LAPORAN PRAKTIKUM
PEMOGRAMAN ALGORITMA PEMOGRAMAN
PEMOGRAMAN GUI 1

disusun Oleh:
LEXI MULIA YUNASPI
2511531006

Dosen Pengampu: DR. Wahyudi, S.T, M.

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Laporan praktikum ini disusun sebagai bentuk pertanggungjawaban atas pelaksanaan kegiatan praktikum mata kuliah Algoritma Pemograman pekan 8 yang membahas tentang Pemograman GUI di Java. Melalui laporan ini penulis dapat lebih memahami materi praktikum dan dengan penulisan laporan ini dapat melatih ketelitian, keteraturan, serta kemampuan menulis sesuai kaidah akademik pada tingkat dasar. Dengan demikian, laporan praktikum yang dihasilkan dapat berfungsi sebagai media pembelajaran, dokumentasi kegiatan, sekaligus sarana untuk melatih keterampilan menulis ilmiah yang akan bermanfaat dalam pembelajaran selanjutnya.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis sangat terbuka terhadap segala bentuk kritik dan saran yang bersifat membangun, demi perbaikan kualitas laporan maupun pemahaman penulis di laporan berikutnya.

Padang, 2025

Lexi Mulia Yunaspi

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam dunia pemrograman, khususnya pada pengembangan aplikasi berbasis desktop, penggunaan *Graphical User Interface* (GUI) sangat penting untuk mempermudah interaksi antara pengguna dan sistem. Java, sebagai bahasa pemrograman yang bersifat *platform-independent*, menyediakan *library Swi*ng yang memungkinkan pembuatan antarmuka grafis secara intuitif dan fleksibel. Salah satu penerapan sederhana namun fundamental dari GUI adalah pembuatan kalkulator atau aplikasi perhitungan aritmatika dasar.

Praktikum ini bertujuan untuk mengimplementasikan sebuah aplikasi GUI sederhana menggunakan Java *Swing* yang mampu melakukan operasi aritmatika dasar seperti penjumlahan, pengurangan, perkalian, dan pembagian. Aplikasi ini dirancang dengan tampilan yang *user-friendly*, meliputi komponen-komponen seperti J*TextField* untuk input angka, J*ComboBox* untuk memilih operator, dan J*Button* untuk memproses perhitungan. Selain itu, aplikasi juga dilengkapi dengan fitur validasi input dan penanganan error, seperti pemberitahuan jika field kosong atau jika input bukan angka, serta pesan peringatan jika terjadi pembagian dengan nol. Melalui praktikum ini, mahasiswa diharapkan mampu memahami cara kerja *event-driven programming* dalam Java *Swing* di mana program bereaksi berdasarkan tindakan pengguna, mengelola komponen GUI, serta menerapkan logika pemrograman untuk menangani berbagai kondisi input dan output. Selain itu, praktikum ini juga menjadi dasar bagi pengembangan aplikasi GUI yang lebih kompleks, sekaligus meningkatkan kemampuan dalam merancang antarmuka yang responsif dan mudah digunakan pengguna.

2.1 Tujuan Praktikum

1. Memahami konsep dasar GUI dalam pemrograman, termasuk komponen seperti tombol, label, textbox, dan event handling.
2. Mampu membuat antarmuka pengguna yang interaktif menggunakan library atau framework (misalnya Java Swing, JavaFX, atau lainnya).
3. Melatih kemampuan menghubungkan logika program dengan elemen GUI, seperti menangani input, output, dan aksi pengguna.
4. Mengembangkan aplikasi sederhana berbasis GUI sebagai implementasi dari konsep OOP dan event-driven programming.
5. Meningkatkan kemampuan perancangan antarmuka agar lebih user-friendly dan fungsional.

