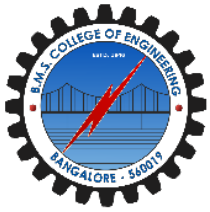
B.M.S. COLLEGE OF ENGINEERING
Basavanagudi, Bengaluru- 560019
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



LAB REPORT

On

Object Oriented Java Programming

Submitted By :
Sanjana Shetty
1BM22CS238

In partial fulfillment of
BACHELOR OF ENGINEERING
In
COMPUTER SCIENCE AND ENGINEERING
2023-24

Department of Computer Science and Engineering
B.M.S College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560019

INDEX

NAME: Sanjana Shetty STD.: 3E SEC.: _____ ROLL NO.: _____ SUB.: OOJ

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
1	12/12/23	Simple programs	10	
2	12/12/23	Quadratic Equation	10	
3	19/12/23	Class	10	
4)	26/12/23	Book Information	10	
5)	2/01/24	Abstract classes		
6)	16/01/24	Bank, Accounts		
7)	23/01/24	Generics, Strings		
8)	23/01/24	Packages	10	
9)	30/01/24	Exception Handling	10	
10)	6/02/24	Lab 8 - MultiThreading	10	6-2-24
11)	20/02/24	Lab 9 - Event Modeling	10	20.2.24

Lab Program 1

. Java program for printing all real solutions to QE $ax^2+bx+c=0$. a, b, c & use the quadratic formula. If discriminate is -ve, display a message saying 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
        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 non-zero 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 & distinct");
        System.out.println("Root 1 = "+r1+"Root 2 = "+r2);
    }
```

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

OUTPUT : ① Enter the coefficients : 3 4 5

Roots are imaginary

Root 1 = 0.0 + i 1.1055415967851332

Root 2 = 0.0 - i 1.1055415967851332

② Enter the coefficients : 1 2 1

Roots are real and equal

Root 1 = Root 2 = -1.0

(3) Enter the coefficients: 0 1 2

Not a quadratic equation
enter a different value of a.

(4) Enter the coefficients: 1 5 6

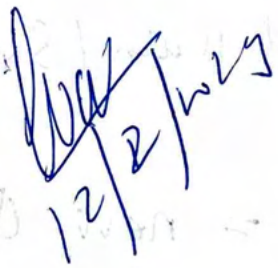
Roots are real & equal

Root 1 = 2.0

Root 2 = -3.0

Sanjana Shetty

1BM22CS238



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

$$SGPA/CGPA = \frac{\Sigma([\text{Grade points})(\text{Course credits})]}{\Sigma(\text{Course credits})}$$

```
import java.util.Scanner;

class Subject{
        int SubjectMarks;
        int credits;
        int grade;
}
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);
```

```
void getStudentDetails()
{
    System.out.print("Enter your name:");
    name = s.next();
    System.out.print("Enter your USN:");
    usn = s.next();
}
void getMarks()
{
    for(i=0; i<q; i++)
    {
        System.out.print("Enter the marks for
        subject " + (i+1) + ":");
        Subject[i].SubjectMarks = s.nextInt();
        System.out.print("Enter credits for
        Subject " + (i+1) + ":");
        Subject[i].credits = s.nextInt();
        if(subject[i].grade == 11)
            Subject[i].grade = 10;
        if(Subject[i].grade <= 4)
            Subject[i].grade = 0;
    }
}
```

```
void ComputeSGPA(){
    int effectiveScore=0;
    int TotalCredits = 0;
    effectiveScore += (subject[i].grade * subject[i].
    credits);
    int TotalCredits += subject[i].credits;

}
SGPA = (double) effectiveScore /(double)totalCredits;

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

    }

}
```

OUTPUT

Enter the name: Sanjana
Enter your usn: 238
Enter marks of subject 1:
84
Enter credits of subject 1:
4
Enter marks of subject 2:
87
Enter credits of subject 2:
4
Enter marks of subject 3:
67
Enter credits of subject 3:
4
Enter marks of subject 4:
91
Enter credits of subject 4:
3
Enter the marks of subject 5:
95
Enter the credits
3
Enter the marks of subject 6:
98
Enter the credits:
1

Enter the marks of subject 7:

94

Enter the credits:

1

Enter the marks of subject 8:

88

Enter the credits:

1

Name: Sanjana Shetty

USN: 238

SGPA: 9.75

Sanjana Shetty

1BM22CS238

Q) LAB 3

```
import java.util.Scanner;

class Books{

    String name;
    String author;
    int price;
    int num_pages;

    Books(String name, String author, int price,
            int num_pages){
        this.name=name;
        this.author=author;
        this.price=price;
        this.num_pages=num_pages;
    }
    public void setName(String name){
        this.name=name;
    }
    public String getName(){
        return "Book name: "+this.name+"\n";
    }
    public String getAuthor(){
        return "Author: "+this.author+"\n";
    }
}
```

```
public void setPrice(int price){
    this.price=price;
}

public int getPrice(){
    return this.price;
}
public void setPages(int num_pages){
    this.num_pages=num_pages;
}
public int getPages(){
    return this.num_pages;
}
}

class Main{
public static void main(String args[]){
    Scanner s=new Scanner(System.in);
    int n;
    String name;
    String author;
    int price;
    int num_pages;
    int i;
    System.out.println("Enter the book entries");
    n=s.nextInt();
    Books b[]=new Books[n];
```

