        else if(SonAge >= fatherAge)
        {
            throw new WrongAge("Son's cannot be older/equal
                                than father");
        }
        else if(fatherAge - SonAge <= 20)
        {
            throw new WrongAge("Difference of age is less");
        }
        else
        {
            System.out.println("Age is valid");
        }
        this.SonAge=SonAge;
    }
}

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

```
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the Age of the Son:");
        int SonAge = sc.nextInt();
        System.out.println("Enter the father age:");
        int fatherAge = sc.nextInt();
        Son son = new Son(fatherAge, SonAge);
      }
      catch(Exception e)
      {
        System.out.println(e.getMessage());
      }
    }
}
```

Output:

```
Enter the Age of Son:
23
Enter the Age of father:
-10
Age of father cannot be negative

Enter the Age of Son:
-20
Enter the Age of father:
25
Age of Son cannot be negative
```

Enter the age of son:
23
Enter the age of father:
56
Age is valid

Enter the age of son:
56
Enter the age of father:
23
Son cannot be older/equal to father

30/1/2024

LAB8

6/02/24

LAB-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.

```
class bms extends Thread {
        public void run() {
            for(int i=1 ; i<=5 ; i++){
                System.out.println("bms college of
                                    engineering");
                try
                {
                    Thread.sleep(10000);
                }
                catch(InterruptedException e){
                    e.printStackTrace();}
            }
        }
}

class cse extends Thread {
{
    public void run(){
        for(int i=1; i<=10 ; i++){
            System.out.println("cse"+i);
            try
            {
                Thread.sleep(2000); }
            catch(InterruptedException e){
                e.printStackTrace();
            }
        }
}
```

```
class lab8{
        public static void main(String a[])
        {
            bms obj1=new bms();
            cse obj2=new cse();
            obj1.start();
            obj2.start();
        }
}
```

Output:

bms college of engineering 1
cse1
cse2
cse 3
cse 4
cse 5
bms college of engineering2
cse 6
cse 7
cse8
cse9
cse10
bms college of engineering 3
bms college of engineering 4
bms college of engineering 5

6/2/2024

LAB9

Date: 20/2/24

LAB 9-

Write a program that creates a user interface to perform integer division. 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 NumberFormat Exception. If Num2 were zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

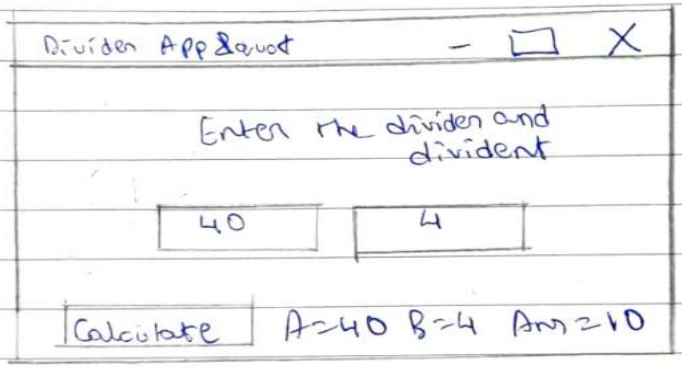
```
import javax.swing.*;
import java.awt.*;
import java.awt.events.*;

class SwingDemo {
    SwingDemo() {
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275, 150);
        jfrm.setLayout(new FlowLayout());
        // to terminate on close
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // text label
        JLabel jlab = new JLabel("Enter the divider and dividat");

        //add text field for both numbers
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);

        //Calc button
        JButton button = new JButton("Calculate");
```

```
//labels
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();

//add in order
jfrm.add(err);
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(anslab);

ActionListener I = new ActionListener() {

    public void actionPerformed(ActionEvent evt)
    {
        System.out.println("Action event from a text fie
    }
};

ajtf.addActionListener(I);
bjtf.addActionListener(I);

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("\n A =" + a);
blab.setText("\n B =" + b);
anslab.setText("\n Ans =" + 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 :-



Definitions :-

JFrame :

A class in Java that has its own modules and constructors

FlowLayout () :

Creates a flow layout with centered alignments and a default 5 unit horizontal and vertical gap.

jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

To terminate on close

JButton ("Calculate:") :
Button with text "Calculate" inside.

JLabel :
To give labels to the objects

jfrm.add(err);
To add error object that has the label

ActionListener ():
Defines what should be done when a user performs a certain operation

getText ();
This method receives a String

SetVisible ():
If you set it to be true, it means you want that thing to be visible in your screen.

JTextField :
A lightweight component that allows the editing of a single line of text.

20/2/2024

LAB10-A

13/2/24

LAB -10 -A-

Demonstrate interprocess communication and deadlock

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

