INDEX

Name Harshvardhan Class 3E

Roll No. Subject Java. College

Sl. No.	Date	Title	Page No.	Teacher Sign / Remarks
1.	12/12	Quadratic Eqn	10	12/12/23
2.	18/12	To find SGPA	10	19/12/23
3.	26/12	n books objects	10	26/12/23
4	2/1/24	Print area of given shape		2-1-2024
5.	02/01/24	LAB-4 : Abstract class		6-2-24
6.	16/1/24	LAB-5 : Bank A/c	10	
7.	23/01/24	LAB-6 Strings		
8.	30/1/24	LAB-7 : Exception		
8.	06/2/24	LAB-8 : Threads		
10.	13/2/24	Deadlocks	10	20/2/2024
9.	20/2/24	Program on event handlers		

12/12/23

LAB-Pg 2

Quadratic Eqn:-

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 nega display a message that there are no real solution

```
importjava.util.Scanner;

class quadratic
{
    int a,b,c;
    double r1,r2,d;
    void getd()
    {
        Scanner s=new Scanner(System.in);
        System.out.println("enter the coefficients of a,b,c
        a=s.nextInt()
        b=s.nextInt()
        c=s.nextInt()

    }
    void compute()
    {
        while(a==0)
        {
            System.out.println("Not a quadratic eqn");
```

```
System.out.println("Enter a non zero 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);
r2 = ((-b) - (math.sqrt(d)))/(double)(2*a);
System.out.println("Roots are real and distinct");
System.out.println("Root =" + "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);
}
```

```
}

}

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

Output

Enter the coefficients of a, b, c

1

2

1

roots are real and equal.

Roots 1 = Roots 2 = -1.0

Enter the coefficients of a, b, c

1

4

1

roots are real & distinct

Roots - 1 = 0.267949 Roots 2 = -3.7320508

12/12/2023

19/12/23

1

LAB-3

//pgm to find USN, SGPA ...

develop a Java Pgm to create a class student with members usn, name, an array credits and array marks

```
import java.until.Scanner;
class subject
{
    int subjectMarks;
    int grade;
    int marks;
}
class subject
{
    Subject subject[];
    String name;
    String usn;
    Scanner s;
    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.print("Enter your name:");
    name = s.next();
    System.out.print("Enter your USN :");
    usn = s.next();
}
void getmarks()
{
    for (i=0 ; i<9 ; i++)
    {
        System.out.print("Enter marks for subject" + (i+1)
                + " :");
        subject[i].subjectmarks = s.nextint();
        System.out.print("Enter credits for subject" + (i+1) + " :
        ");
        subject[i].credits = s.nextint();
        subject[i].grade = (subject[i].subjectmarks/10) + i;
        if(subject[i].grade == 11)
            subject[i].grade = 10;
        if (subject[i].grade <= 4)
            subject[i].grade = 0;
```

```
    }
}
void computeSGPA()
{
    int effectivescore=0;
    int totalcredits=0;
    for(int i=0;i<9;i++)
    {
        effectivescore+=(subject[i].grade*subject[i].credi
        totalcredits+=subject[i].credits;
    }
    SGPA=(double)effectivescore/(double)totalcredits
}
}
class main
{
    public static void main(String args[])
    {
        Student s1=new Student();
        s1.getStudentDetails();
        s1.getmarks();
        s1.computeSGPA();
```

