LAPORAN PRATIKUM

ALGORITMA DAN PEMROGRAMAN

“GUI dan Aplikasi Operasi Aritmatika Sederhana pada Java ”

disusun Oleh:

Muhammad Yasin Habiburrahman

2511532016

Dosen Pengampu:

Dr. Wahyudi. S.T.M.T

Asisten Pratikum:

Muhammad Zaki Al Hafiz

DEPARTEMEN INFORMATIKA FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

TAHUN 2025

KATA PENGANTAR

Puji dan syukur kami panjatkan ke hadirat Allah SWT atas segala rahmat dan karunia-Nya, sehingga kami dapat menyelesaikan tugas kelompok ini dengan baik. Shalawat serta salam semoga senantiasa tercurah kepada junjungan kita Nabi Muhammad SAW, keluarga, sahabat, serta para pengikutnya hingga akhir zaman.

Tugas kelompok ini kami susun dengan tujuan untuk memenuhi salah satu tugas mata kuliah Algoritma dan Pemrograman, dengan tema "GUI dan Aplikasi Operasi Aritmatika Sederhana pada Java", yang mencakup bagaimana membuat aplikasi operasi aritmatika sederhana menggunakan JFrame melalui Eclipse dan program Java.

Kami menyadari bahwa penyusunan tugas ini masih jauh dari sempurna, baik dari segi materi maupun penyajiannya. Oleh karena itu, kami sangat mengharapkan kritik dan saran yang membangun dari semua pihak demi kesempurnaan tugas kami di masa mendatang.

Akhir kata, kami mengucapkan terima kasih kepada dosen pembimbing yang telah memberikan arahan, serta kepada seluruh anggota kelompok yang telah bekerja sama dengan baik sehingga tugas ini dapat terselesaikan tepat waktu. Semoga tugas ini dapat bermanfaat bagi kami khususnya, dan bagi pembaca pada umumnya.

Padang,

Muhammad Yasin Habiburrahman

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pemrograman berorientasi objek pada Java tidak hanya digunakan untuk membangun program berbasis teks, tetapi juga dapat dikembangkan menjadi aplikasi dengan antarmuka grafis yang lebih interaktif. Salah satu teknologi yang digunakan untuk membangun aplikasi berbasis GUI adalah *Java Swing*, yaitu pustaka yang menyediakan berbagai komponen visual seperti *JFrame*, *JPanel*, *JLabel*, *JButton*, dan elemen-elemen lainnya. Pada praktikum minggu ini, mahasiswa mulai diperkenalkan pada pembuatan aplikasi GUI sederhana menggunakan kelas *JFrame* sebagai jendela utama program. Melalui kegiatan ini, mahasiswa belajar bagaimana menyusun komponen antarmuka, mengatur tata letak, serta menghubungkannya dengan logika program. Pemahaman terhadap *Swing* menjadi langkah awal yang penting karena membuka peluang untuk membuat aplikasi Java yang lebih interaktif, seperti kalkulator, form input data, hingga program berbasis event-driven lainnya.

1.2 Tujuan

- Memahami konsep dasar antarmuka grafis (*Graphical User Interface*) pada Java.
- Mengenal fungsi dan penggunaan komponen *Swing*, khususnya *JFrame*.
- Mampu membuat jendela aplikasi menggunakan *JFrame* dan menambahkan komponen GUI ke dalamnya.
- Mempelajari cara kerja event handling menggunakan *ActionListener*.
- Mengembangkan aplikasi Java sederhana berbasis GUI, seperti kalkulator.

1.3 Manfaat Praktikum

- Mahasiswa dapat membuat aplikasi yang lebih interaktif dan mudah digunakan oleh pengguna.
- Memperluas wawasan tentang pengembangan aplikasi desktop menggunakan Jawa.

- Menambah pengalaman dalam mengelola komponen visual dan event pada GUI.

