VISVESVARAYA TECHNOLOGICAL UNIVERSITY **Belgaum -590014, Karnataka.**

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ) *Submitted by*

Mrunalini S M (1BM22CS228)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Dec-2024 to Mar-2024

12/12/23 Object oriented Java

1st Program

```
class hello
{
    public static void main (String args[])
    {
        System.out.println ("hellow world");
    }
}
```

Output: hello world.

Program-2

```
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");
}
Else if (d<0)
{
System.out.Print("Roots are imaginary");
r1 = (-b)/(2*a);
r2 = Math.Sqrt(-d)/(2*a);
System.out.Println("Roots are
real and distinct");
System.Out.Println("Root1 = "+r1+
        "Root 2 = "+r2);
}
```

```
Else if (d<0)
  {
     System.out.Println ("Roots are imaginary");
     r1 = (-b) /(2*a);
     r2 = Math.Sqrt (-d)/(2*a);
     System.Out.Println ("Root 1 = "+r1 + "+i"
                          +r2);
     System.out.Println ("Root 1= "+r1 +"-i"+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"+
     System.Out.Println ("Root 1 = "+r1+"-i"+r
  }
}
}
Class Quadratic Main
{
   Public Static void main (String args[]
   {
      Quadratic q = new Quadratic();
      q.getd();
      q.Compute();
   }
}
```

Output:

Mrunalini 1BM22CS228

Enter the coefficients of a, b, c

1
5
2

Roots are real and distinct

Root 1 = -4.561552180

Root 2 = -0.438447819

Output 2

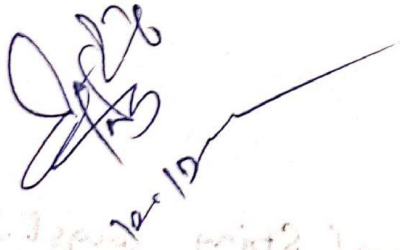
Mrunal 1BM22CS228

Enter the co-efficients of a, b, c

1
2
1

Roots are real & Equal

Root 1 = Root 2 = -1.0



1) Write a Program In Java to find the area of a rectangle and Verify the Same with Various inputs (length, breadth)

```
class RectangleArea {
Public Static Void main (String args[]) {
int length, breadth;
length = Integer.ParseInt (args [0]);
breadth = integer.parseInt (args [1]);
int area = length * breadth
System.out.println ("length of rectangle = "+length);
System.out.println ("breadth of rectangle = "+ breadth
System.out.println ("area of rectangle :"+ area);
}}
```

output:

length of rectangle = 10
breadth of rectangle = 8
area of rectangle = 80;

2) Write a Program to find factorial of given number.

```
class factorial {
Public Static Void main (String args [])
{
int fac = 1;
System.out.println ("Enter a number:");
Scanner sc = new Scanner (System.in);
int n = sc.nextInt ();
for (int i=1; i<=n; i++) {
    fac = fac * i;
}
System.out.println ("The factorial: "+fac);
```

```
    }
}
```

Output: System.out.

Palindrome: Wap to find the given 5 digit int is a palindrome or not

```
Class Palindrome {
Public Static Void main (String args[])
{
 int n, t, rem ,rev = 0
Scanner sc = new Scanner (System.in);
System.out.Println ("Enter a 5 digit number
n = sc.nextInt();
  t = n;
While (t>0) {
rem = t%10;
rev = rev * 10 + rem;
  t = t/10;
}
  if (rev == n) {
System.out.Println ("Palindrome");
}
 else {
System.out.Println (" not palindrome);
}
}
```

```
else
 {
  for (i=2 ; i<=m ; i++)
  {
  if (n%i==0)
  {
     System.out.println(n+" is not a
                              Prime number.");
  break;
  }
 if (flag==0)
  {
    System.out.println(n+" is Prime");
  }
  }
 }
Public Static Void main (String args[])
{
 int i;
Scanner sc = new Scanner (System.in);
System.out.println("Enter the value of i:");
 i = sc.nextInt();
{
 int i;
Scanner sc = new Scanner (System.in);
 System.out.println ("enter the value of i:");
i = sc.nextInt();
 isPrime(i);
}
}
```

Conversion:

```
Class Conversion {
Public Static Void main (String args[])
{
 byte b=1; Short s1=1000, s2;
 Int i1= 100000, i2;
 Long l1= 10000000, l2;
 Char c='t'
   float f1= 25.69f, f2;
 double d1=.53.6987.125, d2;
 System.out.Println(b+" "+s1+" "+i1+" "+l1
                +" "+c+" "+f1+" "+d1);
 s2=b;
 i2=s1;
 i2=i1;
 System.out.println(s2+" "+i2+" "+l2);
}
}
```