3.1 Manfaat Praktikum

1. Mahasiswa dapat membuat aplikasi dengan tampilan visual, bukan hanya berbasis teks.
2. Mempermudah pemahaman bagaimana aplikasi dunia nyata bekerja, karena sebagian besar aplikasi modern menggunakan GUI.
3. Melatih kreativitas desain antarmuka agar lebih menarik dan mudah digunakan.
4. Meningkatkan kemampuan problem solving terutama dalam menangani event, validasi input, dan logika interaksi.
5. Menambah pengalaman yang berguna untuk pengembangan aplikasi desktop, sehingga relevan untuk membuat tugas.

BAB II
PEMBAHASAN

2.1 Praktikum “OperatorAritmatikaGUI”

```
1  package Pekan8_2511531006;
2
3  import java.awt.Color;
19
20 public class OperatorAritmatikaGUI_2511531006 extends JFrame {
21
22     private static final long serialVersionUID = 1L;
23     private JPanel contentPane;
24     private JTextField txtBil1;
25     private JTextField txtBil2;
26     private JTextField txtHasil;
27
28
29     private void pesanPeringatan(String pesan) {
30         JOptionPane.showMessageDialog(this, pesan, "Peringatan", JOptionPane.WARNING_MESSAGE);
31     }
32
33
34     private void pesanError(String pesan) {
35         JOptionPane.showMessageDialog(this, pesan, "Kesalahan", JOptionPane.ERROR_MESSAGE);
36     }
37
38     /**
39      *
40      */
41     public static void main(String[] args) {
42         EventQueue.invokeLater(new Runnable() {
43             public void run() {
44                 try {
45                     OperatorAritmatikaGUI_2511531006 frame = new OperatorAritmatikaGUI_2511531006();
46                     frame.setVisible(true);
47                 } catch (Exception e) {
48                     e.printStackTrace();
49                 }
50             }
51         });
52     }
53
54     /**
55      * Constructor GUI
56      */
57     public OperatorAritmatikaGUI_2511531006() {
58         setResizable(false);
59         setTitle("OPERATOR ARITMATIKA");
60         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
61         setBounds(100, 100, 350, 320);
62
63         contentPane = new JPanel();
64         contentPane.setBackground(new Color(255, 255, 128));
65         contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
66         setContentPane(contentPane);
67         contentPane.setLayout(null);
68
69         JLabel lblJudul = new JLabel("OPERATOR ARITMATIKA");
70         lblJudul.setFont(new Font("Tahoma", Font.BOLD, 16));
71         lblJudul.setBounds(60, 10, 250, 30);
72         contentPane.add(lblJudul);
73