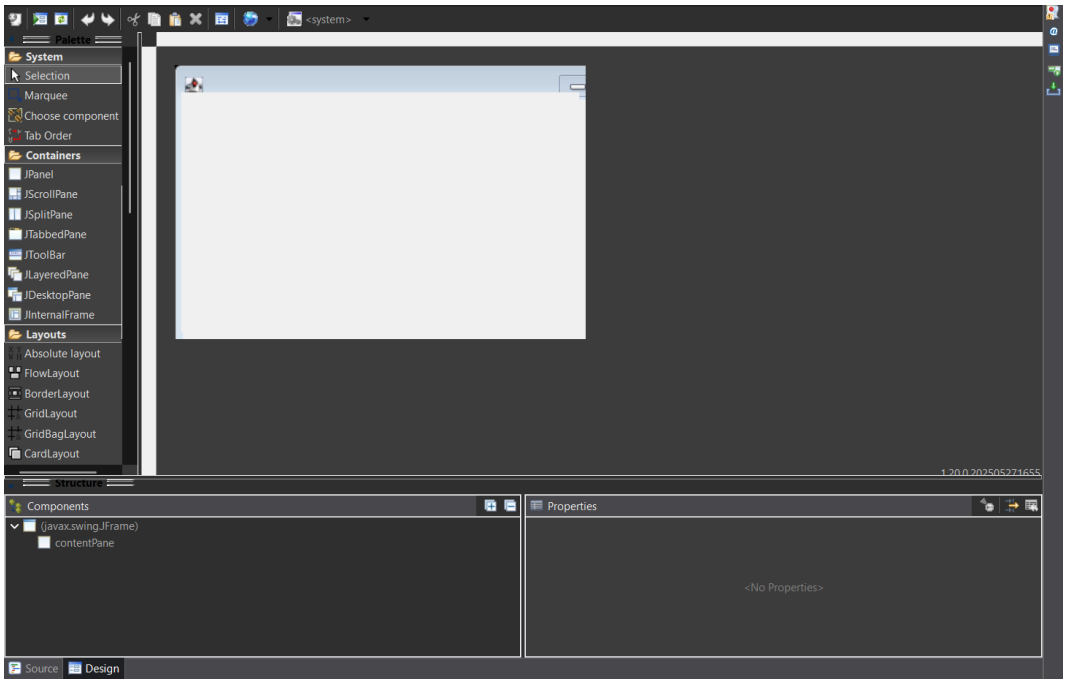
- Menjadi dasar untuk pembuatan aplikasi yang lebih kompleks, seperti form input data, aplikasi perhitungan, dan sistem mini berbasis GUI.

- Membantu memahami hubungan antara logika program dan elemen antarmuka yang digunakan pengguna.

BAB II
PEMBAHASAN

2.1 JFrame

Dalam pembuatan aplikasi berbasis antarmuka grafis di Java, *JFrame* merupakan komponen utama yang berfungsi sebagai jendela tempat seluruh elemen GUI akan ditempatkan. Kelas *JFrame* berasal dari *library Swing*, yaitu kumpulan kelas yang disediakan Java untuk membangun aplikasi desktop dengan tampilan grafis yang modern. Sebuah objek *JFrame* pada dasarnya adalah sebuah jendela seperti aplikasi pada umumnya, lengkap dengan *title bar*, tombol *minimize*, *maximize*, dan *close*. Di dalam *JFrame* inilah programmer menambahkan berbagai komponen lain seperti *label*, *text field*, *button*, dan *panel*. Selain itu, *JFrame* memberikan pengaturan umum seperti ukuran jendela, posisi awal ketika ditampilkan, serta aksi ketika jendela ditutup.



Gambar 2.1. – Antarmuka JFrame pada Eclipse

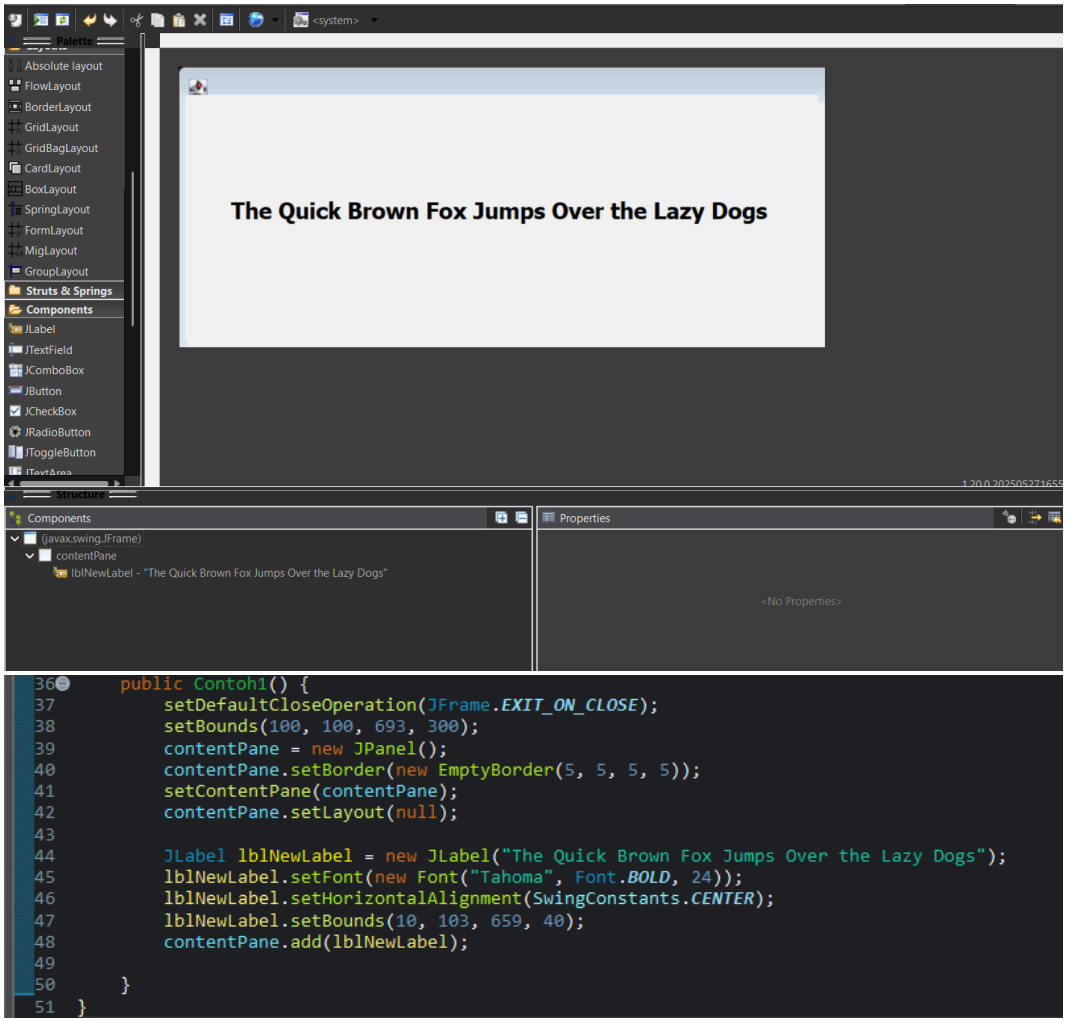
Dalam pengembangan aplikasi GUI menggunakan Eclipse, terutama ketika memanfaatkan fitur *WindowBuilder*, setiap komponen yang ditambahkan melalui tampilan desain secara otomatis akan dihasilkan dalam bentuk kode Java pada bagian *Source*. Ketika pengguna menempatkan *label*, *text field*, *combo box*, *button*, atau komponen lainnya ke dalam *JFrame*, Eclipse akan membuat objek komponen

