ACTIVITY-VI - OOPJ Theory

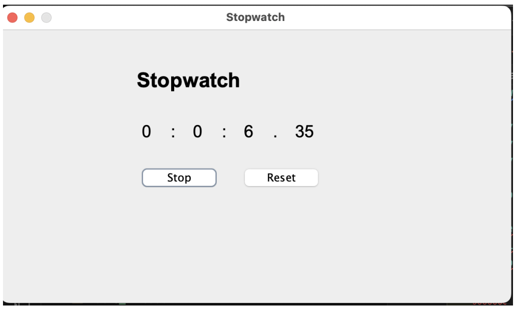
ASSIGNMENT(Group Activity)

💡 Members:
- 2205248 Suket Kamboj
- 2205350 Aditya Sen
- 2205533 Aniruddha Mukherjee
- 2205552 Digvijay Mishra
- 2205568 Mayur Gogoi
- 2205618 Arijit Chowdhury

Stopwatch

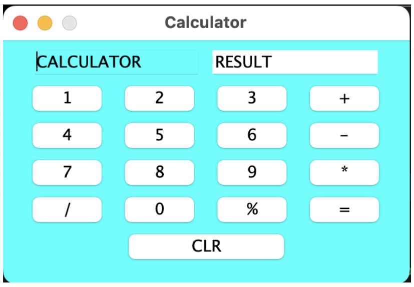
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class Stopwatch {

    JFrame jfm;
    JLabel hrs, min, sec, mili, title, colon1, colon2, colon3;
    JButton startStop, reset;

    Timer timer;
    int hours = 0, minutes = 0, seconds = 0, milliseconds = 0;
    boolean running = false;

    Stopwatch() {

        // Frame
        jfm = new JFrame("Stopwatch");
        jfm.setLayout(null);
        jfm.setSize(600, 350);
        jfm.setResizable(false);
        jfm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Labels
        title = new JLabel("Stopwatch");
        title.setBounds(160, 10, 200, 100);
        title.setFont(new Font("Arial", Font.BOLD, 24));
        jfm.add(title);

        hrs = new JLabel("00");
        hrs.setBounds(160, 100, 40, 40);
        hrs.setFont(new Font("Arial", Font.PLAIN, 20));
        jfm.add(hrs);

        colon1 = new JLabel(":");
        colon1.setBounds(200, 100, 20, 40);
        colon1.setFont(new Font("Arial", Font.PLAIN, 20));
        jfm.add(colon1);

        min = new JLabel("00");
        min.setBounds(220, 100, 40, 40);
        min.setFont(new Font("Arial", Font.PLAIN, 20));
        jfm.add(min);

        colon2 = new JLabel(":");
        colon2.setBounds(260, 100, 20, 40);
        colon2.setFont(new Font("Arial", Font.PLAIN, 20));
        jfm.add(colon2);

        sec = new JLabel("00");
        sec.setBounds(280, 100, 40, 40);
        sec.setFont(new Font("Arial", Font.PLAIN, 20));
        jfm.add(sec);

        colon3 = new JLabel(".");
        colon3.setBounds(320, 100, 20, 40);
        colon3.setFont(new Font("Arial", Font.PLAIN, 20));
        jfm.add(colon3);

        mili = new JLabel("00");
        mili.setBounds(340, 100, 40, 40);
        mili.setFont(new Font("Arial", Font.PLAIN, 20));
        jfm.add(mili);

        // Buttons
        startStop = new JButton("Start");
        startStop.setBounds(160, 160, 100, 30);
        jfm.add(startStop);

        reset = new JButton("Reset");
        reset.setBounds(280, 160, 100, 30);
        jfm.add(reset);

        // Button actions
        startStop.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if (running) {
                    stop();
                } else {
                    start();
                }
            }
        });

        reset.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                reset();
            }
        });

        jfm.setVisible(true);
    }

    private void start() {
        running = true;
        timer = new Timer(10, new ActionListener() { // To count time
            public void actionPerformed(ActionEvent e) {
                milliseconds++;
                if (milliseconds >= 100) {
                    milliseconds = 0;
                    seconds++;
                    if (seconds >= 60) {
                        seconds = 0;
                        minutes++;
                        if (minutes >= 60) {
                            minutes = 0;
                            hours++;
                        }
                    }
                }
                updateLabels();
            }
        });
        timer.start();
        startStop.setText("Stop");
    }