74         JLabel lblBil1 = new JLabel("Bilangan 1");
75         lblBil1.setBounds(20, 60, 100, 14);
76         contentPane.add(lblBil1);
77
78         JLabel lblBil2 = new JLabel("Bilangan 2");
79         lblBil2.setBounds(20, 95, 100, 14);
80         contentPane.add(lblBil2);
81
82         JLabel lblOp = new JLabel("Operator");
```

```
        lblOp.setBounds(20, 130, 100, 14);
        contentPane.add(lblOp);

        JLabel lblHasil = new JLabel("Hasil");
        lblHasil.setBounds(20, 190, 100, 14);
        contentPane.add(lblHasil);

        txtBil1 = new JTextField();
        txtBil1.setHorizontalAlignment(SwingConstants.CENTER);
        txtBil1.setBounds(120, 55, 80, 25);
        contentPane.add(txtBil1);

        txtBil2 = new JTextField();
        txtBil2.setHorizontalAlignment(SwingConstants.CENTER);
        txtBil2.setBounds(120, 90, 80, 25);
        contentPane.add(txtBil2);

        JComboBox<String> cbOperator = new JComboBox<>();
        cbOperator.setModel(new DefaultComboBoxModel<>(new String[] { "+", "-", "*", "/", "%" }));
        cbOperator.setBounds(120, 125, 80, 25);
        contentPane.add(cbOperator);

        txtHasil = new JTextField();
        txtHasil.setHorizontalAlignment(SwingConstants.CENTER);
        txtHasil.setEditable(false);
        txtHasil.setBounds(120, 185, 80, 25);
        contentPane.add(txtHasil);

        JButton btnProses = new JButton("Proses");
        btnProses.setBackground(new Color(255, 128, 64));
        btnProses.setBounds(220, 125, 90, 25);
        contentPane.add(btnProses);


        btnProses.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {

                if (txtBil1.getText().trim().isEmpty()) {
                    pesanPeringatan("Silahkan input Bilangan 1");
                    return;
                }
                if (txtBil2.getText().trim().isEmpty()) {
                    pesanPeringatan("Silahkan input Bilangan 2");
                    return;
                }

                try {
                    int a = Integer.parseInt(txtBil1.getText());
                    int b = Integer.parseInt(txtBil2.getText());
                    int hasil = 0;
                    int op = cbOperator.getSelectedIndex();

                    switch (op) {
                        case 0: hasil = a + b; break;
                        case 1: hasil = a - b; break;
                        case 2: hasil = a * b; break;
                        case 3:
                            if (b == 0) {