Output:

```
1 1000 100000 10000000 +25.69 53.6987
1 1000 1000000
```

Lab Program - 2 19/dec/23

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 Student.

```
import java.util.Scanner;

class Student {
    Private String USN;
    Private String name;
    Private int [] Credits;
    Private int [] marks;
    Public Student1 (String USN, String name,
        int [] Credits, int [] marks ) {
        this.USN = USN;
        this.name = name;
        this.Credits = credits;
        this.marks = marks;
    }
    Public void accept details () {
        Scanner Sc = new Scanner (System.in);
        System.out.Println ("Enter USN:");
        USN = sc.nextline();
        System.out.Println ("Enter Name:");
        name = Sc.nextline();
        System.out.Println ("Enter Number of Subjects:");
        int numSubjects = Sc.nextInt();
        Credits = new int [numSubjects];
        marks = new int [numSubjects];
        for (int i=0; i< numSubjects ; i++) {
```

```
System.out.Println("Enter credit for Subjects"+
                    (i+1)+" : ");
    Credits[i] = Sc.nextInt();
    System.out.Println("Enter marks for
                    Subjects "+(i+1)+" :
    marks[i] = Sc.next Int();

  }
}
Public void display details(){

    System.out.Println("USN : "+USN);
    System.out.Println("Name : "+name);
    for(int i=0; i< credits.length;i++){

      System.out.Println("Subject "+(i+1)+" :
      Credits : "+credits[i]+", Mark : "+marks
    }
}
Public double CalculateSgpa(){
    double totalCreditPoints = 0;
    double totalCredits = 0;

    for(int i=0; i< credits.length;i++){
      totalCreditsPoints += (credits[i]*
      getGradePoints(marks[i]);
      totalCredits += Credits[i];

    }
    return totalCreditPoints/totalCredits;
```

```
}
Private double getGradePoint(int marks){
    if (marks >= 90){
        return 10.0;
    } else if (marks >= 80){
        return 9.0;
    } else if (marks >= 70){
        return 8.0;
    } else if (marks >= 60){
        return 7.0;
    } else if (marks >= 50){
        return 6.0;
    } else if (marks >= 60){
        return 5.0
    } else {
        return 0.0;
    }
}
Public Static void main (String[] args){
    int[] credits = {3, 4, 2};
    int[] marks = {85, 78, 92};
    Student Student = new Student
("BM22CS228", "John", credits, marks);
    Student.acceptdetails();
    Student.displaydetails();
    System.out.println("SGPA: " + Student.
                    calculatorSGPA());
}
```

}

Output:

Enter Name: mrunalini
Enter USN: 1BM22CS228
Enter number of Subjects: 3

Enter credit for subject S1:
3
Enter mark for Subject S1:
90
Enter credit for Subject S2:
4
Enter mark for Subject S2:
90
Enter Credit for Subject S3:
4
Enter marks for Subject S3:
85

USN: 1BM22CS228
Name: mrunalini
Subject Credits : 3, Marks: 90
Subject Credits : 4 Marks: 90
Subject credits : 4 Marks : 85
SGPA: 9.636363

19.12

Lab Program - 3

→ Create a class book which contains four members : name, author, Price, num_Pages Include a Constructor to set the values for the members. include methods to set and get the details of the objects. include a toString() method that could display the complete details of 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 Main
{
    Public Static void main (String args[])
    {
        Scanner s = new Scanner(System.in);
        int n;
        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 (int i = 0; i < n; i++)
        {
            System.out.Println(" Enter the
                name of book");
            name = s.next();
            System.out.Println("Enter the
                author of book");
            author = s.next();
```

```
System.out.println("Enter the number of
                    Pages of book");
num Pages = S.nextInt();
b[i] = new Books (name, author, Price, numPages);
}
    for (int i = 0 ; i<n;i++)
    {
        System.out.println(b[i]);
    }
}
}
```

Output:

```
Enter the number of books
2
Enter the name of the book
Physics
Enter the author of book
Einstein
Enter the Price of book
100
Enter the number of Pages of book
189
Enter the name of book
Chemistry
Enter the author of book
Joseph
Enter the Price of book
200
```

Enter the no number of Pages of book
390

Book name : Physics
Author name : Einstein
Price : 100
Number of Pages : 189

Book name : Chemistry
Author name : Joseth
Price : 390.00
number of Pages : 390.

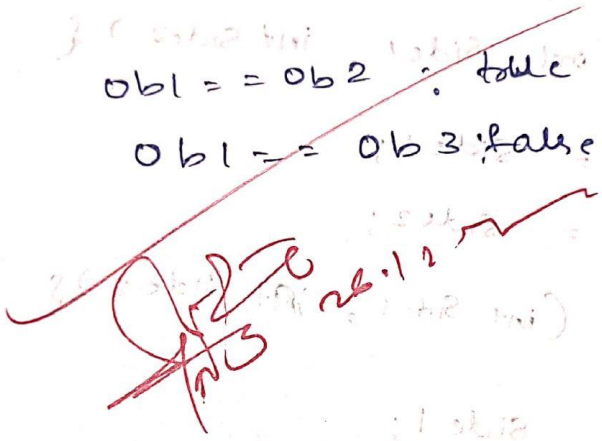
Additional Program

```
import java.util.Scanner;
Class Fast
{
    int a, b;
    Test (int i, int j) {
    a = i; b = j;
    }
    boolean equals (Test o) {
     if (o.a == a && o.b == b) return type;
      else return false;
    }
}
Class Passob
```

```
{ Public Static void main (String args [ ]) {
    Test ob1 = new Test. (100, 22);
    Test ob2 = new Test (100, 22);
    Test ob3 = new Test (-1, -1);
System.out.Println("ob1 == ob2 :" + ob1.
    Equals (ob2));
System.out.Println (" ob1 == ob3" +
    ob1.Equals (ob3));
}
```

Output: ob1 == ob2 : true

ob1 == ob3 : false



```
import java.util.Scanner;
class InputScanner {
    public static Scanner scanner =
        new Scanner(System.in);
}
abstract class Shape extends InputScanner {
    protected int side1;
    protected int side2;

public Shape(int side1, int side2) {
    this.side1 = side1;
    this.side2 = side2;
public Shape(int side1, int side2) {
    this.side1 = side1;
    this.side2 = side2;
}
public abstract void PrintArea();
}
class Rectangle extends Shape {
    public Rectangle(int length,
        int width) {
    super(length, width);
}
```

Lab Prg - 5-4

@ Override
Public void

```
import java.util.Scanner;
class inputScanner
{
    Protected Scanner Scanner;
    Public inputScanner()
    {
      Scanner s= new Scanner (System.in);
      }
}
abstract class Shape extend inputScanner
{
    double a,b,r;
    Scanner s= new Scanner (System.in);
    System.out.Println ("Enter the value of
                          a :" +a);
    a = s.nextDouble ();
    System.out.Println ("Enter the value of
                          b:" +b);
    b= s.nextDouble ();
      PrintArea ();
}
class Rectangle Extends Shape
{
```

```
{
    Rectangle ()
    {
     Super();
    }
      Print Area ()
    {

    double area = a* b;
    System.out.Println ("Area of the
      rectangle is" + area);

