Experiment – 01

Advance Java Lab (5CS4-24)

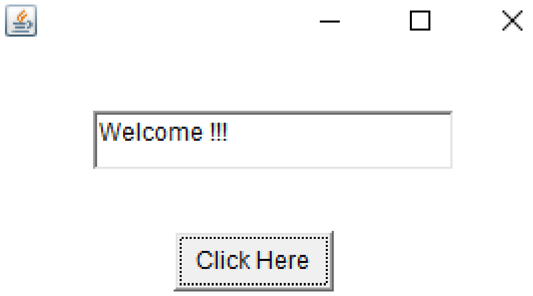
Class – B.Tech III Year, V Sem.

Objective: (a) Write a program in java for event handling using JavaAWT.

Code:

```
import java.awt.*;
import java.awt.event.*;
 public class AEvent extends Frame implements ActionListener{
     TextField tf;
     AEvent(){
         //create components
         tf=new TextField();
         tf.setBounds(60,60,180,30);
         Button b = new Button("Click Here");
         b.setBounds(100, 120,80,30);
         //register listener
         b.addActionListener(this); //passing current instance
         //add components and set size, layout and visibility
         add(b);
         add(tf);
         setSize(300,300);
         setLayout(null);
         setVisible(true);
     }
     public void actionPerformed(ActionEvent e){
         tf.setText("Welcome !!!");
     }
     public static void main(String args[]){
    new AEvent();
    }

}
```

Output:



Welcome !!!

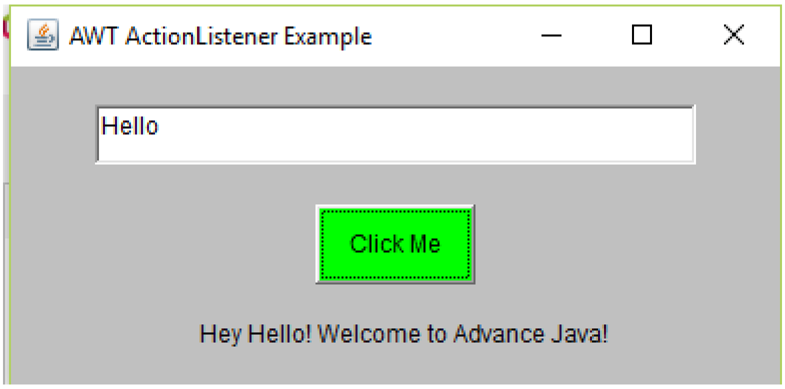
Event Handling in Java

(b) Objective: Write a program in java for implementing Action Listener in Java (on ButtonClick).

```
import java.awt.*;

import java.awt.event.*;


// Driver Class

public class ActionListenerExample {

      // main function

    public static void main(String[] args){


        // Create a frame

        Frame f = new Frame("AWT ActionListener Example");


          // Set the size

        f.setSize(400, 200);


          // Set the layout
        f.setLayout(null);


          // Make the frame visible
        f.setVisible(true);


          // Set the background color of the frame
        f.setBackground(Color.LIGHT_GRAY);


        // Create a button
        Button b = new Button("Click Me");


          // Set the positions
        b.setBounds(160, 100, 80, 40);


          // Add button to the frame
        f.add(b);


          // Set the background color of the button
        b.setBackground(Color.GREEN);


        // Create a text field
        final TextField tf = new TextField();


          // Set the positions
        tf.setBounds(50, 50, 300, 30);


          // Add text field to the frame
        f.add(tf);


        // Create a label
        Label lb = new Label();
```

```
         // Set the positions
        lb.setBounds(100, 150, 300, 30);

         // Add label to the frame
        f.add(lb);

        // Add an action listener to the button
        b.addActionListener(new ActionListener() {

            // Override the actionPerformed() method
            public void actionPerformed(ActionEvent e){

                // Update the text of the label
                lb.setText("Hey " + tf.getText() + "! "
                           + "Welcome to Advance Java!");
            }

        });
    }
}
```

Output;



Hey Hello! Welcome to Advance Java!

Experiment – 02

Advance Java Lab (5CS4-24)

Class – B.Tech III Year, V Sem.

Objective: Write a program in java to implement calculator using swing technology.