```
System.out.println("Enter the name of the
book, author name, price, book pages:");
for(i=0; i<n; i++){
        name = s.next();
        author = s.next();
        price = s.nextInt();
        num_pages = s.nextInt();
        b[i] = new Books(name, author, price,
                num_pages);
    }
for(i=0; i<n; i++){
    System.out.println("Book details are:");
    System.out.println( b[i].getName() + b[i].getAuthor()
      + "Book price: " + b[i].getPrice() +
    "\n Number pages: " + b[i].getPages());
    }
}
}
```

```
System.out.println("Enter the name of the
book, author name, price, book pages:");
for(i=0; i<n; i++){
        name = s.next();
        author = s.next();
        price = s.nextInt();
        num_pages = s.nextInt();
        b[i] = new Books(name, author, price,
                num_pages);
    }
for(i=0; i<n; i++){
    System.out.println("Book details are:");
    System.out.println( b[i].getName() + b[i].getAuthor()
        + "Book price: " + b[i].getPrice() +
"\n Number pages:" + b[i].getPages());
    }
}
}
```

OUTPUT

Enter the number of book entries:

2

Enter the name of the book, author name, price & book pages:

ABC

EFG

300

20

SGD

JSS

345

39

Book details are:

Book name: ABC

Author: EFG

Book price: 300

Number of pages: 20

Book details are:

Book name: SGD

Author: JSS

Book price: 345

Number of pages: 39.

26/12/23

Sanjana Shetty
1BM22CS238

Q) Develop a Java program to create an abstract class named Shape that contains 2 integers & an empty method named printArea(). Provide 3 classes named Rectangle, Triangle, & Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints area of the given Shape.

```
import java.util.Scanner;

class InputScanner{
        Scanner s;
        Input Scanner()
        {
                s = new Scanner(System.in);
        }
}
abstract class Shape extends InputScanner{
        double a;
        double b;
        abstract void printArea();
        abstract void ~~print~~ getInput();
}
```

```
class Triangle extends Shape{
    void getInput(){
        System.out.println("Enter Dimensions of the
        triangle:");
        a=s.nextDouble();
        b=s.nextDouble();
    }
    void printArea(){
        System.out.println("Area of the triangle:"
            + (double)(a*b)/2);
    }
}
class Rectangle extends Shape{
    void getInput(){
        System.out.println("Enter the
        Dimensions of the rectangle:");
        a=s.nextDouble();
        b=s.nextDouble();
    }
    void printArea(){
        System.out.println("Area of Rectangle
        ="+(double)(a*b));
    }
}
```

```
class Circle extends Shape {
    void getInput() {
        System.out.println("Enter the Dimensions
        of the circle :");
        a = s.nextDouble(); }

    Void printArea() {
        System.out.println("Area of Circle=" +
        (double)(3.14 * a * a));
    }
}

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

OUTPUT

Enter the dimensions of the rectangle:

2
3

Enter the dimensions of the triangle:

2
4

Enter the dimensions of the circle:

3

Area of Rectangle = 6.0
Area of Triangle = 4.0
Area of circle = 28.2599

Ava
2-1-24

Sanjana Shetty
1BM22CS238

Lab 5

Q) Bank Account with Current Account & Savings Account

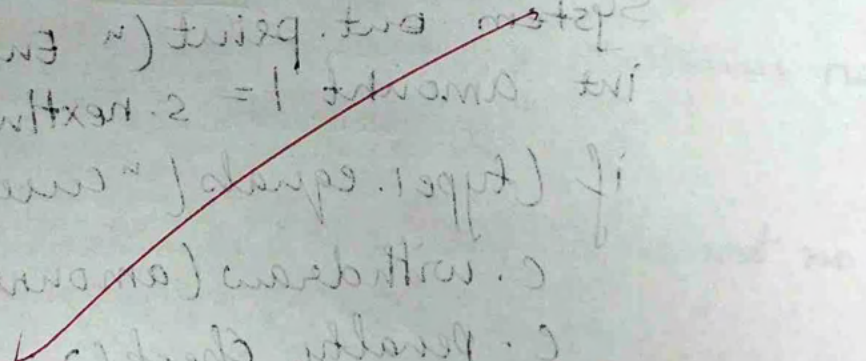
```
impat java.util.Scannee;

abstract class Account{

    String name;
    String accno; String type;

    Account(String name, int accno, String type){
     this.name=name; this.accno=accno; this.type=type;}
    final int min_balance=300;
}

class Savings extends Account{
  int balance;
  Saving(String name, int accno, String type, int balance){
  super(name, accno, type);
  this.balance=balance;
  }
  void deposit(int amount){ balance+=amount;}
  void interest(){
  double interest; final double rate=0.2;
  System.out.println("Compaunt interest:");
  interest=(balance*rate)/100;
  System.out.println(interest);
  }
```

```
void withdraw(int amount) { balance -= amount; }
void display() {
System.out.println("Customer name: " +this.name);
System.out.println("Account number: " +this.accno);
System.out.println("Type of Account: " +this.type);
System.out.println("Balance: " +this.balance);
}
}
class Current extends Account {
int balance;
Current(String name, int accno, String type, int
balance) { super(name, accno, type);
this.balance = balance; }
void deposit(int amount) { balance += amount; }
void withdraw(int amount) { balance -= amount; }
void penaltyCheck() {
if(balance < min_balance) {
int penalty = 300; balance -= penalty;
System.out.println("You have 0 balance or
negative balance due to penalty of
300:" + this.balance); } else {
System.out.println("No penalty");
}
}
```

```
    void display(){
    System.out.println("Customer name:" + this.name);
    System.out.println("Account number:" + this.accno);
    System.out.println("Type of Account:" + this.type);
    System.out.println("Balance:" +balance);
    }
}
class Bank{
    public static void main(String args[])
    {
        Scanner s= new Scanner(System.in);
        System.out.println("Enter customer name:");
        String name=s.next();
        System.out.println("Enter account no:");
        int accno= s.nextInt();
        int balance=0;
        Current c=new Current(name, accno, "current",
                              balance);
        Savings sl= new Savings(name, accno, "savings", balance);
        while(true){
            System.out.println("--MENU--");
            System.out.println("1.Deposit 2.Withdraw
            3. Compute interest 4. Display 5. Exit");
```

```
System.out.println("Enter your choice");
int choice = s.nextInt();

Switch (choice) {
case1: system.out.println("Enter account type:");
       String type = s.next();
       System.out.println("Enter deposit amount");
       int amount = s.nextInt();
       if(type.equals("current")){
           c.deposit(amount); }
       else { si.deposit(amount); }
       break;

Case2: System.out.println("Enter account type:");
       String type1 = s.next();
       System.out.print("Enter withdraw amount");
       int amount1 = s.nextInt();
       if (type1.equals("current")){
          c.withdraw(amount1);
          c.penaltyCheck(); }
       else { si.withdraw(amount1); }
       break;

case 3: System.out.println("Enter the type of
       account:");
       String type2 = s.next();
       if(type2.equals("current")){
           System.out.println("No interest provided");
           break;
       }
```

```
        else { s1.interest(); }
        break;

Case 4: System.out.println("Enter account type:");
        String type3 = s.next();
        if (type3.equals("current")){
            ~~c.penaltycheck();~~
            c.display();
        }
        else { s1.display(); }
        break;

Case 5: System.out.println("Invalid choice");
        System.exit(0);
```



OUTPUT: // Current Account

Enter customer name: Sanjana

Enter account number: 1

--MENU-- 1. Deposit 2. Withdraw 3. Interest 4. Display 5. Exit

Enter your choice: 1

Enter account type: current

Enter the deposit amount: 3000

--MENU--

Enter choice: 3

Enter type of account: current

No interest provided

--MENU--

Enter choice: 2

Enter account type: current.

Enter withdraw amount: 2800

You have 0 or negative penalty due to penalty: -100

--MENU--

Enter choice: 4

Enter account type: current

Customer name: Sanjana

Account number: 1

Type of Account: current

Balance: -100

Sanjana Shetty

1BM22CS238

// Savings account

--MENU--

Enter your choice : 1

Enter type of account: Savings

Enter the deposit amount : 4000

--MENU--

Enter your choice: 3

Enter the type of account: Savings

Compount interest : 8.0

--MENU--

Enter your choice : 4

Enter ~~your choice:~~ type of account : Savings

Customer name : Sanjana

Account number : 1

Type of account : Savings

Balance : 4000

23/1/24

Sanjana Shetty

1BM22CS238

16/01/24

Lab 6

Demonstrate various string constructors w/ java pgms

1.

```
Java
Java
ABCDEF
CDE
cde
```

2. StringLength(), string literal, string concat

```
3
6
He is 9 years old
four: 4
```

3. toString()

```
101  Sanjana  Bangalore
102  Smitha   Bangalore
```

4,5) using getchars(), extract "Bmsce" from "Welcome to Bmsce College"

— getbytes()
— toCharArray()

Bmsce // getchars()

65 // getbytes()

66 ∅

67

68

69

70

71

ABCDEFG // toCharArray()

6,7) .equals() & .regionMatches()

Bmsce equals Bmsce ⟶ true

Bmsce equals College ⟶ false

Bmsce equals BMSCE ⟶ false

Bmsce equalsIgnoreCASE BMSCE ⟶ true

substring matched : Bmsce College

8,9,10) .startsWith() & endsWith() & equals()

vs ==

true

true

false

false

Hello equals Hello ⟶ true
Hello == Hello ⟶ false

11) Perform sorting using compareTo( )

apple
ball
cat
dog
free
watch

String buffer functions

o/p

buffer before = Hello // charAt
charAt(1) before = e
buffer after = Hi
charAt(1) after = i
f // getchars

Hello from Bms College
a = H2!
I like Java!
!avaJ ekil I

After delete : This is a test.
After delete CharAt : his a test
After replace : his ~~awak~~ awatest

