B.M.S COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

OBJECT ORIENTED JAVA PROGRAMMING

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:

Sarthak Gupta
1BM22CS246

Department of Computer Science and Engineering
B.M.S College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019

INDEX

NAME: Sarthak Gupta STD: SEC: ROLL NO: 1BM22CS246

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
1		LAB Prog 1 (quadratic)	10	
2		LAB Prog 2 (calculate SGPA)	10	
3.		LAB Prog 3 (create objects of class)	10	2/1/2024
4.		LAB Prog 4 (Abstract class)	10	
5		LAB Prog 5 (Create a class Bank that maintains two kinds of accounts)	10	
6.		LAB Prog 65 (Programs on Strings)		
7.		LAB Prog 6 (Program of packages)	10	30/1/2024
8.		LAB Prog 7 (Program of Exception	10	
9.		LAB Prog 8 (Program on multithreads)	10	6/2/2024
10.		LAB Prog 10.1 (Program on Inter communication)		13/2/2024
11.		LAB Prog 10.2 (Program on Deadlock)		
12		LAB Prog 9 (Program on Exceptionhandlers)		20/2/2024

LAB PROGRAM - 1

Q. Develop a Java program that prints all real solutions to the quadratic equation $ax^2 + bx + c = 0$. Reading 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)
        {
        r2 = ((-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 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();
    }
}
```

Lab Program-2

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

→
```
import java.util.*;
class Subject
{
    int subjectMarks;
    int credits;
    int grade;
}
class Student {
    Subject subject[];
    String name;
    String usn;
    double gpa;
    Scanner s;
    Student()
    {
        int i;
        subject = new Subject[9];
        for(i=0; i<9; i++)
        subject[i] = new Subject();
        s = new Scanner(System.in);
    }
    void getStudentDetails() {
        System.out.println("Enter student name");
        name = s.next();
        System.out.println("Enter student USN");
        usn = s.next();
    }
```

```
void getMarks() {
    for (int i=0; i<9; i++) {
        System.out.println("Enter marks");
        subject[i].subjectMarks = s.nextInt();
        System.out.println("Enter number of credits");
        subject[i].credits = s.nextInt();
        subject[i].grade = (subject[i].subjectMarks/10)+1;
        if (subject[i].grade == 11)
        subject[i].grade = 10;
        if (subject[i].grade <= 0)
        subject[i].grade = 0;
    }
}

Void computeSGPA() {
    int effectiveScores = 0;
    int totalCredits = 0;
    for (int i=0; i<9; i++) {
        effectiveScores += subject[i].grade * subject[i].credits;
        totalCredits += subject[i].credits;
    }
sgpa = (double) effectiveScores / (double) totalCredits;
}
}

class Main {
    public static void main (String args[]) {
        Student s1 = new Student();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();
        System.out.println("student name:" + s1.name);
        System.out.println("student usn:" + s1.usn);
```

System.out.println("Student SGPA: "+s1.sgpa);
}
}

OUTPUT

Enter Student name
Sarthak
Enter Student USN
1BH22CS246
Enter marks 80
Enter number of credits 4
Enter marks 70
Enter number of credits 3
Enter marks 86
Enter number of credits 4
Enter marks 80
Enter number of credits 4
Enter marks 90
Enter number of credits 3
Enter marks 86
Enter number of credits 4
Enter marks 80
Enter number of credits 4
Enter marks 90
Enter number of credits 4

Student name : Sarthak
Student USN = 1BH22CS246
Student SGPA : 9.033

Lab Program 3

Q. 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: " + this.name + "\n";
        author = "Author name: " + this.author + "\n";
        return name + author + price + numPages;
    }
}
Class BooksMain {
    public static void main(String args[])
    {
```