tersebut, mengatur properti seperti posisi, ukuran, dan teks, serta menambahkannya ke dalam *content pane* tanpa perlu ditulis secara manual. Proses ini mempermudah pembuatan antarmuka karena tidak harus memahami seluruh detail kode untuk setiap komponen secara langsung, tetapi tetap dapat melihat kode lengkapnya jika diperlukan.

```java
package pekan8_2511532016;

import java.awt.EventQueue;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

public class Contoh1 extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Contoh1 frame = new Contoh1();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */
    public Contoh1() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 450, 300);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);

    }

}
```

Gambar 2.2 – Tampilan Kode Program setelah File JFrame Dibuat

Seperti yang sudah dijelaskan sebelumnya, Eclipse secara otomatis akan menuliskan komponen yang telah ditambahkan pada bagian desain tanpa perlu campur tangan pemrogram pada bagian *source*.

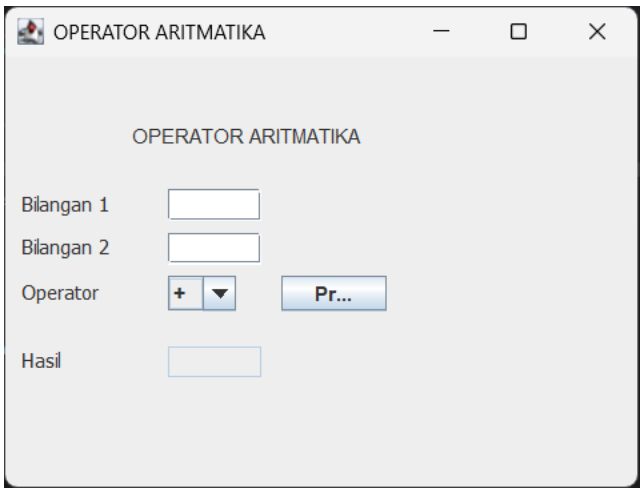


Gambar 2.3 – Perubahan Kode Program setelah Menambahkan Label pada Desain

Pada bagian desain, telah ditambahkan komponen label, kemudian diberi tulisan "The Quick Brown Fox Jumps Over the Lazy Dogs" lengkap dengan *alignment* tengah; dan *font*, *font* style, dan ukuran *font* berturut-turut adalah "Tahoma", *bold*, dan 24. Eclipse juga memberikan ukuran tabel yang memuat tulisan melalui setBounds(x, y, panjang, lebar).

2.2 Program Operasi Aritmatika

Setelah meletakkan komponen-komponen yang diperlukan ke kanvas di bagian desain, program operasi aritmatika yang telah dibuat memiliki tampilan sebagai berikut.



Gambar 2.4 – Tampilan Antarmuka dari Program Operasi Aritmatika

Adapun kode program yang dihasilkan oleh Eclipse adalah sebagai berikut.