```
System.out.println("name: " + s1.name);
System.out.println("USN: " + s1.usn);
System.out.println("SGPA: " + s1.SGPA);
```

Output:

Enter your name : Harsh

Enter your USN: 1BM22CS248.

Enter marks for subject1 : 80

Enter ~~marks~~ credits for subject1 : 5

Enter marks for subject 2 : 70

Enter marks for subject 2 : 6

Enter marks for sub 3 : 50

Enter marks for subject 4 : 5

Enter marks for subject 4 : 90

Enter marks for subject 5 : 9

Enter marks for subject 5 : 88

Enter marks for subject 6 : 7

Enter marks for subject 6 : 6.5

Enter marks for subject 7 : 70

Enter marks for subject 7 : 6

Enter marks for subject 8 : 80

Enter marks for subject-8 : 8

Enter marks for subject - 9 : 50

Enter marks for subject - 9 : 4.

Name : Harsh

USN : 1BM22CS249

SGPA - 8.272

19/12/23

LAB-4 26/12/23 Print Area of given Shape.

Create a class book which contains members : name, author, price, num-pages. Include a constructor to set the values for the values for members. Include methods to set and get the details of objects. Include String() method that could display complete details of book.

Program :-

```
import java.util.Scanner;

class Books{
    String name, author;
    int price, 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;
}

}

class BookMain {
    public static void main (String[] args) {
        Scanner sc = new Scanner(System.in);
        int n;
        String name, author;
        int price, numpages;
        System.out.println("Enter the number of books
        int n;
        String name, author;
        int price, numPages;
        n = sc.nextInt();
        Books b[] = new Books[n];
        System.out.println("Enter Name, author, pri
            number of pages:");
        for (int i=0 ; i<n ; i++) {
            name = sc.nen
            author = sc.ne
            price = sc.ner
            numPages = sc
            b[i] = new Bo
        }
        System.out.pri
        for (int i=0 i
            System.o
        }
    }
}
```

Output

```
Enter number o
1
Enter name, au
BMSCE
Harsh
99
199
Books details:
Book names:
```

```
name = sc.next();
author = sc.next();
price = sc.next();
numPages = sc.nextInt();
b[i] = new Books (name, author, price, numPages);

}
System.out.println("Books details:");
for (int i=0 ; i<n ; i++) {
    System.out.println(b[i].toString());

}

}

}
```

Output

```
Enter number of books:
1
Enter name, author, price, number of pages:
BMSCE
Harsh
99
199
Books details:
Book names: BMSCE
```

Author name : Harsh

Price : 99

Number of pages : 199.

26/12/23

LAB-5

• Write a pg

```
import java.util.*;
class InputScanner
    Scanner sc;
    InputScanner
        sc = n
    }
}
abstract
    da
    d
    a
}
class
```

LAB-5

- Write a pgm to form abstract class for Rectangle.

```
import java.util*;
class InputScanner {
    Scanner sc;
    InputScanner() {
        sc = new Scanner(System.in);
    }
}
abstract class Shape extends InputScanner {
    double a;
    double b;
    abstract void getInput;
    abstract void displayArea;
}
class Rectangle extends Shape {
    void getInput() {
        System.out.println("Enter the length and breadth:");
        a = sc.nextdouble();
        b = sc.nextDouble();
    }
    void displayArea() {
        System.out.println("Area of rectangle is:" + (a*b));
    }
}
```

LAB - 9

1. Write a pgm to find area of rectangle

```
import java.io.*;
class InputStream {
    Scanner sc;
    InputStream {
        sc = new Scanner(System.in);
    }
}
class InputStreamShape implements {
    double a;
    double b;
    abstract void getInput();
    abstract void displayArea();
}
class Rectangle extends InputStreamShape {
    void getInput() {
        System.out.println("Enter the length and breadth");
        a = sc.nextDouble();
        b = sc.nextDouble();
    }
    void displayArea() {
        System.out.println("Area of rectangle is " + (a*b))
    }
}
```