Code:

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

class Calc implements ActionListener
{
                JFrame f;
                JTextField t;
                JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,bdiv,bmul,bsub,badd,bdec,beq,bdel,bclr;

                static double a=0,b=0,result=0;
                static int operator=0;

                Calc()
                {
                    f=new JFrame("Calculator");
                    t=new JTextField();
                    b1=new JButton("1");
                    b2=new JButton("2");
                    b3=new JButton("3");
                    b4=new JButton("4");
                    b5=new JButton("5");
                    b6=new JButton("6");
                    b7=new JButton("7");
                    b8=new JButton("8");
```

```
b9=new JButton("9");
b0=new JButton("0");
bdiv=new JButton("/");
bmul=new JButton("*");
bsub=new JButton("-");
badd=new JButton("+");
bdec=new JButton(".");
beq=new JButton("=");
bdel=new JButton("Delete");
bclr=new JButton("Clear");

t.setBounds(30,40,280,30);
b7.setBounds(40,100,50,40);
b8.setBounds(110,100,50,40);
b9.setBounds(180,100,50,40);
bdiv.setBounds(250,100,50,40);

b4.setBounds(40,170,50,40);
b5.setBounds(110,170,50,40);
b6.setBounds(180,170,50,40);
bmul.setBounds(250,170,50,40);

b1.setBounds(40,240,50,40);
b2.setBounds(110,240,50,40);
b3.setBounds(180,240,50,40);
bsub.setBounds(250,240,50,40);

bdec.setBounds(40,310,50,40);
b0.setBounds(110,310,50,40);
beq.setBounds(180,310,50,40);
badd.setBounds(250,310,50,40);
```

```
bdel.setBounds(60,380,100,40);

bclr.setBounds(180,380,100,40);


f.add(t);

f.add(b7);

f.add(b8);

f.add(b9);

f.add(bdiv);

f.add(b4);

f.add(b5);

f.add(b6);

f.add(bmul);

f.add(b1);

f.add(b2);

f.add(b3);

f.add(bsub);

f.add(bdec);

f.add(b0);

f.add(beq);

f.add(badd);

f.add(bdel);

f.add(bclr);


f.setLayout(null);

f.setVisible(true);

f.setSize(350,500);

f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

f.setResizable(false);


b1.addActionListener(this);

b2.addActionListener(this);

b3.addActionListener(this);
```

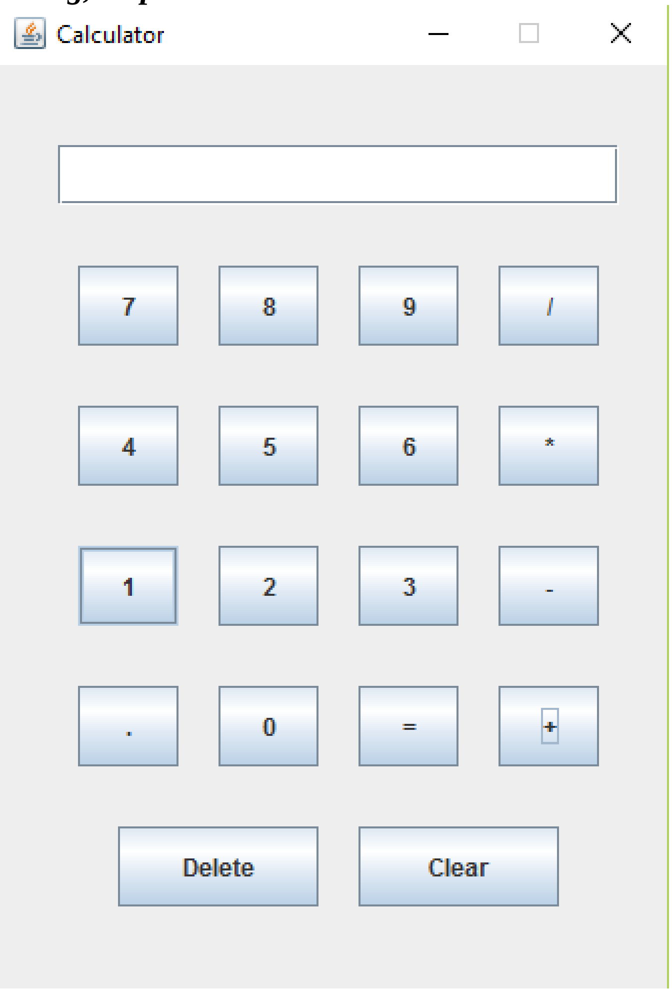
```
    b4.addActionListener(this);
    b5.addActionListener(this);
    b6.addActionListener(this);
    b7.addActionListener(this);
    b8.addActionListener(this);
    b9.addActionListener(this);
    b0.addActionListener(this);
    badd.addActionListener(this);
    bdiv.addActionListener(this);
    bmul.addActionListener(this);
    bsub.addActionListener(this);
    bdec.addActionListener(this);
    beq.addActionListener(this);
    bdel.addActionListener(this);
    bclr.addActionListener(this);
}