    }

}
 Class Triangle extends Shape
 {
     Triangle()
     {
      super();
     }
    PrintArea()
     {
    double area = 3.14 * r * r;
    System.out.Println ("Area of the
triangle is" + area);
    }

}
```

```
class Circle extends Shape
  Circle (double a)
{
  super (a);
}
void area ()
  {
  System.out.println ("area of circle" +
  (3.14 * (radius) * (radius)));
triangle (double a, double b)
    }
}
  super
}
class AbstractAreaMain
{
  Public Static Void Main (String args[])
  {
  Scanner s= new Scanner (System.in);
  System.out.Println (" Enter the lngth and
    breadth of rectangle");
  double l = s.nextInt ();
  double b = s.nextInt ();
  System.out.Println ("Enter the base and
                height");
  double h1 = s.nextInt ();
  double h2 = s.nextInt ();
```

```
System.out.Println("Enter the circle");
double r = s.nextInt();
Shape sh;
Rectangle rect = new Rectangle(l, b);
Triangle cir = new Circle(r);
sh = rect;
sh.area();
sh = tri;
sh.area();
sh = cir;
sh.area();
}
}
```

Output:

Enter the length and breadth of rectangle

9

5

Enter the base and height

3

4

Enter the circle

9

area of rectangle : 45.0

area of triangle : 6.0

area of circle : 254.34

02-07-24

Lab - 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 deposite (double amount)
{
    balance += amount;
}
Void withdrawal (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 :" + 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 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 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 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;
case4: Sa.display();
break.
```

```
Case 5 : System.exit(0);
default : System.out.println("invalid input");
          break;
    }
}
else
{
    System.out.println("\n menu \n1. deposite 2. withdraw. 3. display");
System.out.println("Enter Choice");
    ch = S.nextInt();
    switch(ch)
    {
    Case 1 : System.out.println("enter amount:");
    amount1=S.nextInt();
    ca.deposite(amount1);
    break;
Case 2 : System.out.println("enter 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;

        }
      }
    }
}
```

Output:

Enter the name:
Mounika

Enter the type (Current / Savings)
Current

Enter initial balance
4500

Menu
1) deposit 2 Withdrawal
3) display

Enter the choice

1

Enter amount

23000

menu

1. deposit 2. withdraw 3. display

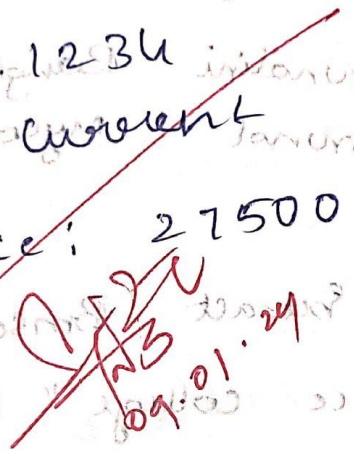
3

name : Navneeth

acc no : 1234

type : current

balance : 27500



Jan/16/2

lab - 6

1) Demonstrate various constructors in String

Str 1 : Hello
Str 2 : world
Str 3 : Or l
Str 4 : Hello
Str 5 : Java

2) Str = "Hello. World"
String length = 12

3) demonstrate toString()

output:

101. mrunalini Bangalore
102. mrunal Mysore.

4) Using getchars(), Extract Bmscc from "Welcome to Bmscc college"

Output : Bmscc

5) Demonstrate getbytes(), tochar Array()
with Proper Java Programs

Output : (i) getbytes()

Hello, world!

(ii) tochar Array()

J a v a P r o g r a m m i n g

6) Check the following output and write the Java Programs using Stringfunction

i. Bmscc Equals Bmscc → true
ii. Bmscc Equals College → false

Bmscc Equals BMSCE → false.

Bmscc Equals IgnoreCase BMSCE → true

Output :

```
Public class StringComparisonExample {
    Public Static Void main (String [] args) {
        String Str1 = "Bmscc";
        String Str2 = "College";
        String Str3 = "BMSCE";
System.out.Println (Str1 + "Equals" + Str1 +
        "->" + Str1.Equals(Str1));
System.out.Println (Str1 + "Equals" + Str2 +
        "->" + Str1.Equals(Str2));
System.out.Println (Str1 + "Equals" + Str3 +
        "->" + Str1.Equals(Str3));
System.out.Println (Str1 + "Equals" + Str4 +
        "->" + Str1.Equals(Str4));
    }
}
```

7) Using regionMatches() find substring "Bmscc College" from String "Welcome to Bmscc College of Engineering", display substring is matched otherwise display not matched

Output :

Substring is matched

8) Demonstrate StartsWith() to give output true & false.

Sol:- import java.lang.String;

Public class Star 2

Public Static Void main

output: Check whether String Starts with Welcome at Pos 11: False

Check whether String Starts With Bmscc to at Pos 11: ~~false~~ true.

9) Demonstrate EndsWith() to give output ~~true~~ and false.

Output: True
False

10) Demonstrate a java Program to show output for Equals() versus ==

Output Using == with String literals:

Str1 == Str2 : true

Str1 == Str3 : False

Using Equals() with String literals:

Str1.Equals(Str2) : true

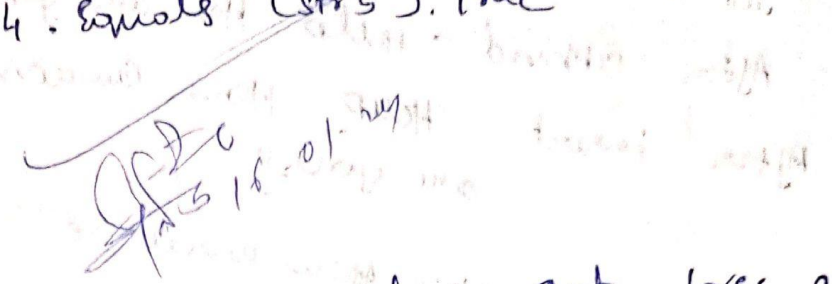
Str1.Equals(Str3) : true

using == with new String Objects:

Str4 == Str5 : false

using Equals() with new String Objects:

Str4.Equals(Str5): true



11) apple, ball, cat, dog, ant, free, gun, teacher, ice, jug, light man, orange, Parrot queen, ring, Star, tree umbrella van, watch, xmas.

