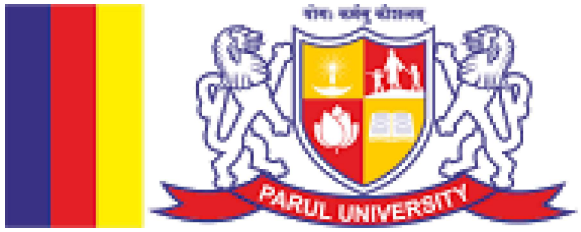


Parul University

FACULTY OF ENGINEERING AND TECHNOLOGY

BACHELOR OF TECHNOLOGY

OBJECT ORIENTED PROGRAMMING WITH JAVA (203105334)

5TH SEMESTER

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

Laboratory Manual

PREFACE

CERTIFICATE

This is to certify that

Mr. **Rahulraj Singh** *with enrolment no.* **190303105570** *and semester/division* **5B8** *has successfully completed his laboratory experiments in the Object-Oriented Programming with Java (203105334) from the department of Computer Science And Engineering during the academic year 2021-2022.*

Date of Submission: …… Staff In Charge: ……

Head of Department: ……

INDEX

6	Write a program to explain static polymorphism in java.	18				
6.1	Write a program to find volume of Box using concept of method overloading.	19				
7	Write a program to find the factorial of a number using interface.	20				
7.1	Write a program to implement multiple inheritance in java using interface.	21				
7.2	Create a package called Mathsoperation1, which must contain classes to perform addition, subtraction, create another package called Mathsoperation2, which must contain classes to perform multiplication and division operation. Create a main class and import the Mathsoperation1, Mathsoperation1 package in it to perform all the operations on the input numbers provided by the user. Finally, display the result of each operation on the console.	22				
8	Write a program to design student registration form using AWT components.	24				
8.1	Write a Java Program for Calculator Operations Using AWT Controls & appropriate layout manager.	27				
9	Write a program to demonstrate array index out of bounds exception.	37				
9.1	Create an interface Account with two methods deposit and withdraw. Create class Savings Account which implements the interface. Write a custom Exception handler for	38				

	Savings Account to handle the scenarios when withdrawn amount is larger than the balance in the account.					
10	Write a program to demonstrate class object locking using method level synchronization.	40				
10.1	Write a program that executes two threads. One thread will print the even numbers and another thread will print odd numbers from 1 to 50.	42				

PRACTICAL – 1

Aim: - Write a program to count the number of words that start with a capital letter.

Code: -

```
import java.util.Scanner;
public class practical_1 {
    public static void main(String[] args) {
        String str1;
        Scanner obj = new Scanner(System.in);
        System.out.println("Enter any String: ");
        str1 = obj.nextLine();
        int len = str1.length();
        char c;
        int count = 0;
        for (int i = 0; i < len; i++) {
            c = str1.charAt(i);
            if (c>=65 && c<=90) {
                count++;
            }
        }
        System.out.println(str1);
        System.out.println("No. of capital letter in given str
ing is " + count);
    }
}
```

Output: -

```
Enter any String:
Ankit Singh
Ankit Singh
No. of captital letter in given string is 2
```

PRACTICAL – 1.1

Aim: - Write a java program to take an array of strings as an input, and arrange strings in ascending order.

Code: -

```
import java.util.Arrays;
import java.util.Scanner;
public class StringSort {
    public static void main(String[] args) {
        String name[];
        Scanner obj = new Scanner(System.in);
        System.out.println("Enter Six Strings: ");
        name = new String[6];
        for (int i = 0; i < 6; i++) {
            name[i] = obj.nextLine();
        }
        Arrays.sort(name);
        System.out.print("Sorted Array: ");
        System.out.println(Arrays.toString(name));

    }
}
```

Output: -

```
Enter Six Strings:
Age
Dash
Line
Text
Game
Play
Sorted Array: [Age, Dash, Game, Line, Play, Text]
```

PRACTICAL – 2

Aim: - Write a program to find the largest number in an array of numbers using command line arguments.

Code: -

```
public class max {
    public static void main(String[] args) {
        int num, maximum;
        num = Integer.parseInt(args[0]);
        int arr[] = new int[num];
        for (int i = 0; i < num; i++) {
            arr[i] = Integer.parseInt(args[i+1]);
        }
        maximum = arr[0];
        for (int i = 0; i < num; i++) {
            if (maximum<arr[i]) {
                maximum = arr[i];
            }
        }
        System.out.println("Maximum value from given array is:
 " + maximum);
    }
}
```

Output: -

```
PS D:\5th Semester\OOPJ\Lab> javac max.java
PS D:\5th Semester\OOPJ\Lab> java max 4 2 3 4 23
Maximum value from given array is: 23
```

PRACTICAL – 2.1

Aim: - Write a program to find factorial of number. Here, take number as command line argument.

