B.M.S COLLEGE OF ENGINEERING BENGALURU

Autonomous Institute, Affiliated to VTU

LAB REPORT

23CS3PCOOJ

Submitted in partial fulfilment of the requirements for Lab

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

Prajwal P

(1BM22CS200)

Department of Computer Science and Engineering,

B.M.S College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019

2023-2024.

INDEX

Sl.No.	Title	Date
1	Complete scanned Observation Book	12/12/2023 - 20/02/2024
2	Lab 1	12/12/2023
3	Lab 2	19/12/2023
4	Lab 3	26/12/2023
5	Lab 4	02/01/2024
6	Lab 5	09/01/2024
7	Lab 6	16/01/2024
8	Lab 7	23/01/2024
9	Lab 8	30/01/2024
10	Lab 9	06/02/2024
11	Lab 10	20/02/2024

→ Develop a Java program that prints all real sol^n's to quad eqn, if dis -ve display no real root.

```
import java.util.Scanner;
class quad {
    public static void main (String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.println("Prajwal P 1bm22cs400");
        System.out.println("Enter coefficients:");
        int a,b,c; double r1,r2;
        a = sc.nextInt();
        b = sc.nextInt();
        c = sc.nextInt();
        while (a==0) {
            System.out.println("Enter non zero a valu
            Scanner s = new Scanner(System.in);
            a = s.nextInt();
        }
        int d = b*b - 4*a*c;
        if (d==0) {
            r1 = (-b)/(2*a);
            System.out.println("Roots are real and 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("Root 1 = "+r1);
    System.out.println("Root 2 = "+r2);
}

else if (d<0){
        System.out.println("Roots are imaginary");
        r1 = (-b) /(2*a);
        r2 = Math.sqrt(-d) / (2*a);
        System.out.println("Root 1 = "+r1+"+"i"+r2);
        System.out.println("Root 2 = "+r1+"-i"+r2);
    }
}
}
```

Output :-

```
Prajwal.P  1BM22CS200
#Case 1 :- Enter coefficients a,b,c
1
2
1
Roots are real and equal
Root 1 = Root 2 = -1.0
```

#Case 2:- Enter coefficients a, b, c

0

1

2

Not a quadratic equation

Enter a non zero value of a:

#Case 3 Enter coefficients a, b, c

1

1

2

): Roots are imaginary

): Root 1 = 0.0 + i 1.322875

Root 2 = 0.0 - i 1.322875

#Case 4 Enter coefficient of a, b, c

1

-3

2

Roots are real and distinct

Root 1 = 2.0

Root 2 = 1.0

12/12

→ SGPA calculator

```
import java.util.Scanner;
class Subject
{  int sub_marks;
   int credits;
   int grade;
}
class SGPA
{
   Subject subject[]
   String name, USN;
   double sgpa;
   Scanner sc= new Scanner(System.in);

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

   void getData()
   { System.out.println("Enter name of Student:");
     name = sc.nextLine();
```

```
        System.out.println("Enter USN : ");
        USN = sc.nextLine();
    }

    void getMarks()
    { for(int i=0; i<8; i++)
        { System.out.println("Enter subject "+(i+1)+" marks");
          subject[i].sub_marks = sc.nextInt();
          System.out.println("Enter subject "+(i+1)+" credits");
          subject[i].credits = sc.nextInt();
        }
      for(int i=0; i<8; i++)
      { if(subject[i].sub_marks >= 90 && subject[i].sub_marks <100)
            subject[i].grade = 10;
        else if(subject[i].sub_marks >= 80 && subject[i].sub_marks <90)
            subject[i].grade = 9;
        else if(subject[i].sub_marks >= 70 && subject[i].sub_marks <80)
            subject[i].grade = 8;
        else if(subject[i].grade_marks >= 60 && subject[i].sub_marks <70)
            subject[i].grade = 7;
        else if(subject[i].grade_marks >= 50 && subject[i].sub_marks <60)
            subject[i].grade = 6;
        else if(subject[i].sub_marks >= 40 && subject[i].sub_marks <50)
            subject[i].grade = 5;
        else {
            System.out.println("Enter valid score");
            break;
```

```
    }
}
    double compute()
    { double sum=0.0, num;
      for(int i=0; i<8; i++)
      { num=(subject[i].grade)*(subject[i].credits);
        sum+=num;
      }
      sgpa=sum/20;
      return sgpa;
    }
  void display()
  { System.out.println("Name of candidate: "+name);
    System.out.println("USN of candidate: "+USN);
    System.out.println("SGPA = "+sgpa);
  }
}
class Main {
  public static void main(String args[]){
    System.out.println("Prajwal P\n1BM22CS200\n\n");
    SGPA ob = new SGPA();
    ob.getData();
    ob.getMarks();
double a=ob.compute();
    ob.display(a);
  }
}
```