                                pesanError("Pembagian dengan 0 tidak diperbolehkan");
                                return;
                            }
                            hasil = a / b;
                            break;
                        case 4:
                            if (b == 0) {
                                pesanError("Modulus dengan 0 tidak diperbolehkan");
                                return;
                            }
                            hasil = a % b;
                            break;
                    }

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

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

Gambar 2.1 Kode Program Praktikum “OperatorAritmatikaGUI”

Program ini dibangun menggunakan WindowBuilder dan Java Swing, sehingga proses pembuatan tampilan GUI menjadi lebih mudah karena komponen dapat disusun secara visual. Aplikasi yang dibuat berfungsi sebagai kalkulator sederhana yang dapat melakukan empat operasi aritmatika dasar: penjumlahan, pengurangan, perkalian, dan pembagian**.**Pengguna cukup memasukkan dua bilangan ke dalam kotak input, kemudian

memilih tombol operasi yang diinginkan. Program akan memproses perhitungan secara otomatis dan menampilkan hasilnya pada field output. Selain itu, aplikasi juga menangani kesalahan sederhana, seperti input kosong atau pembagian dengan nol, sehingga lebih aman dan mudah digunakan.Berikut penjelasan mengenai komponen dan konsep utama yang digunakan dalam program tersebut:

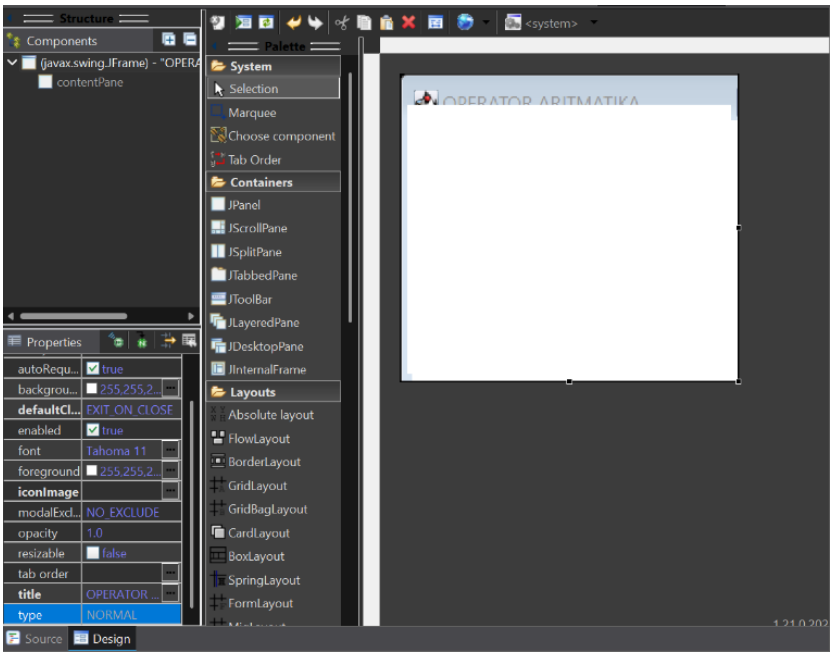
1. Struktur Dasar Program dan Library

Program ini menggunakan beberapa library penting dari Java:

a. a.javax.swing → untuk membuat komponen GUI seperti JFrame, JLabel, JTextField, JComboBox, JButton, dan JOptionPane.
b. java.awt → untuk mengatur warna (*color*), huruf (*font*), dan (ActionListener, ActionEvent).
c. java.awt.event → khusus untuk menangani aksi pengguna seperti klik tombol.
d. javax.swing.border → untuk memberi batas atau border pada panel.

2. Desain dan Inisialisasi Jendela Utama

Secara visual, jendela aplikasi memiliki:



Gambar 2.1 Visual Jendela Aplikasi

- Judul: “OPERATOR ARITMATIKA”

- Ukuran tetap: lebar 330 piksel, tinggi 300 piksel, diposisikan di koordinat layar (100, 100).
- Latar belakang putih dengan border tipis pada panel utama (contentPane).
- Tidak bisa di-*resize* (setResizable(false)) untuk menjaga tata letak tetap sesuai desain.

Secara kode, konfigurasi ini diatur di bagian source class:

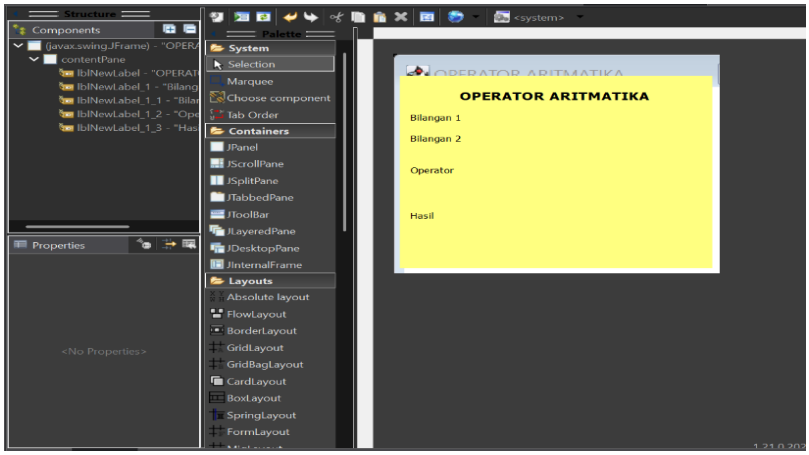
```