```
class Triangle extends shape {
    void getInput() {
        System.out.println("Enter the length and height:");
            a = sc.nextDouble();
            b = sc.nextDouble();
    }
    void displayArea() {
        System.out.println("Area of rectangle
                is:" + (a* a* 3.14));
    }
}

class ShapeMain {
    public static void main (String[] args) {
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
        r.getInput()
        r.displayArea()
        t.displayArea()
        c.getInput();
        c.displayArea();
    }
}
```

Output

Enter length and breadth

78

43

Area of rectangle

Enter length and

43

323

Area of rectan

Que

2-1-

Output

Enter length and breadth:

78
43

Area of rectangle is : 3354.0

Enter length and breadth

43
323

Area of rectangle is : 6944.5

2-1-23

Extra

(LAB-6)

- Develop a Java Pgm to create a class Bank that maintains two kind of account for its customers, one called saving account provides compound intrest and withdrawal facilities but no cheque facility. The current account provides cheque book facility but no intrest. Current account holders should go mantain a minimum balance and if the balance falls below this level, service charge is imposed

```
import java.util.Scanner;
class account
{
  String name;
  int acc no;
  String type;
  double balance;
  account(String name, int acc no, 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 amoun
  {
      if(balance-amount)>=0
      {
          balance -=amoun
      }
      else
      {
          System.out.println
      }
  }
  void display()
  {
      System.out.printlr
  }
}
void display()
{
    System.out.print(
      acno+" "+"t
}
class sav Acct extend
{
```

```
{
    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("Insufficient balance, cant withdraw");
}
}
void display()
{
    System.out.println(" name :" +name+" +accno :" +
    accno+" " + "type:" balance :" +balance);
}
class SavAcct extends account
{
```

```
private static double rate=5;
SavAcct(String name, int accno, double balance)
{
    super(name, accno, "savings", balance);
}
void intrest()
{
    balance += balance*(rate)/100;
    System.out.println("balance:" +balance);
}

class CurAcct extends account
{
    private double minBal =500;
    private double serivecharges = 50;

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

Void checkin()
{
    if(balance<minBal)
```

```
{
    System.out.println("balance is less than min
        balance, service charges imposed :" + service
        charges);

    balance = service charges;
    System.out.println("balance is:" +balance);
  }
}

class accountmain
{
  public static void main(String a[])
  {
    Scanner s=new Scanner(System.in);
    System.out.println("enter name:"
```

O/P

6-2-24

LAB-7

write to pgm to dm.

9

```
import java.util.*;

class WrongAge extends Exception {
    public WrongAge(String s) {
        super(s);
    }
}

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

class Father extends InputScanner {
    int FatherAge;
    public Father() throws WrongAge {
        InputScanner ss = new InputScanner();
        System.out.println("Enter age :")
        fatherage = Scanner.nextInt();
```

```
if (fatherAge <0) {
    throw new wrongAge ("Age cannot be -ve");
}
}

class Son extends Father {
    private int sonAge;

    public Son() throwsn wrongAge {
        ~~Age~~ super();

        System.out.println("Enter son's age");

        sonAge = scanner.nextInt();

        if (sonAge >= fatherAge) {
            new wrongAge ("Son's age cannot be
            greater than father is age");
        }
        else if (son <0) {
            throw new wrongAge ("age cannot be -ve");
```

```
public class ExceptionHandling
    public static void main
    (String [] args) {
        try {
            Son son = new Son
            son.display();
        }
        catch (WrongAge e) {
            System.out.print("Except (1);
        }
    }
}
```

output

Enter Father age :

50

Son age :

24

6/2/24

LAB-8

```
//Write pgm wh
thread displayin
way 10sec a
class Display me

    private St
    private

public Dis
    this
    Thre

}

publi
w
```

6/2/24

LAB - 8

//Write pgm which creates two threads, one thread displaying "BMS College of Engineering" once every 10 sec and another displaying "CSE"

```
class DisplayMessageThread Extend Thread {

    private String message;
    private int interval;

    public DisplayMessageThread (String message, int interval) {
        this.message = message;
        this.interval = interval;
    }
    public void run(){
    while (True) {
        System.out.println(message);
        try {
            Thread.sleep(interval*1000);
        } catch(InterruptedException e) {
            e.printStackTrace()
        }
    }
}
```

