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:

POOJA GAIKWAD

(1BM22CS194)

Department of Computer Science and Engineering,

B.M.S College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019

2023-2024.

INDEX

8. Quadratic formula :

```
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 co-efficie
 of  a, b, c") ;
     a = s. nextInt() ;
     b = s. nextint () ;
     c = s. next int() ;
 }
 void compute()
   {
     while (a==0)
     {
      system. out. println (" Not a quadratic eq
      system. out. print ("Enter a non-zero value
       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 & equ
 System. out. println ( "Root1 = Root2 = "+
}
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 ("Roo 1 = "+ r1 + " root 2 = "+r2);
  }
  else if (d<0)
  {
   system .out print ln ("Roots are Imaginary");
   r1 (-b)/((2*a));
   r2 = Math. sqrt (-d )/(2*a) ;
   system. out . println ("root 1 = "+r1 + " + i"+r2");
  sytem. out. println ( "Root 1 = "+r2 + "-i"+ r2 );
 }
 }
 }
class quadratic Main
   {
    public static void Main (string args [] }
    {
     Quadratic q = new quadratic () ;
        q.getd() ;
        q . compute() ;
   }
 }
```

output : Enter the co-efficient equation

```
        0 4 2
[Not a quadratic equation
[Enter a non-zero value for a]
```

Roots are real & distinct
Root 1 = -0.381966   Root 2 = -2.6180

121 121 121
Roots are imaginary

000
Not a qudratic eqn
Enter a non zero value for a

1 0 0
Roots are real and equal
Root = Root 2 0.0

[illegible] 13/12/23

LAB-2 Prgms :-

Date 19/11/23
Page

8. 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;
}
    class student {
          string name;
          string usn;
          double SGPA;
          scanner s;
          subject [] subject;

student () {
          subjects = new subject [8];
          For (int i=0; i<8; i++) {
                subjects [i] = new subject ();
          }
          s= new scanner (system.in);
}
  void getStudentDetails () {
      system.out.print ("Enter Name :");
       name = s. nextline();
       system .out. print ("Enter USN: ");
        USN = s. nextline();
     }
```

```
void getMarks () {
        for (int i=0 ; i<8 ; i++)
    {
    system.out.println ("Enter details for subject" + (i+
                      + ":");
    system.out.print ("Marks : ");
     subjects [i] . subjectMarks = s.nextInt ();
     system. out.print ("credits : ");
     subjects [i].credits = s.nextInt ();

if (subjects [i]. subjectMarks >= 90) {
     subject [i] . grade = 10;
  } else if (subjects [i] . subjectMarks >= 75) {
     subjects [i] .grade = 9;
  } else if (subjects [i] . subjectMarks >= 60) {
     subjects [i]. grade = 8;
  } else if (subjects [i]. subjectMarks >=50) {
      subjects [i] .grade = 7;
  } else if (subjects [i] . subjectMarks >=40) {
    subjects [i] . grade = 6;
  } else {
    subjects [i] . grade = 0;
  }
 }
}
void computeSGPA () {
     double totalCredits = 0;
     double weightedSum = 0;

for (int i=0; i<8; i++) {
   totalCredits += subjects [i]. credits ;
 weightedSum += subjects [i]. grade * subjects [i].
                                          credits;
}
SGPA = weighted sum / total credits ;
 }
}
public class Main {
        public static void Main (string [] args)
   {
     student s1 = new student ();
     s1. get StudentDetails ();
     s1. get Marks ();
     s1. compute SGPA ();

   system. out. println ("\n Result:");
   system. out. println ("Name:" + s1.name);
   system. out. println ("USN:" + s1.usn);
   system. out. println ("SGPA:" + s1. SGPA);
  }
}
```

⟹ Output :

```
Enter name : Pooja .
Enter USN : 1BM22CS194
Enter details for subject 1 :
 Marks : 85
 credits : 4
Enter details for subject 2 :
 marks : 79
 credits : 4
Enter details for subject 3 :
Marks : 70
Credits : 3
```

LAB - 03 :-

Date 20/02/23
Page

1] create a class book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the numbers. Include methods to set and get the details of the objects. Include a to string() 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 ;

Book (string name, string author, int price, int numPages)
  {
   this. name = name ;
   this. author = author ;
   this. price = price ;
   this. numPages = numPages ;

Public string tostring () {
     string bookDetails = "Book name :" + this.
 name + "\n" + "Author name :" + this. author
 + "\n" + "Price:" + this . price
 return book Details ;
 }
}
public class Main {
  public static void Main (string args []) {
  Scanner s = new Scanner (System. in) ;
```

```
Enter details for subject 5 :
  Marks : 88
  credits : 3
Enter details for subject 6 :
  Marks : 97
  credits : 3
Enter details for subject 7 :
  Marks : 97
  credits : 4
Enter the details for subject 8 :
  Marks : 98
  credits : 2

Result :
Name : Pooja.
USN : 1BM22CS194
SGPA : 9.11923.
```

[illegible] 19/10/23

```
system.out.print("Enter the number of book
  int n = s.nextInt();

  Book[] books = new Book[n];

  for (int i = 0; i<n; i++) {
 system.out.println("Enter details for Book" +(i
 system.out.print("Name:");
 string name = s.next();
 system.out.print(price:");
 int price = s.nextInt();
 system.out.print("Number of pages:");
   int numPages = s.nextInt();

books[i] = new Book(name, author, price, numPage
}
system.out.println("\nDetails of the books:"
 for (int i = 0; i< n; i++) {
  system.out.println("Book" + (i+1) + ":\n"
       + books[i].toString());
  }
 }
}
```

→ Output:

```
Enter the number of books : 2
Enter details for Book 1
Name : harrystyles
Auther : harry
price : 600
Number of pages : 275
Enter details for Book 2
Name : conan gray
Author : conan
```

LAB-04 :-

Date 02/01/24
Page

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;
  class Inputscanner {
      Scanner s = new Scanner (system.in);
      double get Input(string prompt) {
        system .out. print ln (prompt);
          return s.nextDouble(); }}
abstract class shape extends Input Scanner {
      double side 1, side 2 ;
  abstract void area (); }

class Rectangle extends shape {
      Rectangle () {
    side1= get Input ("Enter length of rectangle:");
      side2 = get Input ("Enter breadth of rectangle:");
void area () {
    double area = side1* side 2 ;
  system. out. println ("Area of the rectangle=
                "+ area); }}.
class triangle extends shape {
      triangle () {
    side 1 = get Input ("Enter base of the triangle:");
    side 2 = get Input ("
```

Date 02/01/24

bstract
tegers
ta()
triangle
asses
e
that
);
ngle:");
ctangle:")}
le=
4:");
he

```
void area() {
     double area = side1 * side 2 / 2;
   system. out. print ln ("Area of the triangle ="+area)
                                                        ; } }
class circle extends shape {
     circle () {
        side 1 = get input ("Enter the radius of
                  the circle:"); }
     void area () }
       double area = Math. PI * side1 * side1;
    system .out. println ("Area of the c = "+area);

class Main {
      public static void Main (string args [] ) {
    Rectangle rectangle = new Rectangle ();
    triangle Triangle = new triangle ();
     circle   circle   = new circle ();
   rectangle . area ();
    Triangle. area ();
     circle . area ();
    System. ou. println (" Pooja Gaikwad 184220
  }
}
```

If negative value to be checked

```
class input scanner {
   scanner s = new scanner (system .in);
   int get input (string prompt) {
     double input ;
     double {
       system . out. println (prompt);
        input = s. next Double ;
       if (input <0) {
```

```
system. out. println ("Enter positive values only
}
}while (input <0) ;
    return input ; }}
```

output:

```
Enter length of rectangle :
 20
Enter breadth of rectangle :
 40
Enter base of the triangle ;
 6
Enter the height of the triangle;
8
Enter the radius of the circle :
4
Area of rectangle = 800.0
Area of Δle = 24.0
Area of the circle = 50.26
```

[illegible]
22/01/24

Date ___/___/___
Page ______

price : 450
Number of pages : 450.

[illegible] 20/12/23

```
*[public void deposit (double amount) {
        balance + = amount;
    system. out. println (" De? x

public void display Balance () {
    system. out. println ("Account Balance : $
  }
public void withdraw (double amount) {
    System. out. println (" withdrawal not su
    for this accont type.");
}
}

class curAccount extends Account {
    double Min Balance ;
    double service charge ;

public curAccount (String customer Name, long accou
number, double Min Balance, double service charge)
super (customername, acount Number, "current", ba
this. Min Balane = min Balance ;
this. servicecharge = service charge;
}

public void checkMin Balance() {
    if (balane < Min Balane) {
        balane - = service charge ;
~~service~~ system. out. println ("Minum balance not Mai
service charge + "impesed.");
 display Balance();
}
}
@override.
```

Date 09/01/24
Page

LAB - 05 :-

Q. 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 provide 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

→
```
import java.util.scanner ;

class Account {
        string customer name;
        long accountNumber ;
        string account Type ;
        double balance ;

public Account (string customerName, long accountNum
-ber, String accountType, double balance) {
    this. customerName = customer name;
    this . accountNumber = accountNumber ;
    this . accountType = accountType ;
    this . balance = balance ;
}

public void deposit (double amount ) {
        balance + = amount ;
    System. out. println ("Deposit of $ " + amount
+ " succesful. updated balance : $" + balance);
}
```

```
system. out. print ("Enter service charge for the
    current account :" );
  double service charge = scanner. nextDouble();

user Account = new curAccount ( customer Name,
system. current Time Millis(), initial Amount, min
Balance  service charge ) ;
 } else if (account Typ e choice = = 2) {

system. out. print ("Enter interest rate for the sav
   account :" );
double interestRate = scanner. next Double();

user Account = new savAccount ( customer Name, system.
 current Time Millis(), initial Amount, interest Rate);
} else {
 system.out. println ("Invalid account type choice.
            Exiting the program." );
  scanner . close() ;
  return ;
}
int choice  ;
do {
      system. out. println ("\n select an option
system. out. println ("1. Deposit" \n \2. Display
Balance \ 3. compute interest saving only
   4. withdraw \n . 5. Exit \n
         Enter choices" ) ;
    choice = s. next  + IN + c) ;
```

```
Public void withdraw (double amount) {
    if (amount > balance) {
      system.out.println ("Insufficient funds withdr
-awal failed.");
    } else {
        balance -= amount;
     system.out.println ("withdrawal of $" + amount
    + " succesful. updated balance: $" + balance);
   }
  }
}

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

   system.out.print ("Enter your name: ");
   String customerName = sBanner.nextLine();

  system.out.print ("Enter initial amount: ");
  double initial Amount = scanner.nextDouble();

 System.out.print("select accountType (1. current, 2.
 savings): ");
  int accountTypeChoice = scanner.nexInt();

Account userAccount = null;

if (account TypeChoice == 1) {
  system.out.print ("Enter minimum balance for
      the current account: ");
double minBalance = scanner.nextDouble();
case 4:
  system.out.print ("Enter amount to withdraw");
   double withdraw Amount = scanner.next Double();
 system.out.print ("select account (1.current, 2.savings)
  int acTyp = scanner nextInt();
    if (acTyp = = 2) {
   savings account.withdraw (withdrawAmount);
  } else {
       system.out.println ("Invalid account type
  }
  break;
 case 5:
   system.out.println ("Exiting the program.
                 thank you!");
  break;
  default:
    system.out.println ("invalid choice.please Enter
a valid option.");
  }
} while (choice != 5);
     scanner.close();
  }
}
```

→ <u>output:</u>

```
   select an option : 1.Deposit
                      2. Display Balance.
                      3. compute Interest.
                      4. withdraw.
                      5. Exit
 Enter your choice : 1.
Enter amount to deposit : 500
```

```
ge for the
st Double();
r Name,
nt, ...
) {
the saving
ble();
, system
t Rate);
hoice.
option:]
splay
```

```
switch (choice) {
  case 1:
      System.out.print (Enter amount to deposit:");
      double deposit Amount = scanner.nextDouble();
      System.out.print ("select account (1.current, 2-savings)
      int accountType = scanner.nextInt();
       if (accountType == 1) {
            currentAccount .deposit (deposit (deposit Amount
   } else if (accountType ==2) {
      savingsAccount . deposit (deposit Amount);
   }.
   break;
  case 2:
     system.out.print ("select account (1.current, 2.
  savings): ");
        int acc Type = scanner.nextInt();
     if (acc Type ==1) {
      currentAccount .display Balance();
   }
    else if (acc Type == 2) {
     savings Account. display Balance ();
   } else {
        System.out.println("invalid account type");
  }
  break;
  case 3:
     if (savings Account instance of sav Account)
  { ((sav Account ) savingsAccount ) . compute intrest();
  } else {
    system.out.println ("invalid option for curre
        account ");
  }
  break;
```

```
select account (1. current, 2. savings) : 1
  Deposit of 500 succesful.

 select an option :
1. Deposit
2. Display Balance.
3.  compute Interest
4.  withdraw
5.  Exit .
 Enter your choice : 2.
     select acount (1 - current, 2. savings) : 1
     Account Balance : 15000

select an option :
1. deposit
2. Display balance
3. compute interest
4. withdraw
5. Exit
  Enter your choice : 5
                   Exiting the program
              Thank you !
```

7. add frame

— it is a method in the 'container' class (which 'jframe' extends) that adds a component to the container.

8. add Action listener.

* It is an interface that responds to events, such as button clicks.

* JButton extends it. it adds an 'Action listener' to a button

9. set text :

It is a method which sets the text displayed by the label.

[illegible] 23.02.24

LAB - 05 :-
# strings :

1. ~~+~~ 1. constructor with no. parameters :
→ output : " "

2. costructor with char array
→ output : " Java "

3. constructor with string
→ output : " Hello"

4. constructor with char array and index ;
→ output : "av "

5. constructor with string Buffer.
→ output : "Hello"

6. constructor with stringBuilder ;
→ output : "Hello".

2. a) string length
output :- 5

b) string literal
output :- true

c) string concatenation
output :- "Hello world" ;

3 output :- Person [name = Pooja, age = 21

11. apple
ball
cat
dog
ent
free
gun
hen
ice
jug
kite
lift
man
net
orange
parrot
queen
ring
star
tree
umbrella
van
watch
xmas
yatch
Zee.

12. sorted numbers :
1, 2 3 4 5 6 7 8 9 10.

Date __/__/__
Page ____

13. output :- Thigs is a test . Thisu is, too

14. Helloworld

15. The best college in the world is comrege.

16. Hello Friends.

18. StringBuffer after setlength () :
    stringBuffer after append () ; HelloHello world
    " " insert () ; Hello Hello beautiful world
    " " reverse(); dlrow olleH olleH
    " " delete () ; HelloHellow dlrow
    " " replace() ; Hello Hello good dlrow
    substring : good.

19. Eagle soars high above
    Eagle makes a screeching sound
    Hawk glides smoothly through the air
    Hawk makes a high - pitched sound.

20. circle area : 78 . 539
    circle perimeter : 31 . 41
    Triangle Area : 6
    Triangle perimeter : 12.

17. ~~semester : 3~~
    ~~CGPA : 8.2~~
    Registratiation Number : 1
    Full Name : Pooja.
    semester : 3
    CGP : 3.5

4. output :- Bmsce.

5. output :- BMSCE

6.
```
public class Main {
    public static void main (string[] args) {
        string str1 = "BMSCE";
        string str2 = "college";
        string str3 = "BMSCE"
        string str4 = "BMSCE";

system. out. println (str1. equals (str1));
system. out. println (str1. equals (str2));
system. out. println (str1. equals (str3));
system. out. println (str1. equals Ignore case
            (str3));
}
}
```

7. output :- substring is Matched.

8. output :- False

9. output :- False

10. 1. The two strings are equal using the equals() Method
 2. The two strings are not equal using the == operator