Output :-

Prajwal P

1BM22CS200

Enter name of Student:

Prajwal P

Enter USN:

1BM22CS200

Enter subject 1 marks.

97

Enter subject 1 credit

4

Enter subject 2 marks

93

Enter subject 2 credit

4

Enter subject 3 marks

92

Enter subject 3 credit

3

Enter subject 4 marks

94

Enter subject 4 credit

3

Enter subject 5 marks
87
Enter subject 5 credits
3
Enter subject 6 marks

Enter subject 6 credits
1
Enter subject 7 marks

Enter subject 7 credits
1
Enter subject 8 marks

Enter subject 8 credits
1

Name of candidate: Prajwal P
USN of candidate: 1BM22CS200
SGPA = 9.8

10/10

→ Create a Book class which has 4 members: name, author, price, num_pages. Include constructor to set value for members. Include methods to set and get details of objects. Include toString() method that could display all details

```
import java.util.Scanner;
class Book {
    String name, author;
    int price, num_pages;

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

    public String toString() {
        String n,a,p,N;
        n = "\n" + "Name of Book : " + name + "\n";
        a = "Author of Book : " + author + "\n";
        p = "Price of Book : " + price + "\n";
        N = "Number of Pages : " + num_pages + "\n";
        return n+a+p+N;
    }
}
```

```
class Main {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter no of books: ");
        int n=sc.nextInt();
        Book b[]=new Book[n];
        String name,author;
        int price, num;
        sc.nextLine();
        for(int i=0; i<n; i++){
            System.out.println("Enter name of book: ");
            name = sc.nextLine();
            System.out.println("Enter author's name: ");
            author = sc.nextLine();
            System.out.println("Enter price: ");
            price=sc.nextInt();
            System.out.println("Enter num of pages: ");
            num = sc.nextInt();
            b[i] = new Book(name, author, price, num);
        }
        System.out.println("Book Details: ");
        for(int i=0; i<n; i++) {
            System.out.println(b[i].toString());
        }
    }
}
```

Output:-

Enter number of book:
2
Enter name of book:
Bible
Enter author's name:
God
Enter price:
1000
Enter number of pages:
100

Enter name of Book:
Text
Enter author's name:
Prajwal
Enter price:
100
Enter number of pages:
1000

Book Details:
Name of Book: Bible
Author of Book: God
Price of Book: 1000
Number of Pages: 100

Name of Book : Text
Author of Book : Pragjwal
Enter price : 100
~~Enter~~ Number of Pages : 1000

Ans
26/12/23

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

```
import java.util.Scanner;
class InputScanner{
    Scanner sc = new Scanner (System.in);
}

abstract class Shape{
    double dim1, dim2;
    Shape{ double a, double b) {
        dim1 = a;
        dim2 = b;
    }
    abstract void printArea();
}

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

```
    void printArea() {
        System.out.println("Area of Rectangle is" +d
    }
}

class Triangle extends Shape {
    Triangle(double a, double b) {
        super(a, b);
    }
    void printArea() {
        System.out.println("Area of Triangle is
    }
}

class Circle extends Shape {
    Circle(double a, double b){
        super(a,b);
    } final double pi = 3.14159;
    void printArea(){
        System.out.println("Area of Circle is"+pi*Math.
    }
}
```

```
class Main extends InputScanner {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double a, b;

        System.out.println("Enter sides of Rectangle");
        a = sc.nextDouble();
        b = sc.nextDouble();
        Rectangle r = new Rectangle(a, b);

        System.out.println("Enter height and base of Triangle");
        a = sc.nextDouble();
        b = sc.nextDouble();
        Triangle t = new Triangle(a, b);