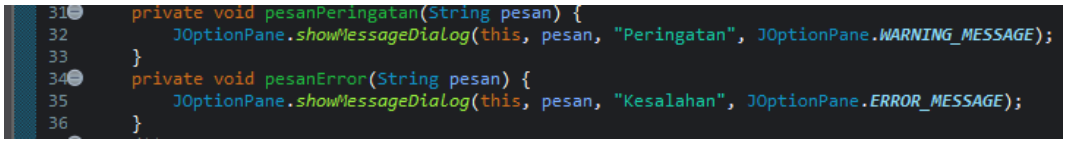
```
57          setTitle("OPERATOR ARITMATIKA");
58          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
59          setBounds(100, 100, 394, 295);
60          contentPane = new JPanel();
61          contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
62          setContentPane(contentPane);
63          contentPane.setLayout(null);
64
65          JLabel lblNewLabel = new JLabel("OPERATOR ARITMATIKA");
66          lblNewLabel.setHorizontalAlignment(SwingConstants.CENTER);
67          lblNewLabel.setFont(new Font("Formula1 Display Bold", Font.PLAIN, 12));
68          lblNewLabel.setBounds(38, 32, 213, 31);
69          contentPane.add(lblNewLabel);
70
71          JLabel lblNewLabel_1 = new JLabel("Bilangan 1");
72          lblNewLabel_1.setFont(new Font("Tahoma", Font.PLAIN, 12));
73          lblNewLabel_1.setBounds(10, 73, 78, 31);
74          contentPane.add(lblNewLabel_1);
75
76          JLabel lblNewLabel_1_1 = new JLabel("Bilangan 2");
77          lblNewLabel_1_1.setFont(new Font("Tahoma", Font.PLAIN, 12));
78          lblNewLabel_1_1.setBounds(10, 99, 78, 31);
79          contentPane.add(lblNewLabel_1_1);
80
81          JLabel lblNewLabel_1_1_1 = new JLabel("Operator");
82          lblNewLabel_1_1_1.setFont(new Font("Tahoma", Font.PLAIN, 12));
83          lblNewLabel_1_1_1.setBounds(10, 126, 78, 31);
84          contentPane.add(lblNewLabel_1_1_1);
85
86          JLabel lblNewLabel_1_1_1_1 = new JLabel("Hasil");
87          lblNewLabel_1_1_1_1.setFont(new Font("Tahoma", Font.PLAIN, 12));
88          lblNewLabel_1_1_1_1.setBounds(10, 167, 78, 31);
89          contentPane.add(lblNewLabel_1_1_1_1);
90
91          txtBil1 = new JTextField();
92          txtBil1.setHorizontalAlignment(SwingConstants.CENTER);
93          txtBil1.setBounds(98, 80, 56, 19);
94          contentPane.add(txtBil1);
95          txtBil1.setColumns(10);
96
97          txtBil2 = new JTextField();
98          txtBil2.setHorizontalAlignment(SwingConstants.CENTER);
99          txtBil2.setColumns(10);
100         txtBil2.setBounds(98, 106, 56, 19);
101         contentPane.add(txtBil2);
102
103         JComboBox cbOperator = new JComboBox();
104         cbOperator.setModel(new DefaultComboBoxModel(new String[] {"+", "-", "*", "/", "%"}));
105         cbOperator.setBounds(98, 132, 41, 21);
106         contentPane.add(cbOperator);
107
108         txtHasil = new JTextField();
109         txtHasil.setEditable(false);
110         txtHasil.setHorizontalAlignment(SwingConstants.CENTER);
111         txtHasil.setColumns(10);
112         txtHasil.setBounds(98, 174, 56, 19);
113         contentPane.add(txtHasil);
```

Gambar 2.5 – Kode Program yang Dihasilkan oleh Eclipse

2.3 Logika Program dan *Error Handling*

Metode pesanPeringatan() digunakan untuk menampilkan peringatan ringan, misalnya ketika pengguna belum mengisi kolom input Bilangan 1 atau Bilangan 2. Metode ini menampilkan *dialog box* dengan ikon kuning (warning icon) menggunakan JOptionPane.WARNING_MESSAGE.

Metode pesanError() digunakan untuk menangani kesalahan yang lebih serius, seperti jika input bukan angka, atau ketika pengguna mencoba melakukan operasi yang tidak valid seperti pembagian dengan 0. Metode ini menggunakan JOptionPane.ERROR_MESSAGE, sehingga kotak dialog tampil dengan ikon merah (error icon) yang menandakan masalah yang lebih penting dibanding peringatan biasa.



Gambar 2.6 – *Method* Pesan Eror dan Peringatan

a. Validasi Input Kosong

Sebelum perhitungan dilakukan, program memastikan bahwa kedua kotak input telah diisi. Pengecekan ini penting agar program tidak mengalami kesalahan ketika mencoba mengonversi string kosong menjadi angka.

```
if(txtBil1.getText().trim().isEmpty())
   pesanPeringatan("Bilangan 1 harus diisi");
else if(txtBil2.getText().trim().isEmpty())
   pesanPeringatan("Bilangan 2 harus diisi");
```

Tabel 2.1 – *Snippet* Logika Input Kosong

b. Konversi Input ke Integer

Setelah input dipastikan tidak kosong, program mencoba mengonversi nilai tersebut menjadi tipe data *int*.

```
try {
   int a = Integer.parseInt(txtBil1.getText());
   int b = Integer.parseInt(txtBil2.getText());
```

