# B.M.S COLLEGE OF ENGINEERING BENGALURU

Autonomous Institute, Affiliated to VTU

## LAB REPORT

## 23CS3PCOOJ

Submitted in partial fulfilment of the requirements for Lab

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:


**PRAJWAL K K**

**(1BM22CS199)**

Department of Computer Science and Engineering,

B.M.S College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019


2023-2024.

# INDEX

# INDEX —: JAVA:—

NAME: PRAJWAL.K. STD _____ SEC: _____ ROLL NO: _____

| S.No. | Date | Title | Page No. | Teacher's Sign/ Remarks |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Bafna Gold
Date: 12/12/23

01 ParseInt :-

{Area of a Rectangle and Verify the Same with Various inputs (length, breadth).

Sol?

```
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);
}
}
```

02 Scanner :-

Sol?

```
import java.util.Scanner;
class HelloWorld
{
public static void main(String args[])
{
int a; float b; String s;
Scanner in = new Scanner(System.in);
System.out.println("Enter a String");
s = in.nextLine();
System.out.println("You entered String" + s);
System.out.println("Enter a integer");
a = in.nextInt();
```

```
System.out.println("You entered integer "+a);
System.out.println("Enter a float");
b=in.nextFloat();
System.out.println("You entered float "+b);
}
}
```

03. Factorial:-

```
import java.util.Scanner;
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);
}
}
```

04. Palindrome:-

```
import java.util.Scanner;
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");
}
}
```

05. Sum of digits in a 5 digit number.

```
import java.util.Scanner;
class sum of digits
{
public static void main (String args[])
{
long number, sum;
Scanner sc = new Scanner (System.in);
System.out.println("Enter a 5-digit number:");
number = sc.nextLong();
```

```
for (sum =0; number!=0; number=
                    number/10)
  {
   sum=sum + number%10;
  }
  System.out.println("Sum of digits:"+sum);
 }
}
```

06. Array:-

```
class AutoArray
{
public static void main(String args[])
 {
 int month_days[]={31,28,31,30, 31,30,31,
                   31,30,31,30,31};
 System.out.println("April has"
                   + month_day[3]+"days");
  }
}
```

07. Type Conversion:-

```
class promote
 {
 public static void main(String args[])
  {
  byte b=42;
  char c='a';
  short s=1024;
  int i=5000;
```

```
float f = 6.74f;
double d = 0.1234;
    double result = (f * b) + (i / c) - (d * s);
System.out.println((f * b) + " + " + (i / c) + " - "
                                    + (d * s));
}
}
```

double result = (f * b) + (i / c) - (d * s);

☆ => Quadratic equation $ax^2+bx+c=0$

```
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 values
                        of 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 & 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 & distinct");
System.out.println("Root1="+r1+"Root2="+r2);
}
else if(d<0)
{
System.out.println("Roots are imaginary");
r1=(-b)/(2*a);
r2=Math.sqrt(-d)/(2*a);
System.out.println("Root1="+r1+"+i"+r2);
System.out.println("Root1="+r1+"-i"+r2);
}
}
}
class Quadratic main
{
public static void main(String args[])
{
Quadratic q=new Quadratic();
q.getd();
q.compute();
}
}
```

:- Output :-

(i) ~~Enter the coefficients of a, b, c:~~
~~0 0 3~~
~~Roots are real and distinct~~
~~Root 1 = Root 2 = 1~~

(ii) Enter the coefficients of a, b, c:
0 2 3
Not a quadratic equation
Enter a non zero value of a

(iii) Enter the coefficients of a, b, c:
1 2 1
Roots are real and Equal
Root 1 = Root 2 = -1

(iii) Enter the coefficient of a, b, c:
1 -3 2
Roots are real & distinct
Root 1 = 2 Root 2 = 1

(iv) Enter the coefficient of a, b, c:
1 1 2
Roots are imaginary
Root 1 = 0.0 + i 0.322875
Root 2 = 0.0 - i 0.322875