```
        catch (InterruptedException e)
        {
            System.out.println("InterruptedException caught");
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put : " + n);
        System.out.println("\n Intimate Consumer\n");
        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<5)
        {
            int r = q.get();
            System.out.println("Consumed :" + r);
            i++;
        }
    }
}

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

Output:

```
Press Control-C to stop
Put : 0
Intimate Consumer
Producer waiting
Got : 0
Intimate Producer
Put : 1
```

Intimate Consumer
Producer waiting
consumed : 0
Got : 1

Intimate Producer
consumed : 1
Put : 2

Intimate Consumer
Producer waiting
Got : 2

Intimate Producer
consumed : 2
Put : 3

Intimate Consumer
Producer waiting
Got : 3
Intimate Producer
consumed : 3
Put : 4
:
:
:
13 Intimate Consumer
Got : 14

Intimate Producer
consumed : 14

LAB10-B

b) Implementing Deadlock

```
class A
{
    Synchronized void foo(B b)
    {
        String name = Thread.currentThread().getName();
        System.out.println(name + "entered A.foo");
        try
        {
            Thread.sleep(5000);
        }
        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[])
    {
        new Deadlock();
    }
}
```

Output:

MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
Inside A.last
Back in mainthread
RacingThread trying to call A.last()
Inside A.last
Back in otherthread

13/2/2024

SOURCE CODE

LAB1

```
import java.util.Scanner;
class quadratic
{
	int a,b,c;
	double r1,r2,d;
	void getd()
	{
		Scanner s=new Scanner(System.in);
		System.out.println("Enter the coefficient of a,b,c");
		a=s.nextInt();
		b=s.nextInt();
		c=s.nextInt();
	}
	void compute()
	{
		while(a==0)
		{
			System.out.println("Not a quadratic equation");
			System.out.println("Enter a non zero number");
			Scanner s=new Scanner(System.in);
			a=s.nextInt();
		}
		d=(b*b)-(4*a*c);
		if(d==0)
		{
			r1=(-b)/(2*a);
			System.out.println("The roots are equal");
			System.out.println("Root1=Root2="+r1);
		}
		else if(d>0)
		{
			r1=((-b)+(Math.sqrt(d))/(double)(2*a));
			r2=((-b)-(Math.sqrt(d))/(double)(2*a));
			System.out.println("The roots are real and distinct");
			System.out.println("Root1= "+r1+"Root 2"+r2);
		}
		else if(d<0)
		{
			System.out.println("The roots are imaginary");
			r1=(-b)/(2*a);
			r2=Math.sqrt(-d)/(2*a);
```

```
                        System.out.println("Root1= "+r1+"+i"+r2);
                        System.out.println("Root1= "+r1+"-i"+r2);
                }
        }
}
class QuadraticMain
{
        public static void main(String args[])
        {
                quadratic q=new quadratic();
                q.getd();
                q.compute();
        }
}
```

LAB 2

```
import java.util.Scanner;
class Subject
{
	int subjectMarks;
	int credits;
	int grade;
}
class Student
{
	String name;
	String usn;
	Subject subject[];
	double sgpa;
	Scanner s;

	Student()
	{
		int i;
		subject=new Subject[9];
		for(i=0;i<9;i++)
			subject[i] = new Subject();
		s = new Scanner(System.in);
	}
	void getStudentDetails()
	{
		System.out.println("Enter the USN :");
		usn=s.nextLine();
		System.out.println("Enter the name :");
		name=s.nextLine();
	}
	void getmarks()
	{
		for(int i=0;i<9;i++)
		{
			System.out.println("Enter the subjectmarks :");
			subject[i].subjectMarks=s.nextInt();
			System.out.println("Enter the credts:");
			subject[i].credits=s.nextInt();
			subject[i].grade=(subject[i].subjectMarks/10)+1;
			if(subject[i].grade>10)
				subject[i].grade=10;
			if(subject[i].grade<=4)
				subject[i].grade=0;
```

```
                }
        }
        void computeSGPA()
        {
                int effectivescore=0;
                int totalcred=0;
                for( int i=0;i<9;i++)
                {
                        effectivescore+=(subject[i].grade*subject[i].credits);
                        totalcred+=subject[i].credits;
                }
                sgpa=(double)effectivescore/(double)totalcred;
        }
}
public class Main
{
        public static void main(String args[])
        {
                Student s1 = new Student();
                s1.getStudentDetails();
                s1.getmarks();
                s1.computeSGPA();
                System.out.println("Name : "+s1.name);
                System.out.println("usn : "+s1.usn);
                System.out.println("SGPA : "+s1.sgpa);
        }
}
```