Tabel 2.2 – *Snippet* Mengubah int menjadi String

c. Penanganan Kesalahan Format Angka

Jika input bukan angka valid, program akan menampilkan dialog kesalahan melalui pesanError(). Dengan cara ini, program lebih aman karena tidak menghentikan eksekusi secara tiba-tiba dan memberikan pesan yang jelas kepada pengguna.

```
} catch (NumberFormatException ex) {
   pesanError("Bilangan 1 dan 2 harus diisi angka");
}
```

Tabel 2.3 – *Snippet* Penanganan Eror Input Non-Angka

d. Pemilihan Operator Menggunakan ComboBox

Baris ini membaca operator yang dipilih pengguna pada JComboBox. Operator direpresentasikan sebagai indeks (0–4), masing-masing mewakili operasi penjumlahan, pengurangan, perkalian, pembagian, dan modulus.

```
int c = cbOperator.getSelectedIndex();
```

Tabel 2.4 – *Snippet* Pemilihan Operator

e. Logika Perhitungan Aritmatika

Setiap operator ditangani dalam blok if terpisah. Setiap kondisi menjalankan operasi aritmatika sesuai pilihan pengguna. Hasil kemudian disimpan ke variabel hasil yang akan ditampilkan pada akhir proses.

```
if(c == 0) hasil = a + b;
if(c == 1) hasil = a - b;
if(c == 2) hasil = a * b;
```

Tabel 2.5 – *Snippet* Pemilihan Operator Penjumlahan, Pengurangan, dan Perkalian

f. Penanganan Kasus Pembagian dan Modulo dengan 0

Pembagian dan modulo dengan nol tidak diperbolehkan dalam matematika maupun dalam Java. Karena itu, program memeriksa nilai pembaginya (b). Jika nilai tersebut nol, program menampilkan pesan kesalahan dan menghentikan proses perhitungan dengan return.

```
if (c == 3) {
   if(b == 0) {
      pesanError("Tidak dapat membagi dengan 0!");
      return;
   }
   hasil = a / b;
}

if(c == 4){
  if(b==0) {
     pesanError(“Tidak dapat melakukan modulo dengan 0!”)
     return;
   }
   hasil = a % b;
}
```

Tabel 2.6 – *Snippet* Penanganan Pembagi 0

g. Menampilkan Hasil ke Output

Setelah perhitungan berhasil dilakukan, hasil akhir dikonversi ke bentuk string dan ditampilkan pada kotak “Hasil”. Bagian ini memberikan umpan balik langsung kepada pengguna mengenai hasil perhitungan yang dilakukan program.

```
txtHasil.setText(String.valueOf(hasil));
```

Tabel 2.7 – Menampilkan Hasil Operasi

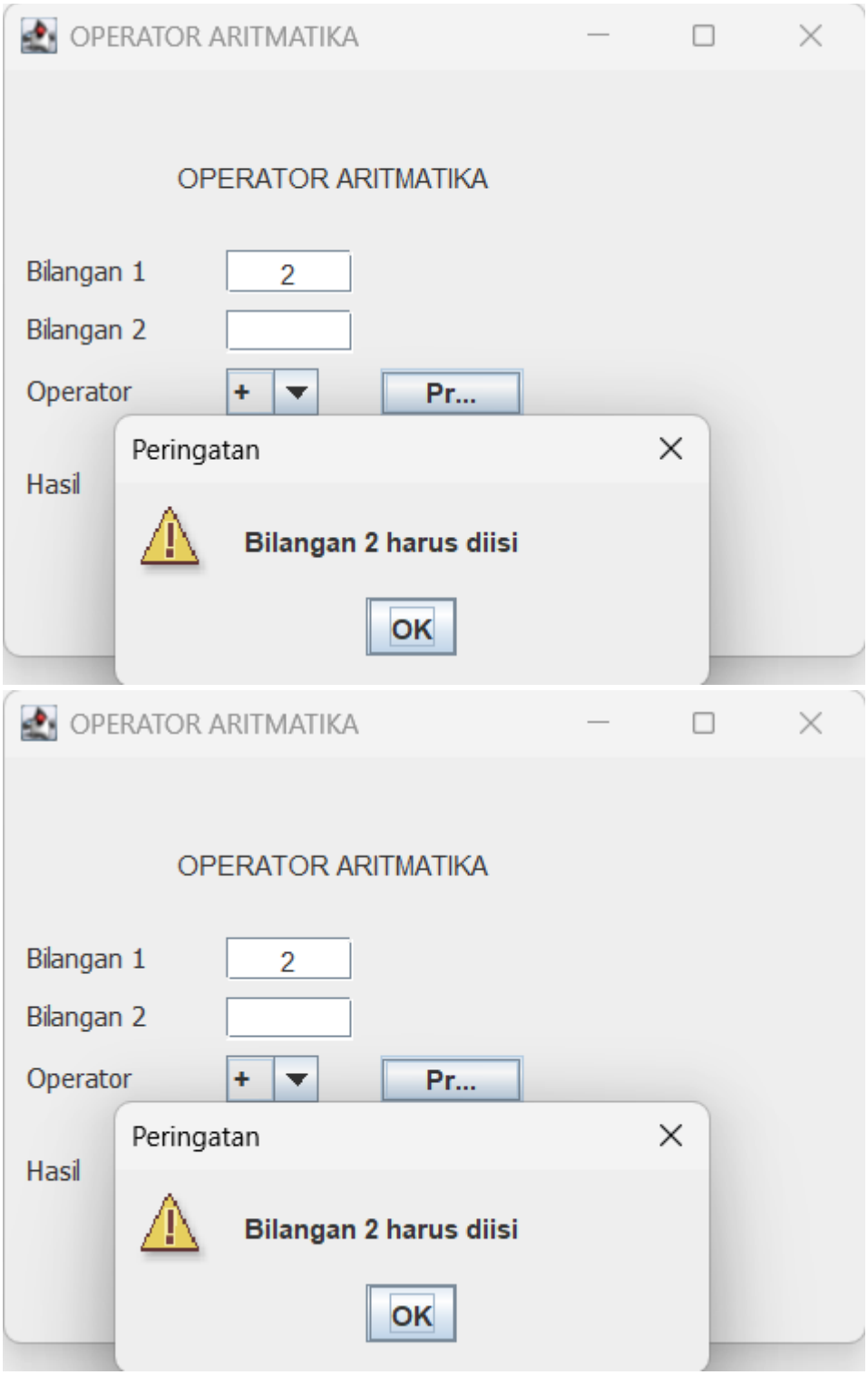
Setelah struktur tersebut disatukan, akan menghasilkan kode program lengkap sebagai berikut.