    private void stop() {
        running = false;
```

```
            timer.stop();
            startStop.setText("Start");
        }

        private void reset() {
            running = false;
            timer.stop();
            hours = 0;
            minutes = 0;
            seconds = 0;
            milliseconds = 0;
            updateLabels();
            startStop.setText("Start");
        }

        private void updateLabels() {
            hrs.setText(" " + hours);
            min.setText(" " + minutes);
            sec.setText(" " + seconds);
            mili.setText(" " + milliseconds);
        }

        public static void main(String[] args) {
            new Stopwatch();
        }
    }
```



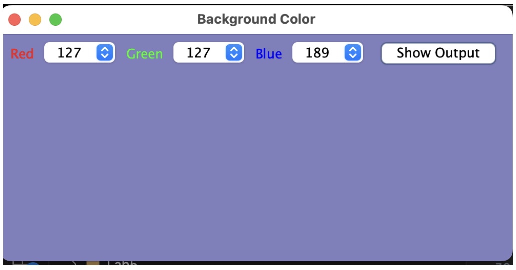
Calculator

```
package swinggg;

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

public class Calculator {
    // Initialize frame, numbers and result
    private JFrame jfrm;
    double num1;
    double num2;
    double result;

    public double get_result(JTextField tf) {
        /*
         * takes num1 and num2 (global variable)
         * extracts which operation is performed
         * returns the result of the operation on num1 num2
         */
        // operations are + - / % *
        String text = tf.getText();
        char operation = ' ';
        if (text.contains("+")) {
            operation = '+';
            result = num1 + num2;
        } else if (text.contains("-")) {
            operation = '-';
            result = num1 - num2;
        } else if (text.contains("*")) {
            operation = '*';
            result = num1 * num2;
        } else if (text.contains("/")) {
            operation = '/';
            result = num1 / num2;
        } else if (text.contains("%")) {
            operation = '%';
            result = num1 % num2;
        }
        return result;
    }

    public double get_second_number(JTextField tf) {
        /*
         * Extracts second number from textfield
         * by looking for the values after the operation character.
         * returns the second number
         */
        String text = tf.getText();
        double secondNumber = 0;
        if (text.contains("+")) {
            secondNumber = Double.parseDouble(text.substring(text.indexOf("+") + 1));
        } else if (text.contains("-")) {
            secondNumber = Double.parseDouble(text.substring(text.indexOf("-") + 1));
        } else if (text.contains("*")) {
            secondNumber = Double.parseDouble(text.substring(text.indexOf("*") + 1));
        } else if (text.contains("/")) {
            secondNumber = Double.parseDouble(text.substring(text.indexOf("/") + 1));
        } else if (text.contains("%")) {
            secondNumber = Double.parseDouble(text.substring(text.indexOf("%") + 1));
        }
        return secondNumber;
    }

    public void set_empty(JTextField tf) {
        /*
         * Sets the Left text field empty when first button is clicked.
         */
        if (tf.getText().equals("CALCULATOR")) {
            tf.setText("");
        }
    }

    JButton sized_button(String cnt) {
        /*
         * creates a button of size 70x25
         * returns the created button
         */
        JButton button = new JButton(cnt);
        button.setPreferredSize(new Dimension(70, 25));
        return button;
    }

    public void add_buttons_to_frame(JFrame jfrm, JTextField tf, JTextField res) {
        result = Double.NEGATIVE_INFINITY;
        /*
         * Adds Button to the Frame
         * Listens for events on buttons
         */
        JButton n1 = sized_button("1");
        jfrm.add(n1);
        JButton n2 = sized_button("2");
        jfrm.add(n2);
        JButton n3 = sized_button("3");
        jfrm.add(n3);
        JButton nplus = sized_button("+");
        jfrm.add(nplus);
        JButton n4 = sized_button("4");
        jfrm.add(n4);
        JButton n5 = sized_button("5");
        jfrm.add(n5);
        JButton n6 = sized_button("6");
        jfrm.add(n6);
        JButton nminus = sized_button("-");
        jfrm.add(nminus);
        JButton n7 = sized_button("7");
        jfrm.add(n7);
        JButton n8 = sized_button("8");
        jfrm.add(n8);
        JButton n9 = sized_button("9");
        jfrm.add(n9);
        JButton nast = sized_button("*");
        jfrm.add(nast);
        JButton ndiv = sized_button("/");
        jfrm.add(ndiv);
        JButton n0 = sized_button("0");
        jfrm.add(n0);
        JButton nmod = sized_button("%");
        jfrm.add(nmod);
        JButton neq = sized_button("=");
        jfrm.add(neq);