21) Generics

Enter elements into the integer stack

10 20 30 40 50

Enter elements in double stack

60.00 70.03 81.23 10.2 23.76

Elements of stack1	Elements of stack 2
50	23.76
40	10.2
30	81.23
20	70.03
10	60.00

live
23-1-24

Lab 6

23/01/24

// Student.java

```
package CIE;

import java.util.Scanner;

public class Student {
    protected String usn = new String();
    protected String name = new String();
    protected int sem;
    void inputStudentDetails() {

        Scanner s = new Scanner(System.in);
        usn = s.next(); name = s.next(); sem = s.nextInt();
    }
    Student() {
    }
    Student(String usn, String name, int sem) {
        this.usn = usn; this.name = name; this.sem = sem;
    }
}
```

// Internals.java

```
import java.util.Scanner;
package CIE;
```

```
public class Internals extends Student {
    protected int[] marks = new int[5];
    String name; String usn; int sem;
    void inputCIEDetails() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter internals marks in 5 subjects");
        for (int i=0; i<5; i++) {
            System.out.println("Subject " + (i+1));
            marks[i] = s.nextInt();
        }
    }
    Internals() {
    }
    Internals(String usn, String name, int sem,
    int marks[]) {
        super(usn, name, usn);
        for (int i=0; i<5; i++) {
            this.marks[i] = marks[i];
        }
    }
    void displayStudentDetails() {
```

```
System.out.println("Name "+this.name+
  " USN: "+this.usn+", Sem: "+this.sem);
System.out.println("Marks:");
for(int i=0; i<5; i++){
  system.out.println("Subject "+(i+1)+": "+
                     marks[i]);
  }
}
}
```

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

public class External extends Internals{
  protected int[] marks1 = new int[5];
  protected int finalMarks = new int[5];
External(){
}
```

```
public void inputSEEmark(){
    System.out.println("Enter SEE marks for 5 subjects");
    for(int i=0; i<5; i++){
        System.out.println("Subject"+(i+1)+"marks:");
        marks1[i] = s.nextInt();
    }
}
public void calcFinalMarks(){
    for(int i=0; i<5; i++){
        finalMarks = marks1[i]/2 + Super.marks[i];
    }
}
public void displayFinalmarks(){
    displayStudentDetails();
    for(int i=0; i<5; i++){
        System.out.println("Subject"+(i+1)+":"+finalsMarks[i]);
    }
}
}
```

```
// main.java

import SEE.Externals;
import java.util.Scanner;

class main{

    public static void main(String args[]){
        Scanner s=new Scanner(System.in);
        int n;
        System.out.println("enter number of
          students:");
        n=s.nextInt();
        External finalMarks[]=new External[n];
        for(int i=0;i<n;i++){

            finalMarks[i]=new External();
            finalMarks[i].inputStudentDetails();
            finalMarks[i].inputCIEDetails();
            finalMarks[i].inputSEEMarks();
        }
        System.out.println("finals details:");
          for(int i=0;i<n;i++){

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

    }

}
```

// Output

Number of students : 2

Enter usn : 238

Enter name : Sanjana

Semester : 3

Enter CIE marks :

Subject 1 : 22

Subject 2 : 27

Subject 3 : 34

Subject 4 : 37

Subject 5 : 36

Enter SEE marks :

Subject 1 : 90

Subject 2 : 92

Subject 3 : 86

Subject 4 : 95

Subject 5 : 93

Displaying data -

Name : Sanjana

USN : 238

Semester : 3

Subject 1 : 78

Subject 2 : 82

Subject 3 : 71

Subject 4 : 84

Subject 5 : 83

Sanjana Shetty

1BM22CS238

25/11/24

Lab 7

2)

```
import java.util.Scanner;

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

class Father{
    int fatherAge;
    Scanner s = new Scanner(System.in)
    Father(){
        System.out.println("Enter Father's age");
        fatherAge = s.nextInt();
    }
    void check1() throws WrongAge {
        if (fatherAge < 0){
            throw new WrongAge("Age cannot be
            negative");
        }
        else{
            System.out.println("Father's age "+fatherAge
        }
    }
}
```

```
class Son extends Father {
    int sonAge;
    Scanner s = new Scanner(System.in);
    Son() {
        super();
        System.out.println("Enter son's age");
        sonAge = s.nextInt();
    }
    void check2() throws WrongAge {
        if (sonAge >= fatherAge){
            throw new WrongAge("Son's age cannot
            be greater than father's age");
        else if (sonAge < 0){
            throw new WrongAge("age can't be
            negative");
        }
        else {
            System.out.println("Son's age: "+sonAge
                                                ;
        }
    }
}
```

```
Class Main {
    public static void main(String args[]) {
        Son s = new Son();
        try {
            s.check2();
        }
        Catch (Wrong Age e) {
            System.out.println(e.getMessage());
        }
    }
}
```

OUTPUT :

Sanjana Shetty
1BM22CS238

1) Enter Father's age
30
Enter son's age
45
Son's age cannot be greater than father's age

2) Enter Father's age
45
Enter Son's age:
15
Son's age: 15

Lab 8

```
class DisplayMessageThread extends Thread {
    private final String message;
    private final long interval;    // in ms

