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:

SUFIYAN DESAI

1BM22CS351

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	SCANNED OBSERVATION BOOK	12/12/2023-20/02/2024
2	LAB1	12/12/24
3	LAB2	19/12/24
4	LAB3	26/12/24
5	LAB4	02/01/24
6	LAB5	09/01/24
7	LAB6	23/1/24
8	LAB7	30/1/24
9	LAB8	06/2/24
10	LAB9	20/2/24
11	LAB10	13/2/24

Expt. No. LAB1 Date 12/12/2023

Page No. 1

1 Quadratic equation

```
import java.util.Scanner;
class Quadratic
{
    Scanner s = new Scanner(System.in);
    int a, b, c;
    double r1, r2, d;

    void getd()
    {
        Scanner s = new Scanner (System.in);
        System.out.println("Enter values for 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 value");
            Scanner s = new Scanner (System.in);
            a = s.nextInt();
        }
```

Teacher's Signature :

Date _____

Expt. No. _____ Page No. _____

```
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+" 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("Root1 = "+r1+" +i"r2);
    System.out.println("Root2 = " + r1 +" -i"r2);
}
}
}
```

Teacher's Signature : _____

Date .....

Expt. No. .....

Page No. .....

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

Output : SUFIYAN (1BM22EC261)

Enter coefficient a,b,c : 1 -5 2

roots are real and distinct

root1 = 4.5615328

root2 = 4.5615528l

Expt. No. LAB 2 Date

Page No.

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

~~class~~ Student

```
import java.util.Scanner;

class Subject
{
    int subjectmarks;
    int credits;
    String grade;
}

class Student
{
    int String name;
    String USN;
    double SGPA;
    Scanner s;
    Subject subject[];

Student() {
        int i;
        subject = new Subject[9];
```

Teacher's Signature :

```
for (i=0; i<a; i++)
        Subject[i] = new Subject();
        s= new Scanner (System.in);
    }
Void getStudentDetails()
{
    System.out.println ("Enter your name");
    name = s.nextLine();
    System.out.println ("Enter your USN");
    usn = s.nextLine();
}
void getMarks()
{
    int i;
    for (i=0; i<8; i++)
    {
      System.out.println ("Enter marks and credits for
                           courses :");
      System.out.println ("Marks :");
      int marks = s.nextInt();
      System.out.println ("credits :");
      int credits = s.nextInt();
```

```
Subject[i].SubjectMarks = marks;
Subject[i].credits = credits;

if (marks >= 90)
{
    Subject[i].grade = "O";
}

else if (marks >= 80)
{
    Subject[i].grade = "A+";
}

else if (marks >= 70)
{
    Subject[i].grade = "A";
}

else if (marks >= 60)
{
    Subject[i].marks = "B+";
}

else if (marks >= 50)
{
    Subject[i].marks = "B";
}
```

```
else if ( marks >=40)
  {   subject[i].grade = "C";
  }
else
    {   subject[i].grade = "F";
    }
  }
}

void computeSGPA() { int i;
                     double SGPA;
                     double totalcredits = 0;
                     double totalgp = 0;
                   for (i=0; i<8; i++)
                   {
                     totalcredits += subject[i].credits;
                     switch (subject[i].grade)
                   {
                     case 'O':
                     totalgp += 10 * subject[i].credits;
                     break;
                     case 'A':
                     totalgp += 9 * subject[i].credits;
                     break;
```

```
case "A":
    total gp += 8 * subject[i].credits;
    break;

case "B+":
    total gp += 7 * subject[i].credits;
    break;

case "B":
    total gp += 6 * subject[i].credits;
    break;

case "C":
    total gp = 5 * subject[i].credits;
    break;

case "F":
    total gp = 0 * subject[i].credits;
    break;
  }
}

sgpa = total gp / total credits;
System.out.println("Sgpa is " + sgpa);
  }
}
```

Teacher's Signature : _____

```
class sgpa
{
    public static void main (String [] a)
    {
        Student s1 = new Student ();
        s1. getStudentDetails ();
        s1. getMarks ();
        s1. computeSGPA();
    }
}
```

output : (SUFIYAN 1BM22EC261)

SGPA is 10.0000

Expt. No. LAB - 3 Date 26/12/23

Page No.