setResizable(false);
setTitle("OPERATOR ARITMATIKA");
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(100, 100, 350, 320);
```

Gambar 2.2 Kode Program Jendela Aplikasi Panel.

Panel utama di dalam aplikasi diatur agar memiliki warna latar belakang putih menggunakan metode setBackground(). Selain itu, panel tersebut juga diberi border melalui setBorder() agar tampilannya lebih rapi dan komponen-komponen di dalamnya terlihat terpisah dengan jelas. Pengaturan ini membantu membuat antarmuka lebih bersih, terstruktur, dan nyaman dilihat oleh pengguna.

3. Komponen Antarmuka dan Implementasinya

A. Label Teks

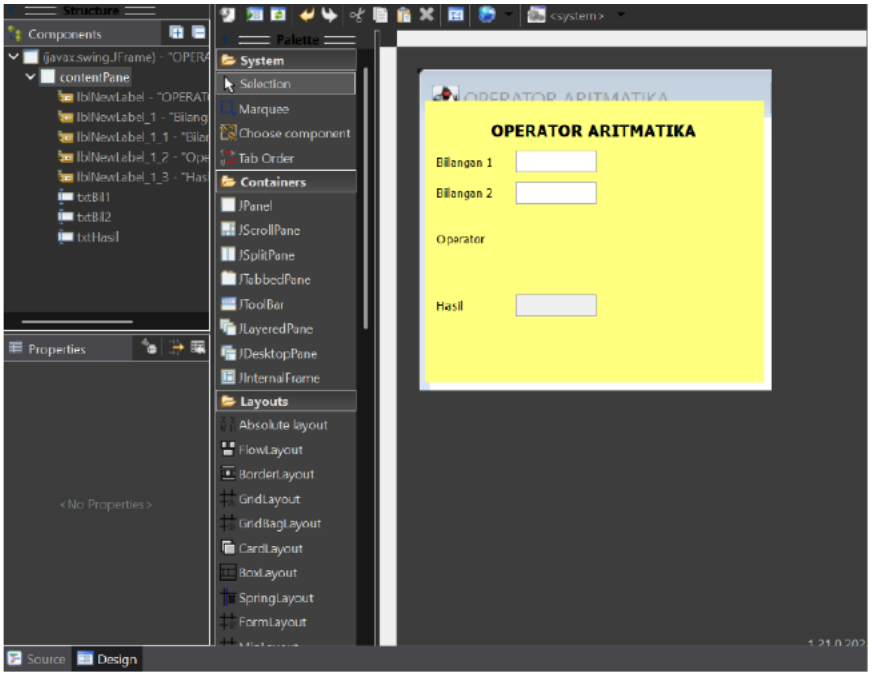


Gambar 2.3 Label Teks

- Pada gambar terlihat beberapa JLabel, yaitu:
- **“Bilangan 1”** → sebagai penanda untuk input angka pertama.
- **“Bilangan 2”** → sebagai penanda untuk input angka kedua.
- **“Operator”** → menunjukkan tempat memilih atau memasukkan jenis operasi (penjumlahan, pengurangan, dll).
- **“Hasil”** → sebagai keterangan untuk menampilkan hasil perhitungan.

Kode : Masing-masing label dibuat sebagai objek JLabel dan ditempatkan dengan setBounds(x, y, width, height).

B. Kotak Input Angka (JTextField)



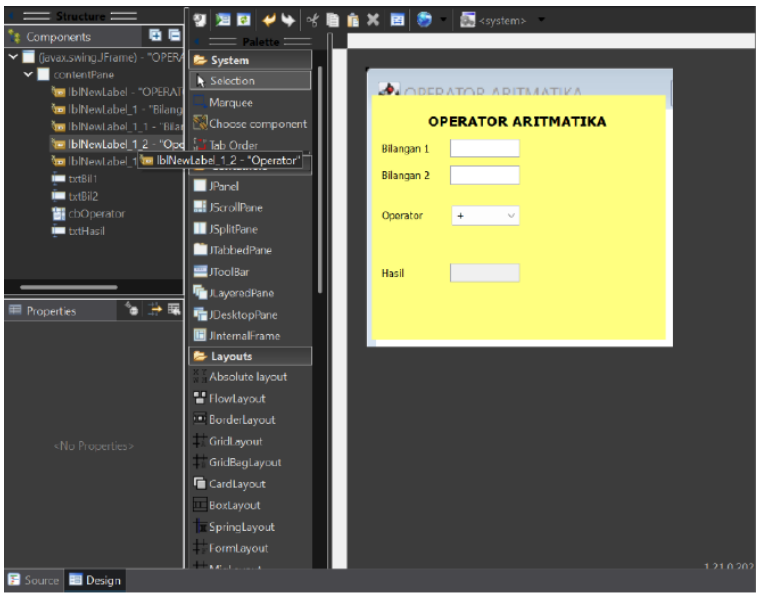
Gambar 2.4 Kotak Input Angka

Gambar tersebut menunjukkan tampilan antarmuka aplikasi Java Swing yang dibuat menggunakan WindowBuilder pada mode Design. Pada sisi kanan terlihat jendela desain GUI dengan latar panel berwarna kuning yang berisi judul “OPERATOR ARITMATIKA” serta beberapa label dan text field seperti Bilangan 1, Bilangan 2, Operator, dan Hasil. Elemen-elemen ini digunakan untuk menerima input angka, memilih operator aritmatika, serta menampilkan hasil perhitungan.Di sisi kiri tampak

panel Structure yang menunjukkan daftar komponen yang digunakan dalam JFrame, seperti lblNewLabel, txtBil1, txtBil2, dan txtHasil. Bagian tengah kiri berisi **Palette**, yaitu kumpulan komponen yang bisa diseret ke tampilan, seperti label, panel, tombol, hingga berbagai jenis layout. Gambar ini menggambarkan proses pembuatan antarmuka aplikasi kalkulator sederhana menggunakan library Swing, di mana setiap komponen ditempatkan secara visual dan dapat diatur propertinya melalui editor desain.

C. Combo Box Operator (JComboBox)

Desain: Drop-down menu berisi empat operator: +, -, *, /. Ditempatkan di sebelah label "Operator".



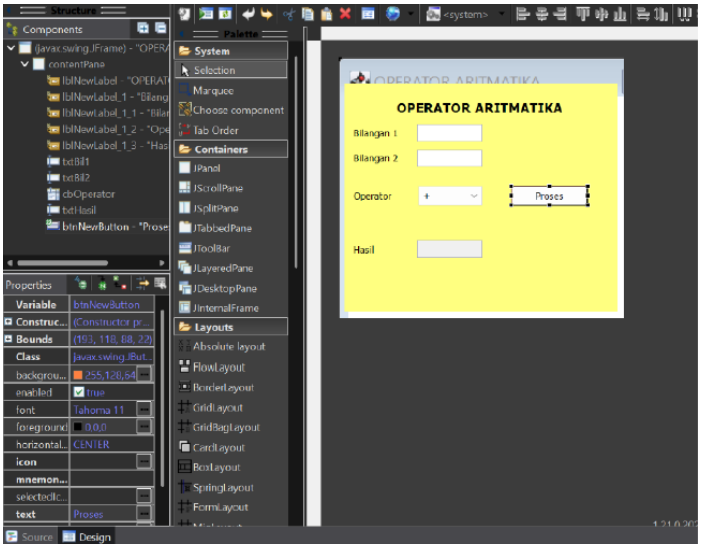
Gambar 2.5 Combo Box Operator

Kode: Dibuat menggunakan DefaultComboBoxModel dan diisi secara eksplisit:

```
cbOperator.setModel(new DefaultComboBoxModel<>(new String[] { "+", "-", "*", "/", "%" }));
```

Gambar 2.6 Kode Program Combo Box Operator

D. Tombol "Proses" (JButton)

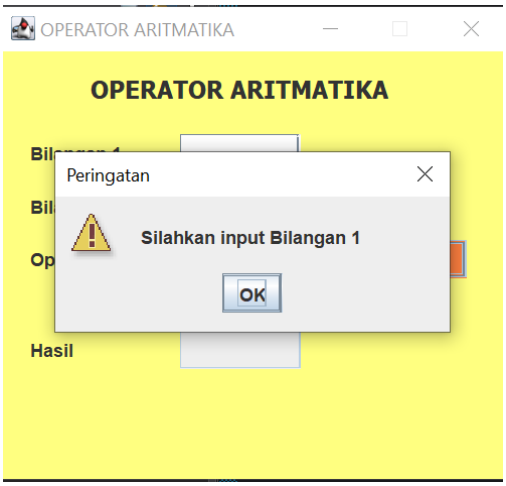


Gambar 2.7 Tombol Proses

a. Desain: Tombol berwarna oranye-biru tua (RGB 255,128,64) dengan teks putih, ditempatkan di sisi kanan, sejajar dengan combo box.

4. Logika Pemrosesan dan Respons Interaksi Pengguna

Ketika tombol “Proses” diklik, sistem menjalankan rangkaian validasi dan perhitungan berikut:



Gambar 2.8 Validasi Input Kosong

Gambar tersebut menunjukkan **popup peringatan** yang muncul ketika kolom Bilangan 1 dibiarkan kosong. Aplikasi menampilkan pesan “Silahkan input Bilangan

1” untuk memaksa pengguna mengisi data sebelum melanjutkan perhitungan. Dialog ini berfungsi sebagai validasi input agar program tidak mengalami error

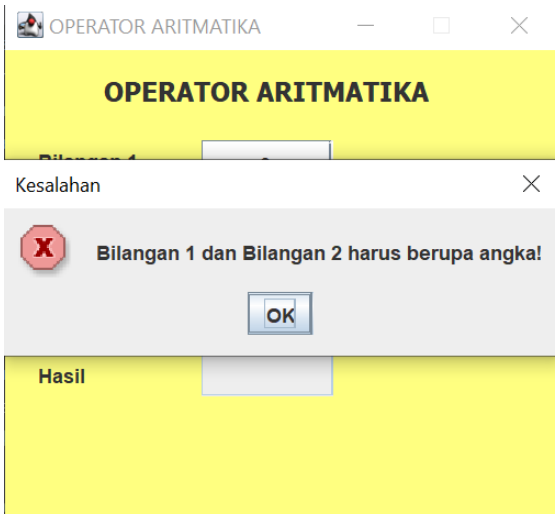
Kode :