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

11/2024

LAB 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 to contain only the method printArea() that prints the area of the given Shape.

-
```
import java.util.Scanner;
class InputScanner{
    Scanner s;
    InputScanner(){
        s= 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 width of
                the rectangle");
        a = s.nextDouble();
        b = s.nextDouble();
    }
```

```
    void displayArea()
    {
        System.out.println("The area of the rectangle
                is" + (a*b));
    }
}
class Triangle extends Shape
{
    void getInput()
    {
     System.out.println("Enter the base and the height of
             the triangle");
     a = S.nextDouble();
     b=S.nextDouble();
    }
    void displayArea()
    {
        System.out.println("The area of the triangle is:"+(a*b*
    }
}
class Circle extends Shape
{
        void getInput(){
        System.out.println("Enter the value of the radius");
        a = S.nextDouble();
        }
    void displayArea(){
    System.out.println("The area of the triangle is:"
                    +(3.14*a*a));
    }
}
```

```
class AbstractAreaMain{
    public static void main(String args[])
    {
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
        r.getInput();
        r.displayArea();
        t.getInput();
        t.displayArea();
        c.getInput();
        c.displayArea();
        System.out.println("Name: Sarthak Gupta");
        System.out.println("USN: 1BH22CS246");
    }
}
```

OUTPUT

```
Enter the length and width of the rectangle
5
4
The area of the rectangle is 20.0
Enter the base and the height of the triangle
10
15
The area of the triangle is: 75.0
Enter the value of the radius
3
The area of the triangle is: 28.2599
Name: Sarthak Gupta
USN: 1BH22CS246
```

2/1/2024

LAB 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 customers 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 deposite interest
(d) Permit withdrawal and update the balance
(e) 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:"+balance);
}
}
class savAcct extends account
{
    private static double rate = 5;
    savAcct(String name, int accno, double balance)
    {
```

```
        super(name, accno, "Savings", balance);
    }

    void insert()
    {
    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 AccountsMain
{
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 initial 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("\n Menu \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;
        }
    }
}
```

O/P?

16/5/2024

LAB - 6

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

→ // inside cie Student.java

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

    Public void importStudentDetails()
    {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter the name:");
        name = s.next();
```

```
        System.out.println("Enter the USN: ");
        usn = s.next();
        System.out.println("Enter the semester: ");
        sem = s.nextInt();
    }
    Public void displayStudentDetails()
    {
        System.out.println("Name: "+name);
        System.out.println("USN: "+usn);
        System.out.println("Sem: "+sem);
    }
}
// Internals.java inside cie
Package cie;
import java.util.Scanner;
Public class Internals extends Students
{
    Protected int marks[] = new int[5];
    Public void inputCIEmarks()
    {
        Scanner s = new Scanner(System.in);
        for(int i=0; i<5; i++)
        {
            System.out.println("Enter subject wise marks "+(i+1));
            marks[i] = s.nextInt();
        }
    }
}
```

```
// Externals.java inside see
package see;
import cie.Internals;
import java.util.Scanner;
public class Externals extends Internals
{
    protected int marks[];
    protected int FinalMarks[];
    public Externals()
    {
        marks = new int[5];
        FinalMarks = new int[5];
    }
    public void inputSEEmarks()
    {
        Scanner S = new Scanner(System.in);
        for(int i = 0; i<5; i++)
        {
            System.out.println("Subject "+(i+1)+" marks");
            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]);
        }
    }
}
// mainfinals.java
import see.Externals;
Public class mainfinals
{
    pubic static void main(String args[])
    {
        int numOfStudents = 2;
        Externals FinalMarks[] = new Externals[numOfStudents];
        for (int i=0; i<numofstudents; i++)
        {
            FinalMarks[i]=new Externals();
            FinalMarks[i].inputStudentDetails();
            System.out.println("Enter CIE Marks:");
            FinalMarks[i].inputCIEmarks();
            System.out.println("Enter SEE marks");
            FinalMarks[i].inputSEEmarks();
        }
        System.out.println("Display the final marks:\n");
        for (int i=0; i<numOfStudents; i++)
        {
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

OUTPUT

Enter the name : Sarthak
Enter the USN : 1BH22CS246
Enter the semester : 3
Enter CIE marks :
Enter Subject 1 marks : 40
Enter Subject 2 marks : 42
Enter Subject 3 marks : 41
Enter Subject 4 marks : 45
Enter Subject 5 marks : 40

Enter SEE marks
Subject 1 marks : 100
Subject 2 marks : 88
Subject 3 marks : 85
Subject 4 marks : 89
Subject 5 marks : 80

Display the final marks :
Name : Sarthak
USN : 1BH22CS246
Sem : 3
Subject 1 : 90
Subject 2 : 86
Subject 3 : 83.5
Subject 4 : 89.5
Subject 5 : 80

30/1/2024

LAB - 7

Q. 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 takes both father and son's age and throws an exception if son's age is >= father's age.

→

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