1 Create a class Book which contains four members name, author, price, num-pages.

```
import java.util.Scanner;
class Books
{
    String name;
    String author;
    int price;
    int num_pages;
}

Books (String name, String author, int price, int
                num_pages)

{   this.name = name;
    this.author = author;
    this.price = price;
    this.num-pages = numpages;
}

public String toString ()
{
    String name, author, price, num-pages;
    name = "Book name :" + this.name + "\n";
    author = " author name :" + author.name + "\n"
    price = "Price :" + this.price + "\n";
    num-pages = " pages :" + this.num-pages + "\n";
```

Teacher's Signature :

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

public class Mainbooks
{
  public static void main (String [] a)
  {
     Scanner s = new Scanner (System.in)
     int n;
     int i;
     String name, author;
     int price;
     int num_pages;
  System.out.println ("Enter number of Books");
     n = s.nextInt ();
     Books b [];
     b = new Books [n];
  for (i=0; i<n; i++)
{
  System.out.println ("Enter details of book");
  System.out.println ("Enter name of book");
     name = s.next();
```

```
System.out.println ("Enter name of author");
    author = S.next();
S.O.P ("Enter the price");
    price = S.nextInt();
S.O.P ("Enter number of pages");
num_pages = S.nextInt();

b[i] = new Books (name, author, price, num_pages);
}

S.O.P ("Book details");
for (i=0; i<n; i++)
{
    Sop (b[i]);
}
}
}
```

Output :

```
Enter no. of Books : 2
Enter details of Book1 :
Enter name of Book : Secret
Enter author name : William
Enter the price : 350
Enter pages : 123
```

Teacher's Signature : _____

Enter details of Book 2 :

Enter name of Book : Revolution

Enter author name : Chetan

Enter price : 200

Enter no. of pages : 301

Enter details of Book 3 :

Enter name of Book : secret

Enter author name : William

Enter Price : 350

Enter No. of pages : 150

Book details :

Book name : Revolution

Book author name : Chetan

Price : 200

Pages : 704

Date 2/1/24

Expt. No. LAB - 4 Page No.

1.

```
import java.util.Scannes;

class InputScannes
{
    protected Scannes s;
    public InputScannes ()
{
    S = new Scanner (System.in));
    }

    public int getInput (String message)
{
    System.out.println (message);
    return scannes.nextInt ();
    }
}

abstract class Shape extends InputScannes {

    protected int a, b;
    public Shape()
    {
        super()
    }
    abstract public void printArea();
}
```

Teacher's Signature :

```
class Rectangle extends shape
{
    protected int a,b ;
    public Rectangle ()
    {   super ();
    }
    public void printArea ()
    {
        a = getInput ("enter the length");
        b = getInput ("enter the breadth");
        int area = a * b ;
        S.O.P ("Area of Rectangle "+area) ;
    }
}

class Triangle extends shape
{
    protected int a,b;
    public Triangle ()
    {
        super ();
    }
    public void printArea ()
    {
    a = getInput ("Enter slide1");
    b = getInput ("Enter
```

```
        double area = 0.5 * a * b;
        S.O.P ("Area of triangle" +area);
    }
}

class circle extends shape
{
    protected int a;
    public circle ()
    {
        super ();
    }

    public void printArea ()
    {
        a = getInput ("Enter the radius");
        double area = 3.14 * a * a;
        S.O.P ("Area of circle" + area);
    }
}

public class MainShape
{
    public static void main (String [] a)
    {
        Rectangle r = new Rectangle ();
        Triangle t = new Triangle ();
        Circle C = new circle ();
```

```
        r.printArea();
        t.printArea();
        c.printArea();
    }
}
```

Output : (SUFIYAN 1BM22EC261)

Enter the length : 2

Enter the breadth : 4

Area of Rectangle : 8

Enter side 1 : 2

Enter side 2 : 2

Area of triangle : 2.0

Enter radius : 5

Area of circle = 78.5

Expt. No. LAB-5 Date 9/1/24

Page No.