```
{public class Main {
    public static void main (string [] args) {
    str 1x
```

Date 11/03/24
Page

Prgm-9 - Deadlock :-

```
class A {
 synchronized void foo (B, b) {
 string name= Thread. currentThread ().get name();
 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 {
 synchorized void bar (A a) {
 string name = Thread. currentThread ().get name();
 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() {
```

Registration Number : 2
Full Name : Nidhi
Semester : 2
CGPA : 3.7

Registration Number : 3
Full Name : Charlie.
Semster : 3
CGPA : 3.8

Registration Number : 4
Full Name : Marya.
Semster : 3
CGPA : 3.6

Registration Number : 5
Full Name : Navya.
Semester : 3
CGPA : 3.9.

[illegible] 16/01/24

```
import java.util.scanner;
import CIE.student ;
 import CIE . internals ;
import SEE . Externals;
 class lab6 {
 public static void main (string args [ ]) {
 scanner sc = new scanner (System. in) ;
 string name ; int usn, sem ;
 System. out. print ln (Enter the number of students ;);
 int n= sc . nextInt ( ) ;
 Internals in [ ] = new Internals [n ];
 Externals ex [ ] = External [n ] ;
 for (int i=0, i<n, i++ ) {
System. out. println ("Enter the name of the student :+(i+1)
 + ";") ;
name = sc next Int ( ) ;
systentem .out. println (" Enter the USN of the student :+(i+1)
 +":") ;
usn = sc.nextInt ( ) ;
system out print ln ("Enter the sem of the student ;+(i+1)
 +';") ;
sem = sc.next Int ( ) ;
in [i] = new Internals (name, usn, sem ) ;
 ex [i] = new External ( ) ;
system. out. print ln ("Enter the internal marks of the
student in 5 subjects :") ;
for (int j = 0 ; j<5 ; ++ )
in [i ] . imarks [j ] = sc. nex Int ( ) ;
System. out . print ln ("Enter the external marks of
the student in 5 subjects ;") ;
for (int j=0 ; j<5 ; j++ )
ex [i] . eMarks [i ] = sc.next Int( ) ;
```

LAB-06 :-

Date 23/01/24
Page

* create a package CIE which has two classes - student and internals. 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.

→

```
Package CIE ;
public class student {
public int usn, sem ;
 public string name ;
  student ()  {
         public student (string name, int usn, int
     sem) {
       this.name = name, int usn, int sem ) {
Super (~~name~~, usn, sem) ;
}        this . name = name ;
}        this . usn = usn ;
         this . sem = sem ; }}
Package CIE ;
public class internals extends CIE . Student {
public int marks [ ] = new int [5] ;
public internals () {}
public internals ( string name, int usn, int
sem)
  super (name, usn, sem ) }}

package SEF ;
public class Externals extends CSE . students
  public int marks [ ] = new int [5 ]
  }
```

```
for (int i=0; i<n ; i++) {
system.out.println ("Details of student "+ (i+1) +":");
system.out.println ("Name :"+ in[i] .name + "\t" +
"USN:" + in [i].usn + "\t" + "sem :"+ in[i].sem);
system.out.println (" Internal marks :");
for (int j=0 ; j <5 ; j++ )
system.out print ("subject " + (j+1) + ":" + in [i].imarks
[j] + "\t");
system.out.println ("External Marks :");
 for (int  j=0 ; j< 5 ; j++)
system.out.print ("subject "+(j+1) + ":"+ex[i].
 emarks [j ]+ "\t");
system.out.println ("\n Total marks :");
 for (int j=0 ; j<5 ; j++ )
system.out.println (" subject " + (j+1) ":"+(in [i].
 imark [j] +ex [i] .emark [j])));
 }
}}.
```

output :

```
 Enter n :
 1
Enter name : Pooja
Enter usn : 34567
Enter sem : 3
Enter Internal Marks : 43
                       45
                       42
                       44
                       47
Enter External Marks : 80
                       78
                       63
                       78
                       85
```

```
Name : Pooja
USN : 34567
Sem : 3
marks of sub1 = 43
marks of sub2 = 45
Marks of sub3 = 42
marks of sub4 = 44
Marks of sub5 = 47
Marks of sub1 = 80
Marks of sub2 = 78
marks of sub3 = 83
Marks of sub4 = 78
Marks of sub5 = 85
Final Marks of sub[1] = 83.0
Final Marks of sub[2] = 84.0
Final Marks of sub[3] = 73.0
Final Marks of sub[4] = 83.0
Final Marks of sub[5] = 89.0
```

[illegible]
23/01/24

Date __/__/__
Page ____

* output:

```
i:)   Enter Father's age : 54
      Enter son's age : 19
      Father age : 54
      son's age : 17

ii) Enter Father's age : 17
    Enter son's age : 54
    Error: son's age cannot be greater than father's
              age

iii) Enter Father's age : -2
     Error : Age cannot be -ve

(iv) Enter Father's age : 12
     Enter son's age : 12
     Error : son's age cannot be equal to father

(v) Enter Father's age : 45
    Enter son's age = -3
    Error : Age cannot be -ve
```

[illegible] 02.02.24

Date 30/01/24
Page

LAB-07 :-

Q. WAP that demonstrates handling of expections inheritance tree. create base class called "Father" and derived class called "son" which extends the base class In Father class, implement a constructor which takes the age & throws the expection wrongAge() when the Input age < 0.

→
```
Import java. util. scanner ;
class wrongsalary extends Exception {
        public wrong salary() {
            super("salary Error");
    }

    Public wrongsalary (string message) {
        super (message);
    }
}

class Input scanner {
    public static in getInput (string prompt) {
    Scanner scanner = new scanner (system . in) ;
    system out . print (prompt);
    return scanner . nextInt () ;
    }
}
Class Father extends Inputscanner {
    Public int FatherAge ;

    Public Father() throws wrongAge {
      FatherAge = get In Input (" Enter Father's
            age:");
      if (FatherAge <0) {
         throw new wrongAge ("Age cannot be
              negative");
}
},}
```

```
system.out.println ("Inside A.last");
    }
  }
class Deadlock implements Runnable {
    A a=new A();
    B b =new B();
    Deadlock () {
  Thread.currentThread().setName ("Main thread");
  Thread t= new Thread (this, "Racing Thread");
    t.start();
    a.foo(b);
  system.out.println ("Back in main thread");
  }
 public void run() {
 b.bar(a);
 system.out.println ("Back in other thread");
 }
 public static void Main (String args[]) {
  new Deadlock ();
  }
 }
```

o/p →
```
Mainthread entered A.foo
Racingthread entered B.bar
MainThread trying to call B.last()
 Inside A.last
 Back in Main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread.
```

13.02.24

```
class son extends father {
   private int sonAge ;