Bafna Gold
Date: 19/12/23

Q. Develop a Java program to create a class Student with members USN, name, an array credits and an array marks.

Sol?

```
import java.util.Scanner;
class Subject
{
  int subjectMarks;
  int Credits;
  int grade;
}
Public class Student
  {
  Subject[] subject;
  String name;
  String USN;
  double SGPA;
  Scanner S;

  Student()
  {
  int i;
  Subject = new Subject[9];
  for (i=0; i<9; i++)
    subject[i] = new subject();
  S = new Scanner(System.in);
  }
  public void getStudentDetails()
  {
  System.out.println("Enter Name:");
  name = S.nextLine();
  System.out.println("Enter USN:");
  name = S.nextLine();
```

```
}
public void getMarks()
{
    for(int i=0; i<8; i++)
    {
        System.out.println("Enter marks for
        Subject" +(i+1)+":");
        Subject[i].SubjectMarks = S.nextInt();
        System.out.println("Enter Credits for
        Subjects" +(i+1)+":");
        if(Subject[i].SubjectMarks >100);
        {
        System.out.println("Invalid marks
            entered");
            i--;
        }
        else if(Subject[i].SubjectMarks >=90)
        {
            Subject[i].grade =10;
        }
        else if(Subject[i].SubjectMarks >=80)
        {
            Subject[i].grade=9;
        }
        else if(Subject[i].SubjectMarks >=70)
        {
            Subject[i].grade=8;
        }
        else if(Subject[i].SubjectMarks >=60)
        {
            Subject[i].grade=7;
        }
```

```
else if(Subject[i].SubjectMarks >=50)
 {
 Subject[i].grade = 6;
 }
else if(Subject[i].SubjectMarks >=40)
 {
 Subject[i].grade = 0;
 }
}
}
public void computeSGPA()
 {
double totalCredits=0;
 double totalGradePoints=0;
 for(int i=0 ;i<8; i++)
 {
    totalCredits += Subject[i].credits;
    totalGradepoints += Subject[i].grade *
                        Subject[i].credits;
 }
 SGPA = totalGradePoints / totalCredits;
 }
 public static void main (String[] args)
  {
  Student S1 = new Student();
  S1.getStudentDetails();
  S1.GetMarks();
  S1.ComputeSGPA();
  System.out.println("Name : "+S1.name);
  System.out.println("USN: "+S1.usn);
  System.out.println("SGPA:"+S1.SGPA);
 }
}
```

:- Output :-

Enter Name:
Prajwal K K
Enter USN:
1BM22CS199
Enter Marks for Subject 1:
89
Enter Credits for Subject 1:
4
Enter Marks for Subject 2:
86
Enter Credits for Subject 2:
4
Enter Marks for Subject 3:
79
Enter Credits for Subject 3:
4
Enter Marks for Subject 4:
90
Enter Credits for Subject 4:
3
Enter Marks for Subject 5:
93
Enter Credits for Subject 5:
3
Enter Marks for Subject 6:
91
Enter Credits for Subject 6:
3
Enter Marks for Subject 7:
87
Enter Credits for Subject 7:
2

Bafna Gold
Date: Page:

Enter marks for subject 8:
98
Enter credits for subject 8:
1
Name: Prajwal KK
USN: 1BM22CS199
SGPA: 9.25

19/12/23

26/12/2023 LAB-03:-

Q. Create a class Book which contains four members: name, author, price, num_pages. 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.

Sol?

```
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 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.numPgs
                                + "\n";
return name + author + price + numPages;
}
}
public 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();
Book[] b = new Book[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 author of
      the book:");
author = S.next();
```

```
System.out.println("Enter the price of
the book: ");
price = S.nextInt();
System.out.println("Enter the number
of pages of the book: ");
numPages = S.nextInt();

b[i] = new Book(name, autha, price,
                numPages);
}
for(int i=0; i<n; i++)
{
    System.out.println(b[i].toString());
}
}
```

Output :-

```
Enter the number of books:
2
Enter name of the book:
ABC
Enter author of the book:
Arvind
Enter the price of the book:
300
Enter the number of pages of the book:
450
```

Enter name of the book:
XYZ
Enter author of the book:
Ravi
Enter the price of the book:
500
Enter the number of pages of the book:
390

Book name: ABC
Author name: Arvind
Price: 300
Number of pages: 450

Book name: XYZ
Author name: Ravi
Price: 500
Number of pages: 390,

26/12/23

02/01/2023

## LAB-04:-

Q 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.

Sol?

```
import java.util.Scanner;

abstract class Shape
{
 int length;
 int breadth;

  abstract void printArea();
}

class Rectangle extends Shape:
 {
  void printArea()
  {
  Scanner s = new Scanner(System.in);
  System.out.println("Enter length and
  breadth of rectangle :");
  length = s.nextInt();
  breadth = s.nextInt();
  System.out.println(" The Area of
    Rectangle is" + (length *breadth));
  }
}
```

```
class Triangle extends Shape
{
void printArea()
{
 Scanner S = new Scanner (System.in);
System.out.println("Enter the dimension
 of Triangle :");
 length = S.nextInt();
 breadth = S.nextInt();
 System.out.println("The Area of
 Triangle is" + (0.5 * length * breadth));
 }
}

class Circle extends Shape
{
 void printArea()
{
Scanner S = new Scanner (System.in);
System.out.println("Enter the dimension
 of Circle :");
length = S.nextInt();
System.out.println("The Area of Circle
  is" + (3.14 * length * length));
  }
}

public class Main
{
  public static void main (String args[])
  {
Shape shape;
```

```
shape = new Rectangle();
shape.printArea();
shape = new Triangle();
shape.printArea();
shape = new Circle();
shape.printArea();
    }
}
```

∴ Output :-

```
Enter the dimensions of the Rectangle :
7 9
Area of rectangle is 63
Enter the dimensions of the Triangle :
5 8
Area of triangle is 20
Enter the dimension of the Circle :
9
Area of circle is 254.34
```

02/01/24

09/01/24 LAB- "05" :-

Q Create a class Account that stores: customer name, account number and type of account. From this derive the classes Current-account and Saving-account 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.