```
import java.util.Scanner;

class account
{
    String name;
    int accno;
    String type;
    double balance;

    account (String name, int accno, String type,
             double balance)
    {
        this.name = name;
        this.accno = accno;
        this.type = type;
        this.balance = balance;
    }

    void deposit (double amount)
    {
        balance = balance + amount;
    }

    void withdraw (double amount)
    {
        if (balance - amount) >= 0)
```

Teacher's Signature :

```
    balance -= amount;
  }
  else
  { S.O.P ("Insufficient balance, cannot
            withdraw");
  }
 }
void display()
{ S.O.P ("name:" + name + "accno:" accno +
        "type" + type + "balance" + balance);
 }
}

class savAcct extends account
{
 private static void main double rate=5;
 savAcct (String name, int accno, double balance)
{
 Super (name, accno, "savings", balance);
 {
  void interest ()
 {
  balance += balance * (rate) / 100;
  S.O.P ("balance:" + balance);
 }
}
```

```
class curAcct extends account {
    private double minBal = 500;
    private double serviceCharges = 50;
    curAcct ( String name, int accno, double balance)
{
    Supes ( name, accno, "current" balance);
    }

Void checkmain ()
{
    if (balance < minBal)
{
    S.O.P ("balance is less than min balance,
    services charges imposed " );
    balance - = service charges;
    S.O.P ("balance is : " + balance);
    }
}
}

class accountmain
{
    public static void main (String a[])
{
    Scannes S = new Scannes (System.in);
    S.O.P ("Enter the name :");
    String name = S.next ();
    S.O.P ("Enter type ( ) );
```

Teacher's Signature : _____

```
String type = S.next();
S.o.p ("Enter accno");
int accno = S.nextInt();
S.o.p ("Enter initial balance");
double balance = S.nextDouble();

int ch;
double amt1, amt2;
account acc = new account (name, accno, type,
                           balance);

SV account sa = new SavAcct (name, accno, balance);
CUS account ca = new curAcct (name, accno, balance);

while (true)
{
    if (acc.type.equals ("savings"))
    {
        S.o.p ("1. deposit \n 2. withdraw \n
                3. compute interest \n 4. display");
        S.o.p ("Enter your choice");
        switch (ch)
        {
            case 1: SOP ("enter amt");
                    amt1 = S.nextInt();
                    sa.deposit(amt);
                    break;
```

```
case 2:  SOP ("enter amount:");
         amt2 = s.nextInt();
         sa.withdraw(amt2);
         break;

case 3:  sa.interest();
         break;

case 4:  sa.display();
         break;

case 5:  System.exit(0);

default: SOP ("Invalid input");
         break;
    }
}

else
{   SOP ("1. deposit \n 2. withdraw \n 3. display");
    SOP ("enter choice");
    ch = s.nextInt();
    switch (ch)
    {
    case 1:
        SOP ("enter amnount");
        amt1 = s.nextInt();
        ca.deposit(amt1);
        break;
```

```
case 2:  sop ("Enter amount");
         amt2 = s.nextInt();
         ca.withdraw(amt2);
         ca.checkmin();
         break;

case 3:  ca.display();
         break;

case 4:  System.exit(0);

default: sop ("Invalid input");
         break;
}
}
}
}
}
```

Output: (SUFIYAN / IBM22EC261)

Enter name: John

Enter type: current

Enter accno: 1

Enter initial balance: 1000

Expt. No.

Date

Page No.

Enter name amount deposit :

500

Enter choice : 2

Enter amount : 600

1. deposit 2. withdraw 3. display

Enter the choice :

4.

16/01/24

Expt. No. LAB-6 Date 16/12/24

Page No.

1) BMSCE college
BMSCE hostel
BMSCE campus

2. String length() → 3
String literal() → abc
String contact() → Hellobrothers

3. Demonetisate toString()

→ The dimensions are 12.0 by 10.0 by 14.0
Box b: Dimensions are 12.0 by 10.0 by 14.0

4. The extracted part is : Bmsce

5
65
66
69
Welcome to BMSCE college

Teacher's Signature : _____

6) Bmsce equals Bmsce → True

Bmsce equals Bmsce → False

Bmsce equals Bmsce → False

Bmsce equals Ignore Case → True

7) Substring ()

S1 = "Bmsce college"

S2 = "welcome to BMSCE"

8) The given string starts with Bmsce : True

9) The given string ends with age : True

10) Hello equals Hello : True

Hello == Hello : False

11) The names in alphabetical are

apple
ball
cat
van
watch

12) sorted Numbers (ascending) = [1, 2, 3, 4, 5, 6, 7, 8, 9]

Expt. No. ---------- Date ----------

Page No. ----------

13 Thwas was a test . There was, too.

14 Hello world

15 Commege

16 Hello fiends

17 Enter no. of students : 2
Enter name : raj
Enter regno : 1
Enter cgpa : 9
Enter name : ram
Enter regno : 2
Enter cgpa : 8

Sorted by name:

name : raj regno : 1 sem : 0 cgpa : 9.0
name : ram regno : 2 sem : 0 cgpa : 8.0

18 Set Length : Hello
char at index 1 : e
After setCharat : Kello
getchars : Hello
After append : Hello How are you?
After insert : Hello Howcome are you?
After delete : Hello How are you?

Teacher's Signature : _____

19.) Eagle : Flying high in the sky

Eagle : Search! Search!

Hawk : Soaring through the air

Hawk : Caw! caw!

20) Circle : Area : 78.53 Perimeter : 31.42

Triangle : Area : 6.0 Perimeter : 12.0

16.01.24

Expt. No. LAB - 6 Date 23/1/24

Page No.

Package program

