Problem 1

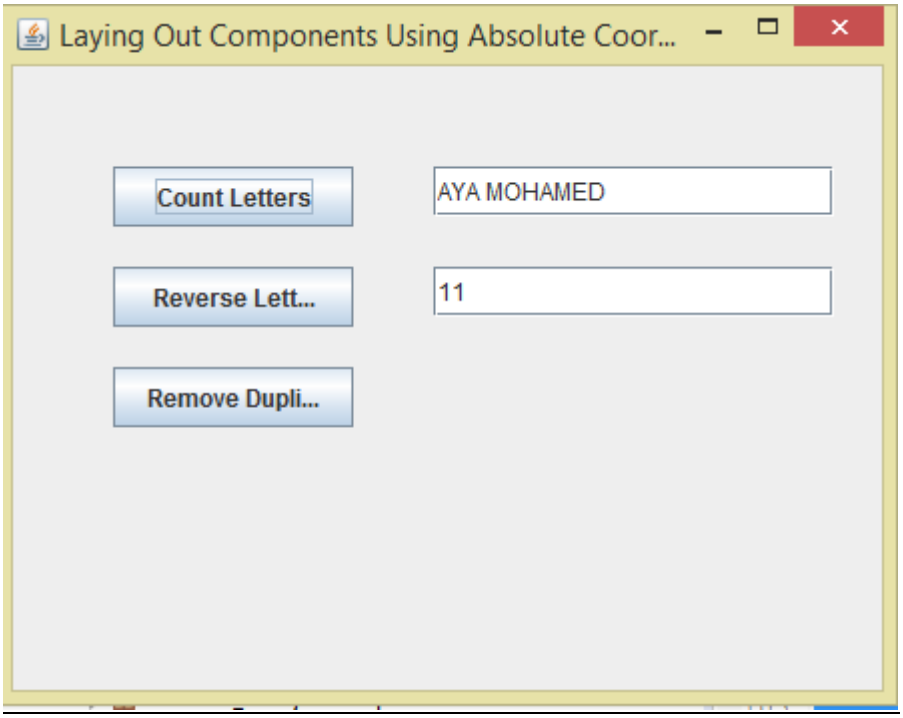
```
package lesson6_Assignment.problem1;

import java.awt.Color;

import java.awt.Container;

import java.awt.FlowLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.JTextField;


public class ProblemOne {


		public static void main(String[] args) {


			JFrame frame = new JFrame("Laying Out Components Using Absolute
Coordinates");


			frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


			JPanel panel = new JPanel();


			panel.setLayout(null);


			JButton button1 = new JButton("Count Letters");

			JButton button2 = new JButton("Reverse Letters");

			JButton button3 = new JButton("Remove Duplicates");

			JTextField input = new JTextField(20);

			JTextField output = new JTextField(20);
```

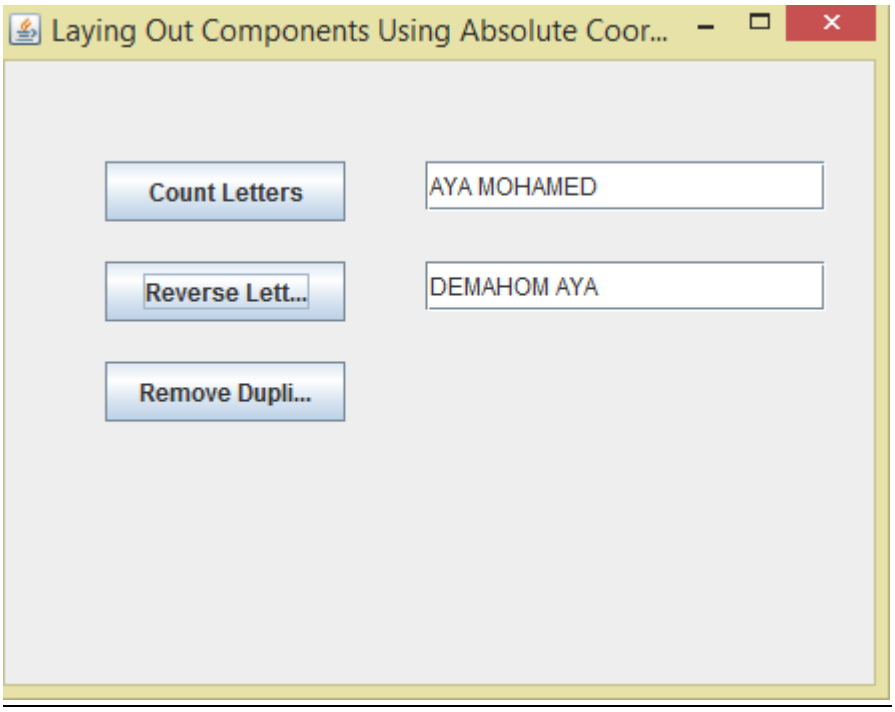
```
button1.setBounds(50, 50, 120, 30);

button2.setBounds(50, 100, 120, 30);

button3.setBounds(50, 150, 120, 30);


input.setBounds(210, 50, 200, 25);

output.setBounds(210, 100, 200, 25);


panel.add(button1);

panel.add(button2);

panel.add(button3);

panel.add(input);

panel.add(output);


frame.add(panel);


frame.setSize(450, 350);


frame.setVisible(true);


// lambda expression

button1.addActionListener(e ->


output.setText(String.valueOf(input.getText().length()))


);


// Anonymous inner class

button2.addActionListener(new ActionListener() {
```