    DisplayMessageThread(String message, long interval) {
        this.message = message;
        this.interval = interval;
    }
    public void run() {
        try {
            while(true) {
                System.out.println(message);
                Thread.sleep(interval);
            }
        } catch (InterruptedException e) {
            System.out.println(Thread.currentThread().getName() + " interrupted");
        }
    }
}
```

```
public class TwoThreeDemo{
    public static void main (String [] args) {
        DisplayMessageThread thread1 = new DisplayMessage
        Thread ("BMS college of Engineering", 10000);

        DisplayMessageThread thread1 = new DisplayMessage
                                       Thread ("CSE", 2000);

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

        thread1.start();
        thread2.start()

        try{
            Thread.sleep(3000);
        }
        Catch (InterruptedException e) {
            System.out.println ("Main thread exception.");
            }

            thread1.interrupt();
            thread2.interrupt();
            System.out.println ("Main thread exiting");
    }
}
```

OUTPUT :

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

Main thread exiting
Thread 2 interrupted
Thread 1 interrupted.

Sanjana Shetty
1BM22CS238

6-2-24

13/2/24

Lab 10

```
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\nIntimate Producer\n");
          notify();
          return n;
      }
  Synchronized void put(int n) {
      while (valueSet)
        try {
          System.out.println("\n Producer Waiting\n");
          wait();
  }
```

```
        catch (InterruptedException e)
        {
            System.out.println("InterruptedException
                                caught");
        }

    }
    this.n = n;
    valueSet = true;
    System.out.println("Put: " + n);
    System.out.println("\nIntimate Consumer\n");
    notify();
  }
}
class Producer implements Runnable {
    Q q;
    Producer(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);
        New Cor
        System.out.println("Press Control-C
          to stop.");
    }
}
```

Output:

```
Put : 1
Got : 1
Put : 2
Got : 2
Put : 3
Got : 3
Put : 4
Got : 4
Put : 5
Got : 5
```

Sanjana Shetty
1BM22CS238

Wk 2-24
13-

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("
        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();
    }

}
```

Sanjana Shetty
1BM22CS238

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
```

Lab 9

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

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

public class DivisionMain1 extends Frame implements
ActionListener {

    TextField num1, num2;
    Button dResultt;
    Label outResult;
    String out = " ";
    double resultNum;
    int flag = 0;

    public DivisionMain1()
{
    setLayout(new FlowLayout());

    dResult = new Button("RESULT");
    Label number1 = new Label("Number 1:",
    Label.RIGHT);

    Label number2 = new Label("Number 2:"
    Label.RIGHT);

    num1 = new TextField(5);
    num2 = new TextField(5);
    outResult = new Label("Result:", Label.RIGHT);
    add(number1);    add(num1);
    add(number2);    add(num2);
```

```
add(dResult);      add(outResult);
num1.addActionListener(this);
num2.addActionListener(this);
dResult.addActionListener(this);
addWindowListener(new WindowAdapter()
{   public void windowClosing(WindowEvent
                                we)
    {
        System.exit(0);
    }
});
}
public void actionPerformed(ActionEvent ae)
{
    int n1, n2;
    try
    {
        if(ae.getSource()==dResult)
        {
            n1 = Integer.parseInt(num1.
                                  getText());
            n2 = Integer.parseInt(num2.getText());
            if(n2==0)
                throw new ArithmeticException();
            out = n1 + " " + n2;
            resultNum = n1/n2;
            out += String.valueOf(resultNum);
            repaint();
    }   }
```

```
Catch (NumberFormatException e1)
{
    flag=1;
    out = "Number Format Exception! "+e1;
    repaint();
}
Catch (ArithmeticException e2)
{
    flag=1;
    out = "Divide by 0 excepteon! "+e2;
    repaint();
}
}
public void paint(Graphics g)
{
    if(flag==0)
    g.drawString(out, outResult.getX() + outRsult.
    getWidth() / outResult.getY() + outResult.get
    Height() - 8);
    else
    g.drawString(out, 100, 200);
    flag=0;
}
public static void main (Steing[] args)
{
    DivisionMain1 dm = new DivisionMain1();
```

```
Catch (NumberFormatException e1)
{
    flag=1;
    out = "Number Format Exception! "+e1;
    repaint();
}
Catch (ArithmeticException e2)
{
    flag=1;
    out = "Divide by 0 excepteai! "+e2;
    repaint();
}
}
public void paint(Graphics g)
{
    if(flag==0)
    g.drawString(out, outResult.getX() + outResult.
    getWidth() / outResult.getY() + outResult.get
    Height() -8);
    else
    g.drawString(out, 100, 200);
    flag=0;
}
public Static void main (Steing[] args)
{
    DivisionMain1 dm = new DivisionMain1();
```

```
dm.setSize(new Dimension(800,400));
dm.setTitle("Division Of Integers");
dm.setVisible(true);
}
}
```

output:

Number 1: 6 Number 2: 2

Result: 6 2 3.0

Sanjana Shetty
1BM22CS238

24/2/2024

1) addActionListener (object_name)

- callback mechanism when any action is perfom
- method of Action Listener interface

2) add Window Listener ( )

- it used to process window events
- it adds the specified window listener to receive window events

3) getText ( )

- function to get text input from the user

4) repaint ( )

- an asynchronous method of applet class.