```
public class ThreadExample {
    public static void main (String [] arg) {
    DisplayMessageThread thread1 =
    new DisplayThread ("BMS college")

    DisplayMessageThread thread2 = new
    DisplayMessageThread ("CSE", 2);

    thread1.start();
    thread2.start();
    }
}
```

Output

```
BMS college of Engineering
CSE
CSE
CSE
```

LAB-10 13/2/24

Q) Implementation of

```
class Q {
int n;
boolean valueset = f
synchronised int
while (!value set)
try {
System.out.priten (
wait ();
} catch (Interrupted
System.out.pri
}
System.out.println
Value set = false
System.out.prit
notify ();
return n;
}
synchronized voi
(value set)
try{
```

LAB-10 13/2/24

Q) Impumentation of produces and consumer

```
class Q {
int n ;
boolean valueset = false ;
synchronised int. get() {
while (!value set)
try {
System.out.priten(" In Consumer waiting \n") ;

wait () ;
} catch (Interrupted Exception e) {
System. out. pritln (" Interrupted Exception caught") ;
}
System .out. pritln (" Crt:" +n) ;
Value set = false
System. out. pritln (" In Intimate procuducer \n") ;

notify ();
return n ;
}
synchronized void put (int n) {
(value set)
try {
```

System.out.println("In Producer waiting \n");

wait();

} catch (InterruptedException e)

System.out.println("Interrupted Exception caught");

}

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;

```
}
}
public
Q q =
new Pro
new Co
System.o
}
}
```

o/p

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

```
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++;

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

Q q=new Q();

new Producer(q);

new Consumer(q);

System.out.println("Press Control-c to p");
}
}
```

O/P

```
Put: 1
Put: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
```

```
Got: 4
```

Deadlocks

```
class A {

synchronized void (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();
}

class B {
synchronised void bar (A a) {
String name = Thread.currentThread()
    getname();
System.out.println(name + "entered B.bar");
try {
Thread.sleep(1000);
} catch (Exception e) {
```

```
}
class D {
}
Aa = n
Bb = n
Deadlock(
Thread t
}
public voi
to bar(a);
System.out
}
new Deadlo
}
}
}
```

```
System.out.println("B interrupted");
}
a last();
}
void last() {
System.out.println("Inside A last");
}
}
class Deadlock implements Runnable
{
A a = new A();
B b = new B();
Deadlock() {
Thread t = new Thread(this, "Racing Thread");
}
public void run() {
to barla();
System.out.println("Back in other thread");
}
new Deadlock();
}
}
```

O/P

Main Thread entered A.foo

main Thread trying to call B.last()

Inside A'last

Back in main thread

Racing Thread trying to call A.last()

Inside A.last

Back in other thread.

imp

public

Act

{

public

{

LAB-9 20/2/24

WAP that creates a user interface to perform integer division. User enters two numbers in text field num1 & num2. Division of Num1 & Num2 is displayed in Result field when Divide button is clicked if Num1 or Num2 were not an integer.

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

public class DivisionMain1 extends Frame implements
  ActionListener
{
    TextField num1, num2;
    Button dResult;
    Label outResult;
    String out="";
    double resultNum;
    int flag=0;

    public DivisionMain1 ()
    {
      setLayout (new FlowLayout ());
      dResult =new Button ("RESULT");
      Label number1 = new Label ("Number 1:", Label.Right);
      Label number2 = new Label ("Number 2, Label.RIGHT);
```

```
num1 = new TextField(5);
num2 = new TextField(5);
outResult = new Label("Result :")

add(number1);
add(num1);
add(number2);
add(num2);
add(Result);

num1.addActionListener(this);
num2.addActionListener(this);

{
    public void windowClosing(WindowEvent we)
    {
        System.exit(0)
    }
}
};
}
public void actionperformed(ActionEvent ae)
{
    int n1,n2;
    try
    {
        if (ae.getSource() == dResult()
```

```
{
    n1 = Integer.PresentInt (num1.getText());
    n2 = Integer.PresentInt (num1.getText());

    out = n1 + " " + n2;
    ResultNum = n1/n2;
    out + = String.valueOf (ResultNum);
    repaint();

  }
}
catch (ArithmeticException e)
{
    flag = 1
    out = "Divide by 0 Exception!" + e;
    repaint();

  }

}
public void paint (Graphics g)
{
    if (flag==0)
    g.draw (out, outResult.getX() + outResult.get
        Width(), outResult.getY());
    else
    g.drawString (Out, 100, 200);
    flag = 0;
}
```

```
public static void main (String [] args)
{
    Division main1 dm = new Division main();
    dm.setTitle("Division of integers");
    dm.setVisible(true);
}
```

