

LAPORAN PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN
TUGAS ALPRO PEKAN 8
OPERATOR ASSIGNMENT GUI



Disusun Oleh :

KINAYA NOVRYA MANDA

(2511531016)

Kelas B Informatika

Dosen Pengampu :

DR. WAHYUDI, S.T, M.T

Asisten Praktikum :

MUHAMMAD ZAKI AL HAFIZ

DEPARTEMEN INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Puji syukur penulis sampaikan kehadirat Allah SWT. Salawat dan salam disampaikan kepada Nabi Muhammad SAW. Karena thaufik dan hidayah-Nya, laporan praktikum Java ini dapat diselesaikan dengan baik dan tepat waktu. Laporan ini disusun sebagai salah satu bentuk pertanggungjawaban dari kegiatan praktikum yang telah dilaksanakan, sekaligus sebagai sarana untuk memperdalam pemahaman mengenai konsep dasar serta penerapan bahasa pemrograman Java.

Melalui praktikum ini, penulis memperoleh pengalaman langsung dalam memahami hubungan antara logika perhitungan dan desain antarmuka yang saling mendukung menggunakan Java. Diharapkan laporan ini dapat memberikan gambaran yang jelas mengenai materi yang telah dipelajari serta hasil dari percobaan yang dilakukan selama praktikum berlangsung.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna, baik dari segi isi maupun penyajiannya. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan demi perbaikan di masa yang akan datang.

Akhir kata, penulis mengucapkan terima kasih kepada dosen pengampu, asisten laboratorium, serta semua pihak yang telah memberikan bimbingan, arahan, dan dukungan sehingga laporan praktikum ini dapat tersusun. Semoga laporan ini dapat bermanfaat bagi pembaca, khususnya dalam memahami dasar-dasar pemrograman Java.

Padang, November 2025

Penulis

DAFTAR ISI

| | |
|--------------------------------|------------|
| KATA PENGANTAR..... | i |
| DAFTAR ISI..... | ii |
| DAFTAR LAMPIRAN..... | iii |
| BAB 1 PENDAHULUAN | |
| 1.1 Latar Belakang..... | 1 |
| 1.2 Tujuan Praktikum..... | 2 |
| 1.3 Manfaat Praktikum..... | 2 |
| BAB 2 PEMBAHASAN | |
| 2.1 Kode Program..... | 3 |
| 2.1.1 Uraian Kode Program..... | 5 |
| 2.2 Langkah Kerja..... | 7 |
| 2.3 Flowchart..... | 14 |
| 2.4 Pseudocode..... | 15 |
| 2.5 Analisis Hasil..... | 18 |
| BAB 3 KESIMPULAN | |
| 3.1 Hasil Praktikum..... | 19 |
| 3.1.1 Output Berhasil..... | 19 |
| 3.1.2 Output Peringatan..... | 20 |
| 3.1.3 Output Kesalahan..... | 21 |
| 3.2 Saran Pengembangan..... | 22 |
| DAFTAR KEPUSTAKAAN..... | 23 |

DAFTAR LAMPIRAN

| | |
|--|----|
| Kode Program 2.1 (Kode Program OperatorAssignmentGUI)..... | 3 |
| Gambar 2.1 (Langkah 1 OperatorAssignmentGUI)..... | 7 |
| Gambar 2.2 (Langkah 2 OperatorAssignmentGUI)..... | 7 |
| Gambar 2.3 (Langkah 3 OperatorAssignmentGUI)..... | 8 |
| Gambar 2.4 (Langkah 4 OperatorAssignmentGUI)..... | 8 |
| Gambar 2.5 (Langkah 5 OperatorAssignmentGUI)..... | 9 |
| Gambar 2.6 (Langkah 6 OperatorAssignmentGUI)..... | 10 |
| Gambar 2.7 (Langkah 7 OperatorAssignmentGUI)..... | 10 |
| Gambar 2.8 (Langkah 8 OperatorAssignmentGUI)..... | 11 |
| Gambar 2.9 (Langkah 9 OperatorAssignmentGUI)..... | 11 |
| Gambar 2.10 (Langkah 10 OperatorAssignmentGUI)..... | 12 |
| Gambar 2.11 (Langkah 11 OperatorAssignmentGUI)..... | 12 |
| Gambar 2.12 (Langkah 12 OperatorAssignmentGUI)..... | 13 |
| Gambar 2.13 (Langkah 13 OperatorAssignmentGUI)..... | 13 |
| Gambar 2.14 (Flowchart)..... | 14 |
| Gambar 3.1 (Hasil Praktikum Output Berhasil)..... | 19 |
| Gambar 3.2 (Hasil Praktikum Output Peringatan)..... | 20 |
| Gambar 3.3 (Hasil Praktikum Output Kesalahan)..... | 21 |
| Tabel 2.1 (Pseudocode)..... | 15 |

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Dalam pemrograman, *operator assignment* merupakan salah satu konsep dasar yang digunakan untuk memberikan nilai pada variabel. Operator ini tidak hanya berupa tanda sama dengan (=), tetapi juga mencakup bentuk lain seperti +=, -=, *=, /=, hingga %= . *Operator assignment* membantu programmer menyederhanakan proses pengubahan nilai variabel sehingga kode dapat ditulis lebih efektif dan mudah dipahami. Oleh karena itu, pemahaman tentang cara kerja *operator assignment* sangat penting sebelum mempelajari konsep pemrograman yang lebih kompleks.

Pada proses pembelajaran, mahasiswa tidak hanya dituntut memahami *operator assignment* secara teori, tetapi juga harus mampu mengimplementasikannya ke dalam program. Salah satu cara untuk membuat pembelajaran lebih menarik adalah dengan menggabungkan penggunaan *operator assignment* ke dalam aplikasi yang memiliki tampilan antarmuka visual atau GUI (*Graphical User Interface*). Dengan GUI, mahasiswa dapat melihat secara langsung bagaimana nilai variabel berubah ketika *operator assignment* dijalankan, sehingga pemahaman menjadi lebih jelas dan interaktif.

Melalui praktikum ini, mahasiswa dilatih untuk menghubungkan konsep *operator assignment* dengan komponen GUI seperti tombol, kotak *input*, dan *output*. Dengan demikian, proses pembelajaran tidak hanya fokus pada logika perhitungan, tetapi juga pengelolaan *input* pengguna serta tampilan hasil secara visual. Praktikum ini diharapkan dapat membantu mahasiswa lebih memahami bagaimana logika program berjalan dan bagaimana nilai variabel dapat diperbarui melalui berbagai macam *operator assignment* dalam sebuah aplikasi sederhana.