Sol?

```
import java.util.Scanner;

Class Amount
{
  String customerName;
  int accountNumber;
  String accountType;
  double balance;

  Account(String customerName, int accountNumb;
  String accountType, double balance)
  {
    this.customerName = customerName;
    this.accountNumber = accountNumber;
    this.accountType = accountType;
```

```
        this.balance = balance;
    }
void deposit (double amount)
    {
    balance += amount;
    System.out.println ("Deposit of" + amount +
" successful");
    }

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

void withdraw (double amount)
    {
    if (balance - amount < 0)
        {
        System.out.println ("Insufficient balance");
        return;
        }
    balance -= amount;
    System.out.println ("Withdrawal of "
        + amount + " Successful");
    }
}

class SavingAccount extends Account
    {
    SavingAccount (String customerName,
int accountNumber, String accountType,
double balance)
```

```
{
super (customerName, accountNumber, accountType,
  balance);
}

Void compoundInterest()
 {
 double rate = 0.05;
 double time = 1.0;
 double interest = balance * Math.pow(1+
 rate, time) - balance;
  balance += interest;
  System.out.println("Interest of " + interest +
   " added");
 }

Void withdraw (double amount)
 {
  if(balance - amount < 0)
  {
    System.out.println("Insufficient
                        balance.");
    return;
  }
  balance -= amount;
  System.out.println("Withdrawal of "
  + amount + " successful");
  }
}

class CurrentAccount extends Account
  {
```

```
    double minimumBalance = 1000;
    double ServiceCharge = 50;

    CurrentAccount (String CustomerName,
int accountNumber, String accountType,
double balance)
    {
    Super (CustomerName, accountNumber,
accountType, balance);
    }

Void withdraw (double amount).
    {
    if (balance - amount < minimumBalance)
    {
    System.out.println ("Insufficient
                        balance");
    return;
    }
    balance -= amount;
System.out.println ("Withdrawal of"
+ amount + "Successful");
    }

Void imposeServiceCharge ()
    {
    if (balance < minimumBalance)
    {
    balance -= ServiceCharges;
    System.out.println ("Service
Charge of " + ServiceCharge + " imposed");
    }
}
}
```

```
public class Bank
{
  public static void main (String args [])
  {
   Scanner scanner = new Scanner (System.in);
   System.out.println ("Enter Customer name: ");
   String CustomerName = Scanner.nextLine ();
   System.out.println ("Enter account number: ");
   int accountNumber = scanner.nextInt();
   System.out.println ("Enter account type
     (Savings / Current): ");
   String accountType = scanner.next();
   System.out.println ("Enter initial balance: ");
   double balance = scanner.nextDouble();

   Account account;
    if (accountType.equals ("savings "))
    {
     account = new SavingsAccount
     (CustomerName, accountNumber,
      accountType, balance);
     }
     else
      {
       account = new CurrentAccount
     (CustomerName, accountNumber,
      accountType, balance);
     }

while (true)
```

```
{
System.out.println ("\n 1. Deposit ");
System.out.println ("\n 2. Display balance ");
System.out.println ("3. Compute and
                    deposit interest ");
System.out.println ("4. Withdrawal ");
System.out.println ("5. Exit ");
System.out.println (" Enter choice: ");
  int choice = Scanner.nextInt();
Switch (choice)
  {
  Case 1:
    System.out.println (" Enter amount to
      be deposit ");
    double amount = scanner.nextDouble
    ();
   account.deposit (amount);
    break;
  Case 2:
    account.displayBalance ();
    break;
  Case 3:
    if (account instance of SavingAccount)
    {
    ((SavingAccount) account).Compound
     Interest ();
     }
    else
     {
     System.out.println (" Interest not
available for current account ");
     }
```

```
break;

Case 4:
 System.out.println("Enter amount to
 withdraw: ");
 amount = Scanner.nextDouble();
 account.withdraw(amount);
 if (account instance of CurrentAccount)
 {
 ((CurrentAccount) account).imposeServiceCharge();
 }
 break;

Case 5:
 System.exit(0);
 }
 }
}
}
```

## :- Output :-

```
Enter customer name: Prajwal K
Enter account number: 199
Enter account type (savings/Current): savings
Enter initial balance: 2890

1. Deposit
2. Display balance
3. Compute and deposit interest
4. Withdraw
5. Exit
```

Enter choice : 3
Interest of 144.5 added

1. Deposit
2. Display balance
3. Compute and deposit interest
4. Withdraw
5. Exit
Enter choice: 4
Enter amount to withdraw : 1770
withdraw of 1770.0 successful

1. Deposit
2. Display balance
3. Compute and deposit interest
4. withdraw
5. Exit
Enter choice : 2
Balance : 1264.5

1. Deposit
2. Display balance
3. Compute and deposit interest
4. withdraw
5. Exit
Enter choice: 5

11/01/24

Bafna Gold
Date: Page:

# LAB-'06' Strings :-

01. Demonstrate various string constructor with proper java program.

Sol?

∴ Output :-

// String
Hello, Man!
Java;

02 Demonstrate string length, string literal, String concat

Sol? ∴ output :- //
length = 8
concatenated : Prajwal k Programming.

03. Demostrate tostring ()

Sol?

∴ output :-

Student : Demo

04 Using getchars(), extract Bmsce from "Welcome to Bmsce college"

Sol?

∴ Output :-

Extracted : Bmsce

05. Demostrate getBytes(), toCharArray() with proper java programs

Sol?