12) Sorted number [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

13) it was a test. it was too

14) Hello world

15) Commerce

16) Hello friend.

17) Enter the no of Students : 3

Enter the name: Rani mounolini

Enter the regd ter no : 567228

Enter the name : Ruchika kushi

Enter the cgpa : -9.8.

Sorted by cgpa : 230

name: Rani mounolini regno : 228

cgpa : 8.1

name : kushi regno: 230

CGPA : 9.8

18) Set length : Hello

Char At index 4 : c

After set char AT : Hello

Set char : Hello

After append : Hello How are you ?

After insert : Hello How awesome are you ?

After delete : Hello How awesome are you ?

After delete : Hello How are you ?

After delete : Hello How are you ?

After delete : Char At : Ello How are you !

After replace : Hello How are you ?

16-01-24

Week - 6.

```
Package CIE;
import java.util.Scanner;
Public class Student {
    Protected String USN = new String();
    Protected String name = new String();
    Protected int Sem;

Public void inputStudentDetails() {
    Scanner S = new Scanner(System.in);
    System.out.Print("Enter USN;");
    usn = S.next();
    System.out.Print("Enter Name:");
    name = S.next();
    System.out.Print("Enter Semester:");
    Sem = S.nextInt();
}
Public void displayStudentDetails() {
    System.out.Println("USN:" + usn);
    System.out.Println("Name:" + name);
    System.out.Println("Semester:" + Sem);
}
}

Package CIE;
Import java.util.Scanner;
Public class Internals extends Student {
```

```
Protected int marks [ ] = new int [5]
Public void input CIEmarks ( ) {
    Scanner s = new Scanner (System.in);
    for (int i = 0; i<5 ; i++) {
System.out.Println(" Enter CIE marks for
    Subject " + (i+1) + " : " );
    marks [i] = s.nextInt( ) ;
        }
    }
Package SEE ;
import CIE. Internals ;
import java.util. Scanner;
Rubic class Externals extends Internals {
    Protected int final marks [ ] = new int [5];
Public void input SEE marks ( ) {
    Scanner s = new Scanner (System.in );
    for (int i = 0; i < 5 ; i++ ) {
System.out.Print (" Enter SEE marks for
                Subject "+ (i+1)+ " : ");
    super.marks [i] = s.next Int( );
    }
}
Public void calculate Final Marks ( ) {
    for (int i = 0; i<5; i++ )
Final Marks[i] = super.marks [i] /2 +
                super.marks [i];
    }
```

```
Public void displayFinalMarks() {
    displayStudentDetails();
    for (int i = 0; i < 5; i++)
        System.out.println("subject " + (i+1) + ":"
            + finalMarks[i]);
    }
}
import SEE.Externals;
class Main {
Public Static void main (String args[]) {
    int numofStudents = 2;
Externals finalMarks[] = new Externals[numofStudents];

for (int i = 0; i < numofStudents; i++) {
    finalMarks[i] = new Externals();
    finalMarks[i].inputStudentDetails();
    System.out.println("Enter CIE Marks");
    finalMarks[i].inputCIEmarks();
    System.out.println("Enter SEE marks");
    finalMarks[i].inputSEEmarks();
}

System.out.println("displaying data:\n");
for (int i = 0; i < numofStudents; i++) {
    finalMarks[i].calculateFinalMarks();
    finalMarks[i].displayFinalMarks();
    }
}
```

SEE :

Sub 1 marks : 30

Sub 2 marks : 70

Sub 3 marks : 90

Sub 4 mark : 80

Sub 5 marks : 30

CIE :

Sub 1 mark = 70

Sub 2 mark : 8

Sub 3 mark :

Sub 4 mar :

Sub 5 marks :

SEE :

Sub 1 mark : 80

Sub 2 mark : 60

Sub 3 mark : 70

Sub 4 mark : 8

Sub 5 marks :

23.01.24

SEE :