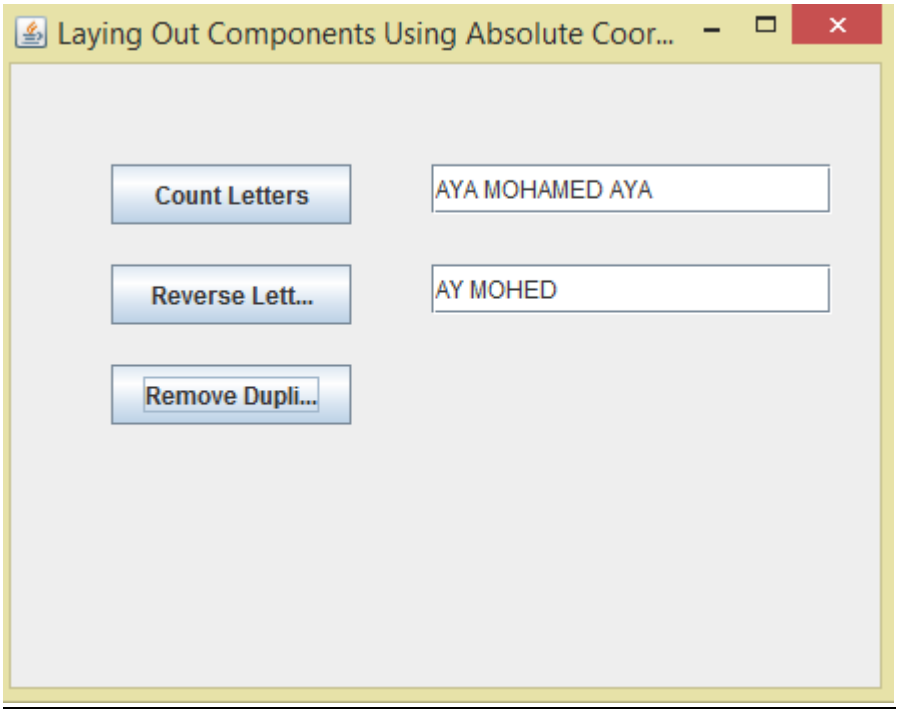
```
                @Override
                public void actionPerformed(ActionEvent e) {
                        String text = input.getText();
                        String reversed = reverse(text);
                        output.setText(String.valueOf(reversed));

                }
            });

            // member class
            class MyEvent implements ActionListener {
                @Override
                public void actionPerformed(ActionEvent e1) {
                        String text = input.getText();
                        String stringAfterRemoveDuplicates =
removeDuplicates(text);

            output.setText(String.valueOf(stringAfterRemoveDuplicates));
                        frame.getContentPane().setBackground(Color.BLACK);

                }
            }

            button3.addActionListener(new MyEvent());

        }

        public static String reverse(String str) {
            if (str == null) {
```

```
                return null;
            }
            if (str.length() <= 1) {
                return str;
            }
            return reverse(str.substring(1)) + str.charAt(0);
        }

        public static String removeDuplicates(String s) {
            if (s.length() <= 1)
                return s;
            if (s.substring(0, s.length() - 1).contains(s.substring(s.length() - 1,
s.length())))
                return removeDuplicates(s.substring(0, s.length() - 1));
            else
                return removeDuplicates(s.substring(0, s.length() - 1)) +
s.substring(s.length() - 1, s.length());

        }

}
```

Output


Laying Out Components Using Absolute Coor...
Count Letters
AYA MOHAMED
Reverse Lett...
11
Remove Dupli...


Laying Out Components Using Absolute Coor...
Count Letters
AYA MOHAMED
Reverse Lett...
DEMAHOM AYA
Remove Dupli...



Problem_2

```
package lesson6_Assignment.problem2;


import java.awt.Color;

import java.awt.Dimension;

import java.awt.FlowLayout;


import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JOptionPane;


public class RainBow extends JFrame {


		JButton square1 = new JButton();

		JButton square2 = new JButton();

		JButton square3 = new JButton();

		JButton square4 = new JButton();
```

```
        JButton square5 = new JButton();
        JButton square6 = new JButton();

        private JButton[] squares = { square1, square2, square3, square4,
square5, square6 };

        private final static Color VIOLET = new Color(128, 0, 128);

        private Color colors[] = { Color.RED, Color.ORANGE, Color.GREEN,
Color.BLUE, Color.YELLOW, VIOLET };

        private String colorsAsString[] = { "RED", "ORANGE", "GREEN", "BLUE",
"YELLOW", "VIOLET" };

        public RainBow() {

           setTitle("Rainbow colors");

           setBounds(0, 0, 400, 100);
           for (int i = 0; i < squares.length; i++) {

                 squares[i].setBackground(colors[i]);

                 squares[i].setPreferredSize(new Dimension(50, 50));

                 setLayout(new FlowLayout(FlowLayout.LEFT));

                 add(squares[i]);
           }
```

```
        square1.addActionListener(e -> showMessage(colorsAsString[0]));
        square2.addActionListener(e -> showMessage(colorsAsString[1]));
        square3.addActionListener(e -> showMessage(colorsAsString[2]));
        square4.addActionListener(e -> showMessage(colorsAsString[3]));
        square5.addActionListener(e -> showMessage(colorsAsString[4]));
        square6.addActionListener(e -> showMessage(colorsAsString[5]));

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public void showMessage(String message) {
        JOptionPane.showMessageDialog(this, message, "Color Meaning",
JOptionPane.INFORMATION_MESSAGE);
    }

    public static void main(String args[]) {
        RainBow f = new RainBow();
        f.setVisible(true);
    }
}
```

Output


Rainbow colors
Color Meaning
GREEN
OK
1
2
3
4
5
6
7