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 C

(1BM22CS198)

Department of Computer Science and Engineering,

B.M.S College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019

2023-2024.

INDEX

INDEX

NAME: Prajwal. C STD ______ SEC: CD ROLL NO: 1BM22CS198

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

1) Parse Int

```
class Rectangle Area{
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 = " + lengt
Syetem. out. println("breadth of rectangle = " + breadt
System. out. println ("area of rectangle = " + area)
}}
```

Output

```
javac Rectangle Area. java
java Rectangle Area 10 8
length of rectangle = 10
breadth of rectangle = 8
area of rectangle = 80.
```

2) Scanner

```
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();
Sytem.out.println("You entered astring "+s);
System.out.println("You entered an 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);
}
}
```

3) Factorial of a given number

```
class factorial {
 public static void main (String args[])
 {
   int fac = 1;
   Sytem.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;
   }
   Sytem.out.println("The factorial : "+fact);
 }
}
```

4) Palindrome

```
class palindrome{
public static void main (String args[])
{
int n, t, rem, rev = 0;
Scanner sc = new Scanner (System.in);
Sytem. out. println ("Ender 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");
}
}
}
```

5) Sum of digits.

```
class sum of digits {
  public static void main (String args []) {
   long number, sum;
   Scanner sc = new Scanner (System.in);
   Sytem.out. print ("Enter a 5-digit number");
   number = sc. next Long ();
  for (sum = 0; number != 0; number = number / 10){
    sum = sum + number % 10;
  }
 System.out.println ("Sum of digits : " + sum);
 }
}
```

6) Arrays - 1D

```
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
_days[3]= 'days.");}
}
```

7) Type conversion

```
class promote {
    public static void main (String args[]){
     byte b=42;
     char c='a';
     short s=1024;
     int i = 5000;
     float f= 6.77f;
     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);~~

12/12/23

Quadratic Equation

```
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 equ
       System.out.println("Enter a non zero valu
            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("Root 1-Root 2 = "+r1);
}
else if (d>0)
 {
     r1 = ((-b) + (Math. sqrt (d))) / (double) (2*a);
   r2 = ((-b) - (Math. sqrt (d))) / (double) (2*a);
  System. out. println ("Roots are real and distinct");
  System. out. println ("Root1 = "+r1 + "Root2 = "
      + r2);
 }
else if (d<0)
 {
    System. out. println ("Roots are imaginary");
    r1 = (-b) / (2*a);
    r2 = Math. sqrt (-d) / (2*a);
    System. out. println ("Root 1 = "+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 ();
   }
}
```

Output

i)

```
Enter the coefficients of a, b, c:
 1  -3  2
Roots are real and distinct
Root 1 = 2   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 coefficient of a, b, c
 1  2  1
Roots are real and Equal
Root 1 = 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
```

19-12-23

Java program to create class Student with members usn, name. Calculate SGPA of 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 (i=0; i<8; i++)
```

```
{
 System.out.print("Enter the Subject
  Marks" + (i+1)+":");
 subject[i]subject Marks = s.nextInt();
 System.out.print("Enter the Credits" + (i+1)+
   ":");
 subject [i].credits = s.nextInt();
 while (subject [i].subject Marks > 100);
 System.out.println("Marks entered are
 invalid, Enter the Marks in between 1 to 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";
}
elseif (Subject [i].SubjectMarks >=50)
{
Subject [i].grade = "6";
}
elseif (subject [i].subjectMarks >=40)
{
```

```
Subject [i]. grade = "0";
  }
 }
}

public void compute SGPA()
 {
  double total Credits = 0;
  double total Grade Point = 0;
  for (int i = 0; i<8; i++)
  {
   total Credits += subject [i]. credits;
   total Grade Point += Subject [i]grade * subject
   [i]. credits;
  }
 SGPA = total Grade Point / total Credits;
}
public static void main (String [] args)
 {
 Student s1 = new Student();
 s1. get Student Details ();
 s1. get Marks ();
 s1. compute SGPA ();
 System. out. println ("Name: " + s1. Name);
 System. out. println ("USN: " + s1. USN);
 System. out. println ("SGPA: " + s1. SGPA);
 }
}
```

Output

```
Enter Name:
Prajwal C

Enter USN:
1BM22CS198

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:
3
Enter marks for subject 4:
90
Enter credits for subject 3:
3
Enter marks for subject 5:
93
Enter credits for subject 5:
3
Enter marks for subject 6:
91
```

```
Enter marks for subject7:
 87
Enter ~~mark~~ credits for subject7:
 2
Enter marks for subject 8:
98
Enter credits for subject 8:
1

Name: Prajwal C
USN: 1BM22CS198
SGPA: 9.25
```