∴ Output :-

Bytes : Prajwal, K!
Chars : Prajwal, K!

06. Check the following output and write the java programs using string functions

Bmsce equals Bmsce -> true
Bmsce equals college -> false
Bmsce equals BMSCE -> false
Bmsce equalsIgnoreCase BMSCE -> true

Sol?

∴ Output :-

true
false
false
true

07. Using regionmatches() find the substring "Bmsce College" from the string "Welcome to Bmsce College of Engineering" if matches display substring is matched otherwise display not matched

:Output:-

Substring is matched

08. Demonstrate startswith() to give output true and false [2nd ends with()]

Sol?

:Output:-

true
true.

09. Demonstrate endswith() to give output true and false.

Sol?

:Output:-

endswith()
false

10. Demostrate a java program to show the output for equals() versus ==

Sol?

equals() versus ==
World equals world -> true
World == World -> false.

11. Write a java program to perform sorting for alphabets using Comparator()

Sol) Compare to()

apple
ball
cat
dog
fish
gum
hen
ice
jug
kite
man
net
orange
parrot
quen
ring
star
yatch
zeu

12 Write a java program to perform sorting of number from 10 to 1 using comparator

Sol)

Comparator()

1
2
3
4
5
6
7
8
9

13 WAP Using substring(), indexof(), +, for replacing "was" to "is". "Thwas was a test. Thwas was, too".

Sol?

Substring(), indexof()
thwas was a test. Thwas was, too.
This was a test. Thwas was, too
This is a test. Thwas was, too
This is a test. This was, too
This iso test. This is, too

14. WAP to demonstrate concat() for S1: "hello" and S2: "world"

Sol?

concat()
helloworld

15. WAP to demonstrate replace(). Replace "college" with "commege".

Sol?

replace()
commege

16. WAP to demonstrate trim() for " Hello Friends"

Sol?

trim()
Hello Friends.

17. Design a class which represents a student. Every student record is made up of the following fields.

(i) Regist no (int)
(ii) Full Name (string)
(iii) Semester (short)
(iv) CGPA (float)

Sol?

Student Records for student 1:
Registration Number : 8
Full Name : Prajwalk
Semester : 2
CGPA : 9.4

Student Record:
Registration Number : 1
Full Name : Prajwalk
Semester : 2
CGPA : 9.4

18. Demonstrate String buffer functions like setlength(), charAt(), setcharAt(), getchars(), append(), Insert(), reverse(), delete(), deletecharAt(), Replace(), substring() with simple java programs

Sol?

String Buffer

After setlength(5): Hello
Character at index 2 : H
After setcharAt(0, 'x') : xello
Getchars(0, 5) : xello.

19. WAP to create an abstract class Bird with abstract methods fly() and makeSound(). Create subclass Eagle and Hawk that extends the Bird class and implement the respective methods to describe how each bird flies and makes a sound.

Sol:

Eagle Actions:
Eagle is soaring high in the sky
Eagle screeches loudly

Hawk Actions:
Hawk is gliding through the air
Hawk makes a sharp cry.

20. WAP to create an abstract class Shape with abstract methods calculateArea() and calculatePerimeter(). Create subclass Circle and Triangle that extends the Shape class and implement the respective methods to calculate the area and perimeter of each shape.

Sol:

Circle. Area: 78.5398; Perimeter: 31.4159
Triangle. Area: 6.0 ; Perimeter: 12.0

# LAB-06 :—

Q) Create a package CIE which has two classes student and Internals. The class Student has members 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 declare the final marks of n students in all five courses.

Sol?

```
// 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 input Student Details ()
  {
   Scanner S = new Scanner (System.in);
   System.out.println("Enter the USN");
     USN = S.nextLine();
   System.out.println("Enter the Student
name");
```

```
    name = S.nextLine();
    System.out.println("Enter the Semester:");
    Sem = S.nextInt();
    }
Public void displayStudentDetails()
    {
    System.out.println("USN = "+Usn);
    System.out.println("Student name = "+name);
    System.out.println("Semester = "+Sem);
    }
}

// Internals.java

Package CIE;
import CIE.Student;
import java.util.Scanner;
Public class Internals extends Students
{
Protected int marks[] = new int[5];
Public void inputCIEmarks()
    {
    Scanner S = New Scanner(System.in);
    System.out.println("Enter the mark of
    each subject");

for(int i=0;i<5;i++)

marks[i] = S.nextInt();
    }
}
```

```
// Externals.java

Package SEE;
import CIE.Internals;
import java.util.Scanner;

Public class Externals extends Internals
{
Protected int marks[];
Protected int final Marks[];
  Public Externals()
  {
   marks = new Int[5];
   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("Subject "+(i+)+
    marks :");
marks[i] = s.next.Int();
   }
  }
  Public void calculate Final marks()
  {
   for(int i=0; i<5; i++)
   final Marks[i] = marks[i]/2 +
     Super.marks[i];
   }
Public void display Final marks()
```

```
{
display Student Details ();
for (int i=0 ; i<5 ; i++)
System out.println ("Subject "+(i+1)+
": "+ final Marks [i]);
  }
}

import SEE.externals;
class Main
{
 public static void main (String args[])
 {
  int num of Students =2;
  Externals final Marks [] = new
  Externals [num of Students ];
  for [int =0; i<nom of Student ; i++)
  {
   final Marks [i] = new Externals ();
   final Marks [i]. input Student Details ();
   System. out. println ("Enter CIE marks ");
   final Marks [i]. input (IE marks ();
   System. out. println ("Enter SEE marks ");
   final Marks [i]. input SEE marks ();
  }
System. out. println ("Displaying data: \n ");
  for (int i=0; i<num of Student ; i++)
  {
     final Marks [i]. Calculate Final
Marks ();
  final Marks [i]. display Final Marks ();
  }
 }
}
```