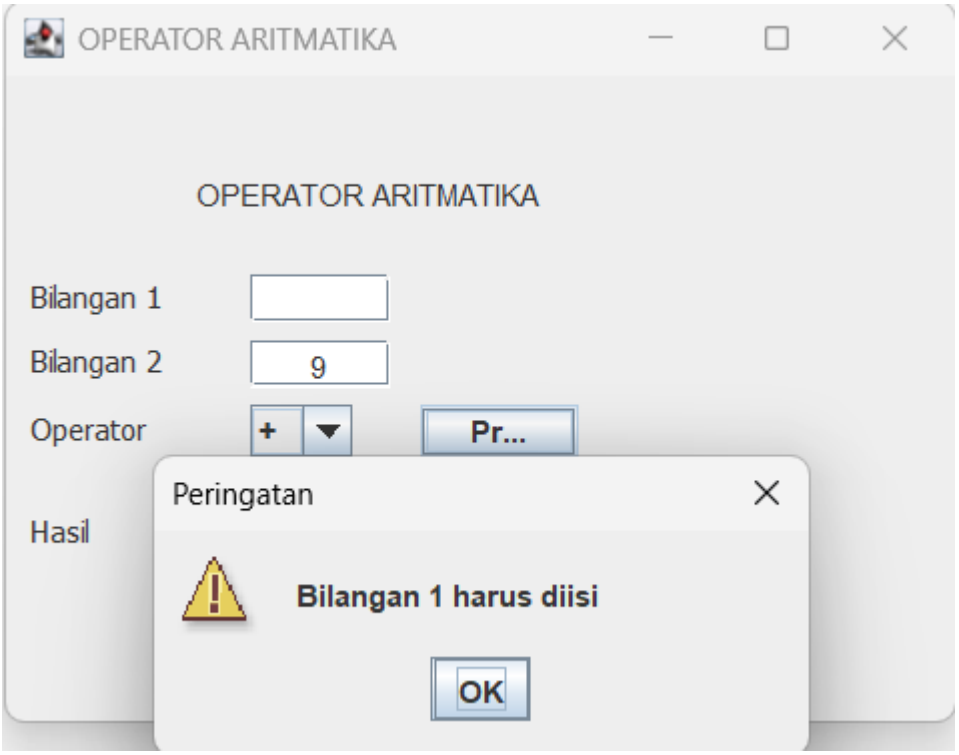
```
117         JButton btnNewButton = new JButton("Proses");
118         btnNewButton.addActionListener(new ActionListener() {
119             int hasil;
120             public void actionPerformed(ActionEvent e) {
121 
122                 if(txtBil1.getText().trim().isEmpty()) pesanPeringatan("Bilangan 1 harus diisi");
123                 else if(txtBil2.getText().trim().isEmpty()) pesanPeringatan("Bilangan 2 harus diisi");
124                 else {
125                     try {
126                         int a = Integer.parseInt(txtBil1.getText());
127                         int b = Integer.parseInt(txtBil2.getText());
128                         int c = cbOperator.getSelectedIndex(); //memilih operator
129 
130                         if(c == 0)hasil= a+b;
131                         if(c == 1) hasil = a-b;
132                         if(c == 2)hasil = a*b;
133                         if (c == 3) {
134                             if(b == 0) {
135                                 pesanError("Tidak dapat membagi dengan 0!");
136                                 return;
137                             }
138                             hasil = a / b;
139                         }
140                         if (c == 4) {
141                             if(b == 0) {
142                                 pesanError("Tidak dapat melakukan modulo dengan 0!");
143                                 return;
144                             }
145                             hasil = a % b;
146                         }
147                     } catch (NumberFormatException ex) {
148                         pesanError("Bilangan 1 dan 2 harus diisi angka");
149                     }
150                 }
151                 txtHasil.setText(String.valueOf(hasil));
152             }
153         });
154         btnNewButton.setBounds(166, 132, 63, 21);
155         contentPane.add(btnNewButton);
```