LAB3

```
import java.util.Scanner;
class Books
{
	String name;
	String author;
	int price;
	int numPages;
	Books(String name, String author, int price, int numPages)
	{

		this.name = name;
		this.author = author;
		this.price = price;
		this.numPages = numPages;
	}
	public String toString()

	{

		String name, author, price, numPages;

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

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

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

		numPages = "Number of pages: " + this.numPages + "\n";

		return name + author + price + numPages;

	}
}
class BooksMain
{
	public static void main(String args[])

	{

		Scanner s = new Scanner(System.in);

		int n;
		String name;
```

```
		String author;
		int price;
		int numPages;
		String ans;
		System.out.println("Enter the amount of books");
		n = s.nextInt();
		Books b[];
		b = new Books[n];
		for(int i=0;i<n;i++)
		{

			System.out.println("Enter the name of the book");
			name=s.next();
			System.out.println("Enter the author of the book");
			author=s.next();
			System.out.println("Enter the price of the book");
			price=s.nextInt();
			System.out.println("Enter the amount of pages of the book");
			numPages=s.nextInt();
			b[i] = new Books(name,author,price,numPages);
		}
		for(int i=0;i<n;i++)
		{
			ans=b[i].toString();
			System.out.println(ans);
		}
	}
}
```

LAB4

```
import java.util.Scanner;
abstract class Shape
{
	double dim1,dim2,radius;
	Shape(double a, double b)
	{
		dim1=a;
		dim2=b;
	}
	Shape(double a)
	{
		radius=a;
	}
	abstract void area();
}
class Rectangle extends Shape
{
	Rectangle(double a, double b)
	{
		super(a,b);
	}
	void area()
	{
		System.out.println("The area of the rectangle is : "+(dim1*dim2)+"units");
	}
}
class Triangle extends Shape
{
	Triangle(double a, double b)
	{
		super(a,b);
	}
	void area()
	{
		System.out.println("The area of the triangle is : "+(dim1*dim2)/2+"units");
	}
}
class Circle extends Shape
{
	Circle(double a)
	{
		super(a);
	}
```

```
	void area()
	{
		System.out.println("The area of the triangle is : "+(3.14*radius*radius)+"units");
	}
}
class AbstractAreaMain
{
	public static void main(String args[])
	{
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter your name : ");
		String name=sc.nextLine();

		System.out.println("Enter your USN : ");
		String usn=sc.nextLine();

		System.out.println("NAME : "+name);
		System.out.println("USN : "+usn);

		System.out.println("Enter the dimensions of the rectangle : ");
		double length=sc.nextInt();
		double breadth=sc.nextInt();
		System.out.println("Enter the dimensions of the triangle(base and height) : ");
		double base=sc.nextInt();
		double height=sc.nextInt();
		System.out.println("Enter the radius of the circle: ");
		double r=sc.nextInt();
		Rectangle rect=new Rectangle(length,breadth);
		Triangle tri=new Triangle(base,height);
		Circle cir=new Circle(r);
		Shape shaperef;
		shaperef=rect;
		shaperef.area();
		shaperef=tri;
		shaperef.area();
		shaperef=cir;
		shaperef.area();
	}
}
```