1.2 Tujuan Praktikum

Tujuan dilakukannya praktikum ini adalah sebagai berikut :

1. Memahami konsep dasar *operator assignment* dalam pemrograman.
2. Mempelajari penggunaan *operator assignment* pada aplikasi berbasis GUI.
3. Melatih mahasiswa menangani input pengguna dan memprosesnya menggunakan *operator assignment*.
4. Mengembangkan kemampuan membuat aplikasi sederhana dengan alur *input–proses–output*.
5. Menghubungkan logika *operator assignment* dengan *event* yang terjadi pada GUI, seperti klik tombol.

1.3 Manfaat Praktikum

Manfaat dilakukannya praktikum ini adalah sebagai berikut :

1. Mahasiswa memahami cara kerja *operator assignment* secara lebih aplikatif.
2. Meningkatkan kemampuan dalam mengimplementasikan *operator assignment* pada tampilan GUI.
3. Melatih keterampilan dalam membuat aplikasi sederhana yang interaktif dan mudah digunakan.
4. Membantu mahasiswa memahami perubahan nilai variabel secara lebih jelas dan visual.
5. Menjadi dasar untuk mempelajari konsep pemrograman tingkat lanjut, seperti manipulasi data dan struktur logika lebih kompleks.

BAB II

PEMBAHASAN

2.1 Kode Program

```

1  package Pekan8_2511531016;
2
3  import java.awt.EventQueue;
18
19  public class OperatorAssignmentGUI_2511531016 extends JFrame {
20
21      private static final long serialVersionUID = 1L;
22      private JPanel contentPane;
23      private JTextField txtBill1;
24      private JTextField txtBill2;
25      private JTextField txtHasil;
26
27      private void pesanPeringatan(String pesan) {
28          JOptionPane.showMessageDialog(this, pesan, "Peringatan", JOptionPane.WARNING_MESSAGE);
29      }
30      private void pesanError(String pesan) {
31          JOptionPane.showMessageDialog(this, pesan, "Kesalahan", JOptionPane.ERROR_MESSAGE);
32      }
33
34      /**
35       * Launch the application.
36       */
37      public static void main(String[] args) {
38          EventQueue.invokeLater(new Runnable() {
39              public void run() {
40                  try {
41                      OperatorAssignmentGUI_2511531016 frame = new OperatorAssignmentGUI_2511531016();
42                      frame.setVisible(true);
43                  } catch (Exception e) {
44                      e.printStackTrace();
45                  }
46              }
47          });
48      }
49
50      /**
51       * Create the frame.
52       */
53      public OperatorAssignmentGUI_2511531016() {
54          setTitle("OPERATOR ASSIGNMENT");
55          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
56          setBounds(100, 100, 450, 300);
57          contentPane = new JPanel();
58          contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
59          setContentPane(contentPane);
60          contentPane.setLayout(null);
61
62          JLabel lblNewLabel = new JLabel("OPERATOR ASSIGNMENT");
63          lblNewLabel.setFont(new Font("Tempus Sans ITC", Font.BOLD, 14));
64          lblNewLabel.setBounds(139, 11, 194, 14);
65          contentPane.add(lblNewLabel);
66
67          JLabel lblNewLabel_1 = new JLabel("Bilangan 1");
68          lblNewLabel_1.setBounds(10, 44, 71, 14);
69          contentPane.add(lblNewLabel_1);
70
71          JLabel lblNewLabel_2 = new JLabel("Bilangan 2");
72          lblNewLabel_2.setBounds(10, 72, 71, 14);
73          contentPane.add(lblNewLabel_2);
74
75          JLabel lblNewLabel_3 = new JLabel("Operator");
76          lblNewLabel_3.setBounds(10, 113, 71, 14);
77          contentPane.add(lblNewLabel_3);
78
79          JLabel lblNewLabel_4 = new JLabel("Hasil");
80          lblNewLabel_4.setBounds(10, 167, 71, 14);

```

```

81     contentPane.add(lblNewLabel_4);
82
83     txtBil1 = new JTextField();
84     txtBil1.setHorizontalAlignment(SwingConstants.CENTER);
85     txtBil1.setBounds(86, 41, 48, 20);
86     contentPane.add(txtBil1);
87     txtBil1.setColumns(10);
88
89     txtBil2 = new JTextField();
90     txtBil2.setHorizontalAlignment(SwingConstants.CENTER);
91     txtBil2.setBounds(86, 69, 48, 20);
92     contentPane.add(txtBil2);
93     txtBil2.setColumns(10);
94
95     txtHasil = new JTextField();
96     txtHasil.setHorizontalAlignment(SwingConstants.CENTER);
97     txtHasil.setEnabled(false);
98     txtHasil.setEditable(false);
99     txtHasil.setBounds(86, 164, 48, 20);
100    contentPane.add(txtHasil);
101    txtHasil.setColumns(10);
102
103    JComboBox cbOperator = new JComboBox();
104    cbOperator.setModel(new DefaultComboBoxModel(new String[] { "+", "-", "*", "/", "% }));
105    cbOperator.setBounds(86, 109, 48, 22);
106    contentPane.add(cbOperator);
107
108    JButton btnNewButton = new JButton("Proses");
109    btnNewButton.addActionListener(new ActionListener() {
110        int hasil;
111        public void actionPerformed(ActionEvent e) {
112            if(txtBil1.getText().trim().isEmpty()) {
113                pesanPeringatan("Bilangan 1 harus diisi");
114
115            } else if (txtBil2.getText().trim().isEmpty()) {
116                pesanPeringatan("Bilangan 1 harus diisi");
117            } else {
118                try {
119                    int a = Integer.parseInt(txtBil1.getText());
120                    int b = Integer.parseInt(txtBil2.getText());
121                    int c = cbOperator.getSelectedIndex(); //operator
122                    if (c==0) {
123                        a+=b;
124                        hasil = a;
125                    }
126                    if (c==1) {
127                        a-=b;
128                        hasil = a;
129                    }
130                    if (c==2) {
131                        a*=b;
132                        hasil = a;
133                    }
134                    if (c==3) {
135                        a/=b;
136                        hasil = a;
137                    }
138                    if (c==4) {
139                        a%=b;
140                        hasil = a;
141                    }
142                } catch (NumberFormatException ex) {
143                    pesanError("Bilangan 1 dan Bilangan 2 harus angka");
144                }
145            }
146            txtHasil.setText(String.valueOf(hasil));
147        }

```



```

148
149     });
150     btnNewButton.setBounds(166, 109, 88, 22);
151     contentPane.add(btnNewButton);
152
153 }
154
155 }
156

```

Kode Program 2.1 (Kode Program OperatorAssignmentGUI)

2.1.1 Uraian Kode Program :

1. *private void pesanPeringatan(String pesan) {*

```

    JOptionPane.showMessageDialog(this,      pesan,      "Peringatan",
    JOptionPane.WARNING_MESSAGE); }

```

→ Method ini digunakan untuk menampilkan kotak dialog peringatan. Setiap kali *method* dipanggil, program akan memunculkan pesan dengan icon *warning* dan judul “Peringatan”.