Sub 1 marks : 30

Sub 2 marks : 70

Sub 3 marks : 90

Sub 4 marks : 80

Sub 5 marks : 30

CIE

Sub 1 mark = 10

Sub 2 mark : 80

Sub 3 mark : 70

Sub 4 mar : 10

Sub 5 marks : 20

SEE

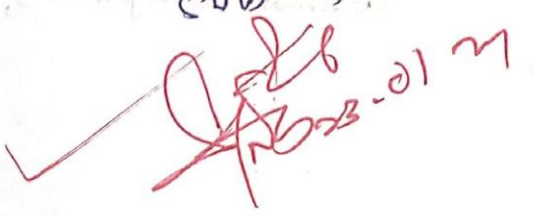
Sub 1 mark : 80

Sub 2 mark : 60

Sub 3 mark : 70

Sub 4 mark : 80

Sub 5 marks : 30



Week-7

30/01/2024

Write a Program that demonstrate handling of Exceptions in inheritance tree. Create a base class called "father" and derived class called "Son", which extends the base 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 calls both Father & Son's age & throws an Exception if Son's age is >= father's age.

```
import java.util.Scanner;
class WrongAge Extends Exception {
    Public WrongAge (String message) {
        Super (message);
    }
}
class Father {
    Protected int FatherAge;
    Public Father (int age) throws WrongAge {
        FatherAge = age;
        if (FatherAge < 0) {
            throw new WrongAge ("Father's age
                                 Cannot be negative");
        }
    }
}
Class Son Extends Father {
    Private int SonAge;
    Public Son (int FatherAge, int SonAge)
        throws WrongAge {
        Super (FatherAge);
```

```
        this . SonAge = SonAge;
        if (SonAge <= SonAge;
          if (SonAge <= 0) {
            throw new WrongAge ("Son's age cannot
                                 be zero or negative");
          }
          if (SonAge >= fatherAge) {
            throw new WrongAge ("Son's age
                 cannot be greater than or
                 Equal to father's age");
          }
        }

Public class Main25 {
    Public Static Void main (String [] args) {
        Scanner Scanner = new Scanner (System.in);
    try {
        System.out.Print ("Enter father's age:");
        int fatherAge = Scanner.nextInt();
        System.out.Print("Enter son's age:");
        int SonAge = Scanner.nextInt();
        Son Son = new Son (fatherAge, SonAge);
        System.out.println ("Father's age:" + fatherAge);
        System.out.println ("Son's age:" + SonAge);
    } Catch (WrongAge e) {
        System.out.println("Exception Caught:" + e);
        System.Out.println("Exception Caught: " + e.getMessage());
    }
```

```
Catch (Exception e) {
    System.out.println ("Error:" + e);
    System.out.println ("Error:" + e.getMessage());
    }
    finally {
        Scanner.close();
    }
}
}
```

Output: Enter father's age: 50

Enter son's age: -10

Exception Caught: WrongAge: Son's age cannot be negative (or) Zero

Enter father's age: 30

Enter Son's age: 40

Exception Caught: WrongAge: Son's age cannot be greater than or Equal to father's age

30.01.24

Week 8

Lab Program-8

```
class BMS extends Thread
{
  public void run()
  {
    for (int i=1; i<=5; i++)
    {
      System.out.println("BMS college of
                          Engineering " + i);
      try
      {
        Thread.sleep(10000);
      }
      catch (InterruptedException e)
      {
        e.printStackTrace();
      }
    }
  }
}
class CS extends Thread
{
  public void run()
  {
    for (int i=1; i<=5; i++)
    {
      System.out.println("CSE " + i);
      try
      {
        Thread.sleep(2000);
      }
      catch (InterruptedException e)
```

```
        {
            e.printStackTrace();
        }
    }
}
class threadMain
{
    public static void main (String args[])
    {
        BMS b1 = new BMS();
        CS c1 = new CS();
        b1.start();
        c1.start();
    }
}
```

Output:

BMS College of Engineering 1
CSE 1
CSE 2
CSE 3
CSE 4
CSE 5
BMS College of Engineering 2
BMS College of Engineering 3
BMS College of Engineering 4
BMS College of Engineering 5

Lab Program-14

```
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("Interrupted Exception caught");
}
System.out.println("Got:" + n);
valueSet = false;
System.out.println("\nIntimate Producer\n");
notify();
return n;
}
Synchronized void put(int n) {
while (valueset)
try {
System.out.println("\nproducer waiting\n")
wait();
} Catch (InterruptedException e) {
System.out.println("Interrupted Exception
caught");
}
```

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

```
Public void run() {
    int i = 0;
    while (i < 15) {
        int r = q.get();
        System.out.println("consumed: " + r);
        i++;
    }
}
}
public 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 1
got 1
Put 2
got 2
Put 3
got 3
Put 4
got 4
Put 5
got 5
Put 6
got 6
Put 7
got 7
Put 8
got 8
Put 9
got 9
Put 10
got 10
Put 11
got 11
Put 12
got 12
Put 13
got 13
Put 14
got 14
```

06.02.24

Lab Program - 10 13/02/24

```
class A {
    Synchronised 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 las() {
        System.out.println ("inside A.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) {
```

```
        System.out.println("B Interrupted");
    }
    System.out.println(name + "trying to
        Call A.last()");
    a.last();
}
void last() {
    System.out.println("Inside A.last");
}
}
class Deadlock implements Runnable {
    A a = new A();
    B b = new B();
Deadlock() {
    Thread.currentThread().setName
        ("Main Thread");
    Thread t = new Thread(this,
                "Racing Thread");
    t.start();
    a.foo(b);
    System.out.println("Back in main
                thread");
}
public void run() {
    b.bar(a);
    System.out.println("Back in Other
                thread");
}
public void run() {
    b.bar(a);
    System.out.println("Back in other
                thread");
}
```

```
Public Static void main (String args[])
    {
        new DeadLock();
    }
}
```

Output: main Thread entered A.bas

Racing Thread entered B.bar

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

Procon.java

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get() {
        while (!valueSet) {
            try {
                System.out.println("Consumer waiting.");
                wait();
            } catch (InterruptedException e) {
                System.out.println(e); }
```

```
System.out.println("got :" + n);
    value set = false;

System.out.println(" Tell producer");
    notify();
}
return n;
}}
Synchronized void put(int n) {
    while (value set) {
    try {
System.out.println("Producer waiting")
    wait();
} catch (Interrupted Exception e) {
    System.out.println(e);
}
this.n = n;
    value set = true;
System.out.println("Put:" + n);
System.out.println("Tell consumer");
    notify();
class Producer implements Runnable {
Q q;
Producers(Q q) {
    this.q = q;
new Thread(this, "Producer").start();
}
Public void run() {
    int i = 0;
    while (i < 3) {
```

```
        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<3) {
        int n = q.get();
    System.out.println("Consumed: " + n);
        i++;
      }
    }
  }
}
```

Output - 1

Put : 1

Got : 1

Put : 2

Got : 2

Put : 3

Got : 3

13.02.24

Lab Prog
Week = 10

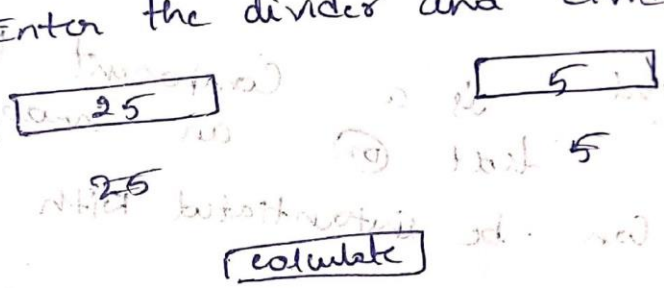
19/02/2024

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

class SwingDemo {
    SwingDemo() {
JFrame jfrm = new JFrame("Divide app");
jfrm.setSize(275, 150);
jfrm.setLayout(new FlowLayout());
jfrm.setDefaultCloseOperation(JFrame.
    EXIT_ON_CLOSE);
JLabel jlab = new JLabel("Enter the divider
                          and dividend:");
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
JButton button = new JButton("calculate");
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();

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