## ÷ Output :-

```
Enter the USN
199

Enter the Student Name
Prajwalk

Enter the semester 3

Enter CIE marks
Enter the marks of each Subject:
 48
 49
 47
 35
 50

Enter SEE marks :
Subject 1 marks : 45
Subject 2 marks : 47
Subject 3 marks : 49
Subject 4 marks : 50
Subject 5 marks : 46

Enter the USN
 211
Enter the student name
Ravi
Enter the semester
 3
```

Enter CIE marks
Enter the marks of each subject
44
42
47
46
41

Enter SEE marks
Subject 1 marks : 49
Subject 2 marks : 44
Subject 3 marks : 47
Subject 4 marks : 50
Subject 5 marks : 42

Display Data:

USN = 1BM22CS199
Student name = Prajwalk
Semsta = 3
Subject 1 : 70
Subject 2 : 72
Subject 3 : 71
Subject 4 : 60
Subject 5 : 73

USN = 211
Student name : Ravi
Semsta : 3
Subject 1 : 93
Subject 2 : 86
Subject 3 : 94

Subject 4: 96
Subject 5: 83

//.

LAB-07:-

Q. WAP that demonstrates handling of exceptions in inheritance tree. Create a base class called "Fatha" 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 fatha and son's age and throws an exception if son's age is >= father's age.

Sol

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

Class Father
  {
  int age;
   Father (int age) throws WrongAge
   {
    if(age<0)
     {
      throw new WrongAge("Age cannot be negative");
     }
```

```
        this.age=age;
    }
}

Class Son extends Father
{
    int Sage;
Son(int Father Age , int Son Age)
throws Wrong Age
    {
        Super(Father Age);
    if(Son Age>=Father Age)
    {
        throw new
Wrong Age("Son's Age should be
    less than Father's Age");
    }
        this.Sage = SonAge;
    }
}
Public class Age
{
  public static void main(String args[])
{
Scanner SC=new Scanner(System.in);
    try{
System.out.println("Enter the Fathers
    age :");
        int b=SC.nextInt();
        Son Son=new Son(a,b);
    }
    catch(Wrong Age e)
```

```
        {
System.out.println("Exception :" +
        e.getMessage());
    }
  }
}
```

∴ Output :-

» Enter the Father's age
58
Enter the sons age
37

Sons age is 37

» Enter the Fathers age
27
Enter the Sons age
27

Father's age should be greater than Son's age.

» Enter the Fathers age
18
Enter the sons age
35

Father's age should be greater than Son's age.

31.01.24

LAB-08:-

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.

Sol?

```
Public class Thread Example
{
Public static void main (string [] args)
{
Thread thread1 = new Thread (new
  Display Every Ten seconds ());
Thread thread2 = new Thread (new
  Display Every Two seconds ());
  thread1.start();
  thread2.start();
  }
}
```

```
Class Display Every Ten seconds implements
  Runnable {
  public void run()
  {
  while (true)
  {
  try {
  Thread.sleep(1000);
  System.out.println("Bms college of
    Engineering");
  }
```

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

Class DisplayEveryTwoSeconds implements
  Runnable
  {
    public void run()
    {
      while(true)
      {
      try
        {
        Thread.sleep(2000);
        System.out.println("CSE");
        }
      catch (InterruptException)
      {
      e.printStackTrace();
        }
      }
    }
  }
```

Output:-

CSE
CSE
CSE
CSE
Bms college of Engineering
CSE
CSE
CSE
CSE
Bms College of Engineering
CSE
CSE
CSE
CSE
Bms College of Engineering.

//

LAB-10:-

Q. Write a Program to Demonstrate Inter process Communication and deadlock.

Sol?

```
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("\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("Interrupted
Exception Caught");
}
this.n=n;
valueset = true;
System.out.println("Put:"+n);
System.out.println("\n Intimate
Consumer\n");
}
}

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

```
}
}
}

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 :-

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

//.

Bafna Gold
Date: Page:

## Deadlock :-

```
class A {
Synchronized void foo (B b) {
  String name =
  Thread.Current.Thread().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.Current Thread().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();
  t.start();
  a.foo(b);
System.out.println("Back in
main thread");
```

```
}

Public void run()
{
b.bar(a);
System.out.println("Back in other
thread");
}
Public Static void main(String args[])
{
new Dead lock();
}
}
```

:- Output :-

```
Main Thread entered A.foo
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.
```

13.02.24

20/02/2024

LAB 09

Q Write a program that creates a user interface to perform integer divisions.

SOL?