    Public son() throws WrongAge {
     super();
    son Age = get Input ("Enter son's age:");
      if (son Age , super. fatherAge) {
    throw new wrongAge ('Son's age cannot be
 greater than father's age");
      } else if (sonAge < 0 ) {
  throw new wrongAge ("Age cannot be negative")
   }
  else if (sonAge == Super. Father Age)
  { throw new wrongAge ("son age cannot be equal
    to father's age !!!!");
  }
  }

    public void display() {
         Super. display();
         system. out. printIn("son's Age : "+ sonAge);
  }
}

  public class ExceptionHandling {
    public static void main (String args[]) {
        try {
           son son = new son();
              son .display ();
         } catch (wrongAge e) {
           system. err. println ("Enter: "+ e. get
           message ();
     }
  }
}
```

LAB - 08 :-

* 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 interv
    {
       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 Thread example {
     public static void main (string[] args) {
```

two threads, one
of Engineering"
nother displaying

ead {

ue;
ssage, int interval

rgs) {

```
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();
}
}
```

Output :-

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

LAB :- 10

* Demonstrate Inter process communication and dead

```
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;
  }
synchronized void put (int n) {
  while (value set)
  try {
  system. out. println (" /n producer waiting \n");
  wait ();
  } catch ( InterruptedException e) {
  System. out. println ("InterruptedException caught");
  }
this. n = n;
value set = true;
System. out. println ("Put :" +n);
System. out. println (" /n Intimate consumer \n
class PCFixed {
  public static void main ()
   {
    Q q = new Q ();
    new producer (q);
    new consumer (q) ;
   System.out.println ("c to stop") ;
  }
 }
```

Output :

```
put:0                 put : 10
Got:0                 Got : 10
put : 1               put : 11
Got : 1               Got : 11
put : 2               put : 12
Got : 2               Got : 12
put : 3               put : 13
Got : 3               Got : 13
put : 4               put : 14
Got : 4               Got : 14
Put : 5
Got : 5
put : 6
Got : 6
pot : 7
got : 7
put : 8
Got : 8
put : 9
Got : 9
```

```
notify();
}
}
class producer implements Runnable {
Q q;
producer (Q q) {
this.q = q;
new Thread (this, "Producer").strart ();
}
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++;
}
}
}
```

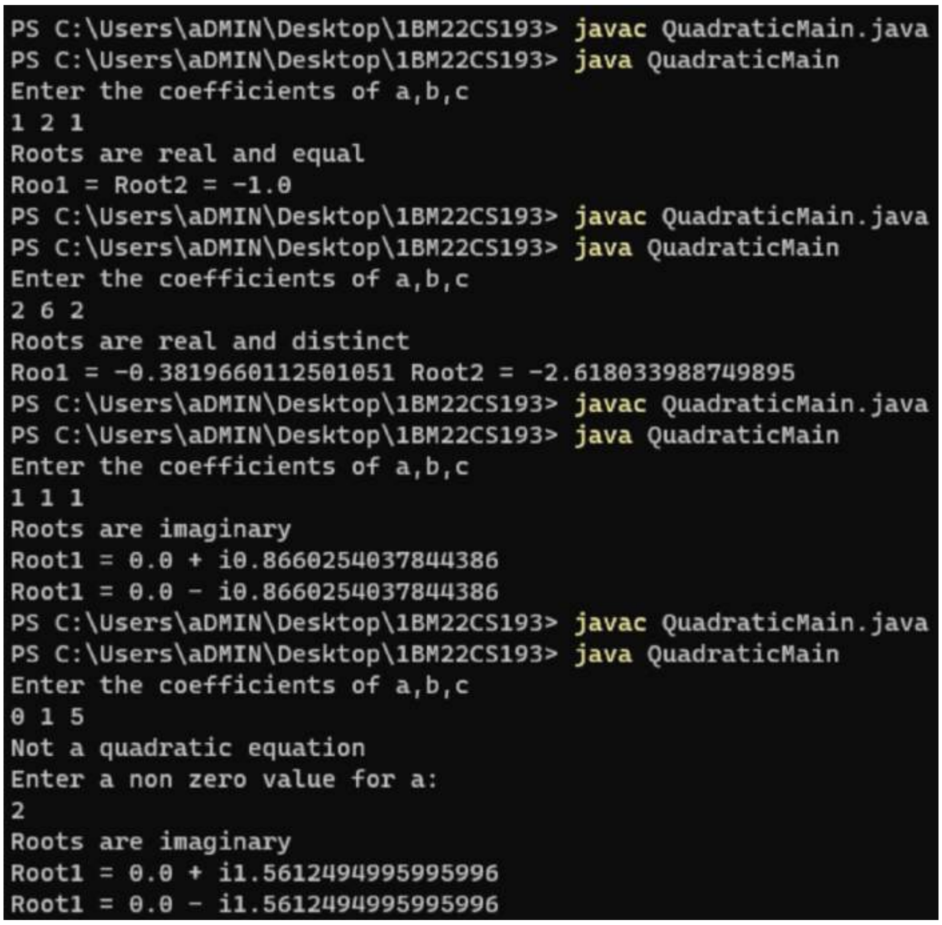
Lab 1

Develop a Java program that prints all real solutions to the quadratic equation ax 2 +bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b^2 – 4ac is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;
class Quadratic
{
int a, b, c;
double r1, r2, d;
void getd()
{
Scanner s = new Scanner(System.in);
System.out.println("Enter the coefficients of a,b,c");
a = s.nextInt();
b = s.nextInt();
c = s.nextInt();
}
void compute()
{
while(a==0)
{
System.out.println("Not a quadratic equation");
System.out.println("Enter a non zero value for a:");
Scanner s = new Scanner(System.in);
a = s.nextInt();
}
d = b*b-4*a*c;
if(d==0)
{
r1 = (-b)/(2*a);
System.out.println("Roots are real and equal");
System.out.println("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();
}
```

}

Output:



```
PS C:\Users\aDMIN\Desktop\1BM22CS193> javac QuadraticMain.java
PS C:\Users\aDMIN\Desktop\1BM22CS193> java QuadraticMain
Enter the coefficients of a,b,c
1 2 1
Roots are real and equal
Roo1 = Root2 = -1.0
PS C:\Users\aDMIN\Desktop\1BM22CS193> javac QuadraticMain.java
PS C:\Users\aDMIN\Desktop\1BM22CS193> java QuadraticMain
Enter the coefficients of a,b,c
2 6 2
Roots are real and distinct
Roo1 = -0.3819660112501051 Root2 = -2.618033988749895
PS C:\Users\aDMIN\Desktop\1BM22CS193> javac QuadraticMain.java
PS C:\Users\aDMIN\Desktop\1BM22CS193> java QuadraticMain
Enter the coefficients of a,b,c
1 1 1
Roots are imaginary
Root1 = 0.0 + i0.8660254037844386
Root1 = 0.0 - i0.8660254037844386
PS C:\Users\aDMIN\Desktop\1BM22CS193> javac QuadraticMain.java
PS C:\Users\aDMIN\Desktop\1BM22CS193> java QuadraticMain
Enter the coefficients of a,b,c
0 1 5
Not a quadratic equation
Enter a non zero value for a:
2
Roots are imaginary
Root1 = 0.0 + i1.5612494995995996
Root1 = 0.0 - i1.5612494995995996
```

Lab 2

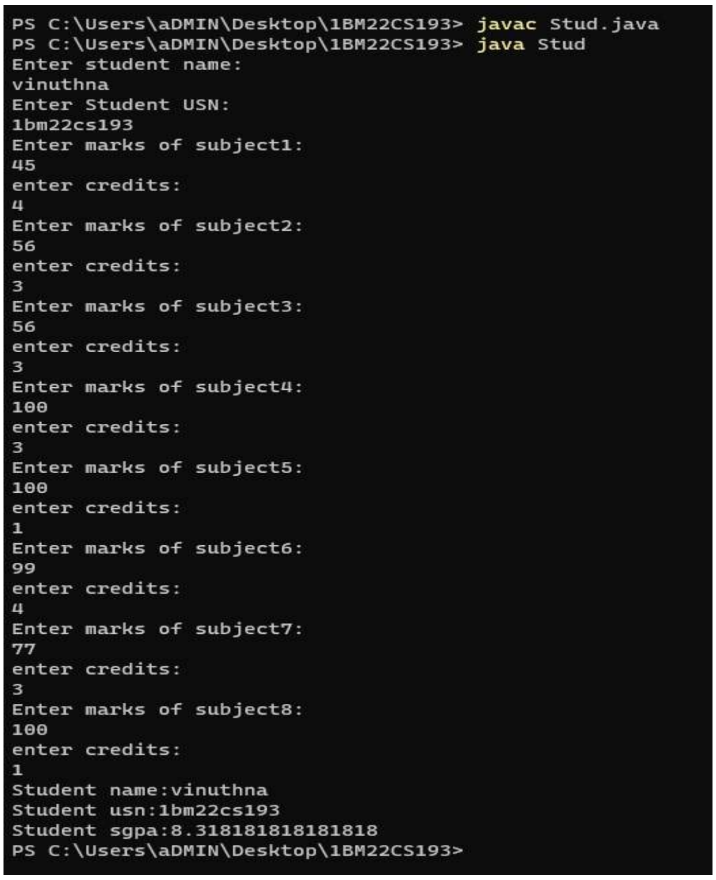
Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

