PRACTICAL – 8

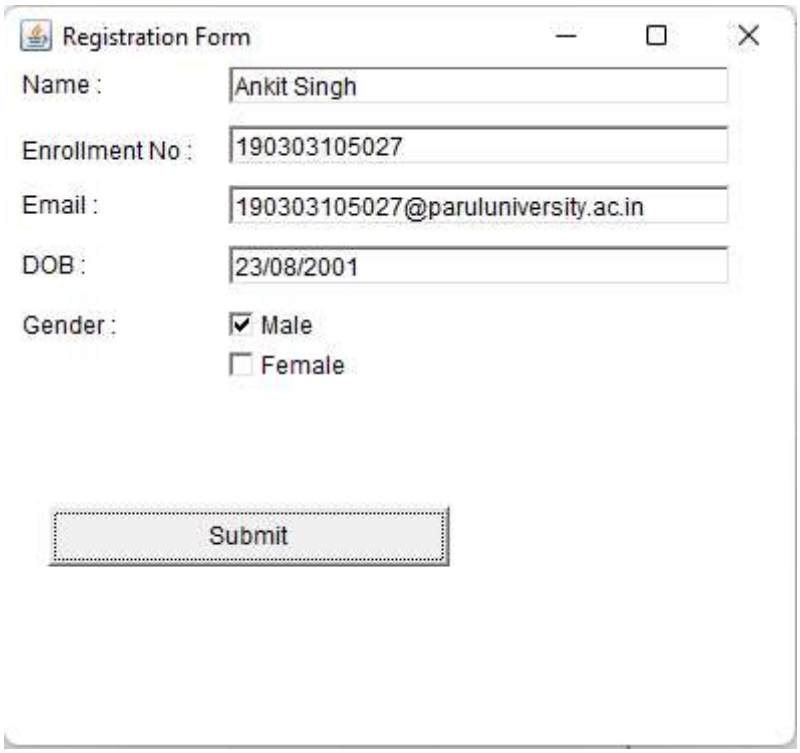
Aim: - Write a program to design student registration form using AWT components.

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 :
Ankit Singh
Enrollment No :
190303105027
Email :
190303105027@paruluniversity.ac.in
DOB :
23/08/2001
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;}
```

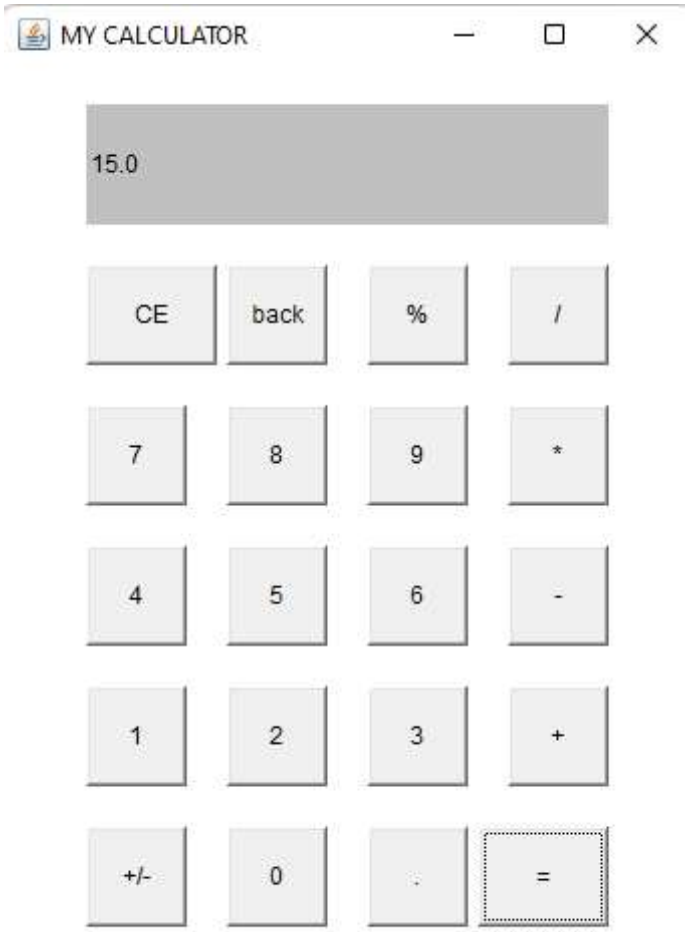
```
    l1.setText(z);
    }
                    //AIRTHMETIC BUTTON
    if(e.getSource()==badd){                          //FOR ADDITIO
N
    try{
        num1=Double.parseDouble(l1.getText());
        }catch(NumberFormatException f){
        l1.setText("Invalid Format");
        return;
        }
    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.