        System.out.println("Enter radius of Circle");
        a = sc.nextDouble();
        Circle c = new Circle(a, 1);

        Shape s;
        s = r;
        s.printArea();
        s = t;
        s.printArea();
        s = c;
        s.printArea();
    }
}
```

Output :-

→ Enter sides of Rectangle
10
10
Enter sides of Triangle
10
10
Enter radius of Circle
25

Area of Rectangle is 100.0
Area of Triangle is 50.0
Area of Circle is 1963.4937

→ Enter sides of Rectangle
-10
-25
Enter sides of Triangle
10
-5
Enter sides of Circle
15

Invalid input for Rectangle
Invalid input for Triangle
Area of circle is 706.85775

02/01/24

→ Develop a java program to create a class Bank that maintains 2 kinds of account for its customers, one called savings account and other current account. Savings account provides compound interest and withdrawal facility but no interest. Current account holders should have minimum balance and if balance falls below charge is imposed.

```
import java.util.Scanner;
class Account{
    String name;
    int acc_no;
    double balance;
    char c;
    Account( String name, int acc_no, char c){
    this.c = c;
    this.name = name;
    this.acc_no = acc_no;
     balance = 0.0;
    }

    void deposit( double amt){
    balance += amt;
    }
```

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

void display(){
    System.out.println("Balance : "+balance);
}

void interest(){
    System.out.println("Interest not calculated for current acc");
}
}

class Cur_acc extends Account{
    int min_bal = 500;
    int penalty = 50;
    Cur_acc(String name, int acc_no){
        super(name, acc_no, 'c');
    }
```

```
void withdraw(double amt){
    if(balance < min_bal){
        System.out.println("Penalty applied");
        balance -= min_bal;
    }
    if(balance > amt){
        balance -= amt;
    }
    else{
        System.out.println("Insufficient funds");
    }
}
}

class Sav_acc extends Account{
    int min_bal = 500;
    int penalty = 50;
    double rate = 5.0;
    double amt;
    Sav_acc(String name, int acc_no){
        super(name, acc_no, 'S');
    }
    void interest(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter time (in years) since account");
```

```
double t = sc.nextDouble();
amount = balance*(Math.pow((1+rate/100),t));
System.out.println("Balance is:" +balance);
balance = amt;
}
}
}
class Bank {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        double amt = 0.0;
        System.out.println("Enter name & account holder");
        String name = sc.nextLine();
        System.out.println("Enter account no");
        int acc_no = sc.nextInt();
        Account obj;
        char a = sc.next().charAt(0);
        switch(a) {
        case 's' : obj = new Sav_acc(name,acc_no);
                   break;
        case 'c' : obj = new Car_acc(name,acc_no);
                   break;
        default: System.out.println("Invalid");
        }
```

```
while (true){
  System.out.println("1: for deposit \n 2\n for withdraw
                      \n 3. for balance \n 4 for interest
                      \n 5. To exit");
  int n = sc.nextInt();
  switch(n){
    case 1: System.out.println("Enter deposit amount:");
            amt = sc.nextDouble();
            obj.deposit(amt);
            break;
    case 2: System.out.println("Enter withdraw amt");
            amt = sc.nextDouble();
            obj.withdraw(amt);
            break;
    case 3: obj.display();
            break;
    case 4: obj.interest();
            break;
    case 5: return;
    default: System.out.println("Invalid");
  }
 }
}
```

Output :-

Enter name & account holder :

Prajwal.P

Enter account no :-

123456

Enter 's' for savings

Enter 'c' for current

s

Enter

1. To deposit
2. To withdraw
3. To display balance
4. To compute interest
5. Exit

1

Enter amount to deposit

10000

2

Enter amount to withdraw

1750

3

Balance is $250.0

4

Enter time (in years) since creating account

12

16/01/24

20 Circle Area : 78.539816
Circle perimeter : 31.4159296
Triagnle area : 12.0
Triangle perimeter : 16.64911

23/01/24

Week 7

Create package CIE with 2 classes Student and Intervals
Internals derived from Student has an array storing CIE marks for 5 courses. Create another package SEE which has external class. Array stores 5 see marks. Import 2 packages and find a final 5 marks.