19/12/23

26/10/23

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

```
import java.util.scanner;
class Book {
    String name;
    String author;
    int price;
    int num pages;

   public Book (String name, String author,
     int price, int numpages) {
     this.name = name;
     this.author = author;
     this.price = price;
     this.numpages = num Pages;
   }
  public String to String () {
   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.num
    pages + "\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 book:");
   name = s.next();

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

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

  System.out.println("Enter the number of
     pages of 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());
 }
}
}
```

Output

```
Enter the number of books:
2
Enter name of book:
A
Enter author of book:
B
Enter the price of the Book:
300
Enter the number of pages of book:
450
Enter name of the book:
X
Enter authour of book:
The
Enter the price of book:
500
Enter the number of pages of book:
390
Book nam : A
Author~~Book~~ name: B
Price : 300
Number of pages : 450

Book Name. x
Author name: The
Price: 500
number of pages: 308
```

2/1/24

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.

=>
```
import java.util.scanner;
abstract class shape{
    int length;
    int breadth;
    abstract void printArea();
  }
class rectangle extends shape{
void printArea(){
Scanner sc = new Scanner(System.in);
System.out.println("Enter the dimensions
of the rectangle (length and breadth):");
length = sc.nextInt();
 breadth = sc.nextInt();
 system.out.println("Area of rectangle is"
  + (length * breadth));
  }
}
class triangle extends shape{
 void printArea(){
 Scanner sc = new Scanner(System.in)
 System.out.println("Enter base ~~length~~ and
 height ~~breadth~~ of triangle ~~rectangle~~ :");
 length = sc.nextInt();
```

```
  breadth = sc.nextint();
  Sytem.out.println ("Area of Triangle
   is" + (0.5*length*breadth));
   }
 }

class circle extends shape{
 void printArea(){
 Scanner sc = new Scanner(System.in);
 system.out.println ("Enter radius of
  circle:");
 length = sc.next int();
 System.out.println ("Area of Circle is"+
 (3.14*length*length));
 }
 }
 public class main{
 public static void main (String []args){
 shape = shape;
 shape = new Rectangle ();
 shape.print Area();
  shape = new Triangle ();
  shape.print Area ();
  shape = new Circle ();
  shape.print Area ();
 }
}
```

Output

```
Enter the dimensions of the rectangle
 (length and breadth):
 2 3
Enter the dimensions of the triangle
 (base and height):
2 4
Enter radius of circle:
 3
Area of Rectangle is 6.0
 Area of triangle is 4.0
  Area of Circle is 28.259999
```

02/01/24

9-1-24

Develop a Java Program to creat 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.

e) 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 customerName, int account
Number, 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 +
     " succesful");
  }

void display Balance(){
  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
+ "succesful");
   }
}

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("Withdrawal of"
    + amount + "succesful");
  }
}

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 < minimum Balance){
     System.out.println ("Insufficient balan");
      return;
   };
  balance -= amount;
  System.out.println ("Withdrawal of "+
   amount+" successful");
 }

void impose service Charge (){
   if (balance < minimum Balance) {
      balance -= service Charge;
    System.out.println ("Service Charge of "
   +service Charge + " imposed");
   }
 }
}

public class Bank{
  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: ");
  int accountNumber = scanner.nextInt();
  System.out.print ("Enter account type
(saving /Current): ");
String accountType = scanner.next();
System.out.print(~~x~~"Enter initial balance: ");
double balance = scanner.nextDouble();

Account account;
if (accountType.equals("Savings")){
    account = new Savings Account
(customerName, accountNumber, account
Type, balance);
} else {
 account = new Current Account
 (customerName, account Number,
 accountType, balance);
}

while (true) {
  System.out.println("\n1. Deposit");
  System.out.println("2. Display balance");
  System.out.println("3. Compute and
  deposit interest");
  System.out.println("4. Withdraw");
System.out.println("5. Exit");
System.out.print("Enter choice: ");
int choice = scanner.nextInt();

switch (~~c~~ choice){
 case 1:
   System.out.println("Enter amount
 to deposit");
 ~~System.out.print.~~
 double amount = scanner.nextDouble
 ();
```