```
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);}
}
```

Output:



```
PS C:\Users\aDMIN\Desktop\1BM22CS193> javac Stud.java
PS C:\Users\aDMIN\Desktop\1BM22CS193> java Stud
Enter student name:
vinuthna
Enter Student USN:
1bm22cs193
Enter marks of subject1:
45
enter credits:
4
Enter marks of subject2:
56
enter credits:
3
Enter marks of subject3:
56
enter credits:
3
Enter marks of subject4:
100
enter credits:
3
Enter marks of subject5:
100
enter credits:
1
Enter marks of subject6:
99
enter credits:
4
Enter marks of subject7:
77
enter credits:
3
Enter marks of subject8:
100
enter credits:
1
Student name:vinuthna
Student usn:1bm22cs193
Student sgpa:8.318181818181818
PS C:\Users\aDMIN\Desktop\1BM22CS193>
```

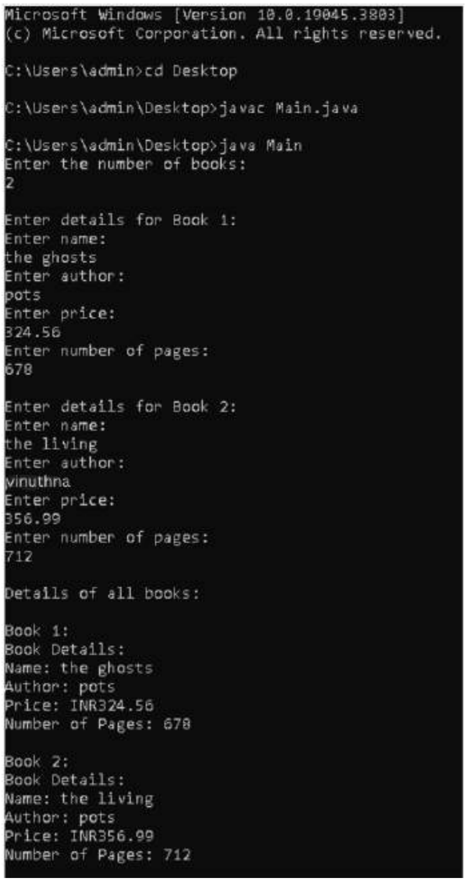
Lab 3

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();
    }
```

}

Output:



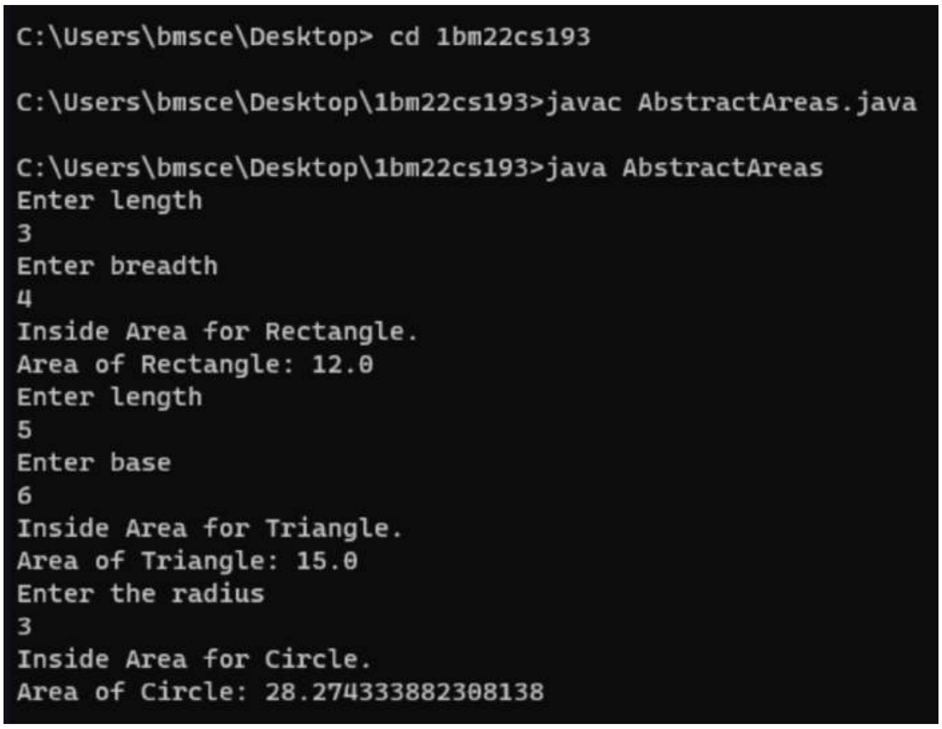
```
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\admin>cd Desktop

C:\Users\admin\Desktop>javac Main.java

C:\Users\admin\Desktop>java Main
Enter the number of books:
2

Enter details for Book 1:
Enter name:
the ghosts
Enter author:
pots
Enter price:
324.56
Enter number of pages:
678

Enter details for Book 2:
Enter name:
the living
Enter author:
vinuthna
Enter price:
356.99
Enter number of pages:
712

Details of all books:

Book 1:
Book Details:
Name: the ghosts
Author: pots
Price: INR324.56
Number of Pages: 678

Book 2:
Book Details:
Name: the living
Author: pots
Price: INR356.99
Number of Pages: 712
```

Lab 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());
    }
}
```

Output:


```
C:\Users\bmsce\Desktop> cd 1bm22cs193

C:\Users\bmsce\Desktop\1bm22cs193>javac AbstractAreas.java

C:\Users\bmsce\Desktop\1bm22cs193>java AbstractAreas
Enter length
3
Enter breadth
4
Inside Area for Rectangle.
Area of Rectangle: 12.0
Enter length
5
Enter base
6
Inside Area for Triangle.
Area of Triangle: 15.0
Enter the radius
3
Inside Area for Circle.
Area of Circle: 28.274333882308138
```

Lab 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

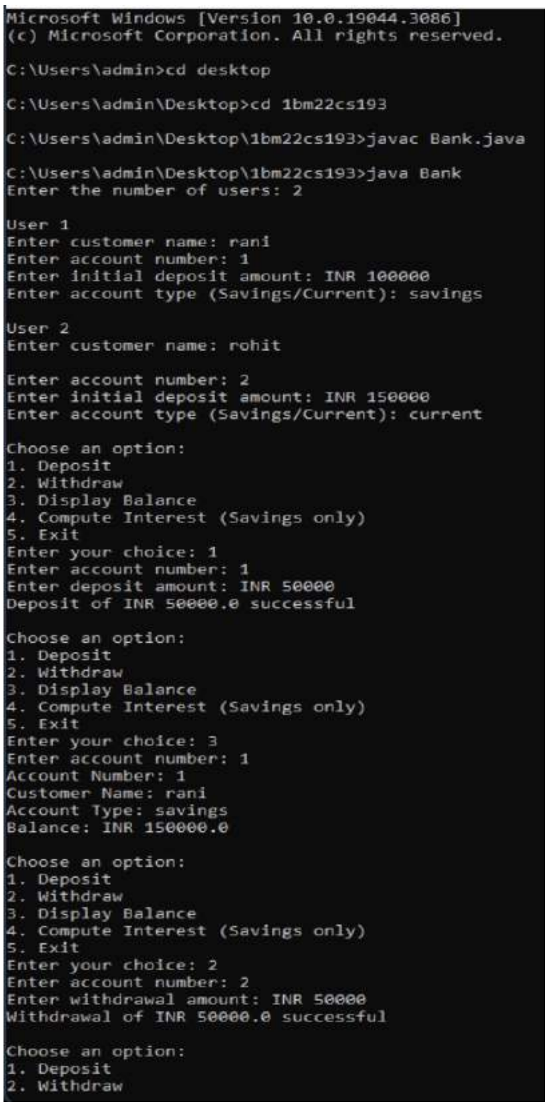
d) Permit withdrawal and update the balance

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

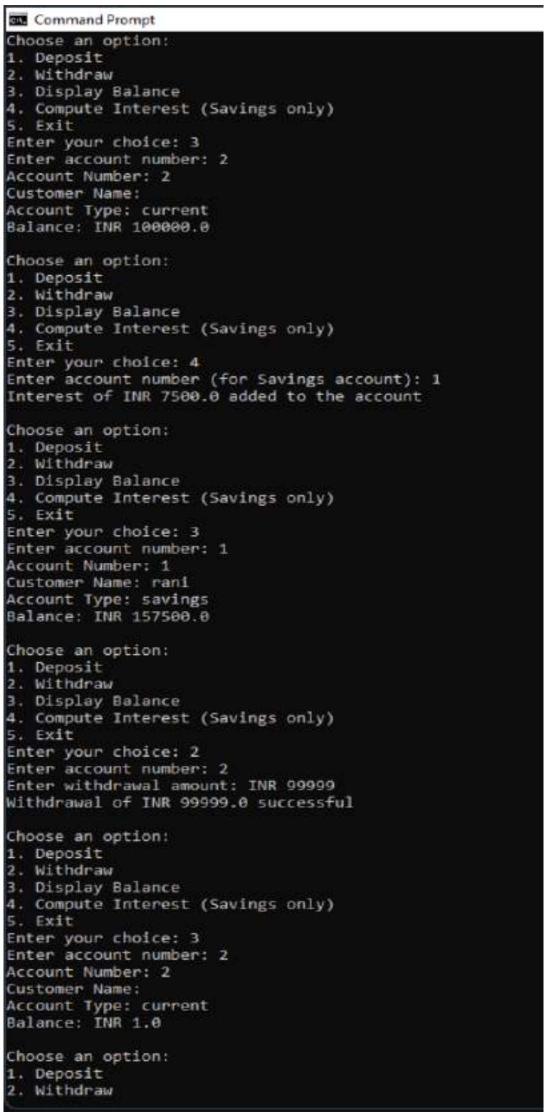
```
import java.util.Scanner;
class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;
    Account(String name, int number, String type, double initialBalance) {
        customerName = name;
        accountNumber = number;
        accountType = type;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

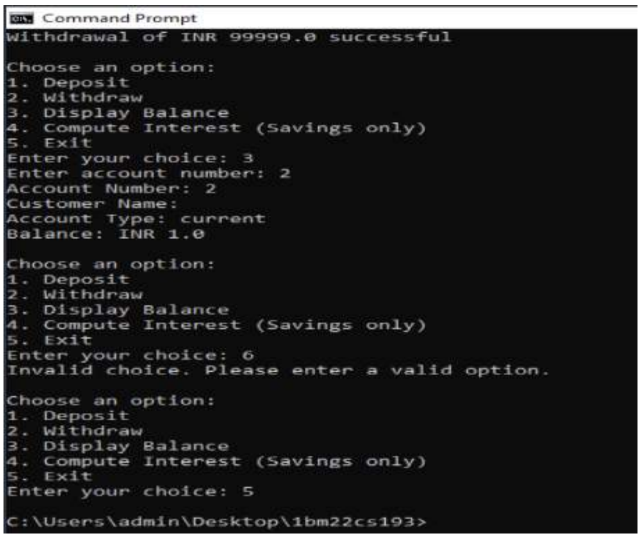
```
                System.out.print("Enter account number: ");
                accNum = scanner.nextInt();
                for (Account acc : accounts) {
                    if (acc.accountNumber == accNum) {
                        acc.displayBalance();
                    }
                }
                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.");
        }
    }
  }
}
```
Output:(Next Page)



```
Microsoft Windows [Version 10.0.19044.3086]
(c) Microsoft Corporation. All rights reserved.

C:\Users\admin>cd desktop

C:\Users\admin\Desktop>cd 1bm22cs193

C:\Users\admin\Desktop\1bm22cs193>javac Bank.java

C:\Users\admin\Desktop\1bm22cs193>java Bank
Enter the number of users: 2

User 1
Enter customer name: rani
Enter account number: 1
Enter initial deposit amount: INR 100000
Enter account type (Savings/Current): savings

User 2
Enter customer name: rohit

Enter account number: 2
Enter initial deposit amount: INR 150000
Enter account type (Savings/Current): current

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 1
Enter account number: 1
Enter deposit amount: INR 50000
Deposit of INR 50000.0 successful

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 3
Enter account number: 1
Account Number: 1
Customer Name: rani
Account Type: savings
Balance: INR 150000.0

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 2
Enter account number: 2
Enter withdrawal amount: INR 50000
Withdrawal of INR 50000.0 successful

Choose an option:
1. Deposit
2. Withdraw
```



```
Command Prompt
Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 3
Enter account number: 2
Account Number: 2
Customer Name:
Account Type: current
Balance: INR 100000.0

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 4
Enter account number (for Savings account): 1
Interest of INR 7500.0 added to the account

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 3
Enter account number: 1
Account Number: 1
Customer Name: rani
Account Type: savings
Balance: INR 157500.0

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 2
Enter account number: 2
Enter withdrawal amount: INR 99999
Withdrawal of INR 99999.0 successful

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 3
Enter account number: 2
Account Number: 2
Customer Name:
Account Type: current
Balance: INR 1.0

Choose an option:
1. Deposit
2. Withdraw
```



```
Command Prompt
Withdrawal of INR 99999.0 successful

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 3
Enter account number: 2
Account Number: 2
Customer Name:
Account Type: current
Balance: INR 1.0

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 6
Invalid choice. Please enter a valid option.

Choose an option:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute Interest (Savings only)
5. Exit
Enter your choice: 5

C:\Users\admin\Desktop\1bm22cs193>
```

Lab 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.