```
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 sc = new Scanner (System.in);
    System.out.println ("enter usn");
     usn = sc.next ();
    System.out.println ("Enter name");
     name = sc.nextLine ();
    System.out.println ("Enter sem");
     sem = sc.nextInt ();
}

public void displayStudentdetails {

S.O.P ("Usn is " + usn + "\n The name is "
        + name + "\n The sem is " + sem);
}
```

Teacher's Signature :

Expt. No. ..... Date .....

Page No. .....

```
public class Internals extends Student
{
    protected int marks[] = new int[5];

public void getmarks()
{
    Scanner s = new Scanner (System.in);
    int i;

    for (i=0; i<5 ; i++)
    {
        S.o.p ("Enter marks of subject"+(i++));
        marks [i] = S.nextInt();
    }
}
```

External.java

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class External extends Internals
{
    protected int marks[];
    protected int final marks [];
```

Teacher's Signature : _____

```
public external C) {
    marks = new int [5];
    final Marks = new int [5];
}

public void input SEE marks () {
    Scanner scanner = new Scanner (System.in);
    S.O.P ("Enter SEE marks for each course");
for (int i=0; i<5; i++)
{   finalmarks [i] = marks [i] /2 + super.internal
    Marks. [i];
    }
}

public void display final Marks ()
{
    display student details ();
    for ( int i=0 ; i<5 ; i++)
    {   S.O.P ("Subject " + (i+1) + finalMarks[i]);
    }
}
```

Main.java

```
import SEE.Externals;
class main {
public static void main (String a[])
{
    int numofStudents = 2;
Externals finalMarks[] = new Externals finalMarks;
                                [numofStudents];

for (int i=0; i<numofstudents; i++)
 {
    finalMarks[i] = new Externals();
    finalMarks[i] = inputStudentDetails();
        S.O.P("Enter CIE marks");
        S.O.P("Enter SEE marks");
    finalMarks[i] = inputSEEmarks();
}

S.O.P("Display data");
for (int i=0; i<numofstudents; i++)
{
    finalMarks[i], calculateFinalMarks[i];
    finalMarks[i]. calculate display. FinalMarks[i];
}
}
```

Expt. No. LAB-7 Date 30/1/24

Page No.

Exception

```
import java.util.Scanner;

class WrongAge extends Exception {

    WrongAge (String message) {
        super (message);
    }

class InputScanner {
    static Scanner sc = new Scanner (System.in);
}

class Father extends InputScanner
{
    int FatherAge;

Father () throws WrongAge {
    S.O.P ( "enter Father age");
     fatherAge = sc.nextInt ();
    if (fatherage < 0)
{
    throw new WrongAge ("Age cannot be neg");
}
}
```

Teacher's Signature : _____

```
void display()
{   S.O.P("Fathers age is" + fatherAge);
}

class Son extends Father {
        int sonAge;
    Son() throws WrongAge {
    S.O.P("Enter son's age");
     SonAge = sc.nextInt();
     if (sonAge > FatherAge)
{
    throw new WrongAge("Son's age cannot be
    greater than Father's");
}
    else if (sonAge < 0) {
        throw new WrongAge("Age cannot be
            negative");
    }
}
```

Date .....

Expt. No. .....

Page No. .....

```
void display()
{
    S.o.p ("Son's age is " + sonAge);
}

public class ExceptionHandling
{
    public static void main (String args[])
{
    try {
        Son son = new Son();
        Son.display();
    }

    catch (WrongAge e)
    {
        S.o.p ("Exception" + e.getMessage());
    }
}
}
```

o/p

20.01.24

Teacher's Signature : _____

Output :

Enter father's age : 10

Enter son's age : 20

Error : Fathers' age must be greater than son's age

Expt. No. LAB 8 Date 6/2/24

Page No.