Code: -

```
public class fact {
    public static void main(String[] a) {
        int number;
        number = Integer.parseInt(a[0]);
        int n = 1;
        for (int i = 1; i <= number; i++) {
            n = n * i;
        }
        System.out.println("The factorial of " + number + " is
 " + n);
    }
}
```

Output: -

```
PS D:\5th Semester\OOPJ\Lab> javac fact.java
PS D:\5th Semester\OOPJ\Lab> java fact 5
The factorial of 5 is 120
```

PRACTICAL – 3

Aim: - Write a program to demonstrate class and objects using the concept of an array object.

Code: -

```
import java.util.Scanner;
public class example {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Number of Students: ");
        int number = sc.nextInt();
        student data[] = new student[number];
        for (int i = 0; i < number; i++) {
            data[i] = new student();
            data[i].dataInsert();
        }
        for (int i = 0; i < number; i++) {
            data[i].display();
        }
    }
}
class student {
    int rollno;
    String name;
    public void dataInsert() {
        Scanner sc1 = new Scanner(System.in);
        System.out.println("Enter Roll No: ");
        rollno = sc1.nextInt();
        System.out.println("Enter Name: ");
```

```
        name = sc1.next();
    }
    public void display() {
        System.out.println("Roll no is : " + rollno + "\tName
is : " + name);
    }
}
```

Output: -

```
Enter Number of Students: 3
Enter Roll No:
1
Enter Name:
Ankit
Enter Roll No:
2
Enter Name:
Rahul
Enter Roll No:
3
Enter Name:
Ashish
Roll no is : 1  Name is : Ankit
Roll no is : 2  Name is : Rahul
Roll no is : 3  Name is : Ashish
```

PRACTICAL – 3.1

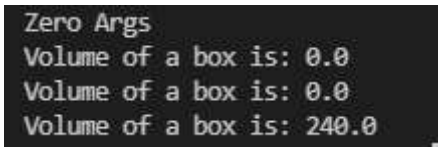
Aim: - Declare a class Box. Overload Box constructors with zero argument, one argument and three argument to initialize the members of the class. Declare a method to find volume of the box.

Code: -

```
public class box_volume {
    public static void main(String[] args) {
        box b = new box();
        box b1 = new box(10);
        box b2 = new box(10,4,6);
        b.volume();
        b1.volume();
        b2.volume();
    }
}
class box{
    double l, b, h, a;
    box() {
        System.out.println("Zero Args");
    }
    box(double l){
        this.l = l;
    }
    box(double b, double h, double a){
        this.a = a;
        this.b = b;
        this.h = h;
    }
    void volume() {
```

```
        double v = a*b*h;
        System.out.println("Volume of a box is: " + v);
    }
}
```

Output: -



PRACTICAL – 4

Aim: - Write a program to demonstrate garbage collection using System.gc() or Runtime.gc().

Code: -

```
public class TestGarbage1{
    public void finalize(){
        System.out.println("object is garbage collected");
    }
    public static void main(String args[]){
     TestGarbage1 s1=new TestGarbage1();
     TestGarbage1 s2=new TestGarbage1();
     s1=null;
     s2=null;
     System.gc();
    }
}
```

Output: -

```
object is garbage collected
object is garbage collected
```

PRACTICAL – 4.1

Aim: - Write a program to show the use of finalize method for garbage collection.

Code: -

```
public class TestGarbage1{
    public void finalize(){
        System.out.println("object is garbage collected");
    }
    public static void main(String args[]){
     TestGarbage1 s1=new TestGarbage1();
     TestGarbage1 s2=new TestGarbage1();
     s1=null;
     s2=null;
     System.gc();
    }
}
```

Output: -

```
object is garbage collected
object is garbage collected
```

PRACTICAL – 5

Aim: - Write a program to demonstrate static constants and final constants.

Code: -

```
public class practical_5 {
    static int subjectCode = 203105334 ;
    final static String subject = "OOPJ";
    public static void main(String[] args)
    {
        System.out.println(subject + "(" + subjectCode + ")");
    }
}
```

Output: -

```
OOPJ(203105334)
```

PRACTICAL – 5.1

Aim: - Write a program to create a class named as Bike which consist one final method called as run(),Declare a subclass Bike & demonstrate the use of final method.