```
class Son extends Father
{
    private int sonAge;
    public Son(int FatherAge, int sonAge) throws WrongAge
    {
        super(FatherAge);
        if (sonAge < 0){
        {
            throw new WrongAge("Son's age cannot be
                                negative");
        }
        else if (sonAge >= FatherAge)
        {
            throw new WrongAge("Son's age cannot be
            greater or equal than father");
        }
        else
        {
            System.out.println("Age is valid");
        }
        this.sonAge = sonAge;
    }
}
}

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

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

OUTPUT

Enter the age of the Son : 50
Enter the age of the father : 20
Son's age cannot be greater than or equal to father's age.

Enter the age of the son : 50
Enter the age of the Father : 50
Son's age cannot be greater than or equal to father's age

30/1/2024

LAB program no. 8

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

→
```
class bms extends Thread{
    public void run(){
        for(int i=1; i<=5; i++){
            System.out.println("bms college of engineering"+i);
            try
            {
                Thread.sleep(10000);
            }
            catch(InterruptedException e){
                e.printStackTrace();
            }
        }
    }
}
class cse extends Thread{
    public void run(){
        for(int i=1; i<=10; i++){
            System.out.println("cse"+i);
            try
            {
                Thread.sleep(2000);
            }
            catch(InterruptedException e){
                e.printStackTrace();
```

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

OUTPUT

cse1 → bms college of engineering1

cse2

cse3

cse4

cse5

bms college of engineering2

cse6

cse7

cse8

cse9

cse10

bms college of engineering 3

bms college of engineering 4

bms college of engineering 5

6/8/2024

13/2/24

LAB 10.1

Q. Demonstrate Inter process communication and deadlock.

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

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

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

OUTPUT

```
Put : 0
Press Control-C to stop
Intimate Consumer
Producer waiting
Got: 0
Intimate Producer
Consumed: 0
Put : 1
Intimate Consumer
Producer waiting
Got: 1
Intimate Producer
Consumed: 1
Put 2
Intimate Consumer
Producer waiting
Got: 2
Intimate Producer
Consumed: 2
:
:
:
:
```

15/2/24

LAB 10.2

Q. Demonstration in 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(2000);
     }catch(Exception e){
          System.out.println("B Interrupted");
     }
     System.out.println(name + "trying to call A.last()");
     a.last();
     }
```

```
    void last(){
      System.out.println("Inside A.last");
    }
}

class Deadlock implements Runnable
{
    A a = new A();
    B b = new B();
    Deadlock(){
      Thread.currentThread().setName("MainThread");
      Thread t = new Thread(this, "RacingThread");
      t.start();
      a.foo(b);
      System.out.println("Back in mainThread");
    }

    public void run(){
      b.bar(a);
      System.out.println("Back in otherThread");
    }

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

OUTPUT

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

Date 20/12/24
Page

LAB Program 9

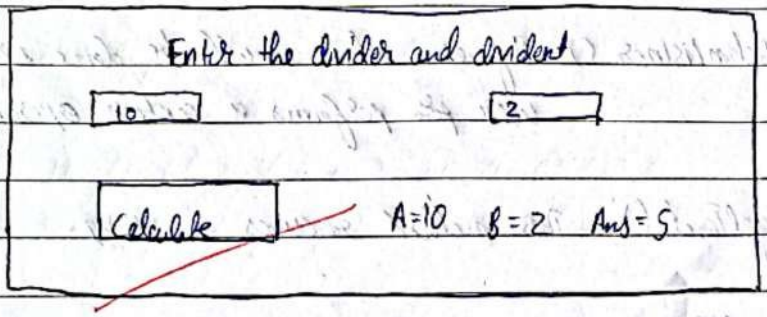
9. Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw 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 SwingDemo{
 SwingDemo(){
 JFrame jfrm = new JFrame("Divider App");
 jfrm.setSize(275, 150);
 jfrm.setLayout(new FlowLayout());
 jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 // text label
 JLabel jlab = new JLabel("Enter the dividend and divisor:");