```
    account.deposit (amount);
    break;
  case 2:
    account.display Balance ();
    break;
  case 3:
    if (account instance of SavingsAccount) {
      ((SavingsAccount) account). compound
      Interest ();
    } else {
      System.out.println ("Interest not
    available for current account");
    }
    break;
  case 4:
    System.out.print ("Enter amount to
      withdraw: ");
    amount = scanner.nextDouble ();
    account.withdraw (amount);
    if (account instance of CurrentAccount) {
    ((CurrentAccount) account). imposeService
    Charge ();
    }
    break;
  case 5:
   System.exit (0);
  }
 }
}
}
```

Output

```
Enter customer name: Prajwal
Enter account number: 198
Enter account type (Savings / Current):
 Savings
 Enter initial balance: 1700
1. Deposit
2. Display balance
 3. Compute and deposit interest
4. Withdraw
5. Exit

Enter choice: 4
Enter amount to withdraw: 500
 withdraw of 500.0 succesful
Enter choice: 2
Balance: 1200.0
```

11/01/24

16-1-24

Practice Programs: string Handling (output)

1) //string constructors

```
BMSCE
MSC
BMSCE
```

2) //string length, string literal, string concat

```
5
7
A year has 52 weeks,
```

3) To string (), ~~charAt~~

```
Person {name = 'John', age = 25} , b
```

4) get chars()

```
Bmsce
```

5) get byte (), to char Array ()

```
72 101 108 108 111
  H e l l o
```

6)

```
Bmsce equals Bmsce -> true
Bmsce equals College -> false
Bmsce equals BMSCE -> false
Bmsce equals Ignore case BMSCE -> true
```

7) region matches ()

```
substring is matched
```

8) Stardwith()
true

9) end swith()
false

10) equals () versus ==

world equals world -> true
world == World -> false

11) compareto () //alphabets
apple
ball
cat
dog
ent
free
gun
hen
ice
jug
bite
man
net
orange
parrot
queen
ring
Octan
yatch
zee

12) compare to () //numbers

```
1
2
3
4
5
6
7
8
9
```

13) substring (), index of ()

```
This as wear a test. This as wear, too
This wear a test. This as wear, too
This is a test. This as wear, too
This is a test. This wear too
This is a test. This is, too
```

14) concat()

```
helloworld
```

15) replace ()

```
Commege
```

16) trim ()

```
Hello Friends
```

17) Student Records

```
enter details for student 1:
Registration number: 18
Full Name: Ajay
Semester: 2
CGPA: 9.15
```

```
Student Record:
Registration Number: 18
 Full Name: Ajay
 Semester: 2
 CGPA: 9.15
```

18) String Buffer

```
After setLength (5): Hello
 character at index 0: H
 After setCharAt (0, 'X'): Xello
 getChar (0, 5): Xello
```

19) Bird Demo

```
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) Shape Demo

```
Circle Area : 78.5398; , Perimeter : 31.4159
Triangle Area : 6.0; Perimeter : 12.0
```

Lab 6

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

```
//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 inputStudentDetails()
{
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 display Student Details ()
{
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 Student
{
protected int marks [] = new int [5];
 public void input CIE marks () .
{
Scanner s = New Scanner (System. in);
System. out. println ("Enter the marks 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 finalMarks [];
public Externals ()
{
marks = new int [5];
finalMarks = new int[5];
}
public void inputSEEmarks ()
{
Scanner s = new Scanner (System.in);
for (int i = 0; i<5; i++)
{
System.out.print ("Subject" + (i+1) + "
marks: ");
marks [i] = s.nextInt ();
}
}
public void calculateFinalMarks ()
{
for (int i = 0; i<5; i++)
finalMarks [i] = marks [i]/2 +
super.marks [i];
}
public void displayFinalMarks ()
{
displayStudentDetails ();
for (int i=0; i<5; i++)
System.out.println ("Subject" + (i+1) +
": " + finalMarks [i]);
}
}
```

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

Output

```
Enter the usn
 198

Enter the student Name
Prajwal

Enter the semester
 3
Enter CIE marks
Enter the marks of each subject
 47
 48
 49
 47
 48
Enter SEE marks :
 Subject 1 marks : 45
Subject 2 marks : 46
 Subject 3 marks : 48
 Subject 4 marks : 47
 Subject 5 marks : 46

 Enter the USN
  198
 Enter the student name
 Anagha
 Enter the semester
 3
Enter CIE marks
Enter the marks of each subject
44
```

```
49
48
47
46
Enter SEE marks
Subject 1 marks : 45
Subject 2 marks : 44
Subject 3 marks : 46
Subject 4 marks : 45
Subject 5 marks : 42

Displaying Data :
USN = 1BM22CS198
Student name = Prajwal
Semester 3
Subject 1 : 72
Subject 2 : 75
Subject 3 : 69
Subject 4 : 81
Subject 5 : 74

USN = 198
Student name = Anagha
Semester = 3
Subject 1 = 93
Subject 2 = 85
Subject 3 = 84
Subject 4 = 87
Subject 5 = 89
```