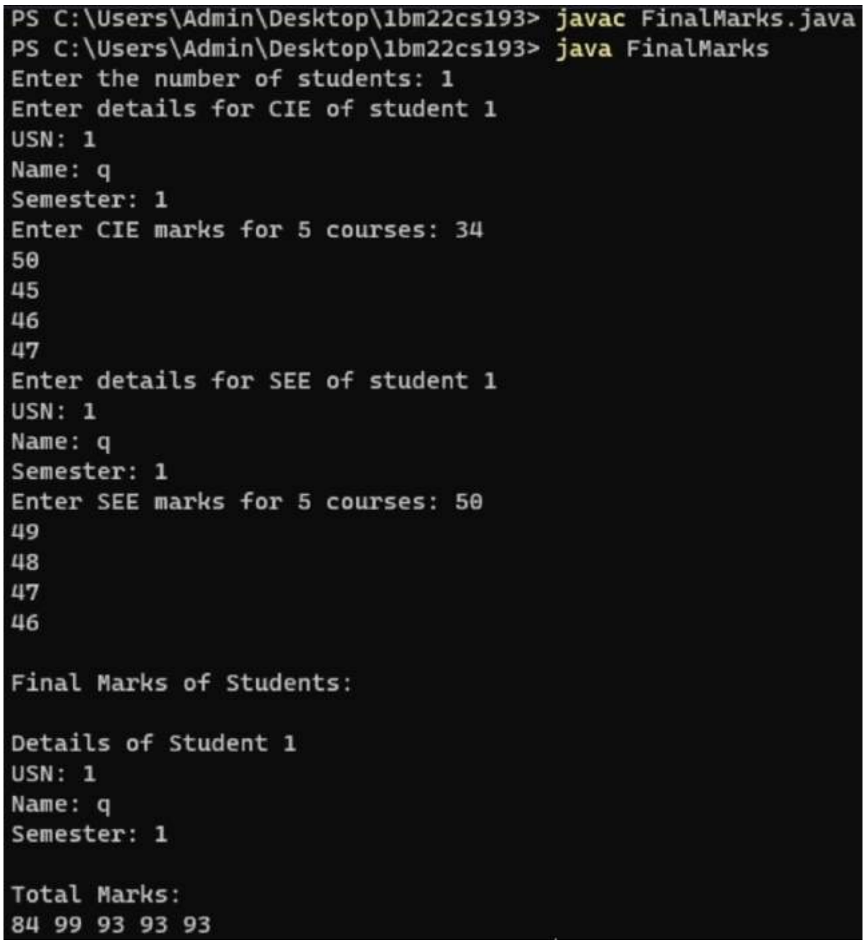
```
package CIE;
public class Student {
   public String usn;
   public String name;
   public int sem;
   public Student(String usn, String name, int sem) {
      this.usn = usn;
      this.name = name;
      this.sem = sem;
   }
}
package CIE;
public class Internals extends Student {
   public int[]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

Output:



```
PS C:\Users\Admin\Desktop\1bm22cs193> javac FinalMarks.java
PS C:\Users\Admin\Desktop\1bm22cs193> java FinalMarks
Enter the number of students: 1
Enter details for CIE of student 1
USN: 1
Name: q
Semester: 1
Enter CIE marks for 5 courses: 34
50
45
46
47
Enter details for SEE of student 1
USN: 1
Name: q
Semester: 1
Enter SEE marks for 5 courses: 50
49
48
47
46

Final Marks of Students:

Details of Student 1
USN: 1
Name: q
Semester: 1

Total Marks:
84 99 93 93 93
```

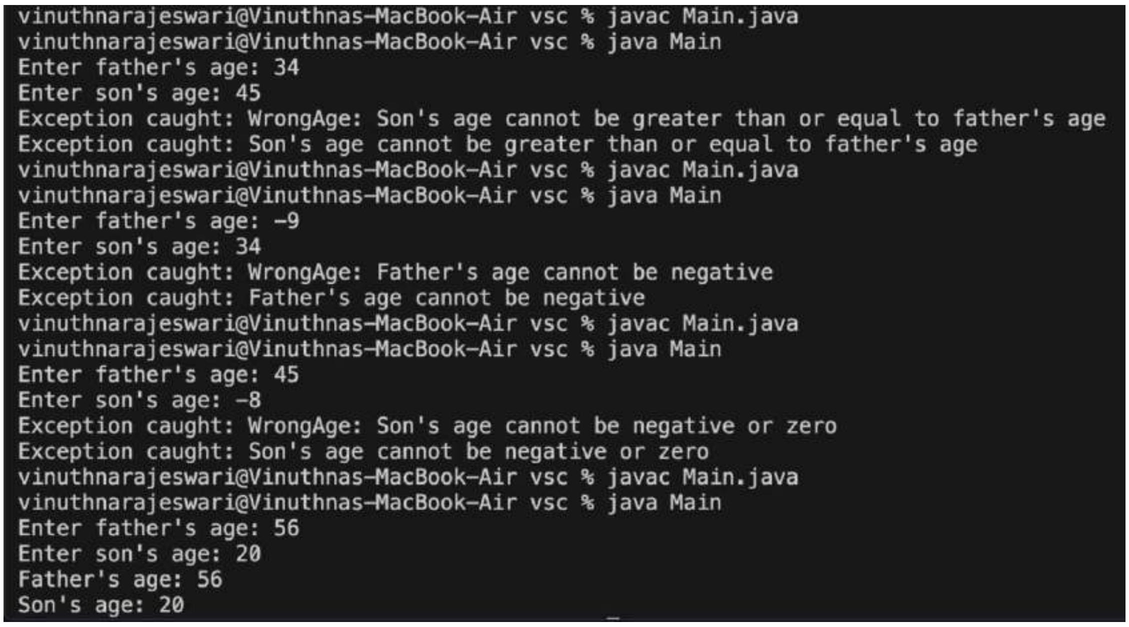
Lab 7

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();
      }
   }
}
```

Output:

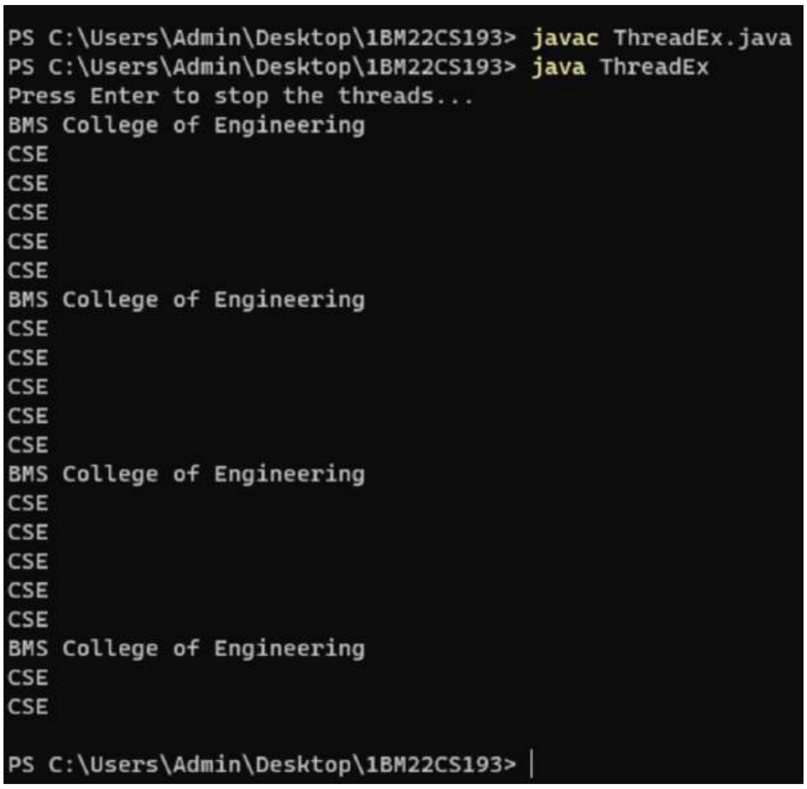

```
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % javac Main.java
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % java Main
Enter father's age: 34
Enter son's age: 45
Exception caught: WrongAge: Son's age cannot be greater than or equal to father's age
Exception caught: Son's age cannot be greater than or equal to father's age
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % javac Main.java
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % java Main
Enter father's age: -9
Enter son's age: 34
Exception caught: WrongAge: Father's age cannot be negative
Exception caught: Father's age cannot be negative
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % javac Main.java
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % java Main
Enter father's age: 45
Enter son's age: -8
Exception caught: WrongAge: Son's age cannot be negative or zero
Exception caught: Son's age cannot be negative or zero
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % javac Main.java
vinuthnarajeswari@Vinuthnas-MacBook-Air vsc % java Main
Enter father's age: 56
Enter son's age: 20
Father's age: 56
Son's age: 20
```