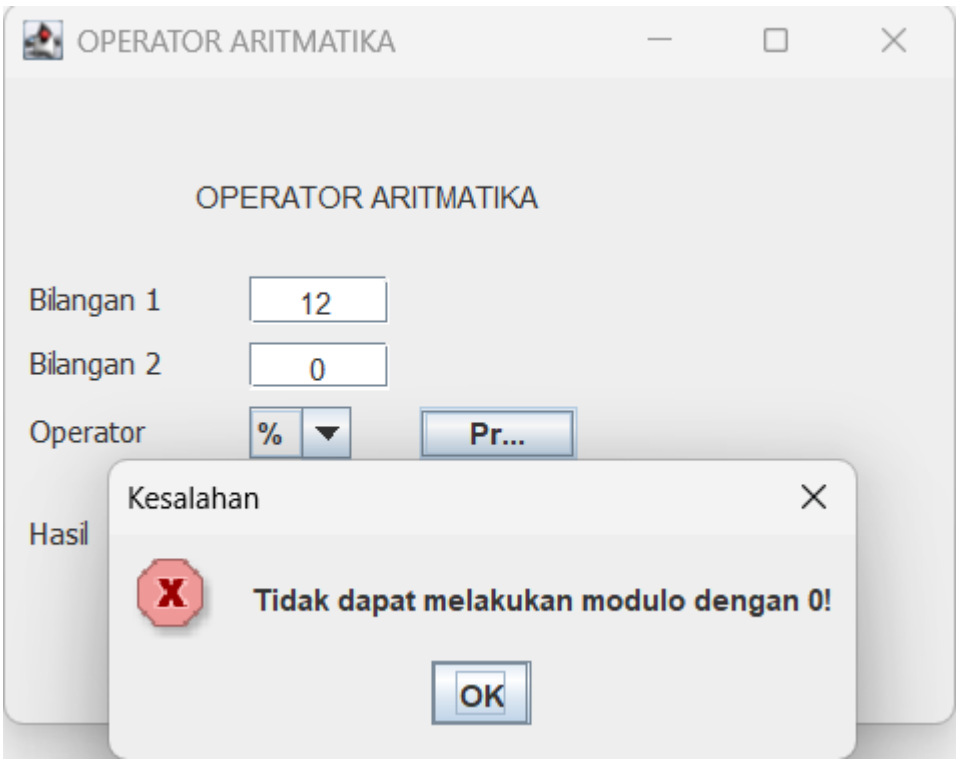
Gambar 2.7 – Kode Program Bagian Logika dan Aritmatika