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

 Button 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(alab);
jfrm.add(blab);
jfrm.add(anslab);

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

button.addActionListener(new ActionListener(){
public void actionPerformed(ActionEvent evt){
  try{
    int a = Integer.parseInt(ajtf.getText());
    int b = Integer.parseInt(bjtf.getText());
    int ans = a/b;
    alab.setText("\n A = "+a);
    blab.setText("\n B = "+b);
    anslab.setText("\n Ans = "+ans);
  }
  catch(ArithmeticException e){
```

```
alab.setText(text:" ");
blab.setText(text:" ");
anslab.setText(text:" ");

err.setText(text:"Enter only Integers!");
}
}
});
jfrm.setVisible(b:true);
}
public static void main(String args[]){
SwingUtilities.invokeLater(new Runnable(){
public void run(){
new SwingDemo();
}
});
}
}
```

OUTPUT:



Definitions of the functions

JFrame: A class in Java that has its own modules and constructors

FlowLayout(): Creates a flow layout with centered alignments and a default 5 unit horizontal and vertical gap.

jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE): To terminate on close

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

JLabel: To give labels to the objects

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

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

getText(): This method receives a String.

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

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

20/3/2024

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

```
   r2=Math.sqrt(-d)/(2*a);
   System.out.println("Root1="+r1+"+i"+r
   2); System.out.println("Root1="+r1+"-
   i"+r2);
  }
 }
}
class QuadraticMain
{
 public static void main(String[] args)
 {
  Quadratic q=new
  Quadratic(); q.getd();
  q.compute();
 }
}
```

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
        subMarks
        ; int
        credits;
        int grade;
}

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

        Student()
        {
                int i;
                subject = new Subject[8];
                for(i=0;i<8;i++)
                        subject[i] = new
                Subject(); s = new
                Scanner(System.in);
```

```
	}

	void getStudentDetails()
	{
		System.out.println("Enter your name:
		"); name = s.next();
		System.out.println("Enter your USN:
		"); usn = s.next();
	}

	void getMarks()
	{
		for(i=0;i<8;i++)
		{
			System.out.println("Enter marks of subject " + (i+1) +
			": "); subject[i].subMarks = s.nextInt();
			System.out.println("Enter credits of subject " + (i+1) +
			": "); subject[i].credits = s.nextInt();
			subject[i].grade = (subject[i].subMarks/10) + 1;
		}
	}
	void computeSGPA()
	{
		int effscore =
		0; int
		totalcreds = 0;
		for(i=0;i<8;i+
		+)
		{
			effscore += subject[i].grade * subject[i].credits;
			totalcreds += subject[i].credits;
		}
		SGPA = (double)effscore/(double)totalcreds;
	}
}

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);
        }
}
```

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: " + 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 books_main
{
        public static void main(String args[])
        {
                Scanner s = new
                Scanner(System.in); int n,i;
```

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

                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 name of book: ");
                        name = s.next();
                        System.out.println("Enter author of book:
                        "); author = s.next();
                        System.out.println("Enter price of book: ");
                        price = s.nextInt();
                        System.out.println("Enter number of pages: ");
                        numPages = s.nextInt();
                        b[i] = new Books(name,author,price,numPages);
                }

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

        }
}
```

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 ofthe given shape.

```
import

java.util.Scanner;

class InputScanner{
int d1, d2;
Scanner sc = new Scanner(System.in);
InputScanner(){
```

```
if(this.getClass() == Circle.class)
{ System.out.println("Enter radius of
circle: "); d1 = sc.nextInt();
}
else{
System.out.println("Enter height and weight:
"); d1 = sc.nextInt();
d2 = sc.nextInt();
}
}
}

abstract class Shape extends
InputScanner{ abstract void printArea();
}

class Triangle extends
Shape{ void printArea(){
System.out.println("Area of triangle is: " + (double)(d1*d2)/2);
}
}

class Rectangle extends
Shape{ void printArea(){
System.out.println("Area of rectangle is: " + (double)(d1*d2));
}
}


class Circle extends
Shape{ void printArea(){
System.out.println("Area of circle: " + (double)(3.14*d1*d1));
}
}

class AreaMain{
public static void main(String
args[]){ Rectangle r = new
Rectangle(); Triangle tr = new
Triangle();
Circle c = new
Circle(); r.printArea();
tr.printArea();
c.printArea();
}
```