        // creates off button
        JButton clr = new JButton("CLR");
        clr.setPreferredSize(new Dimension(140, 25));
        jfrm.add(clr);

        //// Adding Event Listeners
        // listener on 1
        n1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "1");
            }
        });
        // listener on 2
        n2.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "2");
            }
        });
        // listener on 3
        n3.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "3");
            }
        });
        // listener on 4
        n4.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "4");
            }
        });
        // listener on 5
        n5.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "5");
            }
        });
        // listener on 6
        n6.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "6");
            }
        });
        // listener on 7
        n7.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "7");
            }
        });
        // listener on 8
        n8.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "8");
            }
        });
        // listener on 9
        n9.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "9");
            }
        });
        // listner on 0
        n0.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                set_empty(tf);
                tf.setText(tf.getText() + "0");
            }
        });

        // CLR resets the input when CLR button is pressed
        clr.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                tf.setText("CALCULATOR");
                res.setText("RESULT");
                num1 = 0;
                num2 = 0;
                result = Double.NEGATIVE_INFINITY;
            }
        });

        //// Listeners for Operations
        // listener for '+' operation
        nplus.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                if (result == Double.NEGATIVE_INFINITY) {
                    num1 = Double.parseDouble(tf.getText());
                    tf.setText(tf.getText() + " + ");
                } else {
                    tf.setText(Double.toString(num1) + " + ");
                }
            }
        });
        // listerner for '-' operation
        nminus.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                if (result == Double.NEGATIVE_INFINITY) {
                    num1 = Double.parseDouble(tf.getText());
                    tf.setText(tf.getText() + " - ");
                } else {
                    tf.setText(Double.toString(num1) + " - ");
                }
            }
        });
        // listener for '*' operation
        nast.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                if (result == Double.NEGATIVE_INFINITY) {
                    num1 = Double.parseDouble(tf.getText());
                    tf.setText(tf.getText() + " * ");
                } else {
                    tf.setText(Double.toString(num1) + " * ");
                }
            }
        });
        // listner for '%' operation
        nmod.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                if (result == Double.NEGATIVE_INFINITY) {
                    num1 = Double.parseDouble(tf.getText());
                    tf.setText(tf.getText() + " % ");
                } else {
                    tf.setText(Double.toString(num1) + " % ");
                }
            }
        });
        // listener for '/' operation
        ndiv.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                if (result == Double.NEGATIVE_INFINITY) {
                    num1 = Double.parseDouble(tf.getText());
                    tf.setText(tf.getText() + " / ");
                } else {
                    tf.setText(Double.toString(num1) + " / ");
                }
            }
        });

        // listener for '=' button
        neq.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                num2 = get_second_number(tf);
                result = get_result(tf);
                res.setText(Double.toString(result));
                num1 = result;
            }
        });
    }

    public Calculator() {
        prepareGUI();
    }

    private void prepareGUI() {
        /*
         * creates a frame called "Calculator"
         * sets size to 375x225
         * sets bg to cyan
         */
        jfrm = new JFrame("Calculator");
        jfrm.setLayout(new FlowLayout());
        jfrm.setSize(330, 225);
        jfrm.setResizable(false);
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        jfrm.getContentPane().setBackground(Color.CYAN);

        // add Calculator textfield
        JTextField calc = new JTextField("CALCULATOR");
        calc.setEditable(false);
        calc.setBackground(Color.CYAN);
        calc.setPreferredSize(new Dimension(140, 25));
        jfrm.add(calc);

        // add Result text field
        JTextField res = new JTextField("RESULT");
        res.setEditable(false);
        res.setPreferredSize(new Dimension(140, 25));
        jfrm.add(res);