5) drawString()

- takes instance of String class as parameter containing the text to be draw & two integer values specifying the coordinates where text should start.

6) ~~getHeight() / getWidth()~~ setSize()

- Sets the size of the current Dimension object to the specified width & height.

7) setLayout()

- method that allows to set the layout of the container

8) JFrame

- JFrame is a class in java that consists of methods for setting size or visibility

9) JLabel

↳ JLabel instance with the specified text, image, and horizontal alignment.

10) setVisible()

- method makes the Frame appear on the screen.

20/2/24

Sanjana Shetty
1BM22CS238

LABS SOURCE CODE

Lab 1 (12/2/24)

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()
   {
      System.out.println("Sanjana Shetty, 1BM22CS238");
      while(a==0){
         System.out.println("not a quadratic equation");
         System.out.println("enter a non-zero value of a");
         Scanner s=new Scanner(System.in);
         a=s.nextInt();
      }
      d=b*b-4*a*c;
      if(d==0){
         r1=(-b)/(2*a);
         System.out.println("Roots are real and equal");
         System.out.println("Root1=root2="+r1);
      }
      else if(d>0){
```

```
            r1=((-b)+(Math.sqrt(d)))/(2*a);
            r1=((-b)-(Math.sqrt(d)))/(2*a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root1="+r1+"Root2="+r2);
        }
        else if(d<0){
            System.out.println("Roots are imaginary");
            r1=(-b)/(2*a);
            r2=Math.sqrt(-d)/(2*a);
            System.out.println("Root1="+r1+" +i"+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 2 (19/12/23)

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

```
import java.util.*;
class Subject
{
        int subjectMarks;
        int credits;
        int grade;
}
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);
	}
	void getStudentDetails()
	{
		System.out.println("Enter student name");
		name=s.next();
		System.out.println("Enter student USN");
		usn=s.next();
	}
	void getMarks()
	{
		for(int i=0;i<9;i++)
		{
			System.out.println("Enter marks");
			subject[i].subjectMarks=s.nextInt();
			System.out.println("Enter number of +credits");
			subject[i].credits=s.nextInt();
			subject[i].grade=(subject[i].subjectMarks/10)+1;
			if(subject[i].grade==11)
				subject[i].grade=10;
			if(subject[i].grade<=4)
				subject[i].grade=0;

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

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

Lab 3 (26/12/24)

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

```
import java.util.Scanner;

class Books {
   String name;
   String author;
   int price;
   int num_pages;

   Books(String name, String author, int price, int num_pages) {
      this.name = name;
      this.author = author;
      this.price = price;
      this.num_pages = num_pages;
```

```
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return "Book name: " + this.name + "\n";
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    public String getAuthor() {
        return "Author: " + this.author + "\n";
    }

    public void setPrice(int price) {
        this.price = price;
    }

    public int getPrice() {
        return this.price;
    }

    public void setPages(int num_pages) {
        this.num_pages = num_pages;
    }

    public int getPages() {
        return this.num_pages;
    }
}

class Main {
    public static void main(String args[]) {
        Scanner s = new Scanner(System.in);
        int n;
        String name;
        String author;
        int price;
        int num_pages;
        int i;
```

```
        System.out.println("Enter the number of book entries");
        n = s.nextInt();
        Books b[] = new Books[n];

        System.out.println("Enter the name of the book, author name, price and book pages:");
        for (i = 0; i < n; i++) {
            name = s.next();
            author = s.next();
            price = s.nextInt();
            num_pages = s.nextInt();
            b[i] = new Books(name, author, price, num_pages);
        }

        for (i = 0; i < n; i++) {
            System.out.println("Book details are:");
            System.out.println(b[i].getName() + b[i].getAuthor() + "Book price: " + b[i].getPrice() +
"\nNumber of pages: " + b[i].getPages());
        }
    }
}
```

Lab 4 (2/01/24)

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.

```
class Figure{
    int a; int b;
    Figure(int a, int b){
        this.a=a;
        this.b=b;
    }
    double area(){
        return 0;
    }
}
class Rectangle extends Figure{
    Rectangle(int a, int b){
```