```
if (txtBil1.getText().trim().isEmpty()) {
    pesanPeringatan("Silahkan input Bilangan 1");
    return;
}
```

Gambar 2.9 Kode Program Validasi Input Kosong

Validasi Format Angka

Desain Respons: Jika pengguna memasukkan huruf atau simbol, muncul pesan error.



Gambar 2.10 Validasi Format Angka

Kode :

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

Gambar 2.11 Kode Program Validasi Format Angka

Perhitungan Dinamis

Gambar 2.12 Perhitungan Dinamis

Kode :

```
case 0: hasil = a + b; break;
case 1: hasil = a - b; break;
case 2: hasil = a * b; break;
case 3:
    if (b == 0) {
        pesanError("Bilangan 2 tidak boleh 0");
        return;
    }
    hasil = a / b;
    break;
case 4:
    if (b == 0) {
        pesanError("Modulus dengan 0 tidak diperbolehkan");
        return;
    }
    hasil = a % b;
    break;
```

Gambar 2.1 3Kode program Perhitungan Dinamis

Kode tersebut adalah bagian dari switch-case yang menjalankan operasi aritmatika sesuai pilihan. Case 0–2 melakukan penjumlahan, pengurangan, dan perkalian langsung. Pada case 3 (pembagian) dan case 4 (modulus), program terlebih dahulu mengecek apakah *b* bernilai 0. Jika ya, akan muncul pesan error karena pembagian atau modulus dengan nol tidak boleh dilakukan, lalu proses dihentikan. Jika tidak, barulah hasil dihitung. Setiap case ditutup dengan `break` agar tidak berlanjut ke case lain.

BAB III

PENUTUP

3.1 Kesimpulan

Berdasarkan hasil praktikum pembuatan aplikasi GUI menggunakan Java Swing, dapat disimpulkan bahwa mahasiswa mampu memahami dan mengimplementasikan komponen–komponen dasar antarmuka, seperti JLabel, JTextField, JButton, dan struktur logika seperti if serta switch-case dalam sebuah program. Praktikum ini juga menunjukkan bahwa penanganan kesalahan (error handling), seperti mencegah pembagian atau modulus dengan angka nol, sangat penting untuk menjaga program tetap berjalan dengan aman dan sesuai aturan yang benar. Selain itu, penggunaan WindowBuilder mempermudah proses perancangan tampilan sehingga pengembangan aplikasi menjadi lebih cepat, terstruktur, dan efisien.

3.2 Saran

Untuk praktikum selanjutnya, disarankan agar saya lebih memperdalam pemahaman mengenai validasi input, tata letak (layout management), serta pemisahan logika program dari tampilan (MVC) agar aplikasi lebih rapi dan mudah dikembangkan. Mahasiswa juga sebaiknya mencoba menambahkan fitur-fitur tambahan, seperti desain antarmuka yang lebih interaktif atau penggunaan pesan dialog yang lebih informatif.

DAFTAR PUSTAKA

Deitel, P., & Deitel, H. (2015). *Java: How to Program* (10th ed.). Prentice Hall.

GeeksforGeeks. (2023). *Introduction to Java Swing*. https://www.geeksforgeeks.org/introduction-to-java-swing/

S. W. Surya, Pemrograman Berorientasi Objek dengan Java, Yogyakarta, Indonesia: *Graha Ilmu, 2020*