Thread Program

```
class BMS extends Thread {

    public void run()
    {
        while (true) {
            System.out.println("Bms of
                Engeneering");

    try {
            Thread.sleep(1000); // 10 seconds
    }
    Catch (InterruptException e)
    {
            e.printStackTrace();
        }
    }
    }
}

Class CSE extends Thread {
    public void run()
{
    while (true)
    {
    System.out.println ("CSE")
    try {
            Thread.sleep(2000);
```

Teacher's Signature :

```
        }
        catch (InterruptException e) {
            e.printStackTrace();
        }
    }

public class ThreadExample {
    public static void main ()
    {
        BMS B1 = new BMS();
        B1.start();

        CSE C1 = new CSE();
        C1.start();
    }
}
```

Output:

```
CSE
CSE
CSE
CSE
CSE
BMS College
CSE
CSE
CSE
CSE
CSE
Bms College
```

Date 6/2/24

Expt. No. LAB 10 Page No.

Producer – Consumer

1) Incorrect:

```
class Q {
        int n;
    synchronized int get ()
    {     S.o.p (n);
        return n;
    }
     synchronised void put (int n)
    {     this.n = n;
        S.o.P ( n);
    }
}

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++);
        }
    }
}
```

Teacher's Signature :

```
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();
            i++;
        }
    }
}

class PC {
    public static void main ()
    {
        Q q = new Q ();
        new Producer (q);
        new Consumer (q);
        S.o.p ("stop");
    }
}
```

Expt. No. ..... Date .....

Page No. .....

2. correct:

```
class Q2
    int n;
    boolean valueSet = false;
    Synchronised int get()
    {
    while (!valueSet)
    {
    try {
        SOP ("consumer waiting");
    wait();
    }
    catch (InterruptedException e)
    {
    SOP ("caught");
    }

    SOP (n);
    valueSet = false;
    SOP ("Intimate Producer");
    notify();
    return n;
    }
```

Teacher's Signature : _____

```
synchronized void put (int n)
{ while (valueSet)
try {
    sop ( "Producer waiting");
 wait();
}
 catch ( InterruptedException e )
{ sop ( " caught ");
   }
  this.n = n;
  valueSet = true;
  S.op ( " Intimate costumer ");
  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<5) { q.put (i++);
```

Date

Expt. No.

Page No. 1

```
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();
        S.O.P (" " r);
        i++;
        }
    }
}

class PCFixed {
    public static void main()
{
    Q q = new Q();
    new Producer (q);
    new Consumer (q);
    System.out.println ("stop");
}
```

Teacher's Signature :

O/P :

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

06.02.24

Date .....

Expt. No. 10 .....

Page No. .....

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();
}
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)
```

Teacher's Signature : .....

```
{ 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 ("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 Thread");
  }
public static void main (String args[])
{
```

```
new Deadlock();
}
}
```

Main Thread entered.

Output :

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

Exp No - 9

Create a user interface to perform integer divisions.

```
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.setLayout(new EventClose);

        JLabel jlab = new JLabel("Enter divider, dividend");
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);

        JButton button = new JButton("Calculate");

        JLabel err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();

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

Expt. No. 9 Date 26/2/24

Page No.

```
ActionListener I = new ActionListener() {
    public void actionPerformed (ActionEvent evt)
        { System.out.println ("Action event");
    }
};      aftf.addActionListener(I);
        bjtf.addActionListener(I);
button addActionListener (new ActionsListener ()){
    public void ActionPerformed (Action event evt)
    { try {
            int a = Integer.parseInt (jtf.getText());
            int b= Integer.parseInt (btf.getText());
            int ans = a/b;
    alab.setText (a);
    blab.setText (b);
    anslab.setText (ans);
}
    Catch (NumberFormatException e) {
            alab.setText ("");
            blab.setText ("");
            anslab.setText ("");
            err.setText ("Enter only Integers");
    }
    Catch (ArithmeticException e) {
            alab.Text ("");
            blab.Text ("");
            anslab.setText ("");
            err.setText ("B should be Non zero");
        }
    }
}
```

Teacher's Signature :

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

Output:

Enter the divider and dividend

6

3

Calculate A = 6 B = 3 Ans = 2

LAB PROGRAMS (OOJ)

SUFIYAN DESAI (1BM22CS351)