2. *private void pesanError(String pesan) {*

```

    JOptionPane.showMessageDialog(this,      pesan,      "Kesalahan",
    JOptionPane.ERROR_MESSAGE); }

```

→ Method ini menampilkan kotak dialog *error*. Dialog yang muncul bertipe *ERROR_MESSAGE*, biasanya untuk memberi tahu bahwa input tidak valid.

3. Variabel *int hasil*;

→ Variabel ini digunakan untuk menyimpan hasil operasi assignment yang diproses nantinya.

4. *public void actionPerformed(ActionEvent e) {*

→ Method ini berjalan ketika tombol pada GUI ditekan. Semua proses perhitungan akan dilakukan di dalam *method* ini.

5. *if (txtBill1.getText().trim().isEmpty()) {*

pesanPeringatan("Bilangan 1 harus diisi"); }

→ Mengecek apakah input bilangan pertama kosong. Jika kosong, tampil pesan peringatan.

6. *int a = Integer.parseInt(txtBill1.getText()); int b = Integer.parseInt(txtBil2.getText());*

→ Mengkonversi teks dari input menjadi tipe data *integer* agar bisa dihitung.

7. *int c = cbOperator.getSelectedIndex();*

→ Mengambil indeks operator yang dipilih pada *ComboBox*, misalnya 0 = tambah, 1 = kurang, 2 = kali, dst.

8. *} catch (NumberFormatException ex) {*

pesanError("Bilangan 1 dan Bilangan 2 harus angka"); }

→ Jika input bukan angka (misalnya huruf atau simbol), program langsung memunculkan pesan *error*.

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

→ Setelah perhitungan selesai, hasil operasi ditampilkan pada *field* hasil dalam bentuk teks.

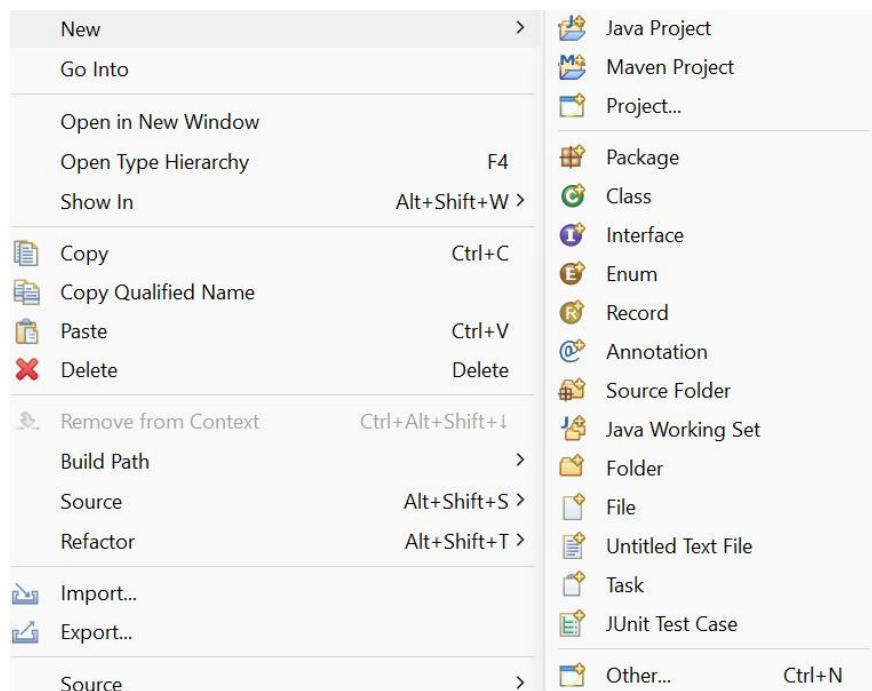
2.2 Langkah Kerja

1. Membuat *package* bernama *Pekan8_2511531016*.



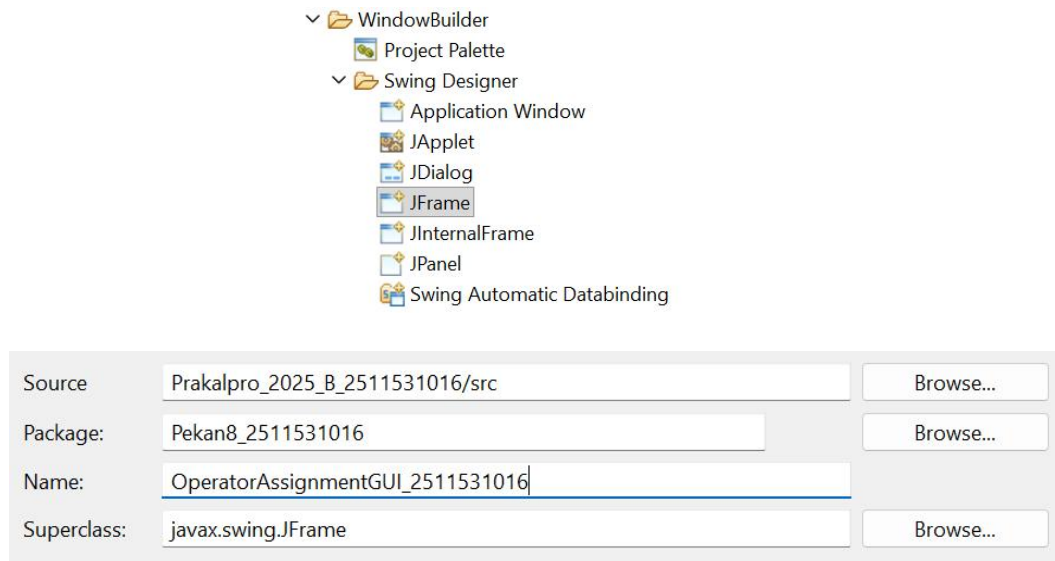
Gambar 2.1 (Langkah 1 OperatorAssignmentGUI)