Code: -

```
class Bike {
    final void run(){
        System.out.println("Bike is Running.");
    }
}
public class Bikes extends Bike{
    public static void main(String[] args) {
        Bikes b = new Bikes();
        b.run();
    }
}
```

Output: -

```
Bike is Running.
```

PRACTICAL – 6

Aim: - Write a program to explain static polymorphism in java.

Code: -

```
class Addition{
    void Add(int a, int b){
        int c = a+b;
        System.out.println(c);
    }
    void Add(int a, int b, int c){
        int d = a+b+c;
        System.out.println(d);
    }
}
public class static_poly {
    public static void main(String[] args) {
        Addition add = new Addition();
        add.Add(5, 6);
        add.Add(5, 6, 7);
    }
}
```

Output: -

```
11
18
```

PRACTICAL – 6.1

Aim: - Write a program to find volume of Box using concept of method overloading.

Code: -

```
class Volume{
    void volume(int l, int b, int h){
        int v = l*b*h;
        System.out.println("Volume of a box is: " + v);
    }
    void volume(int l, double b, int h){
        double v = l*b*h;
        System.out.println("Volume of a box is: " + v);
    }
    void volume(double l, int b, double h){
        double v = l*b*h;
        System.out.println("Volume of a box is: " + v);
    }
}
public class Box {
    public static void main(String[] args) {
        Volume vol = new Volume();
        vol.volume(5, 6, 7);
        vol.volume(5, 6.5, 7);
        vol.volume(6.5, 5, 7.5);
    }
}
```

Output: -

```
Volume of a box is: 210
Volume of a box is: 227.5
Volume of a box is: 243.75
```

PRACTICAL – 7

Aim: - Write a program to find the factorial of a number using interface.

Code: -

```
interface factorial1{
    void fact(int a);
}
class factorial2 implements factorial1{
    public void fact(int a) {
        int n = 1;
        for (int i = 1; i <= a; i++) {
            n = n * i;
        }
        System.out.println("The factorial of " + a + " is " +
n);
    }
}
public class fact_interface {
    public static void main(String[] args) {
        factorial2 f = new factorial2();
        f.fact(5);
    }
}
```

Output: -

```
The factorial of 5 is 120
```

PRACTICAL – 7.1

Aim: - Write a program to implement multiple inheritance in java using interface.

Code: -

```
interface Print1{
    void print();
}
interface Print2{
    void print();
}
class Show implements Print1,Print2{
    public void print() {
        System.out.println("Multiple Inheritance");
    }
}
public class inter_mul_inher {
    public static void main(String[] args) {
        Show s = new Show();
        s.print();
    }
}
```

Output: -

```
Multiple Inheritance
```

PRACTICAL – 7.2

Aim: - Create a package called Mathsoperation1, which must contain classes to perform addition, subtraction, create another package called Mathsoperation2, which must contain classes to perform multiplication and division operation. Create a main class and import the Mathsoperation1, Mathsoperation1 package in it to perform all the operations on the input numbers provided by the user. Finally, display the result of each operation on the console.

Code: -

MathsOperation1.java

```
package MathsOperation;
public class MathsOperation1 {
    public void add(int a, int b) {
        int c = a+b;
        System.out.println("Addition of " + a + " and " + b +
" is: " + c);
    }
    public void sub(int a, int b) {
        int c = a-b;
        System.out.println("Subtraction of " + a + " and " +
b + " is: " + c);
    }
}
```

MathsOperation2.java

```
package MathsOperation;
public class MathsOperation2 {
    public void mul(int a, int b) {
        int c = a*b;
        System.out.println("Multiplication of " + a + " and "
+ b + " is: " + c);
    }
```

```
    public void div(int a, int b) {
        int c = a/b;
        System.out.println("Division of " + a + " and " + b +
" is: " + c);
    }
}
```

Main.java

```
package MathsOperation;
import java.util.*;
public class Main {
    public static void main(String[] args) {
        MathsOperation1 add_sub = new MathsOperation1();
        MathsOperation2 mul_div = new MathsOperation2();
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter two number to perform
Addition, Subtraction, Multiplication and Division");
        int a = sc.nextInt();
        int b = sc.nextInt();
        add_sub.add(a, b);
        add_sub.sub(a, b);
        mul_div.mul(a, b);
        mul_div.div(a, b);
    }
}
```