LAB5

```
import java.util.*;
class account
{
	String name;
	int accnum;
	String typeacc;
	double balance;
	account(String name,int accnum,String typeacc,double balance)
	{
		this.name=name;
		this.accnum=accnum;
		this.typeacc=typeacc;
		this.balance=balance;
	}
	public void deposit(double amount)
	{
		balance+=amount;
		System.out.println("The balance after depositing is:"+balance);
	}
	public void withdraw(double amount)
	{
		if(balance-amount>=0)
		{
			balance-=amount;
			System.out.println("The balance after withdrawing is:"+balance);
		}
		else
			System.out.println("Insufficient value");
	}
	public void display()
	{
		System.out.println("Name: "+name+" Account number: "+accnum+" Type:
"+typeacc+"Balance:"+balance);
	}
}
class savacct extends  account
{
	private double rate=10;
	savacct(String name,int accnum,String typeacc,double balance)
	{
		super(name,accnum,typeacc,balance);
	}
	public void interest()
```

```
	{
		balance+=balance*(rate/100);
		System.out.println("The balance after interest is: "+balance);
	}
}
class curracct extends  account
{
	private double minbal=500;
	private double service=10;
	curracct(String name,int accnum,String typeacc,double balance)
	{
		super(name,accnum,typeacc,balance);
	}
	public void check(double balance)
	{
		if(balance<minbal)
		{
			balance-=service;
			System.out.println("Service charge is imposd since your balance is below
min bal of 500");
			System.out.println("balance after service charge:"+balance);
		}
		else
			System.out.println("No service charge imposed");
	}
}
class bank
{
	public static void main(String args[])
	{
		Scanner s=new Scanner(System.in);
		System.out.println("Enter your name");
		String name=s.next();
		System.out.println("Enter your account nnumber: ");
		int accnum=s.nextInt();
		System.out.println("Enter the type of account (Savings/Current)");
		String typeacc=s.next();
		System.out.println("Enter the current balance value");
		double balance=s.nextDouble();

		double depamt,withamt;
		account acc= new account(name,accnum,typeacc,balance);
		savacct sa=new savacct(name,accnum,typeacc,balance);
		curracct cur=new curracct(name,accnum,typeacc,balance);
```

```
		while(true)
		{
			if(acc.typeacc.equals("Savings"))
			{
				System.out.println("\nMenu\n1.deposit 2.withdraw 3.compute
interest 4.display 5.endloop");
				System.out.println("Enter the choice");
				int ch=s.nextInt();
				switch(ch)
				{
					case 1:
						{System.out.println("enter the amount:");
						depamt=s.nextInt();
						sa.deposit(depamt);
						break;}
					case 2:
						{System.out.println("enter the amount:");
						withamt=s.nextInt();
						sa.withdraw(withamt);
						break;}
					case 3:
						{sa.interest();
						break;}
					case 4:
						{sa.display();
						break;}
					case 5:{System.exit(0);}

					default:
						{System.out.println("invalid input");
						break;}

				}
			}
			else
			{
				System.out.println("\nMenu\n 1.Check penalty 2.end loop");
				System.out.println("Enter the choice");
				int ch=s.nextInt();
				switch(ch)
				{
					case 1:
						{cur.check(balance);break;}
					case 2:
```

```
                    {System.exit(0);}
                default:
                    {System.out.println("invalid input");
                    break;}
                }
            }
        }
    }
}
```

LAB 6

```
//cie folder
package cie;
import java.util.Scanner;
public class Internals extends Student
{
   protected int marks[] = new int[5];
   public void inputCIEmarks()
   {
      Scanner s =new Scanner(System.in);
      for(int i=0;i<5;i++)
      {
         System.out.printf("Enter the %d st/th subject mark : ",i+1);
         marks[i]=s.nextInt();

      }

   }
}
```

```
package cie;

import java.util.Scanner;

public class Student
{

   protected String usn = new String();
   protected String name = new String();
   protected int sem;


   public void inputStudentDetails()
   {
      Scanner s= new Scanner(System.in);
      System.out.println("Enter the name : ");
      name=s.next();
      System.out.println("Enter the usn : ");
      usn=s.next();
      System.out.println("Enter the semester : ");
      sem=s.nextInt();
   }
```

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



//SEE FOLDER


package see;

import cie.Internals;

import java.util.Scanner;

public class Externals extends Internals
{

    protected int marks[];
    protected int finalMarks[];
    public Externals()
    {

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

    public void inputSEEmarks()
    {

        Scanner s = new Scanner(System.in);

        for(int i=0;i<5;i++)
        {

            System.out.print("Subject "+(i+1)+" marks: ");
            marks[i] = s.nextInt();
        }
```

```
    }
    public void calculateFinalMarks()
    {

        for(int i=0;i<5;i++)
            finalMarks[i] = marks[i]/2 + super.marks[i];

    }

    public void displayFinalMarks()
    {
        displayStudentDetails();
        for(int i=0;i<5;i++)
            System.out.println("Subject " + (i+1) + ": " + finalMarks[i]);

    }

}



//MAIN
import see.Externals;
public class mainfinals
{

 public static void main(String args[])
    {

        int numOfStudents = 2;

        Externals finalMarks[] = new Externals[numOfStudents];

        for(int i=0;i<numOfStudents;i++)

        {

            finalMarks[i] = new Externals();

            finalMarks[i].inputStudentDetails();