```
        super(a,b);
    }
    double area(){
        return a*b;
    }
}
class Triangle extends Figure{
    Triangle(int a, int b){
        super(a,b);
    }
    double area(){
        return (0.5)*a*b;
    }
}
class Main{
    public static void main(String args[]){
        Rectangle r=new Rectangle(2,3);
        Triangle t=new Triangle(1,2);
        Figure f=new Rectangle(2,2);
        System.out.println("Area of Rectangle:"+r.area());
        System.out.println("Area of Triangle:"+t.area()+"\n");
        System.out.println(f.area());
    }
}
```

Lab 5 (16/01/24)

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;

abstract class Account {
    String name;
    int accno;
    String type;
    Account(String name, int accno, String type) {
        this.name = name;
        this.accno = accno;
        this.type = type;
    }
    final int min_balance = 300;
}
class Savings extends Account {
    int balance;
    Savings(String name, int accno, String type, int balance) {
        super(name, accno, type);
        this.balance = balance;
    }
    void deposit(int amount) {
        balance += amount;
    }
    void interest() {
        double interest;
        final double rate = 0.2;
        System.out.println("Compound interest:");
        interest = balance * rate / 100;
        System.out.println(interest);
    }
    void withdraw(int amount) {
        balance -= amount;
    }
    void display() {
        System.out.println("Customer name: " + this.name);
```

```
        System.out.println("Account number: " + this.accno);
        System.out.println("Type of Account: " + this.type);
        System.out.println("Balance: " + this.balance);
    }
}

class Current extends Account {
    int balance;
    Current(String name, int accno, String type, int balance) {
        super(name, accno, type);
        this.balance = balance;
    }
    void deposit(int amount) {
        balance += amount;
    }
    void withdraw(int amount) {
        balance -= amount;
    }
    void penaltyCheck() {
        if (balance < min_balance) {
            int penalty = 300;
            balance -= penalty;
            System.out.println("You have 0 or negative balance due to penalty: " + this.balance);
        } else {
            System.out.println("No penalty");
        }
    }
    void display() {
        System.out.println("Customer name: " + this.name);
        System.out.println("Account number: " + this.accno);
        System.out.println("Type of Account: " + this.type);
        System.out.println("Balance: " + balance);
    }
}

class Bank {
    public static void main(String args[]) {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter customer name:");
        String name = s.next();
        System.out.println("Enter account number:");
        int accno = s.nextInt();
        int balance = 0;
        Current c=new Current(name,accno,"current",balance);
```

```
        Savings s1=new Savings(name,accno,"savings",balance);
        while (true) {
            System.out.println("--MENU--");
            System.out.println("1. Deposit 2. Withdraw 3. Compute interest for Savings account 4.
Display account details 5. Exit");
            System.out.println("Enter your choice:");
            int choice = s.nextInt();

            switch (choice) {
                case 1:
                    System.out.println("Enter the type of account:");
                    String type = s.next();
                    System.out.println("Enter the deposit amount:");
                    int amount = s.nextInt();
                    if (type.equals("current")) {
                        c.deposit(amount);
                    } else {
                        s1.deposit(amount);
                    }
                    break;
                case 2:
                    System.out.println("Enter the type of account:");
                    String type1 = s.next();
                    System.out.println("Enter the withdraw amount:");
                    int amount1 = s.nextInt();
                    if (type1.equals("current")) {
                        c.withdraw(amount1);
                        c.penaltyCheck();
                    } else {
                        s1.withdraw(amount1);
                    }
                    break;
                case 3:
                    System.out.println("Enter the type of account:");
                    String type2 = s.next();
                    if (type2.equals("current")) {
                        System.out.println("No interest provided");
                        break;
                    } else {
                        s1.interest();
                    }
                    break;
                case 4:
                    System.out.println("Enter the type of account:");
```

```
                String type3 = s.next();
                if (type3.equals("current")) {
                    c.display();
                } else {
                    s1.display();
                }
                break;
            case 5:
                System.out.println("Invalid choice");
                System.exit(0);
        }
    }
  }
}
```

Lab 6 (23/01/24)

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

```
//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 sc=new Scanner(System.in);
```

```
System.out.println(“Enter student usn");
usn=sc.next();
System.out.println("Enter student name");
name=sc.next();
System.out.println("Enter student semester");
sem=sc.nextInt();
}

public void displayStudentDetails() {
System.out.println("student usn:"+ usn);
System.out.println("student name:"+n ame);
System.out.println(" student sem:"+ sem);
}
}

//Internals.java
package CIE;
import java.util.Scanner;

public class Internals extends Student {

protected int marks[] = new int[5];

public void inputCIEmarks()
{
Scanner sc=new Scanner(System.in);
for (int i=0;i<5;i++)
{
System.out.println("Enter 5 subject marks");
marks[i]=sc.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 sc = new Scanner(System.in);

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

System.out.print("Subject "+(i+1)+" marks: ");

marks[i] = sc.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]);

}

}

//Main1.java
import SEE.Externals;

class Main1 {

public static void main(String args[])

{

int numOfStudents = 2;

Externals finalMarks[] = new
Externals[numOfStudents];

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

{

finalMarks[i] = new Externals();
finalMarks[i].inputStudentDetails();

System.out.println("Enter CIE marks");

finalMarks[i].inputCIEmarks();

System.out.println("Enter SEE marks");
```

```
finalMarks[i].inputSEEmarks();

}
System.out.println("Displaying data:\n");

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

{

finalMarks[i].calculateFinalMarks();

finalMarks[i].displayFinalMarks();

} //end of for loop

} // end of public main

} //end of class main
```

Lab 7 (30/01/2024)