Output: -

```
Enter two number to perform Addition, Subtraction, Multiplication and Division
10
5
Addition of 10 and 5 is: 15
Subtraction of 10 and 5 is: 5
Multiplication of 10 and 5 is: 50
Division of 10 and 5 is: 2
```

PRACTICAL – 8

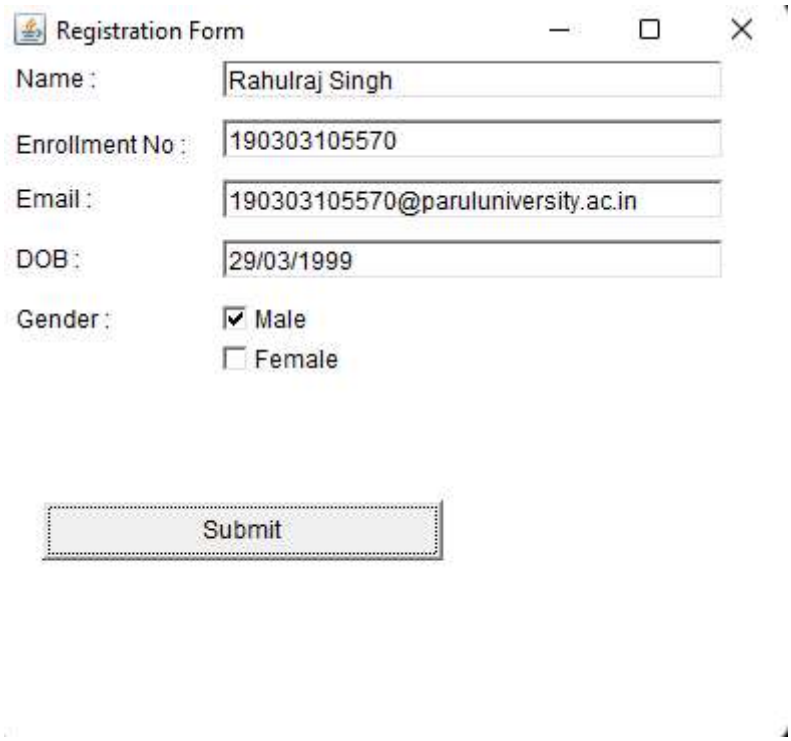
Aim: - Write a program to design student registration form using AWT components.

Code: -

```
import java.awt.*;
class Registration extends Frame{
    Registration() {
        Label name   = new Label("Name : ");
        Label enrollno = new Label("Enrollment No : ");
        Label email  = new Label("Email : ");
        Label gender  = new Label("Gender : ");
        Label dob  = new Label("DOB : ");
        CheckboxGroup cgender = new CheckboxGroup();
        Checkbox male = new Checkbox("Male", false);
        Checkbox female = new Checkbox("Female", false);
        TextField tfname = new TextField();
        TextField tfenrollno = new TextField();
        TextField tfemail = new TextField();
        TextField tfdob = new TextField();
        Button submit = new Button("Submit");
        add(name);
        add(enrollno);
        add(email);
        add(gender);
        add(male);
        add(female);
        add(tfname);
        add(tfenrollno);
        add(tfemail);
```

```
        add(tfdob);
        add(submit);
        add(dob);
        name.setBounds(15, 30, 100, 20);
        tfname.setBounds(120, 30, 250, 20);
        enrollno.setBounds(15, 60, 100, 25);
        tfenrollno.setBounds(120, 60, 250, 20);
        email.setBounds(15, 90, 100, 20);
        tfemail.setBounds(120, 90, 250, 20);
        dob.setBounds(15, 120, 100, 20);
        tfdob.setBounds(120, 120, 250, 20);
        gender.setBounds(15, 150, 100, 20);
        male.setBounds(120, 150, 250, 20);
        female.setBounds(120, 170, 250, 20);
        submit.setBounds(30, 250, 200, 30);
        setTitle("Registration Form");
        setSize(460,390);
        setLayout(null);
        setVisible(true);
    }
    public static void main(String[] args) {
        new Registration();
    }
}
```

Output: -


Registration Form
Name :
Rahulraj Singh
Enrollment No :
190303105570
Email :
190303105570@paruluniversity.ac.in
DOB :
29/03/1999
Gender :
Male
Female
Submit

PRACTICAL – 8.1

Aim: - Write a Java Program for Calculator Operations Using AWT Controls & appropriate layout manager.