        // add all the number buttons and event listeners
        add_buttons_to_frame(jfrm, calc, res);
    }

    public void display() {
        jfrm.setVisible(true);
    }

    public static void main(String[] args) {
        Calculator calculator = new Calculator();
        calculator.display();
    }
}
```



Background Colour Switcher

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class BackGroundColor
{
    BackGroundColor()
    {
        JFrame jfrm = new JFrame("Background Color");
        jfrm.setLayout(new FlowLayout());
        jfrm.setSize(500,250);
        jfrm.setDefaultCloseOperation(jfrm.EXIT_ON_CLOSE);
        JButton Output = new JButton("Show Output");
        JComboBox jcb1 =new JComboBox();
        for(int i=0;i<=255;i++)
        {
            jcb1.addItem(String.valueOf(i));
        }
        JComboBox jcb2 =new JComboBox();
        for(int i=0;i<=255;i++)
        {
            jcb2.addItem(String.valueOf(i));
        }
        JComboBox jcb3 =new JComboBox();
        for(int i=0;i<=255;i++)
        {
            jcb3.addItem(String.valueOf(i));
        }
        JLabel red =new JLabel("Red");
        red.setForeground(Color.red);
        jfrm.add(red);
        jfrm.add(jcb1);
        JLabel green =new JLabel("Green");
        green.setForeground(Color.green);
        jfrm.add(green);
        jfrm.add(jcb2);
        JLabel blue =new JLabel("Blue");
        blue.setForeground(Color.blue);
        jfrm.add(blue);
        jfrm.add(jcb3);
        jfrm.add(Output);
        Output.addActionListener(new ActionListener()
        {
            public void actionPerformed(ActionEvent e)
            {
                int r,g,b;
                r = Integer.parseInt(jcb1.getSelectedItem().toString());
                g = Integer.parseInt(jcb2.getSelectedItem().toString());
                b = Integer.parseInt(jcb3.getSelectedItem().toString());
                Color mycolor = new Color(r,g,b);
                jfrm.getContentPane().setBackground(mycolor);
            }
        });
        jfrm.setVisible(true);
    }
    public static void main(String args[])
    {
        new BackGroundColor();
    }
}
```