→ Externals.java

```
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 inputSEE(){
      for(int i=0; i<5 ; i++){
          marks[i] = sc.nextInt();
      }
```

```
public void cal(){
  for(int i=0; i<5;i++)
      finalmarks[i] = marks[i]/2 + super.marks[i];
  }

public void display(){
    display_StudentDetails();
    for(int i=0; i<5;i++){
       System.out.println(finalmarks[i]);
  }
}
```

→ Internals.java.

```
package CIE;
import java.util.Scanner;
import CIE.student;
public class internals extends Student{
      protected int marks[] = new int[5];
      public void inputCIEmarks(){
        for(int i=0; i<5; i++){
            marks[i] = sc.nextInt();
        }
      }
```

→ Student.java

```
package CIE;
import java.util.Scanner;
public class Student{
    protected String usn;
    protected String name;
    protected int sem;
    public void InputStudentDetails(){
        System.out.println("Enter USN");
        usn = sc.nextLine();
        System.out.println("Enter name");
        name = sc.nextLine();
        System.out.println("Enter sem:");
        sem = sc.nextInt();
        this.usn = usn;
        this.sem = sem;
        this.name = name;
    }
    public void displayStudentDetails(){
        System.out.println("USN"+usn);
        System.out.println("Name "+name);
        System.out.println("Semester "+sem);
    }
}
```

--> Main.java

```
import SEE.externals;
class Main{
    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();
        final.marks[i].InputStudentDetails();
        finalmarks[i].InputCIEmarks();
        finalmarks[i].InputSEE marks");
    }
    for(int i=0;i<num;i++){
        finalmarks[i].calculateFinalmarks();
        finalmarks[i].displayfinalmarks();
    }
  }
}
```

Output:

Enter name : Prajwal P
Enter USN : 1BM22CS200
Enter Sem : 3
CIE marks :
1: 49
2: 48

3 : 48
4 : 42
5 : 49

SEE marks :
1 : 89
2 : 78
3 : 99
4 : 92
5 : 85

USN
Name : Prajwal P
Sem : 3
Final marks : 1 : 93
2 : 88
3 : 97
4 : 88
5 : 91

→ Week 8

Write a program that demonstrates handling of exceptions in inheritance tree. Create base class called Father and subclass Son, which extends the base class. In Father, implement a constructor which takes age & throws exception and age < 0. In son class do same and add exception for son >= father's age.

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

class Father {
    private int age;
    public Father(int age) throws WrongAge {
        if(age<0) {
            throw new WrongAge("Age cant be less than 0");
        }
        this.age=age;
    }
    public int getAge() {
        return age;
    }
```

```
class Son extends Father{
    private int SonAge;
    public son(int fage, int Sage) throws WrongAge {
        super(fage);
        if(sage >= fatherage) {
            throw new("Son's age cant be more
                                than father");
        }
        if(sage<0) {
            throw new("Age cant be negative");
        }
        SonAge = Sage;
    }
    public int getSonAge(){
        return Sage;
    }
}

public class Exception{
    public static void main(String args[]){
        Scanner sc = new Scanner (System.in);
        int Fage, Sage;
        try{
        System.out.println("Enter Father's age :");
        Fage = sc.nextInt();
        Father father = new(Fage);
```

```
System.out.println("Enter Son's Age");
Sage = sc.nextInt();
Son son = new Son(Fage, Sage);

System.out.println("Father's age: " + father.getAge + "\n Son's age:"
                                    + sonAge.getSonAge());
}
catch(WrongAge e) {
        System.out.println("Exception caught: "+e.getMessage());
    }
  }
}
```

O/P

Enter Father's age:
40
Enter Son's age:
20
Father's age: 40
Son's age: 20

Enter Father's age:
-12
Exception caught: Age cant be less than 0

→ Enter Fathers Age:
40
Enter Sons Age:
50
Exception caught : Son's Age cant be more than Father.

30.01.24

→ Threads

Write a program 2 threads one display BMS Clg Engineering every 10s and another showing 'CSE' once every 2s.