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

Class SwingDemo
{
 SwingDemo()
 {
  JFrame jfrm = new JFrame("Divider App");

  jfrm.setSize(275, 150);
  jfrm.setLayout(new FlowLayout());
  jfrm.setDefaultCloseOperation(JFrame.
   EXIT_ON_CLOSE);

: JLabel jlab = new JLabel("Enter the
    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);

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

   button.addActionListener(new ActionListener()
   {
   public void actionPerformed(Action
Event listener)
   {
   public void actionPerformed(Action
Event ext){
   try{
   int a = Integer.ParseInt(ajtf.getText());
   int b = Integer.ParseInt(bjtf.getText());
   int ans = a(b);
```

```
alab.setText("\nA="+a);
blab.setText("\nB="+b);
anilab.setText("\nAm="+
am);
}
catch(NumberFormatException e)
{
alab.setText("");
blab.setText("");
anilab.setText("");
err.setText("Enter only Integers!
");
}
catch(ArithmeticException e)
{
alab.setText("");
blab.setText("");
anilab.setText("");
err.setText("B should be NON
zero!");
}
}
});
jfrm.setVisible(true);
}
public static void main(String args[])
SwingUtilities.invokeLater(new
Runnable()
{
public void run()
{
new SwingDemo();
}
});
}
```

## Output :-

Enter the divider and dividant

[48] [6]

[Calculate] A=48 B=6 Ans=8

## LAB PROGRAM 1 {12/DEC/2023}

**Develop a Java program that prints all real solutions to the quadratic equation ax 2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the 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("Roo1 = Root2 = " + r1);
}
else if(d&gt;0)
{
r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a);
System.out.println("Roots are real and distinct");
System.out.println("Roo1 = " + r1 + " Root2 = " + r2);
}
else if(d&lt;0)
{
System.out.println("Roots are imaginary");
r1 = (-b)/(2*a);
r2 = Math.sqrt(-d)/(2*a);
System.out.println("Root1 = " + r1 + " + i"+r2);
System.out.println("Root1 = " + r1 + " - i"+r2);
}
}
}
class QuadraticMain
{
public static void main(String args[])
{
Quadratic q = new Quadratic();
q.getd();
q.compute();
}
}
```

## LAB PROGRAM 2 {19/DEC/2023}

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

```
import java.util.Scanner;

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

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

    Student() {
        int i;
        subject = new Subject[9];
        for (i = 0; i < 9; i++)
            subject[i] = new Subject();
        s = new Scanner(System.in);
    }

    public void getStudentDetails() {
        System.out.print("Enter Name: ");
        name = s.nextLine();
        System.out.print("Enter USN: ");
        usn = s.nextLine();
    }

    public void getMarks() {
        for (int i = 0; i < 8; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            subject[i].subjectMarks = s.nextInt();
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            subject[i].credits = s.nextInt();
```

```
            if (subject[i].subjectMarks > 100) {
                System.out.println("Invalid marks entered. Please enter marks between 0 and 100.");
                i--;
            } else if (subject[i].subjectMarks >= 90) {
                subject[i].grade = 10;
            } else if (subject[i].subjectMarks >= 80) {
                subject[i].grade = 9;
            } else if (subject[i].subjectMarks >= 70) {
                subject[i].grade = 8;
            } else if (subject[i].subjectMarks >= 60) {
                subject[i].grade = 7;
            } else if (subject[i].subjectMarks >= 50) {
                subject[i].grade = 6;
            } else if (subject[i].subjectMarks >= 40) {
                subject[i].grade = 5;
            } else {
                subject[i].grade = 0;
            }
        }
    }

    public void computeSGPA() {
        double totalCredits = 0;
        double totalGradePoints = 0;
        for (int i = 0; i < 8; i++) {
            totalCredits += subject[i].credits;
            totalGradePoints += subject[i].grade * subject[i].credits;
        }
        SGPA = totalGradePoints / totalCredits;
    }

    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);
    }
}
```

## LAB PROGRAM 3 {26/DEC/2023}

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

```
import java.util.Scanner;

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

        name = "Book name: " + this.name + "\n";
        author = "Author name: " + this.author + "\n";
        price = "Price: " + this.price + "\n";
        numPages = "Number of pages: " + this.numPages + "\n";

        return name + author + price + numPages;
    }
}

public class 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();
```

```
        Book[] b = new Book[n];

        for (int i = 0; i < n; i++) {
            System.out.println("Enter name of the book: ");
            name = s.next();

            System.out.println("Enter author of the book: ");
            author = s.next();

            System.out.println("Enter the price of the book: ");
            price = s.nextInt();

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

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

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

## LAB PROGRAM 4 {02/JAN/2024}

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

```
import java.util.Scanner;
class InputScanner {
    Scanner s = new Scanner(System.in);
    int getInput(String prompt) {
        System.out.println(prompt);
        return s.nextInt();
    }
}
class shape extends InputScanner {
    double dim1;
    double dim2;
    shape(double a, double b) {
        dim1 = a;
        dim2 = b;
    }
}
class Rectangle extends shape {
    Rectangle() {
        super(0, 0);
        dim1 = getInput("Enter length");
        dim2 = getInput("Enter breadth");
    }
    double area() {
        System.out.println("Inside Area for Rectangle.");
        return dim1 * dim2;
```