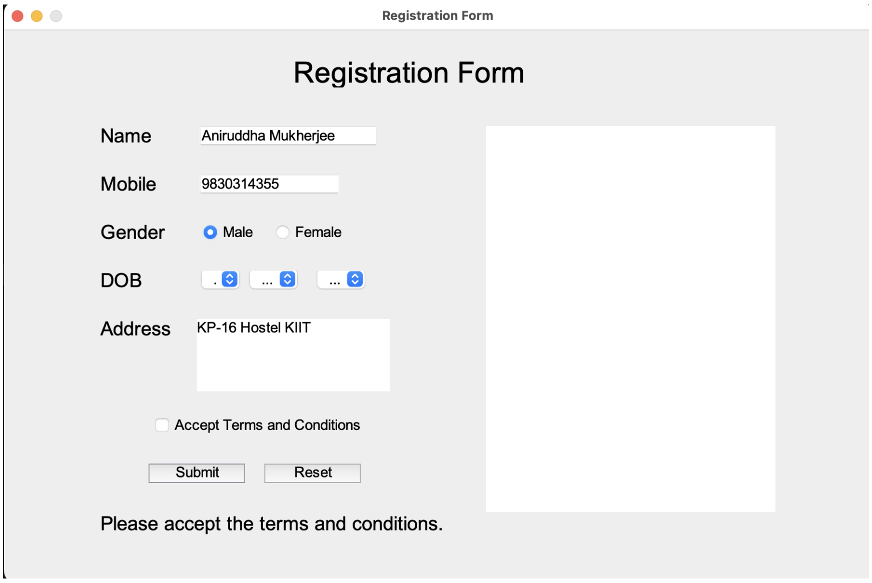


Background Color

Red 127 Green 127 Blue 189

Registration Form

```
package swinggg;

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

public class Registration_Validated {
    public static void main(String[] args) {
        MyFrame frame = new MyFrame();
    }
}

class MyFrame extends JFrame implements ActionListener {
    private JLabel titleLabel, nameLabel, mobileLabel, genderLabel, dobLabel, addressLabel, resultLabel;
    private JTextField nameField, mobileField;
    private JRadioButton maleRadio, femaleRadio;
    private ButtonGroup genderGroup;
    private JComboBox<String> dateCombo, monthCombo, yearCombo;
    private JTextArea addressArea, outputArea, resultArea;
    private JCheckBox termsCheckbox;
    private JButton submitButton, resetButton;

    private String[] dates = { "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13", "14", "15",
            "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28", "29",
            "30", "31" };
    private String[] months = { "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug",
    "Sep", "Oct", "Nov", "Dec" };
    private String[] years = { "1995", "1996", "1997", "1998", "1999", "2000", "2001",
    "2002", "2003", "2004", "2005", "2006", "2007", "2008", "2009" };

    public MyFrame() {
        setTitle("Registration Form");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(900, 600);
        setLocationRelativeTo(null);
        setResizable(false);

        JPanel panel = new JPanel();
        panel.setLayout(null);

        titleLabel = new JLabel("Registration Form");
        titleLabel.setFont(new Font("Arial", Font.PLAIN, 30));
        titleLabel.setBounds(300, 30, 300, 30);
        panel.add(titleLabel);

        nameLabel = new JLabel("Name");
        nameLabel.setFont(new Font("Arial", Font.PLAIN, 20));
        nameLabel.setBounds(100, 100, 100, 20);
        panel.add(nameLabel);
        nameField = new JTextField();
        nameField.setFont(new Font("Arial", Font.PLAIN, 15));
        nameField.setBounds(200, 100, 190, 20);
        panel.add(nameField);

        mobileLabel = new JLabel("Mobile");
        mobileLabel.setFont(new Font("Arial", Font.PLAIN, 20));
        mobileLabel.setBounds(100, 150, 100, 20);
        panel.add(mobileLabel);

        mobileField = new JTextField();
        mobileField.setFont(new Font("Arial", Font.PLAIN, 15));
        mobileField.setBounds(200, 150, 150, 20);
        panel.add(mobileField);

        genderLabel = new JLabel("Gender");
        genderLabel.setFont(new Font("Arial", Font.PLAIN, 20));
        genderLabel.setBounds(100, 200, 100, 20);
        panel.add(genderLabel);

        maleRadio = new JRadioButton("Male");
        maleRadio.setFont(new Font("Arial", Font.PLAIN, 15));
        maleRadio.setBounds(200, 200, 75, 20);
        maleRadio.setSelected(true);
        panel.add(maleRadio);

        femaleRadio = new JRadioButton("Female");
        femaleRadio.setFont(new Font("Arial", Font.PLAIN, 15));
        femaleRadio.setBounds(275, 200, 80, 20);
        panel.add(femaleRadio);

        genderGroup = new ButtonGroup();
        genderGroup.add(maleRadio);
        genderGroup.add(femaleRadio);

        dobLabel = new JLabel("DOB");
        dobLabel.setFont(new Font("Arial", Font.PLAIN, 20));
        dobLabel.setBounds(100, 250, 100, 20);
        panel.add(dobLabel);

        dateCombo = new JComboBox<>(dates);
        dateCombo.setFont(new Font("Arial", Font.PLAIN, 15));
        dateCombo.setBounds(200, 250, 50, 20);
        panel.add(dateCombo);

        monthCombo = new JComboBox<>(months);
        monthCombo.setFont(new Font("Arial", Font.PLAIN, 15));
        monthCombo.setBounds(250, 250, 60, 20);
        panel.add(monthCombo);

        yearCombo = new JComboBox<>(years);
        yearCombo.setFont(new Font("Arial", Font.PLAIN, 15));
        yearCombo.setBounds(320, 250, 60, 20);
        panel.add(yearCombo);

        addressLabel = new JLabel("Address");
        addressLabel.setFont(new Font("Arial", Font.PLAIN, 20));
        addressLabel.setBounds(100, 300, 100, 20);
        panel.add(addressLabel);

        addressArea = new JTextArea();
        addressArea.setFont(new Font("Arial", Font.PLAIN, 15));
        addressArea.setBounds(200, 300, 200, 75);
        addressArea.setLineWrap(true);
        panel.add(addressArea);

        termsCheckbox = new JCheckBox("Accept Terms and Conditions");
        termsCheckbox.setFont(new Font("Arial", Font.PLAIN, 15));
        termsCheckbox.setBounds(150, 400, 250, 20);
        panel.add(termsCheckbox);

        submitButton = new JButton("Submit");
        submitButton.setFont(new Font("Arial", Font.PLAIN, 15));
        submitButton.setBounds(150, 450, 100, 20);
        submitButton.addActionListener(this);
        panel.add(submitButton);

        resetButton = new JButton("Reset");
        resetButton.setFont(new Font("Arial", Font.PLAIN, 15));
        resetButton.setBounds(270, 450, 100, 20);
        resetButton.addActionListener(this);
        panel.add(resetButton);

        outputArea = new JTextArea();
        outputArea.setFont(new Font("Arial", Font.PLAIN, 15));
        outputArea.setBounds(500, 100, 300, 400);
        outputArea.setLineWrap(true);
        outputArea.setEditable(false);
        panel.add(outputArea);

        resultLabel = new JLabel("");
        resultLabel.setFont(new Font("Arial", Font.PLAIN, 20));
        resultLabel.setBounds(100, 500, 500, 25);
        panel.add(resultLabel);

        resultArea = new JTextArea();
        resultArea.setFont(new Font("Arial", Font.PLAIN, 15));
        resultArea.setBounds(580, 175, 200, 75);
        resultArea.setLineWrap(true);
        panel.add(resultArea);

        add(panel);
        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == submitButton) {
            if (termsCheckbox.isSelected() && mobileField.getText().length() == 10) {
                String name = nameField.getText();
                String mobile = mobileField.getText();
                String gender = maleRadio.isSelected() ? "Male" : "Female";
                String dob = dateCombo.getSelectedItem() + "/" + monthCombo.getSelectedItem() + "/"
                        + yearCombo.getSelectedItem();
                String address = addressArea.getText();

                String output = "Name: " + name + "\n"
                        + "Mobile: " + mobile + "\n"
                        + "Gender: " + gender + "\n"
                        + "DOB: " + dob + "\n"
                        + "Address: " + address;

                outputArea.setText(output);
                resultLabel.setText("Registration Successful!");
            } else {
                outputArea.setText("");
                resultArea.setText("");
                if (mobileField.getText().length() != 10) {
                    resultLabel.setText("Mobile number must have exactly 10 digits.");
                } else {
                    resultLabel.setText("Please accept the terms and conditions.");
                }
            }
        } else if (e.getSource() == resetButton) {
            nameField.setText("");
            mobileField.setText("");
            addressArea.setText("");
            outputArea.setText("");
            resultArea.setText("");
            resultLabel.setText("");
            termsCheckbox.setSelected(false);
            dateCombo.setSelectedIndex(0);
            monthCombo.setSelectedIndex(0);
            yearCombo.setSelectedIndex(0);
        }
    }
}
```