Lab 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();
    }
}
```

Output:

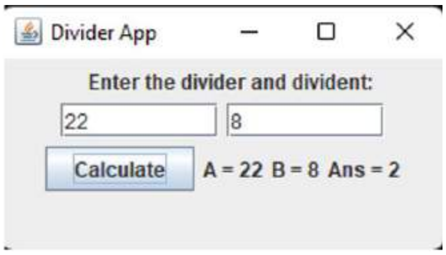

```
PS C:\Users\Admin\Desktop\1BM22CS193> javac ThreadEx.java
PS C:\Users\Admin\Desktop\1BM22CS193> java ThreadEx
Press Enter to stop the threads...
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE

PS C:\Users\Admin\Desktop\1BM22CS193> |
```

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

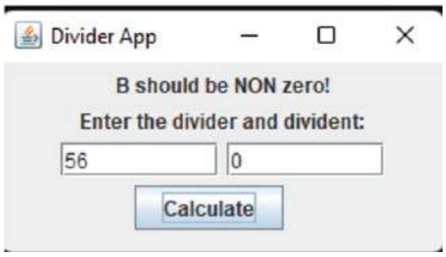
Output:


Divider App

Enter the divider and divident:

22 8

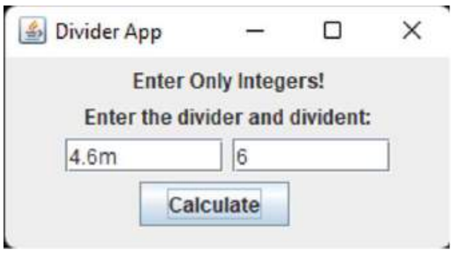
A = 22 B = 8 Ans = 2


Divider App

B should be NON zero!

Enter the divider and divident:

56 0


Divider App

Enter Only Integers!

Enter the divider and divident:

4.6m 6

Lab 10

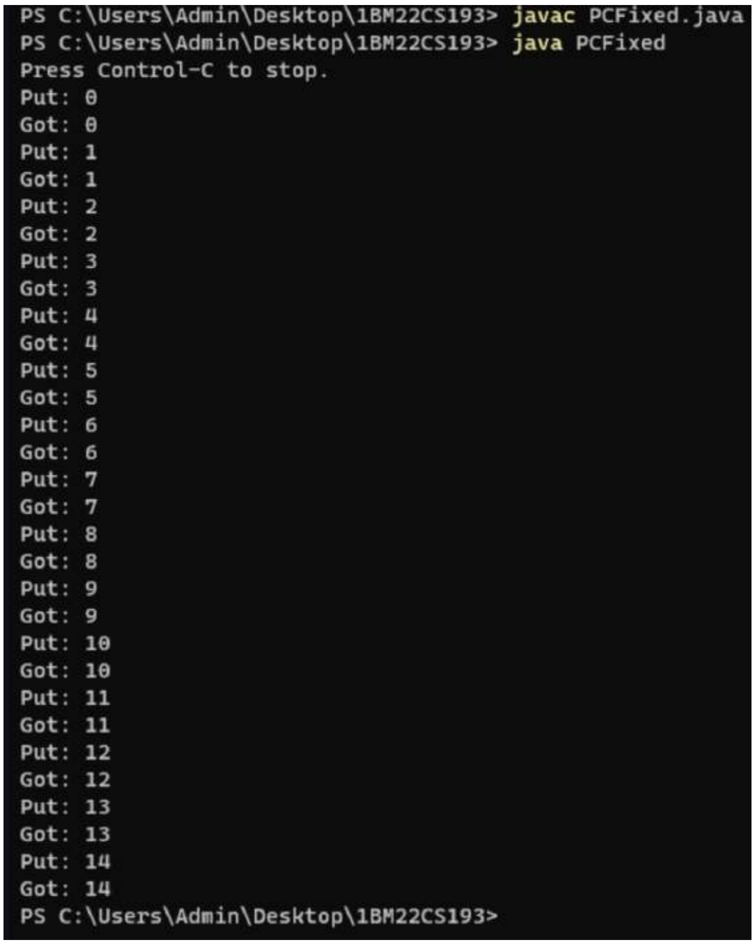
Demonstrate Inter process Communication and deadlock.

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.");
}
}
```

Output:



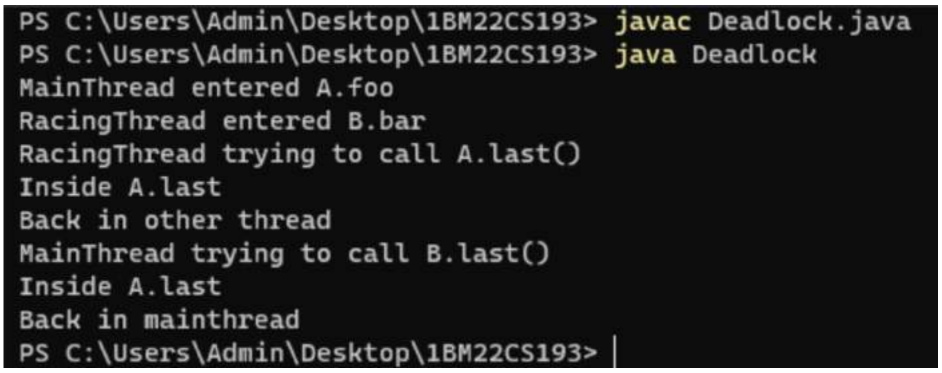
```
PS C:\Users\Admin\Desktop\1BM22CS193> javac PCFixed.java
PS C:\Users\Admin\Desktop\1BM22CS193> java PCFixed
Press Control-C to stop.
Put: 0
Got: 0
Put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
Got: 4
Put: 5
Got: 5
Put: 6
Got: 6
Put: 7
Got: 7
Put: 8
Got: 8
Put: 9
Got: 9
Put: 10
Got: 10
Put: 11
Got: 11
Put: 12
Got: 12
Put: 13
Got: 13
Put: 14
Got: 14
PS C:\Users\Admin\Desktop\1BM22CS193>
```

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();
}
}
```

Output:



```
PS C:\Users\Admin\Desktop\1BM22CS193> javac Deadlock.java
PS C:\Users\Admin\Desktop\1BM22CS193> java Deadlock
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last()
Inside A.last
Back in other thread
MainThread trying to call B.last()
Inside A.last
Back in mainthread
PS C:\Users\Admin\Desktop\1BM22CS193> |
```