```
        }
}
class Triangle extends shape {
    Triangle() {
        super(0, 0);
        dim1 = getInput("Enter length");
        dim2 = getInput("Enter base");
    }
    double area() {
        System.out.println("Inside Area for Triangle.");
        return dim1 * dim2 / 2;
    }
}
class Circle extends shape {
    Circle() {
        super(0, 0);
        dim1 = getInput("Enter the radius");
        dim2 = dim1;
    }
    double area() {
        System.out.println("Inside Area for Circle.");
        return Math.PI * dim1 * dim2;
    }
}
public class Areas {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle();
        System.out.println("Area of Rectangle: " + rectangle.area());

        Triangle triangle = new Triangle();
        System.out.println("Area of Triangle: " + triangle.area());
```

```
        Circle circle = new Circle();
        System.out.println("Area of Circle: " + circle.area());
    }
}
```

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

**a) Accept deposit from customer and update the balance.**

**b) Display the balance.**

**c) Compute and deposit interest**

**d) Permit withdrawal and update the balance**

**Check for the minimum balance, impose penalty if necessary and update the balance.**

```
import java.util.Scanner;
class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;
    Account(String name, int number, String type, double initialBalance) {
        customerName = name;
        accountNumber = number;
        accountType = type;
        balance = initialBalance;
    }
    void deposit(double amount) {
        balance += amount;
```

```
}
    void displayBalance() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Customer Name: " + customerName);
        System.out.println("Account Type: " + accountType);
        System.out.println("Balance: INR " + balance);
    }
    void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawal of INR " + amount + " successful");
        } else {
            System.out.println("Insufficient funds");
        }
    }

    void computeInterest() {
    }
    void checkMinimumBalance(double minBalance, double serviceCharge) {
    }
}
class SavAcct extends Account {
    double interestRate = 0.05;
    SavAcct(String name, int number, String type, double initialBalance) {
        super(name, number, type, initialBalance);
    }
    void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest of INR " + interest + " added to the account");
    }
```

```
}
class CurAcct extends Account {
    double minBalance = 1000;
    double serviceCharge = 50;
    CurAcct(String name, int number, String type, double initialBalance) {
        super(name, number, type, initialBalance);
    }
    void checkMinimumBalance(double minBalance, double serviceCharge) {
        if (balance < minBalance) {
            System.out.println("Service charge of INR " + serviceCharge + " imposed");
            balance -= serviceCharge;
        }
    }
}
public class Bank {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of users: ");
        int numUsers = scanner.nextInt();
        Account[] accounts = new Account[numUsers];
        for (int i = 0; i < numUsers; i++) {
            System.out.println("\nUser " + (i + 1));
            System.out.print("Enter customer name: ");
            scanner.nextLine();
            String name = scanner.nextLine();
            System.out.print("Enter account number: ");
            int accNumber = scanner.nextInt();
            System.out.print("Enter initial deposit amount: INR ");
            double initialDeposit = scanner.nextDouble();
            System.out.print("Enter account type (Savings/Current): ");
            scanner.nextLine();
```

```
    String accType = scanner.nextLine();
    if (accType.equalsIgnoreCase("Savings")) {
        accounts[i] = new SavAcct(name, accNumber, accType, initialDeposit);
    } else if (accType.equalsIgnoreCase("Current")) {
        accounts[i] = new CurAcct(name, accNumber, accType, initialDeposit);
    } else {
        System.out.println("Invalid account type entered. Defaulting to Account.");
        accounts[i] = new Account(name, accNumber, "Account", initialDeposit);
    }
}
boolean exit = false;
while (!exit) {
    System.out.println("\nChoose an option:");
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Display Balance");
    System.out.println("4. Compute Interest (Savings only)");
    System.out.println("5. Exit");
    System.out.print("Enter your choice: ");
    int choice = scanner.nextInt();
    switch (choice) {
        case 1:
            System.out.print("Enter account number: ");
            int accNum = scanner.nextInt();
            System.out.print("Enter deposit amount: INR ");
            double depositAmount = scanner.nextDouble();
            for (Account acc : accounts) {
                if (acc.accountNumber == accNum) {
                    acc.deposit(depositAmount);
                }
            }
```

```
    break;
case 2:
    System.out.print("Enter account number: ");
    accNum = scanner.nextInt();
    System.out.print("Enter withdrawal amount: INR ");
    double withdrawAmount = scanner.nextDouble();
    for (Account acc : accounts) {
        if (acc.accountNumber == accNum) {
            acc.withdraw(withdrawAmount);
        }
    }
    break;
case 3:

    System.out.print("Enter account number: ");
    accNum = scanner.nextInt();
    for (Account acc : accounts) {
        if (acc.accountNumber == accNum) {
            acc.displayBalance();
        }
    }
    break;
case 4:
    System.out.print("Enter account number (for Savings account): ");
    accNum = scanner.nextInt();
    for (Account acc : accounts) {
        if (acc.accountNumber == accNum && acc instanceof SavAcct) {
            ((SavAcct) acc).computeInterest();
        }
    }
    break;
case 5:
```