```
class BMS extends Thread{
    public void run(){
        while(true){
        System.out.println("BMS Clg of Engineering");
        try{
            Thread.sleep(10,000);
            }
        catch(InterruptedException e){
            System.out.println(e);
            }
        }
    }}

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

```
public class Main {
    public static void main(String[] args) {
        BMS bms = new BMS();
        CS cs = new CS();
        bms.start();
        cs.start();
    }
}
```

Output :-

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

-> Wak to

Demonstrate IPC

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

```
catch (InterruptedException e) {
    System.out.println(e);
} this.n = n;
  valueSet = true;
  System.out.println("Put " + n);
  System.out.println("\n Intimate consumer\n");
  notify();
}
}

class Consumer implements Runnable {
  Q q;
  Consumer(Q 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 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 Main{
    public static void main(String[] args){
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
    }
}
```

o/p

Put : 1
Got : 1
Put : 2
Got : 2
Put : 3
Got : 3
Put : 4
Got : 4
Put : 5
Got : 5

Deadlock

```
class A {
  synchronized void foo(B b) {
    String name = Thread.current().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 Main implements Runnable{
    A a = new A();
    B b = new B();
    Main(){
        Thread.currentThread().setName("Main Thread");
        Thread t = new Thread(this, "Racing Thread");
        t.start();
        a.foo(b);
        System.out.println("Back in main thread");
    }
    public void run(){
        b.bar(a);
        System.out.println("Back in other thread");
    }
    public static void main(String args[]){
        new Main();
    }
}
```

o/p

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

13.02.24

7).

d4).

Week 9

Write a program that creates a user interface to perform integer divisions. The users enter 2 two numbers in the text field, Num1 and Num2. The division of Num1 or Num2 were not integer, Throw a exception, if 2 wq 0 the program would throw an exception. Display exception in box

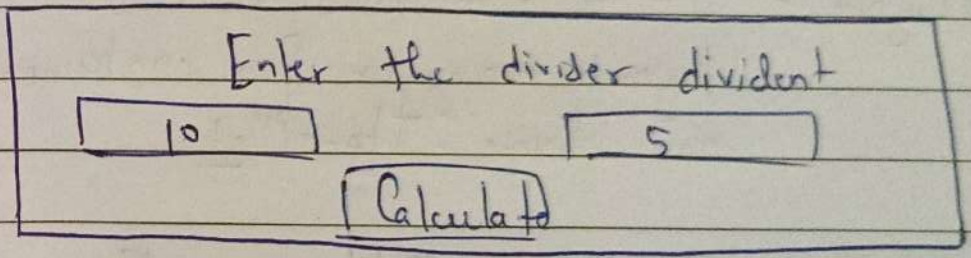
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

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 divid
                                  and divident");

        JTextField ajtf = new JtextField(8);
        JTextField bjtf = new JtextField(8);
```

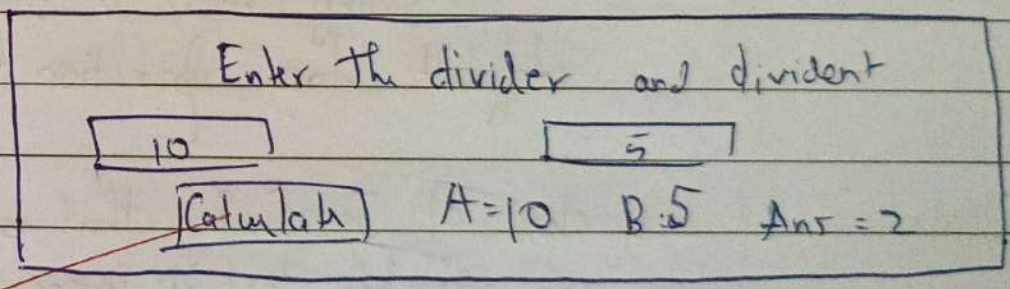
```
JButton button = new JButton("Calculate");
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel answlab = new JLabel();

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

ActionListener l = new ActionListener() {
    public void actionPerformed(ActionEvent ae) {
        System.out.println("Action event from a text field");
    }
};
atft.addActionListener(l);
batft.addActionListener(l);

button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent ae)
```

```
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();
        }
    });
}
}
```

o/p





20.08.24

Lab 1 [12-12-2023]

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