public void actionPerformed(ActionEvent e)
{
    if(e.getSource()==b1)
        t.setText(t.getText().concat("1"));

    if(e.getSource()==b2)
        t.setText(t.getText().concat("2"));

    if(e.getSource()==b3)
        t.setText(t.getText().concat("3"));

    if(e.getSource()==b4)
        t.setText(t.getText().concat("4"));

    if(e.getSource()==b5)
```

```
            t.setText(t.getText().concat("5"));

        if(e.getSource()==b6)
            t.setText(t.getText().concat("6"));

        if(e.getSource()==b7)
            t.setText(t.getText().concat("7"));

        if(e.getSource()==b8)
            t.setText(t.getText().concat("8"));

        if(e.getSource()==b9)
            t.setText(t.getText().concat("9"));

        if(e.getSource()==b0)
            t.setText(t.getText().concat("0"));

        if(e.getSource()==bdec)
            t.setText(t.getText().concat("."));

        if(e.getSource()==badd)
        {
            a=Double.parseDouble(t.getText());
            operator=1;
            t.setText("");
        }

        if(e.getSource()==bsub)
        {
            a=Double.parseDouble(t.getText());
            operator=2;
            t.setText("");
```

```
                }

                if(e.getSource()==bmul)
                {
                    a=Double.parseDouble(t.getText());
                    operator=3;
                    t.setText("");
                }

                if(e.getSource()==bdiv)
                {
                    a=Double.parseDouble(t.getText());
                    operator=4;
                    t.setText("");
                }

                if(e.getSource()==beq)
                {
                    b=Double.parseDouble(t.getText());

                    switch(operator)
                    {
                        case 1: result=a+b;
                            break;

                        case 2: result=a-b;
                            break;

                        case 3: result=a*b;
                            break;

                        case 4: result=a/b;
```

```
                                                            break;

                                                    default: result=0;
                                            }

                                            t.setText(""+result);
                                    }

                                    if(e.getSource()==bclr)
                                            t.setText("");

                                    if(e.getSource()==bdel)
                                    {
                                            String s=t.getText();
                                            t.setText("");
                                            for(int i=0;i<s.length()-1;i++)
                                            t.setText(t.getText()+s.charAt(i));
                                    }
                            }

                            public static void main(String...s)
                            {
                                    new Calc();
                            }
}
```

Output:


Calculator
7
8
9
/
4
5
6
*
1
2
3
-
.
0
=
+
Delete
Clear

Experiment – 03

Advance Java Lab (5CS4-24)

Class – B.Tech III Year, V Sem.

Objective:

3.1 - Write a Java program that makes a Connection with database using JDBC and prints metadata of this connection.

Code:

```
import java.sql.*;

class MysqlCon{

public static void main(String args[]){ try{ Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection( "jdbc:mysql://localhost:3306/db","root","root");

//here db is database name, root is username and password

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery("select * from emp");

while(rs.next())

System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3));

con.close();

}

catch(Exception e){ System.out.println(e);}

}

}
```

Objective – 3.2 Include the database Connectivity in the program to insert, update, delete and display of information.