            System.out.println("Enter CIE marks");

            finalMarks[i].inputCIEmarks();
```

```
            System.out.println("Enter SEE marks");

            finalMarks[i].inputSEEmarks();
        }
        System.out.println("Displaying the final marks :\n");

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

            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

LAB 7

```
import java.util.*;
class WrongAge extends Exception
{
   public WrongAge(String s)
   {
      super(s);
   }
}
class Father
{
   private int age;
   public Father(int age) throws WrongAge
   {
      this.age=age;
      if(age<0)
      {
         throw new WrongAge("Age of father cannot be negative ");
      }
   }
}
class Son extends Father
{
   private int SonAge;
   public Son(int FatherAge,int SonAge) throws WrongAge
   {
      super(FatherAge);
      if(SonAge<0)
      {
         throw new WrongAge("Son's age cannot be negative");
      }
      else if(SonAge>=FatherAge)
      {
         throw new WrongAge("Son cannot be older/equal than father");

      }
      else if(FatherAge-SonAge<=20)
      {
         throw new WrongAge("Difference of age is very less");

      }
      else
      {
```

```
            System.out.println("Age is Valid");
        }
        this.SonAge=SonAge;
    }

}

class FatherSonAge
{
    public static void main(String args[])
    {
        try
        {   Scanner sc=new Scanner(System.in);
            System.out.println("Enter the Age of the son ");
            int SonAge=sc.nextInt();
            System.out.println("Enter the father's age: ");
            int FatherAge=sc.nextInt();
            Son son=new Son(FatherAge,SonAge);
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }
    }
}
```

LAB 8

```
class bms extends Thread{
	public void run(){
		for(int i=1;i<=5;i++){
			System.out.println("bms college of engineering"+i);
			try
			{
				Thread.sleep(10000);
			}
			catch(InterruptedException e){
				e.printStackTrace();
			}
			}
		}
}
class cse extends Thread{
	public void run(){
		for(int i=1;i<=10;i++){
			System.out.println("cse"+i);
			try
			{
				Thread.sleep(2000);
			}
			catch(InterruptedException e){
				e.printStackTrace();
			}
			}
		}
}
class lab8{
	public static void main(String a[]){
		bms obj1=new bms();
		cse obj2=new cse();
		obj1.start();
		obj2.start();
	}
}
```

LAB 9

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

class SwingDemo{
	SwingDemo()
	{

	JFrame jfrm = new JFrame("Divider App&amp;quot");
	jfrm.setSize(275, 150);
	jfrm.setLayout(new FlowLayout());
	// to terminate on close
	jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

	// text label
	JLabel jlab = new JLabel("Enter the divider and divident");

	// add text field for both numbers
	JTextField ajtf = new JTextField(8);
	JTextField bjtf = new JTextField(8);

	// calc button
	JButton button = new JButton("Calculate");

	// labels
	JLabel err = new JLabel();
	JLabel alab = new JLabel();
	JLabel blab = new JLabel();

	JLabel anslab = new JLabel();

	// add in order :)
	jfrm.add(err); // to display error bois
	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!");
        }
        }
        });

        // display frame
        jfrm.setVisible(true);
    }

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

```
        });
    }
}
```

LAB 10A

```
class Q
{

   int n;

   boolean valueSet = false;

   synchronized int get()
   {

      while(!valueSet)
      {
         try {

            System.out.println("\nConsumer waiting\n");

            wait();

         }
         catch(InterruptedException e)
         {

         System.out.println("InterruptedException caught");
         }
      }



      System.out.println("Got: " + n);

      valueSet = false;

      System.out.println("\nIntimate Producer\n");

      notify();

      return n;

   }
   synchronized void put(int n)
   {

      while(valueSet)
```

```
        try
        {
            System.out.println("\nProducer waiting\n");
            wait();
        }
        catch(InterruptedException e)
        {
            System.out.println("InterruptedException caught");
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\nIntimate Consumer\n");
        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[])
    {

        Q q = new Q();

        new Producer(q);

        new Consumer(q);

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

    }

}
```

LAB 10 B

```
class A
{

   synchronized void foo(B b)
   {

      String name =Thread.currentThread().getName();

      System.out.println(name + " entered A.foo");

      try
      {

         Thread.sleep(1000);

      }
      catch(Exception e)
      {

         System.out.println("A Interrupted");

      }

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

      b.last();

   }

   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); // get lock on a in this thread.

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

    }
    public void run()
    {

        b.bar(a); // get lock on b in otherthread.

        System.out.println("Back in otherthread");

    }

    public static void main(String args[])
    {
        new Deadlock();
    }

}
```