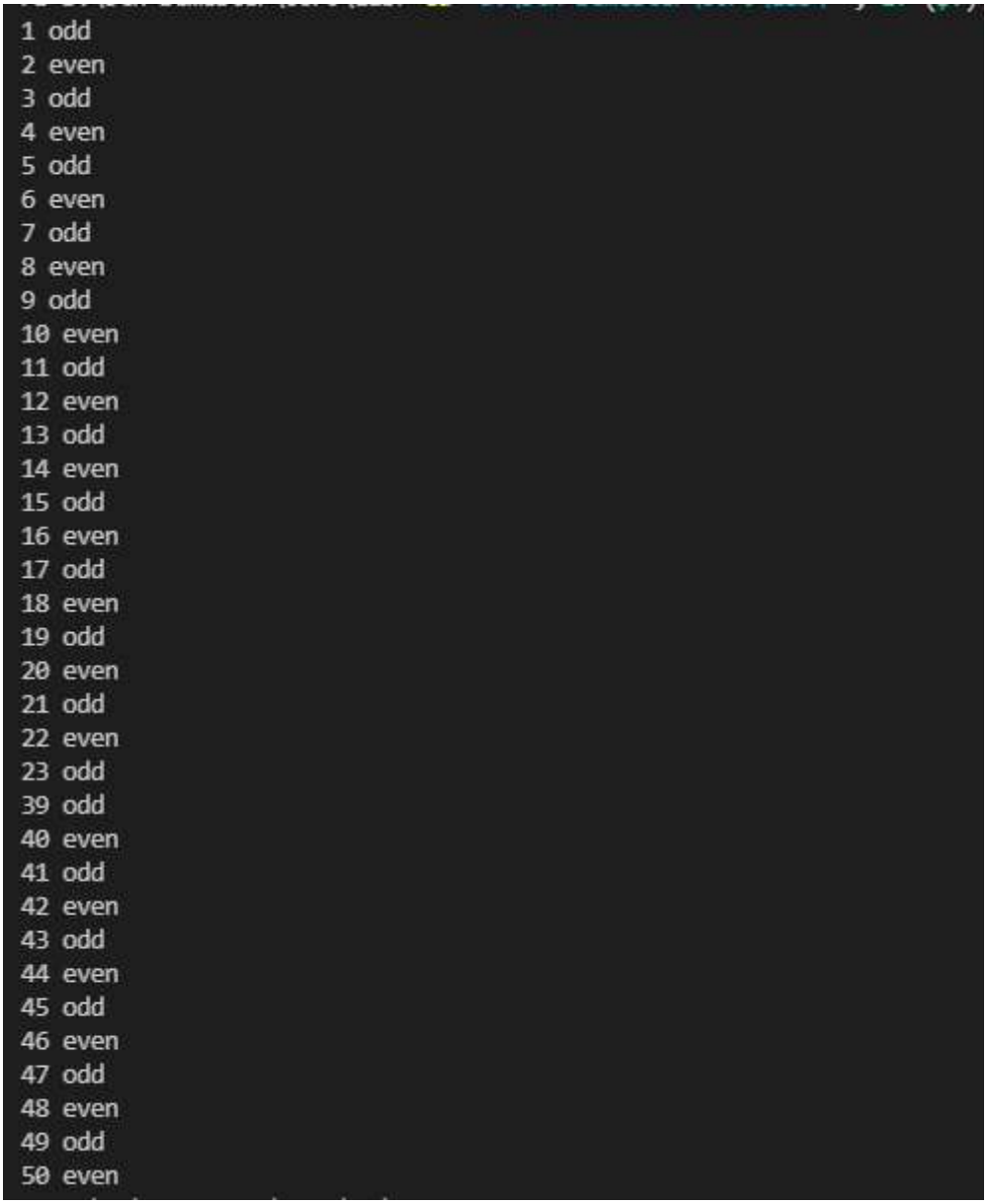
Code: -

```
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