```
ActionListener l = new ActionListener() {
    Public Void actionPerformed (ActionEvent ae)
    {
    System.out.Println ("Action Event from a
    text field");
    }
};
ajtf.AddActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
    Public void actionPerformed (ActionEvent ae){
    try {
    int a = Integer.ParseInt(ajtf.getText());
    int b = Integer.ParseInt(bjtf.getText());
    int ans = a/b;
    alab.setText("\n A =" + a);
    blab.setText("\n B = " + b);
    ansLab.setText("\n Ans = " + ans);
} catch (NumberFormatException e) {
    alab.setText(" ");
    blab.setText(" ");
    anslab.setText(" ");
    Err.SetText("Enter only Integers!");
} catch (ArithmeticException e){
    alab.SetText(" ");
    blab.SetText(" ");
    anslab.setText(" ");
    Err.SetText("B Should be NON Zero!");
```

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

output:



Report

① JFrame

JFrame is a class defined in javax.swing Package that represents a window or a frame in a GUI application. JFrame serves as a Container to hold various GUI components such as buttons, textfields labels etc

SetSize: This method is used to set size of frame in pixels.
(int width, int height)

SetLayout: This method is used to set the layout manager for the frame, which determines how components are arranged with frame.

SetDefaultCloseOperation: This method sets the default operation that will be performed when the user closes the frame, such as exiting the application (or) hiding the frame.

JLabel: is a component used to display text (or) an image on GUI
it can be instantiated with 'JLabel(String text)'

getText(): This method retrieves the text entered by user in text field

Adding components to JFrame

'ActionListener' is an interface used to handle events triggered by user actions, such as clicking a button.

'addActionListener(ActionListener listener)': This method adds an ActionListener to component.

'setText()': Method is commonly used to text content of various components in Graphical User Interface (GUI) Programming, Particularly with components like JLabel, JTextField, Javax.Swing.text.JtextComponent and javax.Swing.JLabel.

WEEK 1

1. Develop a Java program that prints all real solutions to the quadratic equation ax2+bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2 -4ac is negative, display a message stating that there are no real solutions.

Program:

```
import java.util.Scanner;

class Quadratic {
   int a, b, c;
   double r1, r2, d;

   void getCoefficients() {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter the coefficients of a, b, and c:");
      a = scanner.nextInt();
      b = scanner.nextInt();
      c = scanner.nextInt();
   }

   void computeRoots() {
      while (a == 0) {
         System.out.println("Not a quadratic equation");
         System.out.println("Enter a non-zero value for a:");
         Scanner scanner = new Scanner(System.in);
         a = scanner.nextInt();
      }

      d = b * b - 4 * a * c;

      if (d == 0) {
         r1 = -b / (2.0 * a);
         System.out.println("Roots are real and equal");
         System.out.println("Root1 = Root2 = " + r1);
      } else if (d > 0) {
         r1 = (-b + Math.sqrt(d)) / (2.0 * a);
```

```
            r2 = (-b - Math.sqrt(d)) / (2.0 * a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root1 = " + r1 + " Root2 = " + r2);
        } else {
            System.out.println("Roots are imaginary");
            r1 = -b / (2.0 * a);
            r2 = Math.sqrt(-d) / (2.0 * a);
            System.out.println("Root1 = " + r1 + " + i" + r2);
            System.out.println("Root2 = " + r1 + " - i" + r2);
        }
    }
}

class QuadraticMain {
    public static void main(String[] args) {
        System.out.println("My Name is Mrunalini  S M");
        System.out.println("My USN is 1BM22CS228");
        Quadratic quadratic = new Quadratic();
        quadratic.getCoefficients();
        quadratic.computeRoots();
    }
}
```

WEEK2

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.

Program:

```
import java.util.Scanner;
import java.io.*;

class Student {
    private String usn;
    private String name;
    private int[] credits;
    private int[] marks;

    // Constructor
    public Student(String usn, String name, int numSubjects) {
        this.usn = usn;
        this.name = name;
        this.credits = new int[numSubjects];
        this.marks = new int[numSubjects];
    }

    // Method to accept details
    public void acceptDetails() {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter details for student " + name + " (USN: " + usn +
")");
        for (int i = 0; i < credits.length; i++) {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = scanner.nextInt();

            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }
```

```
    // Method to display details
    public void displayDetails() {
        System.out.println("Details for student " + name + " (USN: " + usn + ")");
        for (int i = 0; i < credits.length; i++) {
            System.out.println("Subject " + (i + 1) + " - Credits: " + credits[i] + ",
Marks: " + marks[i]);
        }
    }

    // Method to calculate SGPA
    public double calculateSGPA() {
        double totalCredits = 0;
        double totalGradePoints = 0;

        for (int i = 0; i < credits.length; i++) {
            totalCredits += credits[i];
            totalGradePoints += calculateGradePoints(marks[i]) * credits[i];
        }

        return totalGradePoints / totalCredits;
    }

    // Helper method to calculate grade points based on marks
    private double calculateGradePoints(int marks) {
        if (marks >= 90) {
            return 10.0;
        } else if (marks >= 80) {
            return 9.0;
        } else if (marks >= 70) {
            return 8.0;
        } else if (marks >= 60) {
            return 7.0;
        } else if (marks >= 50) {
            return 6.0;
        } else {
            return 0.0;
        }
    }
}
```