O/P

Number 1: 6 Number 2: 4

Result 6 41.0

20/2/2024

LAB PROGRAM 1:

```
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(&quot;Enter the coefficients of a,b,c&quot;);
a = s.nextInt();
b = s.nextInt();
c = s.nextInt();
}
void compute()
{
while(a==0)
{
System.out.println(&quot;Not a quadratic equation&quot;);
System.out.println(&quot;Enter a non zero value for a:&quot;);
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(&quot;Roots are real and equal&quot;);
System.out.println(&quot;Roo1 = Root2 = &quot; + r1);
}
```

```
else if(d&gt;0)
{
r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a);
System.out.println(&quot;Roots are real and distinct&quot;);
System.out.println(&quot;Roo1 = &quot; + r1 + &quot; Root2 = &quot; + r2);
}
else if(d&lt;0)
{
System.out.println(&quot;Roots are imaginary&quot;);
r1 = (-b)/(2*a);
r2 = Math.sqrt(-d)/(2*a);
System.out.println(&quot;Root1 = &quot; + r1 + &quot; + i&quot;+r2);
System.out.println(&quot;Root1 = &quot; + r1 + &quot; - i&quot;+r2);
}

}
}

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

q.compute();
}
}
```

LAB PROGRAM 2:

```
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();
        }
}
```

LAB PROGRAM 3:

```
import java.util.Scanner;

class Books
```

```
{
        String name;
        String author;
        int price;
        int numPages;

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

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

public class Mainbook
{
        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]);
```

```
                }
        }
}
```

LAB PROGRAM 4:

```
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);
                System.out.println();
        }
}

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

```
        {
                super();
        }

        public void printArea()
        {
                a=getInput("Enter the side 1: ");
                b=getInput("Enter the side 2: ");
                double area=0.5*a*b;
                System.out.println("Area of the Triangle: " +area);
                System.out.println();
        }

}

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);
                System.out.println();

        }
```

```
}

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();
	}
}
```

LAB PROGRAM 5:

```
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:"+balance);
        }
}
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;
            }
        }
    }
  }
}
```

LAB PROGRAM 6:

```
//Student.java
package CIE;
import java.util*;
public class Student
{
   protected string usn=new string();
   protected string name= new string();
   protected int

   public void input StudentDetails()
   {
```

```
        Scanner sc=new scanner(system.in);
        System.out.println("Enter student USN");
        usn=sc.next();
        System.out.println("Enter student name");
        name = sc.next();
        System.out.println("Enter semester");
        Scanner=sc.nextInt();
    }
    public void display StudentDetails()
    {
        System.out.println("Student usn:"+usn);
        System.out.println("Student name:"+name);
        System.out.println("Student semester:"+sum);
    }
}
//Internals.java
package CIE;
import java.util*
public class Internals extends student
{
    protected int marks[]=new int[5];
    public void input CIE marks()
    {
        scanner sc=new scanner(system.in);
        for(int i=0;i<5;i++)
        {
            System.out.println("Enter 5 subject marks");
            marks[i]=sc.nextInt();
        }
    }
}
```