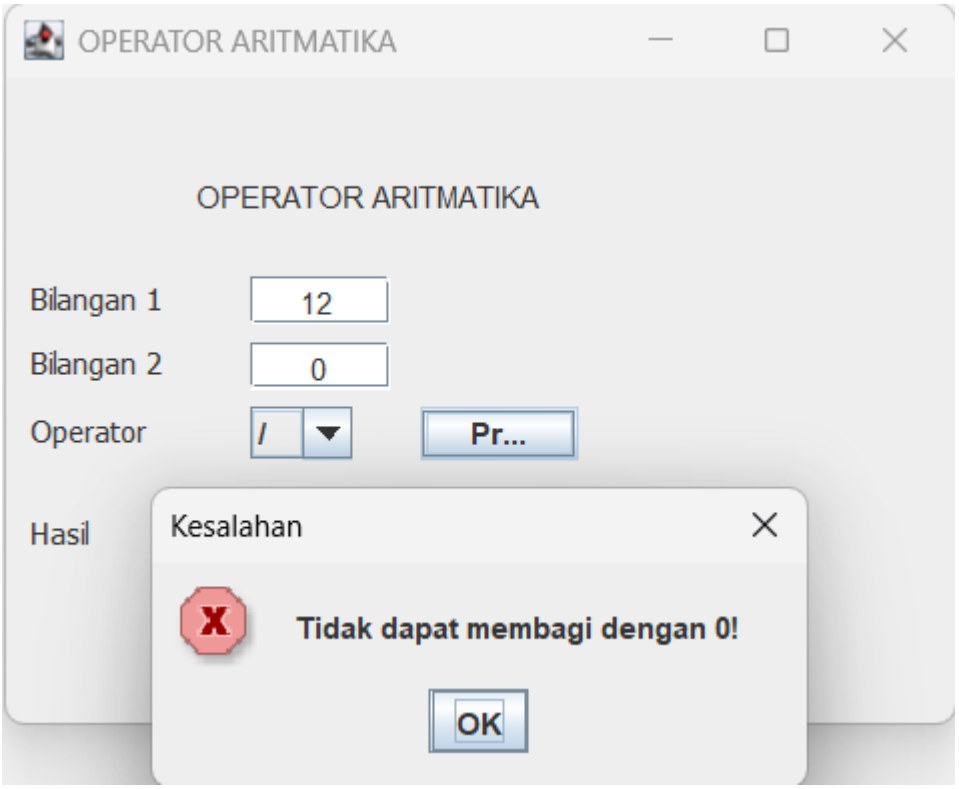
2.4 Contoh Output

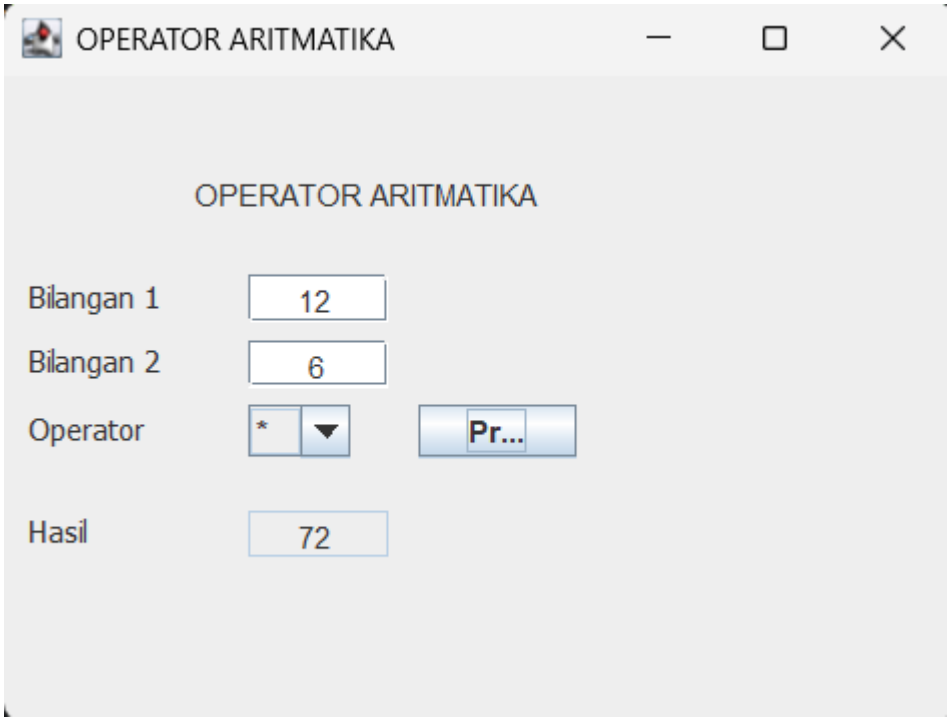
Berikut merupakan contoh output hasil operasi aritmatika, pesan peringatan, dan pesan eror.




OPERATOR ARITMATIKA
OPERATOR ARITMATIKA
Bilangan 1
Bilangan 2
9
Operator
+
Pr...
Hasil
Peringatan
Bilangan 1 harus diisi
OK


OPERATOR ARITMATIKA
OPERATOR ARITMATIKA
Bilangan 1
12
Bilangan 2
0
Operator
%
Pr...
Hasil
Kesalahan
Tidak dapat melakukan modulo dengan 0!
OK


OPERATOR ARITMATIKA
OPERATOR ARITMATIKA
Bilangan 1
12
Bilangan 2
0
Operator
/
Pr...
Hasil
Kesalahan
Tidak dapat membagi dengan 0!
OK


OPERATOR ARITMATIKA
OPERATOR ARITMATIKA
Bilangan 1
12
Bilangan 2
6
Operator
*
Pr...
Hasil
72

BAB III
KESIMPULAN

Pada praktikum pekan ini, mahasiswa mempelajari bagaimana konsep operasi aritmatika dan operator assignment dapat diimplementasikan dalam sebuah aplikasi berbasis *Graphical User Interface* menggunakan *Java Swing*. Melalui pembuatan form input, pemilihan operator, serta penampilan hasil secara interaktif, mahasiswa memahami bahwa logika program tidak hanya bergantung pada perhitungan matematis semata, tetapi juga pada validasi data, penanganan kesalahan, serta interaksi pengguna. Penggunaan metode khusus seperti pesanPeringatan() dan pesanError() membantu menciptakan aplikasi yang lebih aman dan nyaman digunakan, karena program mampu memberikan respons yang jelas ketika terjadi kesalahan input atau operasi yang tidak valid, seperti pembagian atau modulo dengan nol. Selain itu, mahasiswa dapat melihat secara langsung bagaimana desain antarmuka, komponen-komponen GUI, dan logika perhitungan saling terintegrasi dalam sebuah aplikasi yang utuh. Secara keseluruhan, praktikum ini memperkuat pemahaman konsep dasar operator dalam Java sekaligus mengenalkan praktik pembuatan aplikasi GUI yang lebih terstruktur dan interaktif.