```
public class StudentSGPA {
    public static void main(String[] args) {
        System.out.print("my name is Mrunalini  S M");
        System.out.print("my USN is 1BM22CS228");

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of subjects: ");
        int numSubjects = scanner.nextInt();

        System.out.print("Enter the USN: ");
        String usn = scanner.next();

        System.out.print("Enter the name: ");
        String name = scanner.next();

        Student student = new Student(usn, name, numSubjects);
        student.acceptDetails();

        // Displaying details and SGPA
        student.displayDetails();
        System.out.println("SGPA: " + student.calculateSGPA());
    }
}
```

WEEK3

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.

Program:

```
import java.util.Scanner;

class Book {
    String name;
    String author;
    int price;
    int numPages;

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

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

public class BookStore{

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of books: ");
```

```
        int n = scanner.nextInt();

        Book[] books = new Book[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter name of the book: ");
            scanner.nextLine(); // consume the newline character
            String name = scanner.nextLine();

            System.out.print("Enter author of the book: ");
            String author = scanner.nextLine();

            System.out.print("Enter the price of the book: ");
            int price = scanner.nextInt();

            System.out.print("Enter the number of pages of the book: ");
            int numPages = scanner.nextInt();

            books[i] = new Book(name, author, price, numPages);
        }

        System.out.println("\nBook Details:");
        for (int i = 0; i < n; i++) {
            System.out.println("Book " + (i + 1) + ":\n" + books[i]);
        }
    }
}
```

WEEK4

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

Program:

```
import java.util.Scanner;

abstract class Shape {
    protected int side1;
    protected int side2;

    public Shape(int side1, int side2) {
        this.side1 = side1;
        this.side2 = side2;
    }

    public abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }

    @Override
    public void printArea() {
        int area = side1 * side2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }
```

```
    @Override
    public void printArea() {
        double area = 0.5 * side1 * side2;
        System.out.println("Triangle Area: " + area);
    }
}
class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }
    @Override
    public void printArea() {
        double area = Math.PI * side1 * side1;
        System.out.println("Circle Area: " + area);
    }
}
public class Area {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter length and width for Rectangle:");
        int rectLength = scanner.nextInt();
        int rectWidth = scanner.nextInt();
        Rectangle rectangle = new Rectangle(rectLength, rectWidth);
        rectangle.printArea();

        System.out.println("Enter base and height for Triangle:");
        int triBase = scanner.nextInt();
        int triHeight = scanner.nextInt();
        Triangle triangle = new Triangle(triBase, triHeight);
        triangle.printArea();

        System.out.println("Enter radius for Circle:");
        int circleRadius = scanner.nextInt();
        Circle circle = new Circle(circleRadius);
        circle.printArea();
        scanner.close();
    }
}
```

WEEK5

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.

Program:

```
import java.util.Scanner;

abstract class Account {
    String customerName;
    long accountNumber;
    String accountType;
    double balance;

    public Account(String customerName, long accountNumber, String
accountType, double balance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }

    public void displayBalance() {
        System.out.println("Account Balance: $" + balance);
    }
```

```
    // Abstract method for withdrawal
    public abstract void withdraw(double amount);
}

class CurrAcct extends Account {
    double minimumBalance;
    double serviceCharge;

    public CurrAcct(String customerName, long accountNumber, double balance) {
        super(customerName, accountNumber, "Current Account", balance);
        this.minimumBalance = 1000; // Set minimum balance
        this.serviceCharge = 50;    // Set service charge
    }

    @Override
    public void withdraw(double amount) {
        if (balance - amount >= minimumBalance) {
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining balance: $" +
balance);
        } else {
            System.out.println("Insufficient funds. Service charge of $" +
serviceCharge + " applied.");
            balance -= serviceCharge;
            System.out.println("Remaining balance after service charge: $" + balance);
        }
    }
}

class SavAcct extends Account {
    double interestRate;

    public SavAcct(String customerName, long accountNumber, double balance) {
        super(customerName, accountNumber, "Savings Account", balance);
        this.interestRate = 0.05; // Set interest rate (5%)
    }

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

```
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining balance: $" +
balance);
        } else {
            System.out.println("Insufficient funds. Cannot complete withdrawal.");
        }
    }

    public void depositInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest deposited. Updated balance: $" + balance);
    }
}

public class Bank1 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter account number: ");
        long accountNumber = scanner.nextLong();

        System.out.print("Enter initial balance: ");
        double initialBalance = scanner.nextDouble();

        System.out.print("Enter account type (Current/Savings): ");
        String accountType = scanner.next();

        Account account;
        if (accountType.equalsIgnoreCase("Current")) {
            account = new CurrAcct(customerName, accountNumber, initialBalance);
        } else if (accountType.equalsIgnoreCase("Savings")) {
            account = new SavAcct(customerName, accountNumber, initialBalance);
        } else {
            System.out.println("Invalid account type. Exiting program.");
            return;
        }
```

```
        int choice;
        do {
            System.out.println("\n1. Deposit");
            System.out.println("2. Display Balance");
            System.out.println("3. Deposit Interest for Savings Account");
            System.out.println("4. Withdraw");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter deposit amount: ");
                    double depositAmount = scanner.nextDouble();
                    account.balance += depositAmount;
                    System.out.println("Deposit successful. Updated balance: $" +
account.balance);
                    break;

                case 2:
                    account.displayBalance();
                    break;

                case 3:
                    if (account instanceof SavAcct) {
                        ((SavAcct) account).depositInterest();
                    } else {
                        System.out.println("This option is applicable for Savings Account
only.");
                    }
                    break;

                case 4:
                    System.out.print("Enter withdrawal amount: ");
                    double withdrawalAmount = scanner.nextDouble();
                    account.withdraw(withdrawalAmount);
                    break;

                case 5:
```

```
                System.out.println("Exiting program. Goodbye!");
                break;

            default:
                System.out.println("Invalid choice. Please enter a valid option.");
        }

    } while (choice != 5);

    scanner.close();
  }
}
```