//PROGRAM 1 Develop a Java program that prints all real solutions to the quadratic equation ax 2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate 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 equation");
			System.out.println("enter a non zero value for a : ");
			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("roots are real and equal");
                        System.out.println("root1 = root2 ="+r1);
                }
                else if(d>0)
                {
                        r1=((-b)+(Math.sqrt(d)))/(double)(2*a);
                        r1=((-b)-(Math.sqrt(d)))/(double)(2*a);
                        System.out.println("roots are real and distinct");
                        System.out.println("root 1 ="+r1+"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 1 ="+r1+"-i"+r2);
                }
        }
}
class quadraticMain
{
        public static void main(String args[])
        {
```

```
                quadratic q = new quadratic();
                q.getd();
                q.compute();
        }
}
```

//PROGRAM 2 Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

```
import java.util.Scanner;

class Subject
{
        int subjectMarks;
        int credits;
        String grade;
}

class Student
{
        String name;
        String usn;
        double SGPA;
        Scanner s;
        Subject subject[];
        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 your name : ");
		name = s.nextLine();
		System.out.println("enter your usn : ");
		usn = s.nextLine();
	}

	void getMarks()
	{
		int i;
		for(i=0;i<8;i++)
		{
			System.out.println("enter the marks and credits for course " + (i+1) +
":");
			System.out.println("marks : ");
			int marks = s.nextInt();
			System.out.println("credits : ");
			int credit = s.nextInt();
			subject[i].subjectMarks = marks;
			subject[i].credits = credit;

			if(marks >= 90 && marks<=100)
			{
				subject[i].grade = "O";
```

```
                }
                else if(marks>=80 && marks<90)
                {
                        subject[i].grade = "A+";
                }
                else if(marks>=70 && marks<80)
                {
                        subject[i].grade = "A";
                }
                else if(marks>=60 && marks<70)
                {
                        subject[i].grade = "B+";
                }
                else if(marks>=50 && marks<60)
                {
                        subject[i].grade = "B";
                }
                else if(marks>=40 && marks<50)
                {
                        subject[i].grade = "C";
                }
                else if(marks>=0 && marks<40)
                {
                        subject[i].grade = "F";
                }
        }
}
void computeSGPA()
{
```

```
        int i;
        double sgpa;
        double totalcredits = 0;
        double totalgradepoints = 0;

        for(i=0;i<8;i++)
        {
            totalcredits += subject[i].credits;
            switch(subject[i].grade)
            {
                case "O" : totalgradepoints += 10*subject[i].credits;
                break;
                case "A+" : totalgradepoints += 9*subject[i].credits;
                break;
                case "A" : totalgradepoints += 8*subject[i].credits;
                break;
                case "B+" : totalgradepoints += 7*subject[i].credits;
                break;
                case "B" : totalgradepoints += 6*subject[i].credits;
                break;
                case "C" : totalgradepoints += 5*subject[i].credits;
                break;
                case "F" : totalgradepoints += 0*subject[i].credits;
                break;
            }
        }
        sgpa = totalgradepoints/totalcredits;
        System.out.println("the sgpa is : "+sgpa);
}
```

```
}
class sgpa
{
	public static void main(String args[])
	{
		Student s1 = new Student();
		s1.getStudentDetails();
		s1.getMarks();
		s1.computeSGPA();
	}
}
```

//PROGRAM 3 Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to create n book objects.

```
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 s: " + 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;
    }
}

public class Mainbooks
 {
    public static void main(String[] args)
    {
        Scanner s = new Scanner(System.in);
        int n;
            int i;
        String name;
            String author;
        int price;
            int numPages;
        System.out.println("Enter the number of books:");
        n = s.nextInt();
        Books b[];
```

```
        b=new Books[n];
    for (i = 0; i < n; i++)
    {
        System.out.println("Enter the details of book" + (i+1) + ":");

      System.out.println("Enter the name of the book:");
        name = s.next();
        System.out.println("Enter the author name:");
      author = s.next();
        System.out.println("Enter the price:");
      price = s.nextInt();
        System.out.println("Enter the number of pages:");
      numPages = s.nextInt();

      b[i] = new Books(name, author, price, numPages);
    }
    System.out.println("Book details:");
    for (i = 0; i < n; i++)
    {
      System.out.println(b[i]);
    }
  }
}
```

//PROGRAM 4 Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea( ) that prints the area of the given shape.