```
import java.util.Scanner;

class quad {
   public static void main(String args[]) {
      int a, b, c;
      double r1, r2, d;
      Scanner s = new Scanner(System.in);
      System.out.println("Prajwal P\t1BM22CS200");
      System.out.println("Enter the coefficients of a,b,c");
      a = s.nextInt();
      b = s.nextInt();
      c = s.nextInt();
      while (a == 0) {
         System.out.println("Not a quadratic equation");
         System.out.println("Enter a non zero value for a:");
         a = s.nextInt();
      }
      d = b * b - 4 * a * c;
      if (d == 0) {
         r1 = (-b) / (2 * a);
         System.out.println("Roots are real and 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("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("Root1 = " + r1 + " - i" + r2);
      }}}
```

Lab 2 [19-12-2023]

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
{
   int sub_marks;
   int credits;
   int grade;
}

class SGPA
{
   Subject subject[];
   String USN,name;
   double sgpa;
   Scanner sc = new Scanner(System.in);

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

   void getData()
   {
      System.out.println("Enter name of Student.");
      name=sc.nextLine();
      System.out.println("Enter USN of Student.");
      USN=sc.nextLine();
   }

   void getMarks()
   {
      for(int i=0;i<8;i++)
      {
         System.out.println("Enter subject "+(i+1)+" marks");
         subject[i].sub_marks=sc.nextInt();
         System.out.println("Enter subject "+(i+1)+" credits");
```

```
        subject[i].credits=sc.nextInt();
    }

    for(int i=0;i<8;i++)
    {
        if(subject[i].sub_marks>=90 && subject[i].sub_marks<=100)
        {
            subject[i].grade=10;
        }
        else if(subject[i].sub_marks>=80 && subject[i].sub_marks<90)
        {
            subject[i].grade=9;
        }
        else if(subject[i].sub_marks>=70 && subject[i].sub_marks<80)
        {
            subject[i].grade=8;
        }
        else if(subject[i].sub_marks>=60 && subject[i].sub_marks<70)
        {
            subject[i].grade=7;
        }
        else if(subject[i].sub_marks>=50 && subject[i].sub_marks<60)
        {
            subject[i].grade=6;
        }
        else if(subject[i].sub_marks>=40 && subject[i].sub_marks<50)
        {
            subject[i].grade=5;
        }
        else if(subject[i].sub_marks>=0 && subject[i].sub_marks<40)
        {
            subject[i].grade=0;
        }
        else
        {
            System.out.println("Enter valid score");
            break;
        }
    }
}

double compute()
{
    double sum=0.0,num;
    for(int i=0;i<8;i++)
```

```
    {
      num=(subject[i].grade)*(subject[i].credits);
      sum+=num;
    }
    sgpa=sum/20;
    return sgpa;
  }

  void display(double sgpa)
  {
    System.out.println("Name of candidate: "+name);
    System.out.println("USN of candidate: "+USN);
    System.out.println("SGPA= "+sgpa);
  }
}

class Main
{
  public static void main(String args[])
  {  System.out.println("Prajwal P\n1BM22CS200\n\n");
    SGPA ob=new SGPA();
    ob.getData();
    ob.getMarks();
    double a=ob.compute();
    ob.display(a);
  }
}
```

Lab 3 [26-12-2023]

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.

```
import java.util.Scanner;
class Book
        {
        String name,author;
        int price;
        int num_pages;

        Book(String name,String author,int price,int num_pages){
```

```
		this.name=name;
		this.author=author;
		this.price=price;
		this.num_pages=num_pages;
	}

	public String toString(){
		String n,a,p,N;
		n="\n"+"Name of Book: "+name+"\n";
		a="Author of Book: "+author+"\n";
		p="Price of Book: "+price+"\n";
		N="Number of pages: "+num_pages+"\n";
		return n+a+p+N;
	}

}

class Main
	{
	public static void main(String args[]){
		Scanner sc=new Scanner(System.in);
		System.out.println("\nEnter number of books: ");
		int n=sc.nextInt();
		Book b[]=new Book[n];
		String name,author;
		int price,num;
		sc.nextLine();
		System.out.println(“Prajwal P\t1BM22CS200”)
		for(int i=0;i<n;i++){
			System.out.println("Enter name of book: ");
			name=sc.nextLine();
			System.out.println("Enter author's name: ");
			author=sc.nextLine();
			System.out.println("Enter price: ");
			price=sc.nextInt();
			System.out.println("Enter number of pages: ");
			num=sc.nextInt();
			b[i]= new Book(name,author,price,num);
		}
		System.out.println("Book Details: ");
		for(int i=0;i<n;i++){
			System.out.println(b[i].toString());
		}}}
```