2. Klik kanan pada package dan pilih *new* lalu klik *other*.



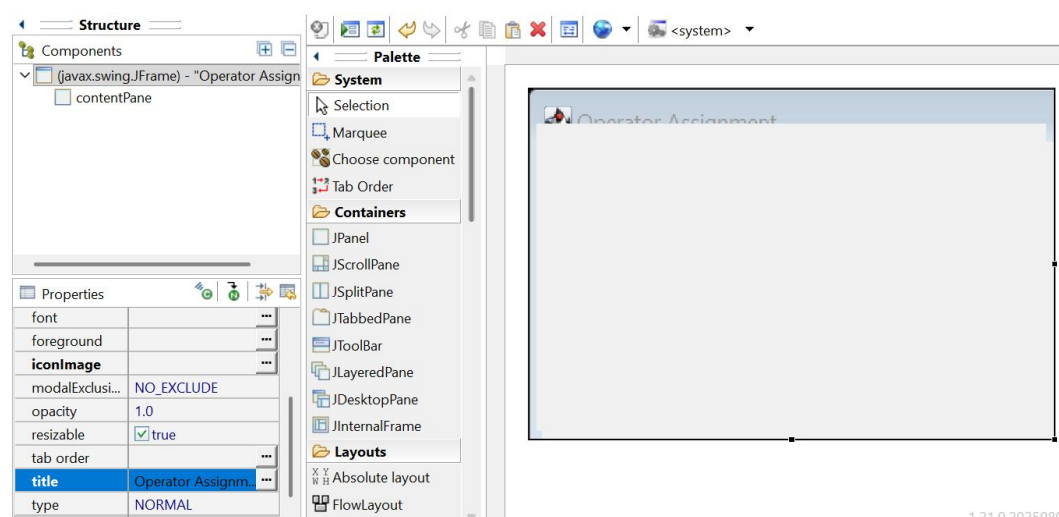
Gambar 2.2 (Langkah 2 OperatorAssignmentGUI)

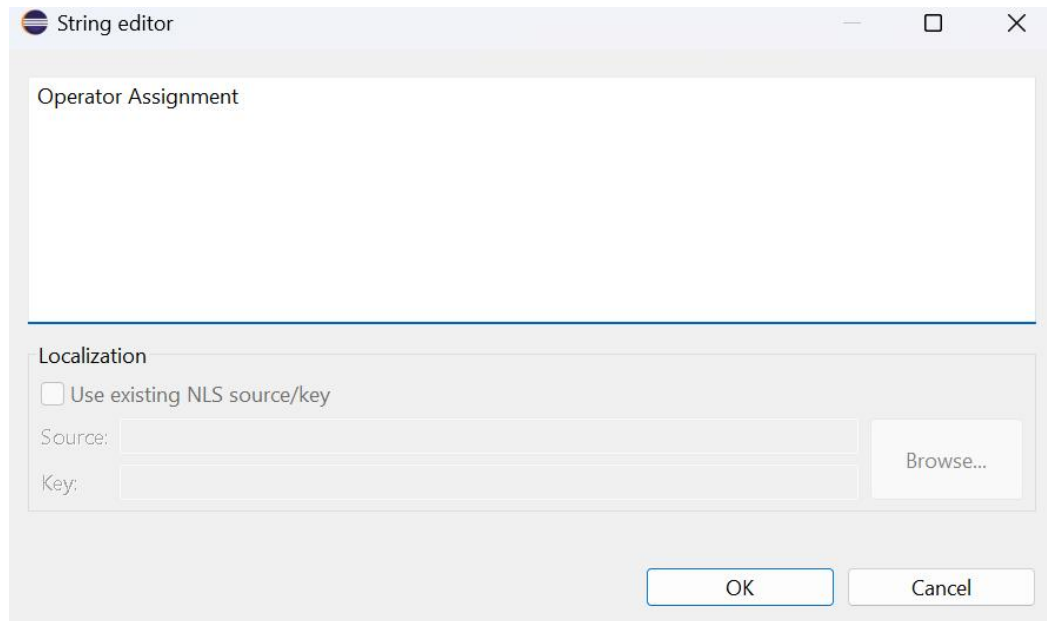
3. Klik *JFrame* pada menu *WindowBuilder* dan beri nama *OperatorAssignmentGUI_2511531016*.



Gambar 2.3 (Langkah 3 OperatorAssignmentGUI)

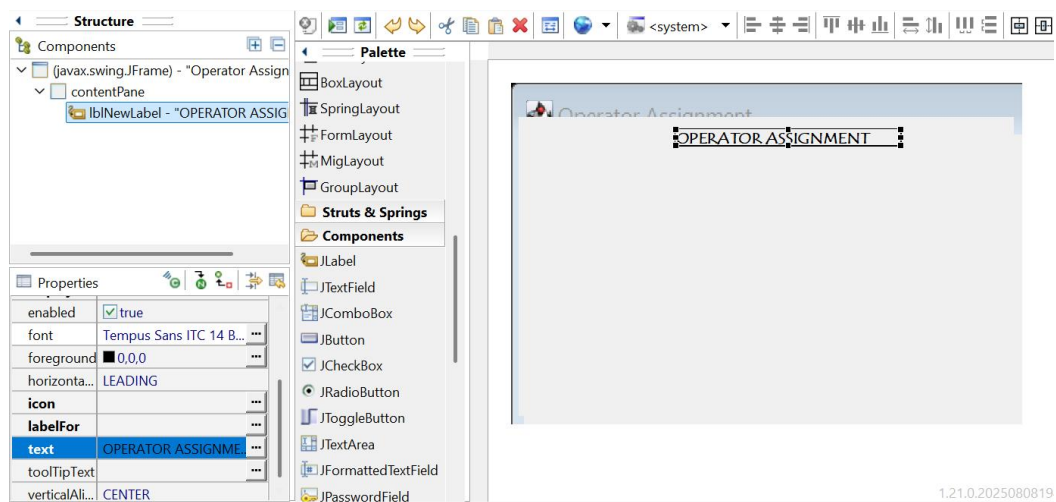
4. Akan muncul tampilan seperti ini, lalu buat judul (*title*) yaitu Operator Assignment. Setelah itu, klik *contentPane* lalu ubah *layout* nya menjadi *absolute layout*.