Code: -

```
import java.awt.*;
import java.awt.event.*;
class MyCalc extends WindowAdapter implements ActionListener{
  Frame f;
Label l1;
Button b1,b2,b3,b4,b5,b6,b7,b8,b9,b0;
Button badd,bsub,bmult,bdiv,bmod,bcalc,bclr,bpts,bneg,bback;
double xd;
double num1,num2,check;

MyCalc(){
    f= new Frame("MY CALCULATOR");
    l1=new Label();
    l1.setBackground(Color.LIGHT_GRAY);
    l1.setBounds(50,50,260,60);
    b1=new Button("1");
    b1.setBounds(50,340,50,50);
    b2=new Button("2");
    b2.setBounds(120,340,50,50);
    b3=new Button("3");
    b3.setBounds(190,340,50,50);
    b4=new Button("4");
    b4.setBounds(50,270,50,50);
    b5=new Button("5");
    b5.setBounds(120,270,50,50);
```

```
b6=new Button("6");
b6.setBounds(190,270,50,50);
b7=new Button("7");
b7.setBounds(50,200,50,50);
b8=new Button("8");
b8.setBounds(120,200,50,50);
b9=new Button("9");
b9.setBounds(190,200,50,50);
b0=new Button("0");
b0.setBounds(120,410,50,50);
bneg=new Button("+/-");
bneg.setBounds(50,410,50,50);
bpts=new Button(".");
bpts.setBounds(190,410,50,50);
bback=new Button("back");
bback.setBounds(120,130,50,50);

badd=new Button("+");
badd.setBounds(260,340,50,50);
bsub=new Button("-");
bsub.setBounds(260,270,50,50);
bmult=new Button("*");
bmult.setBounds(260,200,50,50);
bdiv=new Button("/");
bdiv.setBounds(260,130,50,50);
bmod=new Button("%");
bmod.setBounds(190,130,50,50);
bcalc=new Button("=");
bcalc.setBounds(245,410,65,50);
```

```
bclr=new Button("CE");
bclr.setBounds(50,130,65,50);

b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
b5.addActionListener(this);
b6.addActionListener(this);
b7.addActionListener(this);
b8.addActionListener(this);
b9.addActionListener(this);
b0.addActionListener(this);

bpts.addActionListener(this);
bneg.addActionListener(this);
bback.addActionListener(this);

badd.addActionListener(this);
bsub.addActionListener(this);
bmult.addActionListener(this);
bdiv.addActionListener(this);
bmod.addActionListener(this);
bcalc.addActionListener(this);
bclr.addActionListener(this);

f.addWindowListener(this);
//ADDING TO FRAME
f.add(l1);
```

```
    f.add(b1); f.add(b2); f.add(b3); f.add(b4); f.add(b5);f.ad
d(b6); f.add(b7); f.add(b8);f.add(b9);f.add(b0);


    f.add(badd); f.add(bsub); f.add(bmod); f.add(bmult); f.add
(bdiv); f.add(bmod);f.add(bcalc);


    f.add(bclr); f.add(bpts);f.add(bneg); f.add(bback);


    f.setSize(360,500);
    f.setLayout(null);
    f.setVisible(true);
    }
                        //FOR CLOSING THE WINDOW
    public void windowClosing(WindowEvent e) {
    f.dispose();
    }


    public void actionPerformed(ActionEvent e){
    String z,zt;
                              //NUMBER BUTTON
    if(e.getSource()==b1){
    zt=l1.getText();
    z=zt+"1";
    l1.setText(z);
    }
    if(e.getSource()==b2){
    zt=l1.getText();
    z=zt+"2";
    l1.setText(z);
```

```
}
if(e.getSource()==b3){
zt=l1.getText();
z=zt+"3";
l1.setText(z);
}
if(e.getSource()==b4){
zt=l1.getText();
z=zt+"4";
l1.setText(z);
}
if(e.getSource()==b5){
zt=l1.getText();
z=zt+"5";
l1.setText(z);
}
if(e.getSource()==b6){
zt=l1.getText();
z=zt+"6";
l1.setText(z);
}
if(e.getSource()==b7){
zt=l1.getText();
z=zt+"7";
l1.setText(z);
}
if(e.getSource()==b8){
zt=l1.getText();
z=zt+"8";
```

```
l1.setText(z);
}
if(e.getSource()==b9){
zt=l1.getText();
z=zt+"9";
l1.setText(z);
}
if(e.getSource()==b0){
zt=l1.getText();
z=zt+"0";
l1.setText(z);
}

if(e.getSource()==bpts){  //ADD DECIMAL PTS
zt=l1.getText();
z=zt+".";
l1.setText(z);
}
if(e.getSource()==bneg){ //FOR NEGATIVE
zt=l1.getText();
z="-"+zt;
l1.setText(z);
}

if(e.getSource()==bback){  // FOR  BACKSPACE
zt=l1.getText();
try{
    z=zt.substring(0, zt.length()-1);
    }catch(StringIndexOutOfBoundsException f){return;}
```