```
import java.util.Scanner;

class inputScanner
```

```
{

	protected Scanner s;

	public inputScanner()
	{
		s = new Scanner(System.in);
	}

	public int getInput(String message)
	{
		System.out.println(message);
		return s.nextInt();
	}



}

abstract class Shape extends inputScanner
{
	protected int a,b;

	public Shape()
	{
		super();
	}

	abstract public void printArea();
}

class Rectangle extends Shape
```

```
{
	protected int a,b;
	public Rectangle()
	{
		super();
	}

	public void printArea()
	{

		a=getInput("Enter the length:");
		b=getInput("Enter the bredth:");
		int area= a*b;
		System.out.println("Area of the Rectangle:" +area);
	}
}

class Triangle extends Shape
{
	protected int a,b;
	public Triangle()
	{
		super();
	}

	public void printArea()
	{
		a=getInput("Enter the side1:");
		b=getInput("Enter the side2:");
		double area=0.5*a*b;
		System.out.println("Area of the Triangle:" +area);
```

```
	}

}

class Circle extends Shape
{
	protected int a;
	public Circle()
	{
		super();
	}

	public void printArea()
	{
		a=getInput("Enter the radius:");
		double area=3.14*a*a;
		System.out.println("Area of the Circle:" +area);

	}

}

public class MainShape
{
	public static void main(String[] args)
	{
		Rectangle r=new Rectangle();
		Triangle t=new Triangle();
		Circle c=new Circle();

		r.printArea();
```

```
            t.printArea();
            c.printArea();
        }
}
```

//PROGRAM 5 Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest.

Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

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

a) Accept deposit from customer and update the balance.

b) Display the balance.

c) Compute and deposit interest

d) Permit withdrawal and update the balance Check for the minimum balance, impose penalty if necessary and update the balance.

```
import java.util.Scanner;
class account
{
        String name;
        int accno;
        String type;
        double balance;

        account(String name,int accno,String type,double balance)
        {
            this.name=name;
            this.accno=accno;
            this.type=type;
```

```
		this.balance=balance;
	}
	void deposit(double amount)
	{
		balance+=amount;
	}
	void withdraw(double amount)
	{
		if((balance-amount)>=0)
		{
			balance-=amount;
		}
		else
		{
			System.out.println("insufficient balance,cant withdraw");
		}
	}

	void display()
	{

	System.out.println("name:"+name+"accno:"+accno+"type:"+type+"balance:"+balanc
e);
	}
}
class savAcct extends account
{

	private static double rate=5;
	savAcct(String name,int accno,double balance)
```

```
	{
		super(name,accno,"savings",balance);

	}

	void interest()
	{
		balance+=balance*(rate)/100;
		System.out.println("balance:"+balance);
	}


}
class curAcct extends account
{

	private double minBal=500;
	private double serviceCharges=50;

	curAcct(String name,int accno,double balance)
	{
		super(name,accno,"current",balance);

	}


	void checkmin()
	{
```

```
                if(balance<minBal)
                {
                        System.out.println("balance is less than min balance,service charges imposed:"+serviceCharges);
                        balance-=serviceCharges;
                        System.out.println("balance is:"+balance);
                }

        }

}
class accountMain
{
        public static void main(String a[])
        {
                Scanner s=new Scanner(System.in);
                System.out.println("enter the name :");
                String name=s.next();
                System.out.println("enter the type(current/savings):");
                String type=s.next();
                System.out.println("enter the account number:");
                int accno=s.nextInt();
                System.out.println("enter the intial balance:");
                double balance=s.nextDouble();
                int ch;
                double amount1,amount2;
                account acc=new account(name,accno,type,balance);
                savAcct sa=new savAcct(name,accno,balance);
                curAcct ca=new curAcct(name,accno,balance);
                while(true)
```

```
                {
                        if(acc.type.equals("savings"))
                        {
                                System.out.println("\nMenu\n1.deposit 2.withdraw
3.compute interest 4.display");
                                System.out.println("enter the choice:");
                                ch=s.nextInt();
                                switch(ch)
                                {
                                        case 1:System.out.println("enter the amount:");
                                                amount1=s.nextInt();
                                                sa.deposit(amount1);
                                                break;
                                        case 2:System.out.println("enter the amount:");
                                                amount2=s.nextInt();
                                                sa.withdraw(amount2);
                                                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\n1.deposit 2.withdraw
3.display");
```