```
package com.devdaily.sqlprocessortests;

import java.sql.*;

public class BasicJDBCDemo
```

```
{

Connection conn;

public static void main(String[] args)

{

new BasicJDBCDemo();

}

public BasicJDBCDemo()

{

try{

Class.forName("com.mysql.jdbc.Driver").newInstance();

String url = "jdbc:mysql://localhost/coffeebreak";

conn = DriverManager.getConnection(url, "username", "password");

doTests();

conn.close();
}
catch (ClassNotFoundException ex) {

System.err.println(ex.getMessage());

} catch (IllegalAccessException ex) {

System.err.println(ex.getMessage());

} catch (InstantiationException ex) {

System.err.println(ex.getMessage());

} catch (SQLException ex){

System.err.println(ex.getMessage());

}

}

private void doTests()
```

```
{
doSelectTest();
doInsertTest();

doSelectTest();

doUpdateTest();

doSelectTest();

doDeleteTest();

doSelectTest();
}
private void doSelectTest()
{
System.out.println("[OUTPUT FROM SELECT]");
String query = "SELECT COF_NAME, PRICE FROM COFFEES";
try
{
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery(query);
while (rs.next())
{
String s = rs.getString("COF_NAME");
float n = rs.getFloat("PRICE");
System.out.println(s + " " + n);
}
}
```

```
catch (SQLException ex)

{

System.err.println(ex.getMessage());

}}
private void doInsertTest()

{

System.out.print("\n[Performing INSERT] ... "); try

{

Statement st = conn.createStatement();

st.executeUpdate("INSERT INTO COFFEES " +

"VALUES ('BREAKFAST BLEND', 200, 7.99, 0, 0)");

}

catch (SQLException ex)

{

System.err.println(ex.getMessage());

}

}

private void doUpdateTest()

{

System.out.print("\n[Performing UPDATE] ... "); try

{

Statement st = conn.createStatement();

st.executeUpdate("UPDATE COFFEES SET PRICE=4.99 WHERE
COF_NAME='BREAKFAST BLEND'");

}

catch (SQLException ex){

System.err.println(ex.getMessage());}
```

```
}
private void doDeleteTest()
{
System.out.print("\n[Performing DELETE] ... "); try
{
Statement st = conn.createStatement();
st.executeUpdate("DELETE FROM COFFEES WHERE COF_NAME='BREAKFAST BLEND'");
}
catch (SQLException ex)
{
System.err.println(ex.getMessage());
}
}
```

Experiment – 04

Advance Java Lab (5CS4-24)

Class – B.Tech III Year, V Sem.

Objective:

Write a java program for two way TCP communication for server and client. It should look like a simple chat application.

Code:

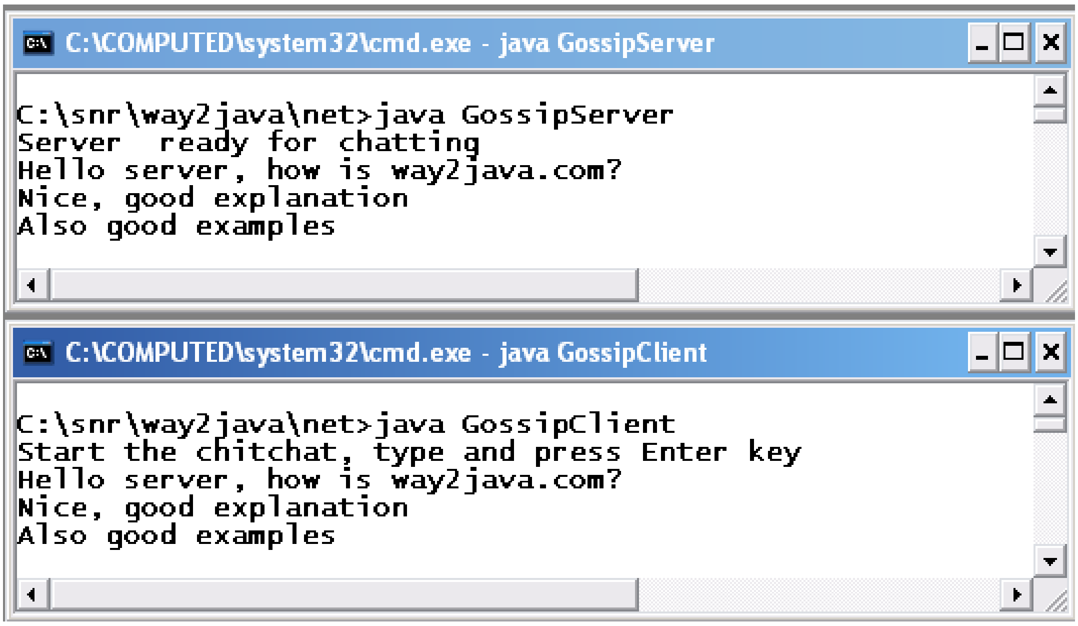
GossipClient.java