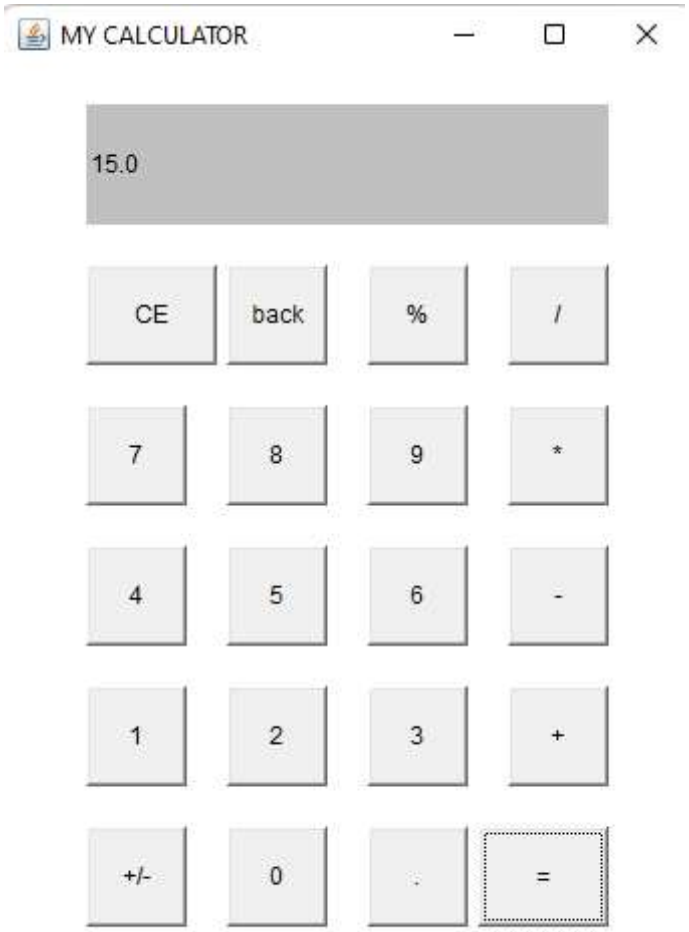
```
    l1.setText(z);
    }
                    //AIRTHMETIC BUTTON
    if(e.getSource()==badd){                          //FOR ADDITIO
N
    try{
        num1=Double.parseDouble(l1.getText());
        }catch(NumberFormatException f){
        l1.setText("Invalid Format");
        return;
        }
    z="";
    l1.setText(z);
    check=1;
    }
    if(e.getSource()==bsub){                        //FOR SUBTRACT
ION
    try{
        num1=Double.parseDouble(l1.getText());
        }catch(NumberFormatException f){
        l1.setText("Invalid Format");
        return;
        }
    z="";
    l1.setText(z);
    check=2;
    }
    if(e.getSource()==bmult){                       //FOR MULTIPLI
CATION
    try{
```

```
        num1=Double.parseDouble(l1.getText());
        }catch(NumberFormatException f){
        l1.setText("Invalid Format");
        return;
        }
    z="";
    l1.setText(z);
    check=3;
    }
    if(e.getSource()==bdiv){                    //FOR DIVISION
    try{
        num1=Double.parseDouble(l1.getText());
        }catch(NumberFormatException f){
        l1.setText("Invalid Format");
        return;
        }
    z="";
    l1.setText(z);
    check=4;
    }
    if(e.getSource()==bmod){                    //FOR MOD/REMAIN
DER
    try{
        num1=Double.parseDouble(l1.getText());
        }catch(NumberFormatException f){
        l1.setText("Invalid Format");
        return;
        }
    z="";
```

```
l1.setText(z);
check=5;
}
                        //RESULT BUTTON
if(e.getSource()==bcalc){
try{
    num2=Double.parseDouble(l1.getText());
    }catch(Exception f){
    l1.setText("ENTER NUMBER FIRST ");
    return;
    }
if(check==1)
    xd =num1+num2;
if(check==2)
    xd =num1-num2;
if(check==3)
    xd =num1*num2;
if(check==4)
    xd =num1/num2;
if(check==5)
    xd =num1%num2;
l1.setText(String.valueOf(xd));
} //FOR CLEARING THE LABEL and Memory
if(e.getSource()==bclr){
num1=0;
num2=0;
check=0;
xd=0;
z="";
```

```
    l1.setText(z);
    }

    }
    //MAIN METHOD where objects of MyCalc is instantaiated
    public static void main(String args[]){
        new MyCalc();
    }
}
```

Output: -



PRACTICAL – 9

Aim: - Write a program to demonstrate array index out of bounds exception.