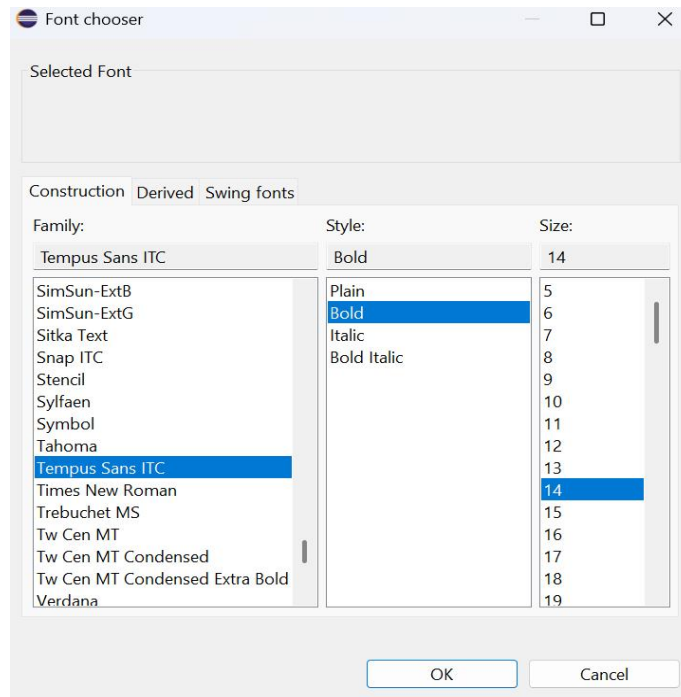
Gambar 2.4 (Langkah 4 OperatorAssignmentGUI)

5. Pada menu *component* klik *JLabel* lalu buat *NewLabel* berjudul “OPERATOR ASSIGNMENT”.



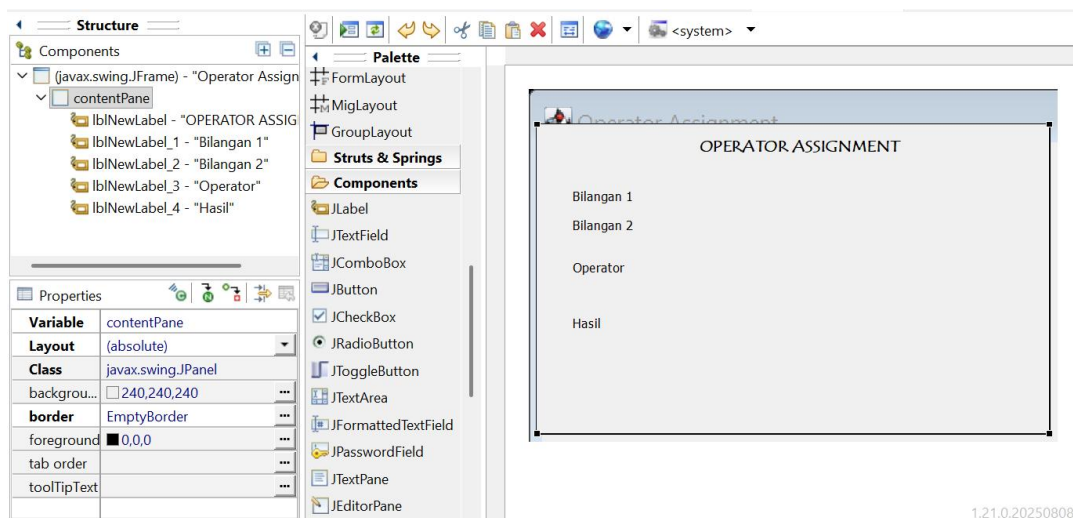
Gambar 2.5 (Langkah 5 OperatorAssignmentGUI)

6. Pilih gaya *font* dan ukuran teks sesuai keinginan.



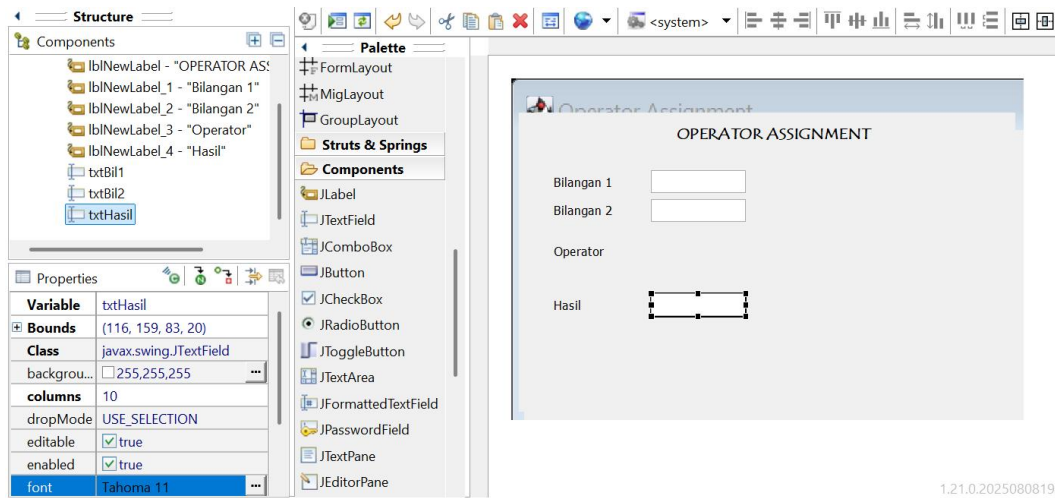
Gambar 2.6 (Langkah 6 OperatorAssignmentGUI)

7. Tambahkan lagi *JLabel* untuk membuat teks kedalam *design* aplikasi yaitu “Bilangan 1”, “Bilangan 2”, “Operator”, dan “Hasil”.