```
import java.io.*;

import java.net.*;

import java.util.Scanner;


public class GossipClient {

    private static final String SERVER_ADDRESS = "localhost";

    private static final int SERVER_PORT = 12345;


    public static void main(String[] args) {

        try {

            Socket socket = new Socket(SERVER_ADDRESS, SERVER_PORT);

            System.out.println("Connected to the chat server!");


            // Setting up input and output streams

            PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

            BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));


            // Start a thread to handle incoming messages

            new Thread(() -> {

                try {

                    String serverResponse;

                    while ((serverResponse = in.readLine()) != null) {

                        System.out.println(serverResponse);
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }).start();

            // Read messages from the console and send to the server
            Scanner scanner = new Scanner(System.in);
            String userInput;
            while (true) {
                userInput = scanner.nextLine();
                out.println(userInput);
            }

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

GossipServer.java

```
//Server program to handle multiple
// Clients with socket connections
import java.io.*;
import java.net.*;
import java.util.concurrent.CopyOnWriteArrayList;

public class GossipServer {
    private static final int PORT = 1234;
    private static CopyOnWriteArrayList<ClientHandler> clients = new CopyOnWriteArrayList<>();

    public static void main(String[] args) {
        try {
            ServerSocket serverSocket = new ServerSocket(PORT);
            System.out.println("Server is running and waiting for connections..");

            // Accept incoming connections
            while (true) {
                Socket clientSocket = serverSocket.accept();
                System.out.println("New client connected: " + clientSocket);

                // Create a new client handler for the connected client
                ClientHandler clientHandler = new ClientHandler(clientSocket);
                clients.add(clientHandler);
                new Thread(clientHandler).start();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    // Broadcast a message to all clients except the sender
    public static void broadcast(String message, ClientHandler sender) {
        for (ClientHandler client : clients) {
            if (client != sender) {
                client.sendMessage(message);
            }
        }
    }

    // Internal class to handle client connections
    private static class ClientHandler implements Runnable {
        private Socket clientSocket;
        private PrintWriter out;
        private BufferedReader in;
        private String Username; // Use Username consistently


        // Constructor
        public ClientHandler(Socket socket) {
            this.clientSocket = socket;


            try {
                // Create input and output streams for communication
                out = new PrintWriter(clientSocket.getOutputStream(), true);
                in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
            } catch (IOException e) {
                e.printStackTrace();
            }
        }


        // Run method to handle client communication
        @Override
        public void run() {
            try {
                // Get the username from the client
                Username = getUsername(); // Use Username consistently
                System.out.println("User " + Username + " connected."); // Use Username consistently


                out.println("Welcome to the chat, " + Username + "!"); // Use Username consistently
                out.println("Type Your Message");
                String inputLine;


                // Continue receiving messages from the client
                while ((inputLine = in.readLine()) != null) {
                    System.out.println("[" + Username + "]: " + inputLine); // Use Username consistently
                    // Broadcast the message to all clients
                    broadcast("[" + Username + "]: " + inputLine, this); // Use Username consistently
                }

                // Remove the client handler from the list
                clients.remove(this);

                // Close the input and output streams and the client socket
                in.close();
                out.close();
                clientSocket.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        // Get the username from the client
        private String getUsername() throws IOException {
            out.println("Enter your username:");
            return in.readLine();
        }

        public void sendMessage(String message) {
            out.println(message);
            out.println("Type Your Message");
        }
      }
    }
```

Output:


C:\COMPUTED\system32\cmd.exe - java GossipServer
C:\COMPUTED\system32\cmd.exe - java GossipClient

```
C:\snr\way2java\net>java GossipServer
Server  ready for chatting
Hello server, how is way2java.com?
Nice, good explanation
Also good examples
```

```
C:\snr\way2java\net>java GossipClient
Start the chitchat, type and press Enter key
Hello server, how is way2java.com?
Nice, good explanation
Also good examples
```