```
                exit = true;
                break;
            default:
                System.out.println("Invalid choice. Please enter a valid option.");
        }
    }
  }
}
```

## LAB PROGRAM 6 {16/JAN/2024}

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

```
package CIE;
public class Student {
    public String usn;
    public String name;
    public int sem;
    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
}
package CIE;
public class Internals extends Student {
    public int[] internalMarks;
    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
}
```

```
package SEE;
```

```
import CIE.Student;
public class External extends Student {
    public int[] seeMarks;
    public External(String usn, String name, int sem, int[] seeMarks) {
        super(usn, name, sem);
        this.seeMarks = seeMarks;
    }
}

import CIE.Internals;
import SEE.External;
import java.util.Scanner;
public class FinalMarks {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = scanner.nextInt();
        Internals[] cieStudents = new Internals[n];
        External[] seeStudents = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for CIE of 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();
            int[] cieMarks = new int[5];
            System.out.print("Enter CIE marks for 5 courses: ");
            for (int j = 0; j < 5; j++) {
```

```
            cieMarks[j] = scanner.nextInt();
        }
        cieStudents[i] = new Internals(usn, name, sem, cieMarks);
    }
    for (int i = 0; i < n; i++) {
        System.out.println("Enter details for SEE of 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();
        int[] seeMarks = new int[5];
        System.out.print("Enter SEE marks for 5 courses: ");
        for (int j = 0; j < 5; j++) {
            seeMarks[j] = scanner.nextInt();
         }
        seeStudents[i] = new External(usn, name, sem, seeMarks);
    }
    System.out.println("\nFinal Marks of Students:");
    for (int i = 0; i < n; i++) {
        System.out.println("\nDetails of Student " + (i + 1));
        System.out.println("USN: " + cieStudents[i].usn);
        System.out.println("Name: " + cieStudents[i].name);
        System.out.println("Semester: " + cieStudents[i].sem);
        System.out.println("CIE Marks: ");
        for (int j = 0; j < 5; j++) {
            System.out.print(cieStudents[i].internalMarks[j] + " ");
        }
        System.out.println("\nSEE Marks: ");
```

```
            for (int j = 0; j < 5; j++) {
                System.out.print(seeStudents[i].seeMarks[j] + " ");
            }
        }
    }
}
```

## LAB PROGRAM 7 {23/JAN/2024}

**Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is >=father’s age.**

```
import java.util.Scanner;
class WrongAge extends Exception {
    public WrongAge(String 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 <= 0) {
            throw new WrongAge("Son's age cannot be negative or zero");
        }
```

```
            throw new WrongAge("Son's age cannot be greater than or equal to father's age");
        }
    }
}
 public class Main {
    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();
        }
    }
 }
```

## LAB PROGRAM 8 {30/JAN/2024}

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

```
class DisplayThread extends Thread {
    private String message;
    private int interval;
    private boolean running = true;
    public DisplayThread(String message, int interval) {
        this.message = message;
        this.interval = interval;
    }
    public void run() {
        while (running) {
            System.out.println(message);
            try {
                Thread.sleep(interval);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
    public void stopThread() {
        running = false;
    }
}
public class ThreadEx {
    public static void main(String[] args) {
        DisplayThread bmsThread = new DisplayThread("BMS College of Engineering", 10000);
        DisplayThread cseThread = new DisplayThread("CSE", 2000);
```

```
        bmsThread.start();
        cseThread.start();
        System.out.println("Press Enter to stop the threads...");
        try {
            System.in.read();
        } catch (Exception e) {
            e.printStackTrace();
        }
        bmsThread.stopThread();
        cseThread.stopThread();
    }
}
```

## LAB PROGRAM 9 {067/FEB/2024}

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

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo{
SwingDemo(){
JFrame jfrm = new JFrame("Divider App");
jfrm.setSize(275, 150);
jfrm.setLayout(new FlowLayout());
jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JLabel jlab = new JLabel("Enter the divider and divident:");
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
JButton button = new JButton("Calculate");
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
JLabel anslab = new JLabel();
jfrm.add(err); // to display error bois
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
```

```
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);
ActionListener l = new ActionListener() {
public void actionPerformed(ActionEvent evt) {
System.out.println("Action event from a text field");
}
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent evt) {
try{
int a = Integer.parseInt(ajtf.getText());
int b = Integer.parseInt(bjtf.getText());
int ans = a/b;

alab.setText("\nA = " + a);
blab.setText("\nB = " + b);
anslab.setText("\nAns = "+ ans);
}
catch(NumberFormatException e){
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("Enter Only Integers!");
}
catch(ArithmeticException e){
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("B should be NON zero!");
```

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

# LAB PROGRAM 10

## Demonstrate Inter process Communication and deadlock.

## IPC

```
class Q {
int n;
boolean valueSet = false;
synchronized int get() {
while(!valueSet)
try {
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
System.out.println("Got: " + n);
valueSet = false;
notify();
return n;
}
synchronized void put(int n) {
while(valueSet)
try {
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
this.n = n;
valueSet = true;
System.out.println("Put: " + 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);
System.out.println("Back in mainthread");
}
public void run() {
b.bar(a);
System.out.println("Back in other thread");
}
public static void main(String args[]) {
new Deadlock();
}
}
```