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

```
import java.util.Scanner;

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


class Father{
int fatherAge;
Scanner s=new Scanner(System.in);
```

```
Father(){
System.out.println("Enter Father's age");
fatherAge=s.nextInt();
}
void check1() throws WrongAge{
if(fatherAge<0){
throw new WrongAge("Age cannot be negative");
}
else{
System.out.println("Father's age"+fatherAge);
}
}
}
class Son extends Father{

int sonAge;
Scanner s=new Scanner(System.in);
Son(){
super();
System.out.println("Enter son's age");
sonAge=s.nextInt();
}
void check2() throws WrongAge{
if(sonAge>=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{
System.out.println("Son's age: "+sonAge);
}
}
}
class Main{
public static void main(String args[]){
Son s=new Son();
try{
s.check2();
}
catch(WrongAge e){
System.out.println(e.getMessage());
}
```

```
}
}
```

Lab 8 (6/02/24)

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 DisplayMessageThread extends Thread {
    private final String message;
    private final long interval; // in milliseconds

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

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

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

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

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

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

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

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

Lab 9 (20/02/24)

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

The user enters two numbers in the text fields, Num1 and Num2. The division of

Num1 and Num2 is displayed in the Result field when the Divide button is clicked.

If Num1 or Num2 were not an integer, the program would throw a

NumberFormatException. If Num2 were Zero, the program would throw an

Arithmetic Exception Display the exception in a message dialog box.

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

public class DivisionMain1 extends Frame implements ActionListener
{
        TextField num1,num2;
        Button dResult;
        Label outResult;
```

```
String out="";
double resultNum;
int flag=0;

public DivisionMain1()
{
	setLayout(new FlowLayout());

	dResult = new Button("RESULT");
	Label number1 = new Label("Number 1:",Label.RIGHT);
	Label number2 = new Label("Number 2:",Label.RIGHT);
	num1=new TextField(5);
	num2=new TextField(5);
	outResult = new Label("Result:",Label.RIGHT);

	add(number1);
	add(num1);
	add(number2);
	add(num2);
	add(dResult);
	add(outResult);

	num1.addActionListener(this);
	num2.addActionListener(this);
	dResult.addActionListener(this);
	addWindowListener(new WindowAdapter()
	{
		public void windowClosing(WindowEvent we)
		{
			System.exit(0);
		}
	});
}

public void actionPerformed(ActionEvent ae)
{
	int n1,n2;
	try
	{
		if (ae.getSource() == dResult)
		{
			n1=Integer.parseInt(num1.getText());
			n2=Integer.parseInt(num2.getText());
```

```
                        /*if(n2==0)
                                throw new ArithmeticException();*/
                        out=n1+" "+n2;
                        resultNum=n1/n2;
                        out+=String.valueOf(resultNum);
                        repaint();

                }
        }
        catch(NumberFormatException e1)
        {
                flag=1;
                out="Number Format Exception!  "+e1;
                repaint();
        }
        catch(ArithmeticException e2)
        {
                flag=1;
                out="Divide by 0 Exception!  "+e2;
                repaint();
        }

    }

    public void paint(Graphics g)
    {
        if(flag==0)

g.drawString(out,outResult.getX()+outResult.getWidth(),outResult.getY()+outResult.getHeight()-
8);
        else
        g.drawString(out,100,200);
        flag=0;
    }

    public static void main(String[] args)
    {
        DivisionMain1 dm=new DivisionMain1();
        dm.setSize(new Dimension(800,400));
        dm.setTitle("DivionOfIntegers");
        dm.setVisible(true);
    }

}
```

Lab 10 (13/02/24)

10.A)Demonstrate Inter process Communication and deadlock

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

			}
		System.out.println("Got: " + n);
		valueSet = false;
		System.out.println("Intimate Producer\n");
		notify();
		return n;
	}
	synchronized void put(int n) {
		while(valueSet){
			try {
				System.out.println("Producer waiting");
				wait();
				} catch(InterruptedException e) {
					System.out.println("InterruptedException caught");
				}

			}
		this.n = n;
		valueSet = true;
		System.out.println("Put: " + n);
```

```
                System.out.println("Intimate 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<5) {
                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<5) {
                        int r=q.get();
                        System.out.println("consumed:"+r);
                        i++;
                }
        }
}
class PCFixed {
        public static void main(String args[]) {
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
        }
}
```

10B)
DEADLOCK

```
class A {

synchronized void foo(B b) {

String name =Thread.currentThread().getName();

System.out.println(name + " entered A.foo");

try {

Thread.sleep(1000);

} catch(Exception e) {

System.out.println("A Interrupted");

}

System.out.println(name + " trying to call B.last()");

b.last();

}

void last() {

System.out.println("Inside A.last");

}

}

class B {
```

```
synchronized void bar(A a) {

String name =Thread.currentThread().getName();

System.out.println(name + " entered B.bar");

try {

Thread.sleep(1000);

} catch(Exception e) {
System.out.println("B Interrupted");
}
System.out.println(name + " trying to call A.last()");
a.last();
}
void last() {
System.out.println("Inside A.last");
}
}
class Deadlock implements Runnable
{
A a = new A();
B b = new B();

Deadlock() {
Thread.currentThread().setName("MainThread");
Thread t = new Thread(this,"RacingThread");
t.start();
a.foo(b); // get lock on a in this thread.
System.out.println("Back in main thread");
}
public void run() {
b.bar(a); // get lock on b in other thread.
System.out.println("Back in other thread");
}
public static void main(String args[]) {
new Deadlock();

}

}
```