Code: -

```
public class fact {
    public static void main(String[] a) {
        int number;
        number = Integer.parseInt(a[0]);
        int n = 1;
        for (int i = 1; i <= number; i++) {
            n = n * i;
        }
        System.out.println("The factorial of " + number + " is
 " + n);
    }
}
```

Output: -

```
PS D:\5th Semester\OOPJ\Lab> javac fact.java
PS D:\5th Semester\OOPJ\Lab> java fact
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 0 out of bounds for length 0
        at fact.main(fact.java:5)
```

PRACTICAL – 9.1

Aim: - Create an interface Account with two methods deposit and withdraw. Create class Savings Account which implements the interface. Write a custom Exception handler for Savings Account to handle the scenarios when withdrawn amount is larger than the balance in the account.

Code: -

```
interface Account {
    void deposit(int amount);
    void withdraw(int amount) throws InsufficientFundsExceptio
n;
}
class SavingAccount implements Account {
    int Balance = 3000;
    public void deposit(int amount) {
        Balance = Balance + amount;
        System.out.println("Balance after deposit is : " + Bal
ance);
    }
    public void withdraw(int amount) throws InsufficientFundsE
xception {
        if (amount > Balance) {
            throw new InsufficientFundsException("Insufficient
 Funds");
        } else {
            Balance = Balance - amount;
            System.out.println("Balance after deposit is : " +
 Balance);
        }
    }
}
class InsufficientFundsException extends Exception {
```

```
    public InsufficientFundsException(String msg) {
        super(msg);
    }
}
public class practical_9_1 {
    public static void main(String[] args) throws Insufficient
FundsException {
        SavingAccount sA = new SavingAccount();
        sA.deposit(5000);
        sA.withdraw(3000);
        sA.withdraw(6000);
    }
}
```

Output: -

```
Balance after deposit is : 8000
Balance after deposit is : 5000
Exception in thread "main" InsufficientFundsException: Insufficient Funds
        at SavingAccount.withdraw(practical_9_1.java:13)
        at practical_9_1.main(practical_9_1.java:30)
```

PRACTICAL – 10

Aim: - Write a program to demonstrate class object locking using method level Synchronization.

Code: -

```
public class practical_10 implements Runnable{
    public void run(){
        Lock();
    }
    public void Lock() {
        System.out.println(Thread.currentThread().getName());
        synchronized(this) {
            System.out.println("in block " + Thread.currentThr
ead().getName());
            System.out.println("in block " + Thread.currentThr
ead().getName() + " end");
        }
    }
    public static void main(String[] args) {
        practical_10 p = new practical_10();
        Thread t1 = new Thread(p);
        Thread t2 = new Thread(p);
        practical_10 p1 = new practical_10();
        Thread t3 = new Thread(p1);
        t1.setName("t1");
        t2.setName("t2");
        t3.setName("t3");
        t1.start();
        t2.start();
        t3.start();
```

```
        }
}
```

Output: -

```
t1
t2
t3
in block t1
in block t3
in block t1 end
in block t3 end
in block t2
in block t2 end
```

PRACTICAL – 10.1

Aim: - Write a program that executes two threads. One thread will print the even numbers and another thread will print odd numbers from 1 to 50.