Lab 4 [02-01-2024]

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 contains only the method printArea( ) that prints the area of the given shape.

```
import java.util.Scanner;

abstract class Shape{
   double dim1,dim2;
   Shape(double a,double b){
      dim1=a;
      dim2=b;
   }
   abstract void printArea();
}

class Rectangle extends Shape{
   Rectangle(double a,double b){
      super(a,b);
   }
   void printArea(){
      if(dim1<0 || dim2<0){
         System.out.println("Invalid input for Rectangle");
      }
      else{
      System.out.println("\nArea of Rectangle is "+(dim1*dim2));
      }
   }
}

class Triangle extends Shape{
   Triangle(double a, double b){
      super(a,b);
   }
   void printArea(){
      if(dim1<0 || dim2<0){
         System.out.println("Invalid input for Triangle");
      }
      else{
      System.out.println("Area of Triangle is "+(dim1*dim2)/2);
      }
   }
}
```

```
class Circle extends Shape{
   Circle(double a, double b){
      super(a,b);
   }
   final double pi=3.14159;
   void printArea(){
      if(dim1<0 || dim2<0){
         System.out.println("Invalid input for Circle");
      }
      else{
      System.out.println("Area of Circle is "+pi*Math.pow(dim1,2));
      }
   }
}

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

      Scanner sc=new Scanner(System.in);
      double a,b;

      System.out.println("Enter sides of rectangle: ");
      a=sc.nextDouble();
      b=sc.nextDouble();
      Rectangle r=new Rectangle(a,b);

      System.out.println("Enter height and base length of Triangle: ");
      a=sc.nextDouble();
      b=sc.nextDouble();
      Triangle t=new Triangle(a,b);

      System.out.println("Enter radius of Circle: ");
      a=sc.nextDouble();
      Circle c=new Circle(a,1);

      Shape s;
      s=r;
      s.printArea();
      s=t;
      s.printArea();
      s=c;
      s.printArea();

   }}
```

Lab 5 [09-01-2024]

Develop a Java program to create a class of Bank that maintains two kinds of accounts for its customers, one called a savings account and the other a current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides a check 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 deposits from customers and update the balance.
b) Display the balance.
c) Compute and deposit interest
d) Permit withdrawal and update the balance
Check for the minimum balance, impose a penalty if necessary and update the balance.

```
import java.util.Scanner;
class Account {
 String name;
 int acc_no;
 double balance;
 char c;
 Account(String name, int acc_no, char c) {
  this.c = c;
  this.name = name;
  this.acc_no = acc_no;
  balance = 0.0;
 }
 void deposit(double amt) {
  balance += amt;
 }
 void withdraw(double amt) {
  if (balance > amt) {
   balance -= amt;
  } else {
   System.out.println("Insufficiant funds");
  }
 }
 void display() {
  System.out.println("Balance is " + balance);
 }
 void interest() {
  System.out.println("Interest not calculated for Current account;");
```

```
  }
}

class Cur_acc extends Account {
  int min_bal = 500;
  int penalty = 50;
  Cur_acc(String name, int acc_no) {
    super(name, acc_no, 'c');
  }
  void withdraw(double amt) {
    if (balance < min_bal) {
      System.out.println("Penalty applied");
      balance -= penalty;
    }
    if (balance > amt) {
      balance -= amt;
    } else {
      System.out.println("Insufficiant funds");
    }

  }

}
class Sav_acc extends Account {
  int min_bal = 500;
  int penalty = 50;
  double rate = 5.0;
  double amount;
  Sav_acc(String name, int acc_no) {
    super(name, acc_no, 's');
  }
  void interest() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter time(in years) since conception of account: ");
    double t = sc.nextDouble();
    amount = balance * (Math.pow((1 + rate / 100.0), t));
    System.out.println("Balance is " + amount);
    balance = amount;
  }
 }


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