```
//External.java
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends Internals
{
   protected int marks[];
   protected int final marls[];
   public Externals()
   {
      marks=new int[5];
      final marks = new int[5];
   }
   public void input SEE marks()
   {
      Scanner sc=new scanner(System.in);
      for(int i=0;i<5;i++)
      {
         System.out.println("Subject"+(i+1)+"marks:");
         marks[i]=sc.nextInt();
      }
   }
   public void Calculate Final Marks[]
   {
      for(int i=0;i<5;i++)
      {
         final marks[i]=Marks[i]/2+super.marks[i];
      }
   }
   public void display FinalMarks()
   {
```

```
        display StudentDetails();
        for(int i=0;i<5;i++)
        {
            System.out.println("Subject"+(i+1)+"+final Marks[i]");
        }
    }
}
//Main.java
import SEE.Externals
class Main
{
    public static void main(string args[])
    {
        int num of students=2;
        Externals final Marks[]=new Externals[num of students];
        for(int i=0;i<num of students;i++)
        {
            final marks[i]=newExternals();
            final marks[i].input student Details();
            System.out.println("Enter CIE marks");
            final marks[i].input CIE marks();
            System.out.println("Enter SEE marks");
            final marks[i].input SEE marks();
        }
        System.out.println("Displaying data:/n");
        for(int i=0;i<num of students;i++)
        {
            final Marks[i].Calculate FinalMarks();
            final Marks[i].display FinalMarks();
        }
    }
```

```
}
```

LAB PROGRAM 7:

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

LAB PROGRAM 8:

```
class BMSThread extends Thread {
```

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

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

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

LAB PROGRAM 9:

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

public class DivisionMain1 extends Frame implements ActionListener
{
        TextField num1,num2;
        Button dResult;
        Label outResult;
        String out="";
        double resultNum;
        int flag=0;

        public DivisionMain1()
        {
                setLayout(new FlowLayout());

                dResult = new Button("RESULT");
                Label number1 = new Label("Number 1:",Label.RIGHT);
                Label number2 = new Label("Number 2:",Label.RIGHT);
                num1=new TextField(5);
                num2=new TextField(5);
                outResult = new Label("Result:",Label.RIGHT);

                add(number1);
                add(num1);
```

```
        add(number2);
        add(num2);
        add(dResult);
        add(outResult);

        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new WindowAdapter()
        {
                public void windowClosing(WindowEvent we)
                {
                        System.exit(0);
                }
        });
}

public void actionPerformed(ActionEvent ae)
{
        int n1,n2;
        try
        {
                if (ae.getSource() == dResult)
                {
                        n1=Integer.parseInt(num1.getText());
                        n2=Integer.parseInt(num2.getText());

                        /*if(n2==0)
                                throw new ArithmeticException();*/
                        out=n1+" "+n2;
                        resultNum=n1/n2;
```

```
                    out+=String.valueOf(resultNum);
                    repaint();

                }
            }
            catch(NumberFormatException e1)
            {
                flag=1;
                out="Number Format Exception!  "+e1;
                repaint();
            }
            catch(ArithmeticException e2)
            {
                flag=1;
                out="Divide by 0 Exception!  "+e2;
                repaint();
            }

        }

        public void paint(Graphics g)
        {
            if(flag==0)

        g.drawString(out,outResult.getX()+outResult.getWidth(),outResult.getY()+outResult.getHeig
ht()-8);
            else
            g.drawString(out,100,200);
            flag=0;
        }

        public static void main(String[] args)
```

```
        {
                DivisionMain1 dm=new DivisionMain1();
                dm.setSize(new Dimension(800,400));
                dm.setTitle("DivionOfIntegers");
                dm.setVisible(true);
        }

}
```

LAB PROGRAM 10:

```
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<2) {
        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.");
        }
}
```

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();
        }
}
```