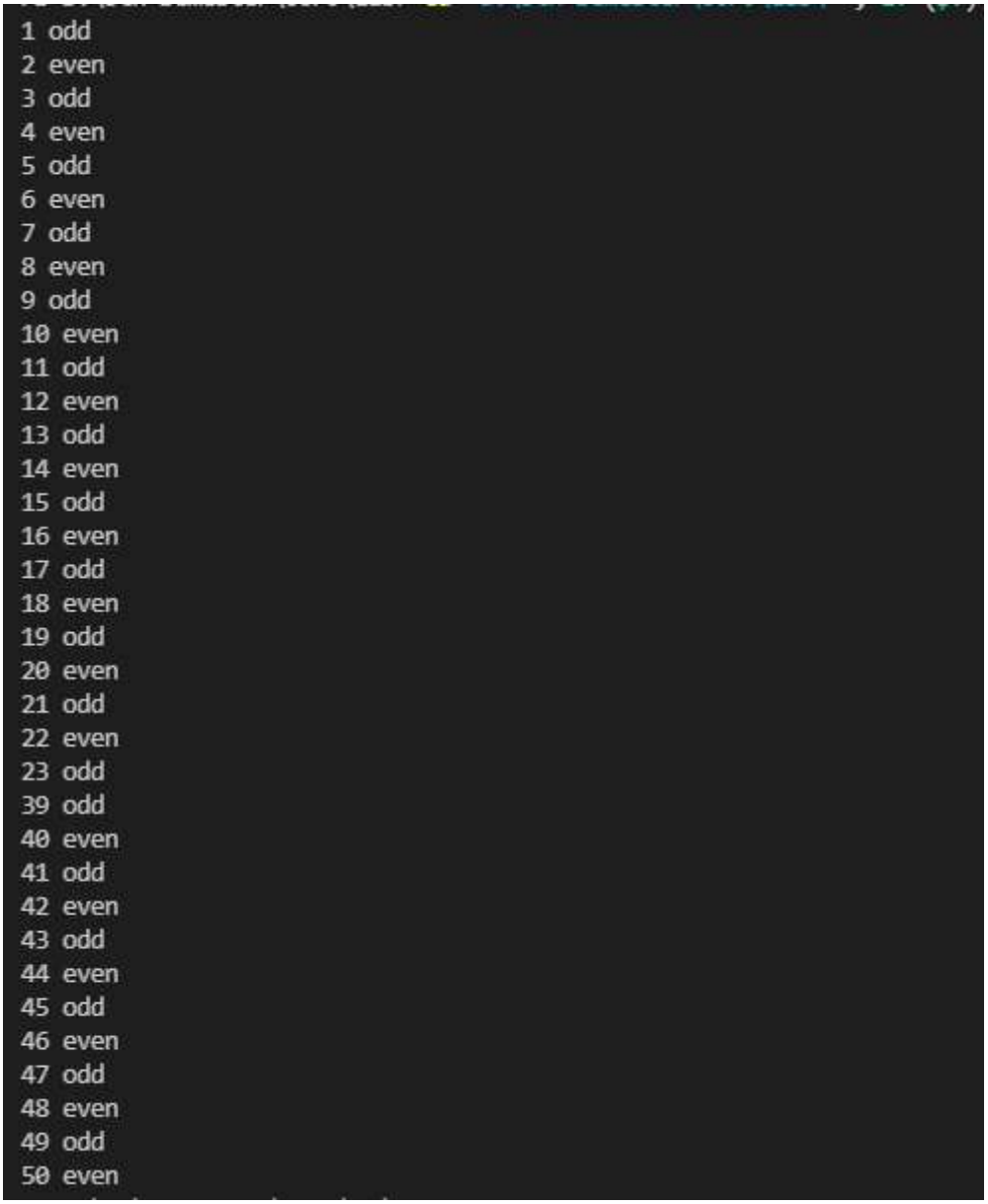
Code: -

```
public class practical_10_1 {
    int counter = 1;
    static int N;

    public void odd(){
        synchronized (this){
            while (counter < N){
                while (counter % 2 == 0) {
                    try {
                        wait();
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }
                System.out.println(counter + " " + "odd");
                counter++;
                notify();
            }
        }
    }
    public void even(){
        synchronized (this){
            while (counter < N){
                while (counter % 2 == 1) {
                    try {
```

```
                    wait();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
            System.out.println(counter + " " + "even");
            counter++;
            notify();
        }
    }
}
public static void main(String[] args) {

    practical_10_1 oE = new practical_10_1();
    N = 50;
    Thread t1 = new Thread(new Runnable() {
        @Override
        public void run() {
            oE.odd();
        }
    });
    Thread t2 = new Thread(new Runnable() {
        @Override
        public void run() {
            oE.even();
        }
    });
    t1.start();
    t2.start();
```

```
        }
}
```

Output: -


1 odd
2 even
3 odd
4 even
5 odd
6 even
7 odd
8 even
9 odd
10 even
11 odd
12 even
13 odd
14 even
15 odd
16 even
17 odd
18 even
19 odd
20 even
21 odd
22 even
23 odd
39 odd
40 even
41 odd
42 even
43 odd
44 even
45 odd
46 even
47 odd
48 even
49 odd
50 even