Registration Form

Registration Form

Name Aniruddha Mukherjee

Mobile 9830314355

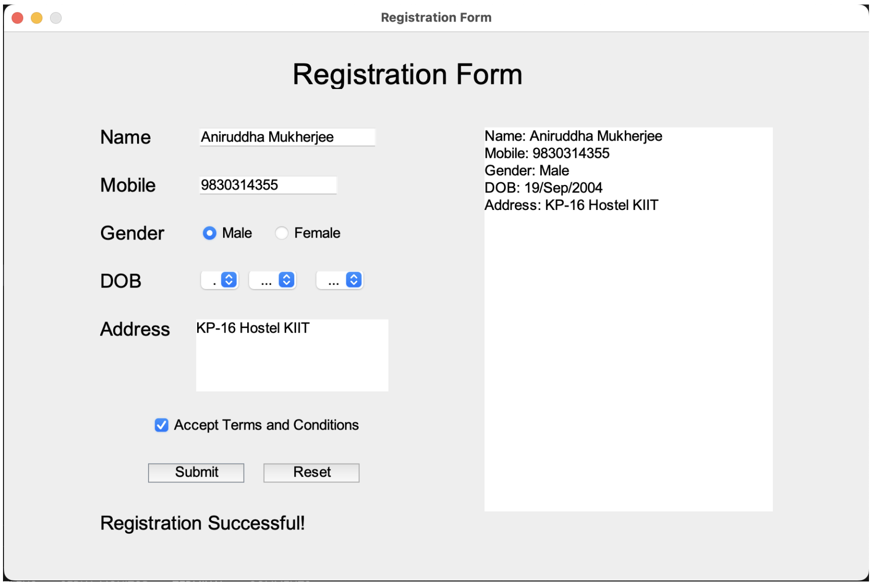
Gender ● Male ○ Female

DOB . ...

Address KP-16 Hostel KIIT

□ Accept Terms and Conditions

Please accept the terms and conditions.



Registration Form

Registration Form

Name Aniruddha Mukherjee

Mobile 9830314355

Gender ● Male ○ Female

DOB . ...

Address KP-16 Hostel KIIT

☑ Accept Terms and Conditions

```
Name: Aniruddha Mukherjee
Mobile: 9830314355
Gender: Male
DOB: 19/Sep/2004
Address: KP-16 Hostel KIIT
```

Registration Successful!