WEEK6

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

Program:

```
// MainProgram.java
import CIE.Internals;
import SEE.External;
import java.util.Scanner;

public class MainProgram {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter the number of students:");
        int numStudents = scanner.nextInt();

        Internals[] internalsArray = new Internals[numStudents];
        External[] externalsArray = new External[numStudents];

        for (int i = 0; i < numStudents; i++) {
            System.out.println("Enter details for Student " + (i + 1));
            System.out.print("USN: ");
            String usn = scanner.next();

            System.out.print("Name: ");
            String name = scanner.next();

            System.out.print("Semester: ");
            int sem = scanner.nextInt();
```

```
            System.out.println("Enter Internal Marks for 5 courses:");
            int[] internalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                internalMarks[j] = scanner.nextInt();
            }

            internalsArray[i] = new Internals(usn, name, sem, internalMarks);

            System.out.println("Enter SEE Marks for 5 courses:");
            int[] seeMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                seeMarks[j] = scanner.nextInt();
            }

            externalsArray[i] = new External(usn, name, sem, seeMarks);
        }

        // Calculate and display final marks
        System.out.println("\nFinal Marks of Students:");
        for (int i = 0; i < numStudents; i++) {
            int[] internalMarks = internalsArray[i].getInternalMarks();
            int[] seeMarks = externalsArray[i].getSeeMarks();

            int[] finalMarks = new int[5];
            int totalMarks = 0;

            System.out.println("Student " + (i + 1) + " - " +
internalsArray[i].getName());

            for (int j = 0; j < 5; j++) {
                finalMarks[j] = internalMarks[j] + seeMarks[j];
                totalMarks += finalMarks[j];
                System.out.println("Course " + (j + 1) + ": " + finalMarks[j]);
            }

            System.out.println("Total Marks: " + totalMarks + "\n");
        }
    }
```

```
}


// CIE/Internals.java
package CIE;

public class Internals extends Student {
    protected int[] internalMarks;

    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }

    public int[] getInternalMarks() {
        return internalMarks;
    }
}

// CIE/Student.java
package CIE;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    // Parameterized constructor
    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }

    // Default constructor
    public Student() {
        this("", "", 0);
    }

    // Getter for name
```

```
    public String getName() {
        return name;
    }
}

// SEE/External.java
package SEE;

import CIE.Student;

public class External extends Student {
    protected int[] seeMarks;

    public External(String usn, String name, int sem, int[] seeMarks) {
        super(usn, name, sem);
        this.seeMarks = seeMarks;
    }

    public int[] getSeeMarks() {
        return seeMarks;
    }
}
```

WEEK7

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.

Program:

```
import java.util.Scanner;

class WrongAge extends Exception {
    public WrongAge() {
        super("Age Error");
    }

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

class InputScanner {
    protected Scanner scanner;

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

    public int nextInt() {
        return scanner.nextInt();
    }
}

class Father extends InputScanner {
    protected int fatherAge;

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

```
        fatherAge = super.nextInt();
        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }

    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 = super.nextInt();

        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to 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 ExceptionHandlingDemo {
    public static void main(String[] args) {
        try {
            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

WEEK8

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.

Program:

```
class DisplayThread extends Thread {
    private String message;
    private int interval;

    public DisplayThread(String message, int interval) {
        this.message = message;
        this.interval = interval;
    }

    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(interval);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}

public class week8 {
    public static void main(String[] args) {
        DisplayThread thread1 = new DisplayThread("BMS College of Engineering",
10000); // 10 seconds
        DisplayThread thread2 = new DisplayThread("CSE", 2000); // 2 seconds

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

WEEK9

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.

Program:

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

class SwingDemo{
   SwingDemo(){
      // create jframe container
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 200);
      jfrm.setLayout(new FlowLayout());
      // to terminate on close
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

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

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

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

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

```
// add in order :)
jfrm.add(err);  // to display error message
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

ActionListener l = new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        // No need to handle action event from text fields separately
        // because we handle it in the button's action listener
    }
};
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());
            if (b == 0) {
                throw new ArithmeticException();
            }
            int ans = a/b;

            alab.setText("A = " + a);
            blab.setText("B = " + b);
            anslab.setText("Ans = "+ ans);
            err.setText(""); // Clear any previous error message
        }
        catch(NumberFormatException e){
            clearLabels();
            err.setText("Enter Only Integers!");
        }
        catch(ArithmeticException e){
            clearLabels();
```

```
                err.setText("B should be NON zero!");
            }
        }

        private void clearLabels() {
            alab.setText("");
            blab.setText("");
            anslab.setText("");
        }
    });

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

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

WEEK10

10. Demonstrate Inter process Communication and deadlock

Program:

A. Deadlock

```
public class DeadlockExample {

    public static void main(String[] args) {
        final SharedResource sharedResource = new SharedResource();

        Thread process1 = new Thread(() -> {
            try {
                sharedResource.method1();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        });

        Thread process2 = new Thread(() -> {
            try {
                sharedResource.method2();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        });

        process1.start();
        process2.start();
    }
}

class SharedResource {
    private final Object lock1 = new Object();
    private final Object lock2 = new Object();

    public void method1() throws InterruptedException {
        synchronized (lock1) {
```

```
            System.out.println("Method 1 acquired lock1");
            Thread.sleep(1000);

            synchronized (lock2) {
                System.out.println("Method 1 acquired lock2");
                // Perform some task using both lock1 and lock2
            }
        }
    }

    public void method2() throws InterruptedException {
        synchronized (lock2) {
            System.out.println("Method 2 acquired lock2");
            Thread.sleep(1000);

            synchronized (lock1) {
                System.out.println("Method 2 acquired lock1");
                // Perform some task using both lock1 and lock2
            }
        }
    }
}
```

B. InterprocessCommunication:

```
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("\nNotify 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("\nNotify 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++);
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

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++;
            try {
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

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