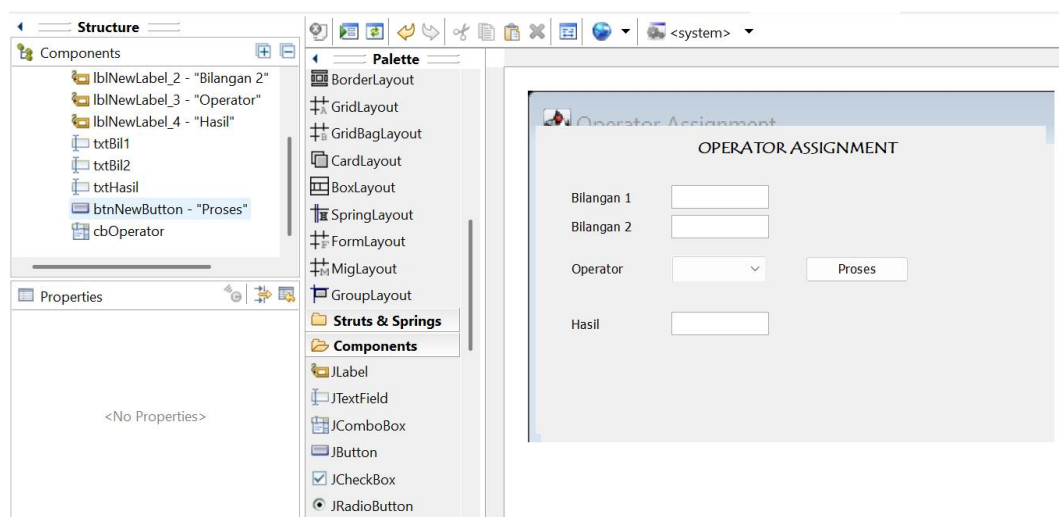
Gambar 2.7 (Langkah 7 OperatorAssignmentGUI)

8. Tambahkan *TextField* disamping kanan label Bilangan 1, Bilangan 2 dan Hasil, lalu ubah variabel nya menjadi “txtBil1”, “txtBil2”, dan “txtHasil”.



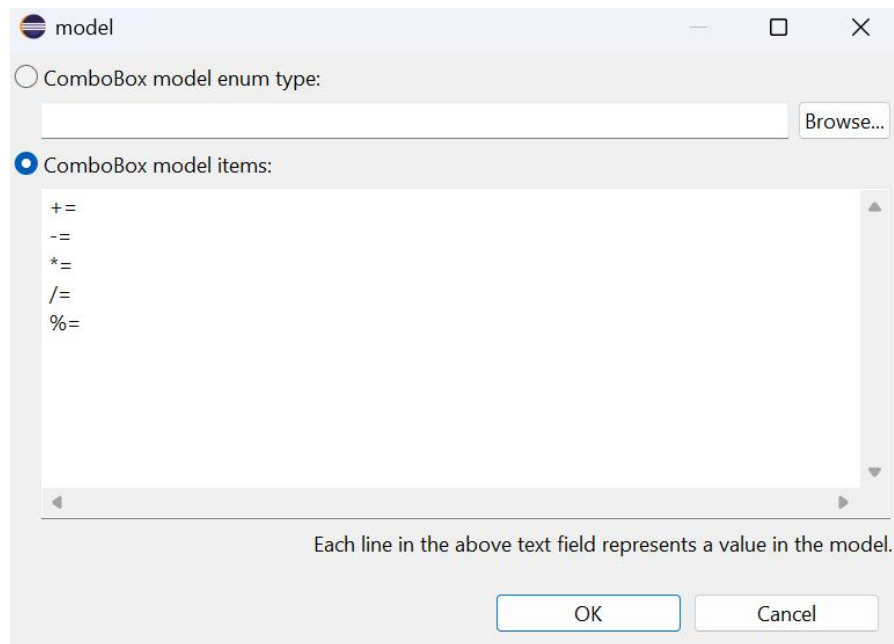
Gambar 2.8 (Langkah 8 OperatorAssignmentGUI)

9. Tambahkan *JComboBox* disamping kanan label Operator, lalu ubah variabel nya menjadi "cbOperator". Setelah itu, tambahkan *JButton* lalu ubah teks menjadi “Proses”.



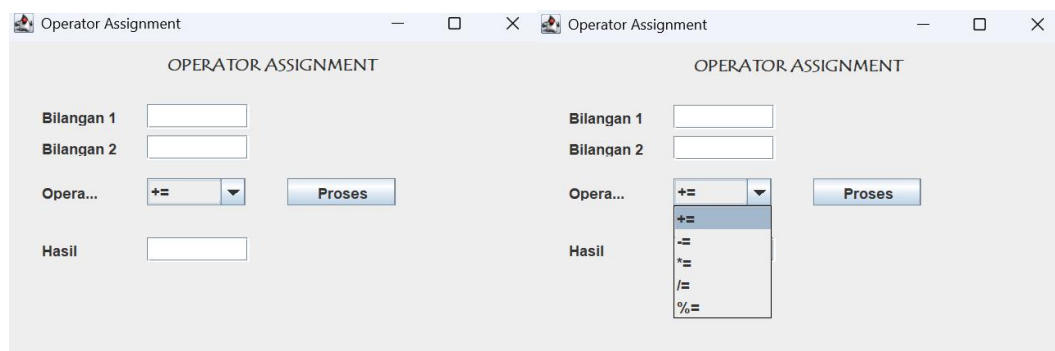
Gambar 2.9 (Langkah 9 OperatorAssignmentGUI)

10. Tambahkan operator-operator aritmatika (+ - * / %) pada model.



Gambar 2.10 (Langkah 10 OperatorAssignmentGUI)

11. Tampilan akan menjadi seperti ini, lalu kita akan lanjutkan ke *source code* untuk menjalankan program.



Gambar 2.11 (Langkah 11 OperatorAssignmentGUI)

12. Tambahkan kode program *pesanPeringatan* dan *pesanKesalahan* seperti dibawah ini jika ada kesalahan input.

```
private void pesanPeringatan(String pesan) {
    JOptionPane.showMessageDialog(this, pesan, "Peringatan", JOptionPane.WARNING_MESSAGE);
}
private void pesanError(String pesan) {
    JOptionPane.showMessageDialog(this, pesan, "Kesalahan", JOptionPane.ERROR_MESSAGE);
}
```

Gambar 2.12 (Langkah 12 OperatorAssignmentGUI)