}

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.*;
import
java.lang.Math;
abstract class
Bank{
        abstract void withdraw(double
        amt); abstract void
        deposit(double amt); abstract
        void display();
        abstract void menudisp();
}

class Account extends Bank{
        String name;
        int
        acc_num
        ; String
        type;
        double
        bal;
        String menu = " ";
        Account(String name, int acc_num, String type, double bal, String menu)
                { this.name = name;
                this.acc_num = acc_num;
                this.type = type;
                this.bal = bal;
                this.menu = menu;
        }
        public void withdraw(double amt){
                if(amt>bal){
                        System.out.println("Withdraw declined! Max amount you can withdraw
                        is:
" + bal);
                }
```

```
                else
                {       bal -= amt;
                        System.out.println("Updated balance is: " + bal);

                }
        }


        public void deposit(double
                amt){ bal += amt;
                System.out.println("Updated balance is: " + bal);
        }


        public void display(){
                System.out.println("Account number: " + name);
                System.out.println("Account name: " + acc_num);
                System.out.println("Account type: " + type);
                System.out.println("Balance: " + bal);
        }
        public void menudisp(){
                menu = "---------MENU---------\n1. Deposit\n2. Withdraw\n3. Display";
        }
}

class Savings extends
        Account{ double
        interest;
        Savings(String name, int acc_num, String type, double bal,String menu, double
                interest){ super(name, acc_num, type, bal,menu);
                this.interest = interest;
        }
        public double interest(int time)
                { double comp;
                comp = bal + Math.pow((bal*(1+(interest/100))),time);
                return comp;
        }
        public void menudisp(){
                super.menudisp();
                menu += "\n4. Compute Interest\n5. Exit";
        }
}

class Current extends
        Account{ double
        minbal = 10000;
        Current(String name, int acc_num, String type, double bal, String menu, double
                minbal){ super(name, acc_num, type, bal,menu);
```

```
                this.minbal = minbal;
        }

        public void menudisp(){
                super.menudisp();
                menu += "\n4. Cheque Book\n5. Exit";
        }

        public void withdraw(double amt){
                if(amt>bal){
                        System.out.println("Withdraw declined! Max amount you can withdraw
                        is:

" + bal);
                }
                else{
                        bal -= amt;
                        System.out.println("Updated balance is: " + bal);
                }
        }

}


class BankMain{
        public static void main(String args[])
                { Scanner s = new
                Scanner(System.in); String name;
                String menu =
                " "; int
                acc_num;
                String type;
                double bal =
                0; double
                interest; int
                choice;
                int time;
                double
                money;
                System.out.println("Enter customer name:
                "); name = s.next();
                System.out.println("Enter account number:
                "); acc_num = s.nextInt();
                System.out.println("---------ACCOUNT TYPE ---------\n1.Savings
Account\n2.Current Account\nPlease select account type:
                "); type = s.next();

                if(type.equals("savings")){
                        System.out.println("Enter interest amount: ");
                        interest = s.nextDouble();
```

```
                    Savings accs = new Savings(name, acc_num, type, bal, menu,
                    interest); do{
                            accs.menudisp();
                            System.out.println(accs.menu);
                            System.out.println("Enter choice: ");
                            choice = s.nextInt();
                        switch(choice){
                                case 1:

                                                System.out.println("Enter amount to be deposited:
");
                                                money = s.nextDouble();
                                                accs.deposit(money);
                                                break;
                                        case
                                        2:      System.out.println("Enter amount to be withdrawn:
");
                                                money = s.nextDouble();
                                                accs.withdraw(money);
                                                break;

                                        case    accs.display();
                                        3:      break;

                                                System.out.println("Enter time to calculate interest
in years: ");                           case
                                        4:
                                                time = s.nextInt();
                                                money = accs.interest(time);
                                                System.out.println("Compund interest is: "
money);
                                                + break;