```
                        System.out.println("enter the choice:");
                        ch=s.nextInt();
                        switch(ch)
                        {
                                case 1:System.out.println("enter the amount:");
                                        amount1=s.nextInt();
                                        ca.deposit(amount1);
                                        break;
                                case 2:System.out.println("enter the amount:");
                                        amount2=s.nextInt();
                                        ca.withdraw(amount2);
                                        ca.checkmin();
                                        break;

                                case 3:ca.display();
                                        break;
                                case 4:System.exit(0);
                                default:System.out.println("invalid input");
                                        break;
                        }
                }
        }
    }
}
```

//PROGRAM 6 Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of 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 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.print("Enter USN: ");
        usn = s.next();
        System.out.print("Enter Name: ");
        name = s.next();
        System.out.print("Enter Semester: ");
        sem = s.nextInt();
    }

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

```
package CIE;

import java.util.Scanner;

public class Internals extends Student
{
   protected int marks[] = new int[5];

   public Internals() {}

   public void inputCIEmarks()
           {
      Scanner s = new Scanner(System.in);
      System.out.println("Enter Internal Marks for " + name);
      for (int i = 0; i < 5; i++)
                      {
         System.out.print("Subject " + (i + 1) + " marks: ");
         marks[i] = s.nextInt();
      }
   }
}
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);
        System.out.println("Enter SEE Marks for " + name);
        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]);
    }
}
import SEE.Externals;

public 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();
```

```
                }
        }
}
```

//PROGRAM 7 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 message)
 {
  super(message);
 }
}

class InputScanner
{
 protected Scanner s;
 public InputScanner()
 {
  s = new Scanner(System.in);
 }
}

class Father extends InputScanner
```

```
{
 protected int fatherAge;
 public Father() throws WrongAge
 {
  System.out.println("Enter Father's Age:");
  fatherAge=s.nextInt();

  if(fatherAge<0)
  {
   throw new WrongAge("Age cannot be negetive:");
  }
 }

 public void display()
 {
  System.out.println("Father's Age:" + fatherAge);
 }

}


class Son extends Father
{
 private int sonAge;

 public Son() throws WrongAge
 {
  super();
  System.out.println("Enter Son's age:");
```

```
 sonAge=s.nextInt();



 if(sonAge>fatherAge)
 {
  throw new WrongAge("Son's age cannot be greater than father's age");
 }
 else if (sonAge<0)
 {
  throw new WrongAge("Age cannot be negative");
 }
}

 public void display()
 {
  super.display();
  System.out.println("Son's Age: " + sonAge);
 }

}



public class FatherSonAge
{
 public static void main(String args[])
 {
  try
  {
   Son son=new Son();
```

```
  son.display();
 }

 catch (WrongAge e)
 {
  System.out.println("Error: " + e.getMessage());
 }
}

}
```

//PROGRAM 8 Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displaying "CSE" once every two seconds.

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

```
        }
}

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


public class ThreadExample
{
        public static void main(String[] args)
        {
                BMSThread bmsThread=new BMSThread();
                bmsThread.start();
```

```
            CSEThread cseThread=new CSEThread();
            cseThread.start();
        }
}
```

//PROGRAM 9 Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

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

class UserInterface {
   UserInterface() {
      // 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 dividend:");

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

ActionListener calculateListener = new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
      try {
         int a = Integer.parseInt(ajtf.getText());
         int b = Integer.parseInt(bjtf.getText());
         if (b == 0) {
            throw new ArithmeticException();
         }
         int ans = a / b;
```

```
            alab.setText("\nA = " + a);
            blab.setText("\nB = " + b);
            anslab.setText("\nAns = " + ans);
            err.setText(""); // Clear any previous error message
        } catch (NumberFormatException e) {
            displayErrorMessage("Enter Only Integers!");
        } catch (ArithmeticException e) {
            displayErrorMessage("B should be non-zero!");
        }
    }

    private void displayErrorMessage(String message) {
        alab.setText("");
        blab.setText("");
        anslab.setText("");
        err.setText(message);
    }
  };

  button.addActionListener(calculateListener);

  // 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 UserInterface();
        }
      });
   }
}
```

//PROGRAM 10 (Part 1) Demonstrate Inter process Communication

```
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<5)
		{
			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.");
        }
}
```

//PROGRAM 10 (Part 2) Demonstrate 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 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 other thread.
                System.out.println("Back in other thread");
```

```
        }

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

}
```

END OF REPORT