23/01/24

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

```
import java.util.*;
class WrongAge extends Exception {
    WrongAge (String 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 fatherAge, int sonAge)
    throws WrongAge {
            super (fatherAge);
        if (sonAge >= fatherAge) {
            throw new
    WrongAge ("Son's age should be less
    than father's age");
        }
            this.sage = sonAge;
        }
    }
  public class Error {
      public static void main (String args[])
  {
  Scanner sc = new Scanner (System.in);
        try {                              Father
  System.out.println ("Enter the ~~daddy's~~
   age:");
   int a = sc.nextInt();
            Father father = new
   Father (a);
     System.out.println ("Enter the
   Mother Son's
   ~~Sonny's~~ age:");
           int b = sc.nextInt();
         Son son = new Son (a, b);
        } catch (WrongAge e) {

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

Output

```
Enter the Fathers age
 55
Enter the Sons age
 29

Sons age is 24

Enter the Fathers age
 32
Enter the Sons age
 38

```

Sons age ~~cannot be greater than~~ should be less than Fathers age

```
Enter the Fathers age
 40
Enter the Sons age
 40

Son age should be less than Fathers age
```

31.01.24

6/2/24

Pgm8

```
public class ThreadExample (
 public static void main (String [] args)
 {
Thread thread1 = new Thread (new Display
Every Ten Seconds ());
Thread thread 2 = new Thread (new
Display Every Two Seconds ());
 thread 1. Start ();
 thread 2. Start ();
 }
}

Class Display Every Ten Seconds implements
 Runnable {
 public void run ()
 {
while (true)
 {
try {
 Thread. sleep (1000);
 Sytem. out. println ("BMS collég of
  Engineering");
} catch (Interrupted Exception e)
 {
 e. printStackTrace ();
 }
 }
 }
}
```

```
class Display Every Two Seconds implements
 Runnable {
  public void run(){
   while(true)
   {
   try {
     Thread. Sleep (2000);
      System. out. println ("CSE");
    }
    catch (Interrupt Exception e) {
    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
```

Pgrm:10

```
class Q {
int n;
boolean value Set = false;
synchronized int get(){
while (!value Set)
 try {
System.out.println("\n Consumer Waiting \n");
wait();
} catch (InterruptException e) {
System.out.println("\nIntimate Producer\n");
notify }
    System.out.println("Got: "+n);
    value Set = false;
   System.out.println("\nIntimate Producer
  \n");
 notify ();
  return n;
 }

synchronized void put(int n) {
while (Value Set)
 try {
System.out.println("\n Producer waiting \n");
wait();
} catch (InterruptException e) {
System.out.println("Interrupted Exception
 caught");
}
this.n=n;
value Set = 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 x= q.get();
System.out.println ("consumed: "+x);
 i++;
 }
}
}
```

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

Output

```
Press Control-C to stop
 Put : 0

Intimate Consumer
 Producer waiting
 Got : 0

Intimate Producer
put : 1

Intimate consumer
 Producer waiting
 consumed: 0
 Got : 1

Intimate Producer
Consumed: 1
Put : 2
```

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 cal
    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.current Thread(). SetName ("M
 ain 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 static void main (String args []){
new Deadlock ();
}
}
```

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

Prgm 9.

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

class Swing Demo {
    SwingDemo (){
        JFrame jfrm = new JFrame (&quot; Divider
App&quot; );
jfrm.set Size (275, 150);
jfrm. set Layout (new Flow layout ());
jfrm. set Default Close Operation (JFrame.
Exit_ON_CLOSE);

Jlabel jlab = new Jlabel (&quot; Enter the
divider and dividend: &quot; );
JTextField ajtf = new JTextField (8);
JText Field bjtf = new JText Field (8);
JButton button = new JButton (&quot;
Calculate&quot; );
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);
```

```
 jfm.add (anslab);