13. Untuk menjalankan program, buat kode dibawah ini, maka program sudah bisa dijalankan.

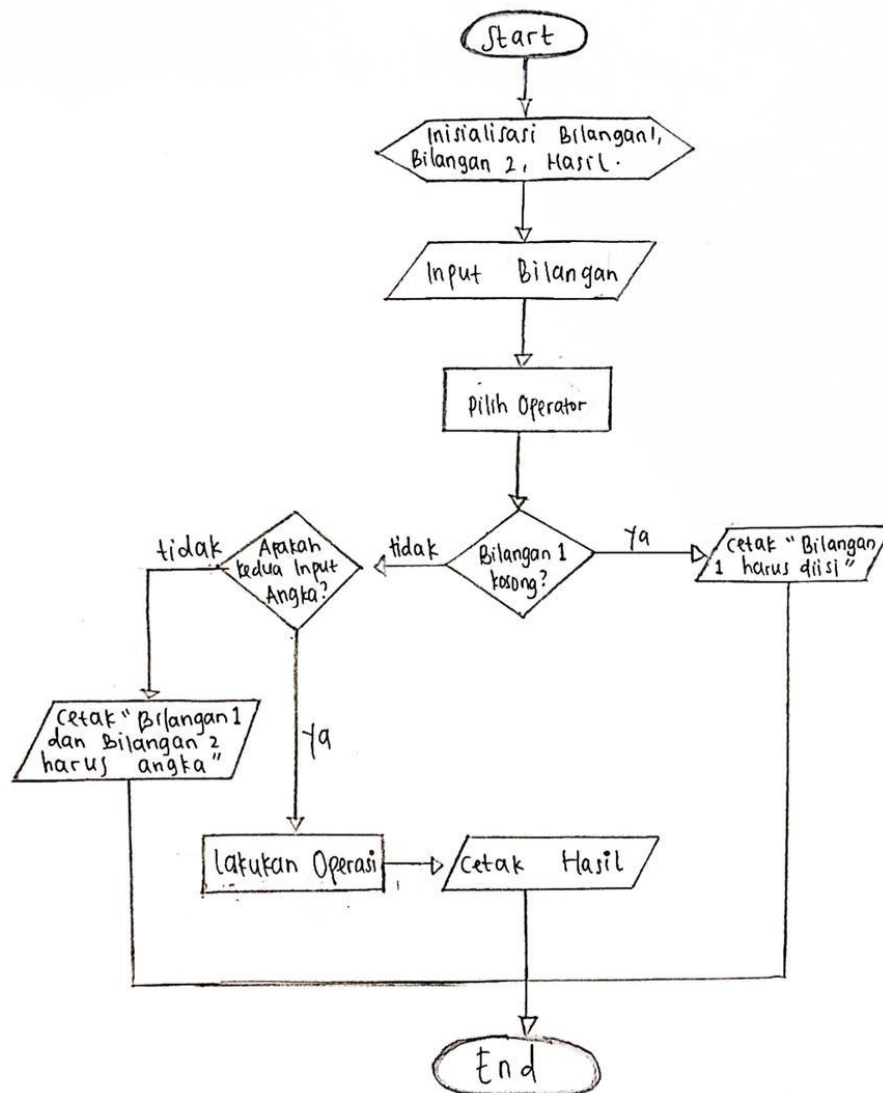
```
int hasil;
public void actionPerformed(ActionEvent e) {
    if(txtBil1.getText().trim().isEmpty()) {
        pesanPeringatan("Bilangan 1 harus diisi");
    } else if (txtBil2.getText().trim().isEmpty()) {
        pesanPeringatan("Bilangan 1 harus diisi");
    } else {
        try {
            int a = Integer.parseInt(txtBil1.getText());
            int b = Integer.parseInt(txtBil2.getText());
            int c = cbOperator.getSelectedIndex(); //operator
            if (c==0) {
                a+=b;
                hasil = a;
            }
            if (c==1) {
                a-=b;
                hasil = a;
            }
            if (c==2) {
                a*=b;
                hasil = a;
            }
            if (c==3) {
                a/=b;
                hasil = a;
            }
            if (c==4) {
                a%=b;
                hasil = a;
            }
        } catch (NumberFormatException ex) {
            pesanError("Bilangan 1 dan Bilangan 2 harus angka");
        }
    }
}
```

Gambar 2.13 (Langkah 13 OperatorAssignmentGUI)

2.3 Flowchart

Flowchart
Operator Assignment Gui

Kinaya Novrya Manda
2511531016
Informatika (B)



Gambar 2.14 (Flowchart)

Flowchart di atas menggambarkan alur dasar program Operator Assignment GUI, dimulai dari Start lalu masuk ke proses input data, yaitu pengguna memasukkan dua angka dan memilih operator yang ingin digunakan. Setelah data diterima, program masuk ke bagian percabangan (*decision*) yang memeriksa operator apa yang dipilih, seperti +, -, *, /, atau %. Setiap pilihan operator akan membawa program ke proses perhitungan yang sesuai, lalu hasilnya ditampilkan kepada pengguna. Jika operator yang dimasukkan tidak valid, program akan menampilkan pesan peringatan atau kesalahan. Setelah hasil atau pesan *error* muncul, flowchart menutup proses dengan simbol End, menandakan bahwa program telah selesai dijalankan.

2.4 Pseudocode

| |
|--|
| <p>Judul:</p> <p>Program Operator Assignment GUI</p> <p>{Program untuk memproses operasi assignment pada input pengguna melalui tampilan GUI}</p> |
| <p>Deklarasi:</p> <p>Bilangan 1 : Integer;</p> <p>Bilangan 2 : Integer;</p> <p>Operator : String;</p> <p>Hasil : Integer;</p> |
| <p>Pseudocode:</p> <ol style="list-style-type: none"> 1. Start 2. read bil1 3. read bil2 |

```

4. read operator
5.  if operator = "+" then
6.    hasil ← bil1 + bil2
7.    print "Hasil Penjumlahan = ", hasil
8.
9.  else if operator = "-" then
10.   hasil ← bil1 - bil2
11.   print "Hasil Pengurangan = ", hasil
12.
13. else if operator = "*" then
14.   hasil ← bil1 * bil2
15.   print "Hasil Perkalian = ", hasil
16.
17. else if operator = "/" then
18.   if bil2 = 0 then
19.     print "KESALAHAN: Pembagian dengan angka 0 tidak
        diperbolehkan!"
20.   else
21.     hasil ← bil1 / bil2
22.     print "Hasil Pembagian = ", hasil
23.
24. else if operator = "%" then
25.   if bil2 = 0 then
26.     print "KESALAHAN: Modulus dengan angka 0 tidak
        diperbolehkan!"
27.   else
28.     hasil ← bil1 % bil2
29.     print "Hasil Modulus = ", hasil
30.   end if
31.

```

```
32.  else
33.    print "PERINGATAN: Bilangan 1 harus diisi"
34.    print "KESALAHAN: Bilangan 1 dan Bilangan 2 harus angka"
35.  end if
36. End
```

Tabel 2.1 (Pseudocode)

Pseudocode di atas menggambarkan alur kerja program yang membaca dua angka dan satu operator dari pengguna, lalu memeriksa operator tersebut menggunakan percabangan *If*. Setiap operator seperti $+=$, $-=$, $*=$, $/=$, dan $\%=$ memiliki proses perhitungan masing-masing, dengan pengecekan khusus saat pembagian atau *modulus* agar tidak menggunakan nol. Jika operator tidak sesuai, program menampilkan pesan peringatan. Secara ringkas, pseudocode ini menunjukkan langkah-langkah untuk memproses operasi assignment berdasarkan input pengguna.