                                                break;
                                        case
                                        5:

                                }

                    }while(choice!=5);
                }

        }
}
```

6) Create a package CIE which has two classes- Student and Internals. The class Student has members like usn, name, sem. The class Internals derived from Student has an array that stores the internal marks scored in five courses of the current semester of the

student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

```
package
CIE; import
java.util.*;
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);
  this.usn=s.nextLine();
  this.name=s.nextLine();
  this.sem=s.nextInt();
 }

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

```
package CIE;
import java.util.Scanner;
public class Internals extends
 Student{ protected int
 marks[]=new int[5]; public void
 inputCIEmarks(){
  Scanner s=new
  Scanner(System.in); for(int
  i=0;i<5;i++){ 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);
  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.*;

class

Main1{

 public static void main(String
  args[]){ int num=2;
  Externals finalMarks[]=new Externals[num];
  for(int i=0;i<num;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<num;i++)
  { finalMarks[i].calculateFinalMarks();
```

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


 }
}
```

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

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

class Father extends
    InputScanner{ int fatherAge;
    Father() throws WrongAge
        { System.out.println("Enter father's age: ");
        fatherAge = s.nextInt();
        if (fatherAge < 0)
            throw new WrongAge("Age cannot be negative");
    }

    void displayf(){
        System.out.println("\nFather's  age: " + fatherAge);
    }
}

class Son extends
    Father{ int sonAge;
    Son() throws
        WrongAge{ super();
        System.out.println("Enter son's age: ");
```

```
        sonAge =
        s.nextInt(); if
        (sonAge < 0)
            throw new WrongAge("Age cannot be negative.");
        else if(sonAge > fatherAge)
            throw new WrongAge("Son's age cannot be greater than father's age.");
        else{
            displayf();
            displaySon();
            throw new WrongAge("Valid age.");
        }
    }

    void displaySon(){
        System.out.println("\nSon's age : "+ sonAge);
    }
}

class exceptionsmain {
    public static void main(String[] args)
        { try{
            Son son = new Son();
        } catch(WrongAge e)
            { System.err.println(e.getMessage());
        }
    }

}
```

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.

```
import java.lang.*;
class DisplayMessageThread extends Thread
    { private final String message;
    private final long interval; // in milliseconds

    DisplayMessageThread(String message, long interval)
        { this.message = message;
        this.interval = interval;
    }

    public void
        run() { try {
```

```
            while (true)
                { System.out.println(message);
                Thread.sleep(interval);
            }
        } catch (InterruptedException e)
            { System.out.println(Thread.currentThread().getName() + " interrupted.");
        }
    }
}

public class TwoThreadDemo {
    public static void main(String[] args) {
        DisplayMessageThread thread1 = new DisplayMessageThread("BMS College of
Engineering", 10000); // 10 seconds
        DisplayMessageThread thread2 = new DisplayMessageThread("CSE", 2000); // 2
        seconds

        thread1.setName("Thread 1");
        thread2.setName("Thread 2");

        thread1.start();
        thread2.start();

        try {
            // Let the threads run for a while
            Thread.sleep(30000); // Let the program run for 30 seconds
        } catch (InterruptedException e)
            { System.out.println("Main thread
            interrupted.");
        }

        // Interrupt both threads to stop
        them thread1.interrupt();
        thread2.interrupt();

        System.out.println("Main thread exiting.");
    }
}
```

10)Demonstrate Inter process Communication and deadlock
a)Inter process Communication
b)Deadlock

```
import
java.lang.*;
class Q {
    int n;
```

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

class Producer implements
    Runnable { Q q;
    Producer(Q q)
        { this.q = q;
        new Thread(this, "Producer").start();
    }
    public void
        run() { int i =
        0;
        while(i<15) {
            q.put(i++);
        }
```

```
    }
}

class Consumer implements
    Runnable { Q q;
    Consumer(Q q)
        { this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void
        run() { int
        i=0;
        while(i<15) {
            int
            r=q.get()
            ; 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 otherthread.

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

}

public static void main(String args[])

{ new Deadlock();

}
```

```
}
```

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 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.getHeight()- 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);
          }

}
```