```
   double amt = 0;
   System.out.println("Enter name of account holder: ");
   String name = sc.nextLine();
   System.out.println("Enter account number: ");
   int acc_no = sc.nextInt();
   System.out.println("Enter 's' for Svaings Account \nEnter 'c' for Current Account");
   Account obj;
   char a = sc.next().charAt(0);
   switch (a) {
   case 's':
    obj = new Sav_acc(name, acc_no);

    break;
   case 'c': obj = new Cur_acc(name, acc_no);
    break;
   default:
    System.out.println("Invalid input");
    return;
   }
   while (true) {
    System.out.println("Enter\n1. To deposit\n2. To Withdraw\n3. To Display balance\n4. To compute
interest\n5. To exit");
    int n = sc.nextInt();
    switch (n) {
    case 1:
     System.out.println("Enter amount to deposit: ");
     amt = sc.nextDouble();
     obj.deposit(amt);
     break;
    case 2:
     System.out.println("Enter amount to withdraw: ");
     amt = sc.nextDouble();
     obj.withdraw(amt);
     break;
    case 3:
     obj.display();
     break;
    case 4:
     obj.interest();
     break;
    case 5:
     return;
    default:  System.out.println("Invalid input");
}}}}
```

Lab 6 [23-01-2024]

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

```
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 sc = new Scanner(System.in);
      for(int i=0;i<5;i++) {
         System.out.print("Enter marks of Subject "+(i+1)+" : ");
         marks[i] = sc.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]);
      }
   }

}

package CIE;
import java.util.Scanner;
import CIE.student;
public class internals extends student{
   protected int marks[] = new int[5];
```

```
    public void InputCIEMarks(){
        Scanner sc = new Scanner(System.in);
        for(int i = 0; i < 5; i++){
            System.out.println("Enter Marks of Subject " + (i+1) + ":");
            marks[i] = sc.nextInt();
        }
    }
}

package CIE;
import java.util.Scanner;
public class student{
    protected String usn;
    protected String name;
    protected int sem;
    public void InputStudentDetails(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter USN: ");
        String usn = sc.next();
        System.out.println("Enter name: ");
        String name = sc.next();
        System.out.println("Enter sem: ");
        int sem = sc.nextInt();
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
    public void displayStudentDetails(){
        System.out.println("USN: " + this.usn);
        System.out.println("Name: " + this.name);
        System.out.println("Sem: " + this.sem);
    }
}

import SEE.externals;
class Main {
    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 data:\n");
        for(int i=0;i<numOfStudents;i++){
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

Lab 7 [30-01-2024]

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 the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is >=father’s age.

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

// Base class
class Father {
    private int age;

    // Constructor in Father class
    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Age cant be less than 0");
        }
        this.age = age;
    }

    public int getAge() {
        return age;
    }
```

```
}

// Derived class
class Son extends Father {
   private int sonAge;

   // Constructor in Son class
   public Son(int fatherAge, int sonAge) throws WrongAge {
      super(fatherAge);

      if (sonAge >= fatherAge) {
         throw new WrongAge("Son's age cant be more than Father's age");
      }

      if(sonAge<0){
         throw new WrongAge("Age cant be negative");
      }
      this.sonAge = sonAge;
   }

   public int getSonAge() {
      return sonAge;
   }
}

public class ExceptionInheritanceDemo {
   public static void main(String[] args) {
      Scanner sc=new Scanner(System.in);
      int Fage,Sage;
      try {


         System.out.println("Enter Father's age: ");
         Fage=sc.nextInt();
         Father father = new Father(Fage);

         System.out.println("Enter Son's age: ");
         Sage=sc.nextInt();
         Son son = new Son(Fage,Sage);

         System.out.println("Father's age: "+father.getAge()+"\nSon's Age: "+son.getSonAge());
      } catch (WrongAge e) {
         System.out.println("Exception caught: " + e.getMessage());
      }
   }}
```

Lab 8 [06-02-2024]

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

public class Main{
   public static void main(String args[]){
      BMS bms=new BMS();
      CS cs=new CS();
      bms.start();
      cs.start();
   }
}
```

Lab 9 [20-02-2024]

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.

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

class SwingDemo{
   SwingDemo(){
      // create jframe container
      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 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 10 [06-02-2024]

Demonstrate Inter-Process Communication and deadlock

IPC

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

DeadLock

```
class A {
    synchronized void foo(B b) {
```

```
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered A.foo");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println("A 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 Main implements Runnable {
    A a = new A();
    B b = new B();
    Main() {
        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 Main();
    }
}
```