Action listener l = new ActionListener() {
  public void action Performed (Action Event
 evt) {
 System.out.println (&quot; Action event from
 a text field &quot;);
  }
};
aitf. add Action Listener ~~new~~ (l);
bitf. add Action Listener (l);
 button. add Action listener (new Action listener ()
 (
public void action Performed (Action Event evt){
 try {
   int a = Integer. ParseInt (aitf. getText ());
   int b = Integer. ParseInt (bitf. getText ());
    int ans = a/b;
  alab. set Text (&quot; \nA = &quot; + a);
  blab. set Text (&quot; \nB = &quot; + b);
 anslab. set Text (&quot; \n Ans = &quot; +
  ans);
}
catch (Number Format Exception e){
    alab. set Text ( &quot; &quot; );
    blab. set Text (&quot; &quot; );
    anslab. setText ( &quot; &quot; );
    err. set Text (&quot; Enter Only Integers!
    &quot;);
}
catch (Arithmetic Exception e) {
    alab. set Text ( &quot; &quot; );
lblab. setText ("Equal: "Equal:);
ans lab. setText ("Equal: "Equal:);
err. setText ("Equal: B should be non
  zero" Equal:);
    }
  }
});
  jfrm. setVisible (true);
}
public static void main (String args[]){
 SwingUtilities. invokeLater (new Runnable(){
     public void run (){
         new SwingDemo();
     }
   });
 }
}
```

Output

Enter the divider and divident:

[18] [6]

[Calculate] A = 18 B = 6 Ans = 3

29.02.24

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

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, credits, grade;}
class Student {
   String name;
   String usn;
   double SGPA;
   Scanner s;
   subject subjects[];
Student()
{
int i;
subjects = new subject[9];
for(i=0;i<8;i++)
subjects[i] = new subject();
s = new Scanner(System.in);
}
public void getStudentDetails(){
System.out.println("Enter student name:");
name=s.nextLine();
System.out.println("Enter Student USN:");
usn=s.nextLine();}
public void getMarks(){
int i;
for(i=0;i<8;i++){
```

```
System.out.println("Enter marks of subject"+(i+1)+":");
subjects[i].subjectMarks= s.nextInt();
if(subjects[i].subjectMarks>=40&&subjects[i].subjectMarks<=100){
subjects[i].grade=calculateGrade(subjects[i].subjectMarks);}
else{
System.out.println("Invalid Marks. Marks should be between 40 and 100");}
System.out.println("enter credits:");
subjects[i].credits=s.nextInt();
}
}
public int calculateGrade(int marks){
if (marks>=90)
return 10;
else if(marks>=70&&marks<=80)
return 9;
else if(marks>=60&&marks<=70)
return 8;
else if(marks>=50&&marks<=60)
return 7;
else
return 6;
}
public void computeSGPA() {
    int totalscore = 0;
    int totalcred = 0;
    for (int i = 0; i < 8; i++) {
        totalscore += subjects[i].grade * subjects[i].credits;
        totalcred += subjects[i].credits;
    }
    SGPA = (double) totalscore / (double) totalcred;
  }
```

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

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

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 the book. Develop a Java program to create n book objects.

```
import java.util.Scanner;

class Book {

    private String name;

    private String author;

    private double price;

   private int numPages;

    public Book(String name, String author, double price, int numPages) {

        this.name = name;

        this.author = author;

        this.price = price;

        this.numPages = numPages;

    }

    public void setName(String name) {

        this.name = name;

    }

    public String getName() {

        return name;

    }

    public void setAuthor(String author) {

        this.author = author;

    }

    public String getAuthor() {

        return author;

    }

    public void setPrice(double price) {
        this.price = price;
    }
    public double getPrice() {
        return price;
    }
    public void setNumPages(int numPages) {
        this.numPages = numPages;
    }
    public int getNumPages() {
        return numPages;

    }
    public String toString() {
        return "Book Details: \nName: " + name + "\nAuthor: " + author + "\nPrice: INR" + price + "\nNumber of Pages: " + numPages;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number of books: ");
        int n = scanner.nextInt();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Book " + (i + 1) + ":");
            scanner.nextLine();
            System.out.println("Enter name: ");
            String name = scanner.nextLine();
            System.out.println("Enter author: ");
            String author = scanner.nextLine();
            System.out.println("Enter price: ");
            double price = scanner.nextDouble();
            System.out.println("Enter number of pages: ");
            int numPages = scanner.nextInt();
            books[i] = new Book(name, author, price, numPages);
        }
        System.out.println("\nDetails of all books:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nBook " + (i + 1) + ":\n" + books[i]);
        }
        scanner.close();
    }
}
```

LAB PROGRAM 4

Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the 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;
        System.out.println("Deposit of INR " + amount + " successful");
    }
    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

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

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");
        }
        if (sonAge >= fatherAge) {
            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

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

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