2.5 Analisis Hasil

Berdasarkan hasil percobaan dari kode program yang telah dijalankan, dapat dilihat output yang ditampilkan menunjukkan bahwa program operator assignment telah menerapkan validasi input dengan baik. Saat pengguna memasukkan data yang tidak sesuai, seperti karakter huruf pada kolom bilangan, program langsung menampilkan pesan kesalahan berupa *popup*. Hal ini membuktikan bahwa program mampu mendeteksi input yang tidak valid sebelum melakukan proses perhitungan, sehingga mencegah terjadinya *error* dan menjaga alur program tetap stabil. Validasi ini sangat penting agar aplikasi tidak salah memproses data atau menampilkan hasil yang tidak akurat.

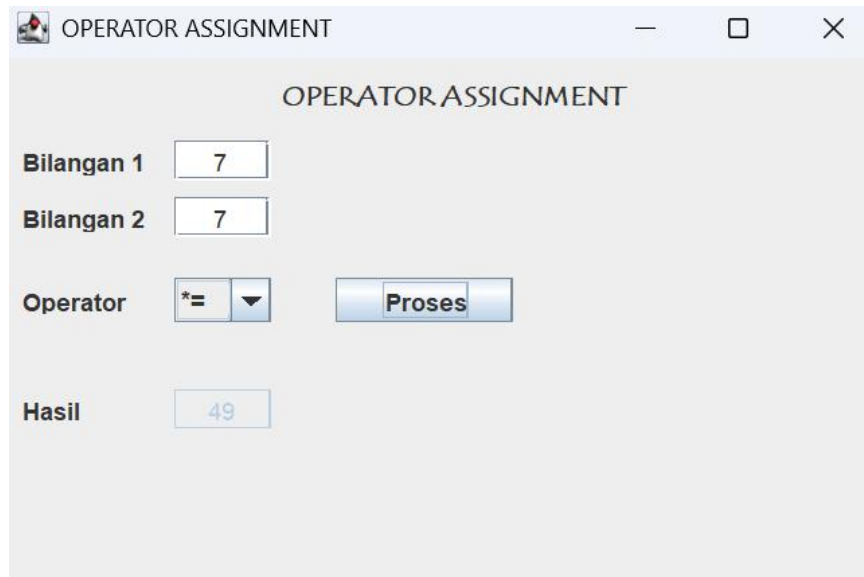
Selain itu, penggunaan *popup* sebagai media pemberitahuan membuat pesan kesalahan menjadi jelas dan mudah dipahami oleh pengguna. Pesan seperti “Bilangan 1 dan Bilangan 2 harus angka” memberi informasi yang spesifik mengenai apa yang harus diperbaiki, sehingga pengguna dapat langsung memperbaiki inputnya tanpa kebingungan. Penyampaian pesan secara langsung ini juga sesuai dengan prinsip *user-friendly*, di mana aplikasi harus memberikan umpan balik yang cepat dan informatif ketika terjadi kesalahan dalam pengisian data.

Berdasarkan hasil tersebut, dapat disimpulkan bahwa program ini sudah memenuhi prinsip dasar pengembangan perangkat lunak, terutama dalam hal validasi input dan penanganan kesalahan. Dalam teori pemrograman, validasi input diperlukan untuk memastikan data sesuai dengan tipe yang diharapkan agar proses perhitungan berjalan benar. Selain itu, pemberian pesan kesalahan termasuk bagian dari prinsip *usability* pada desain antarmuka. Dengan demikian, penerapan validasi dan *feedback* pada program ini sudah sesuai dengan dasar teori dan mendukung keandalan aplikasi secara keseluruhan.

BAB III KESIMPULAN

3.1 Hasil Praktikum

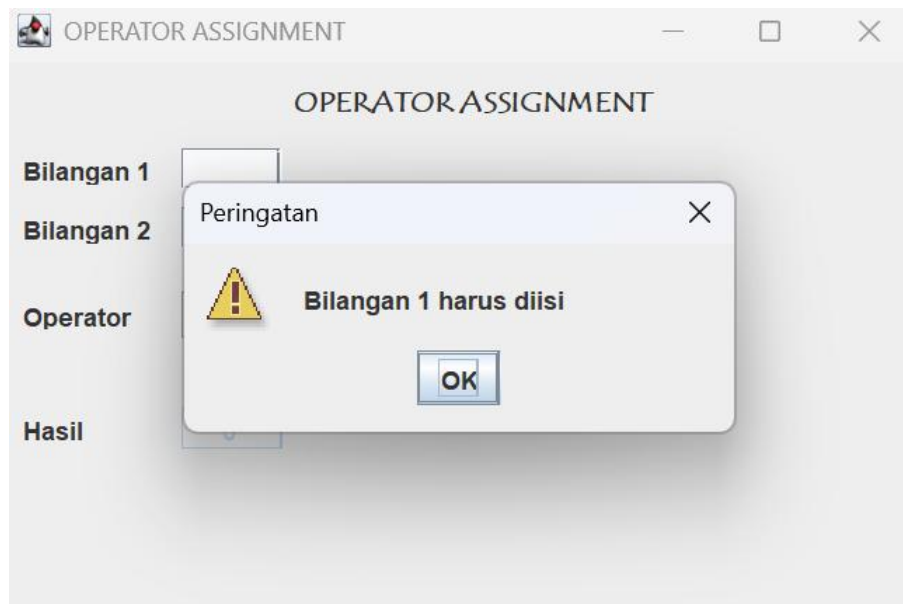
3.1.1 Output Berhasil



Gambar 3.1 (Hasil Praktikum Output Berhasil)

Output di atas menampilkan hasil perhitungan dari program “Operator Assignment”, di mana pengguna memasukkan dua angka yaitu 7 dan 7, lalu memilih operator * (kali). Operator perkalian digunakan untuk mencari hasil perkalian antara dua bilangan. Jadi ketika tombol Proses ditekan, program menghitung $7 * 7$, dan menghasilkan nilai 49. Hasil ini kemudian langsung ditampilkan pada kolom “Hasil”. Program ini membantu pengguna memahami cara kerja operasi assignment sederhana dalam aplikasi Java GUI.

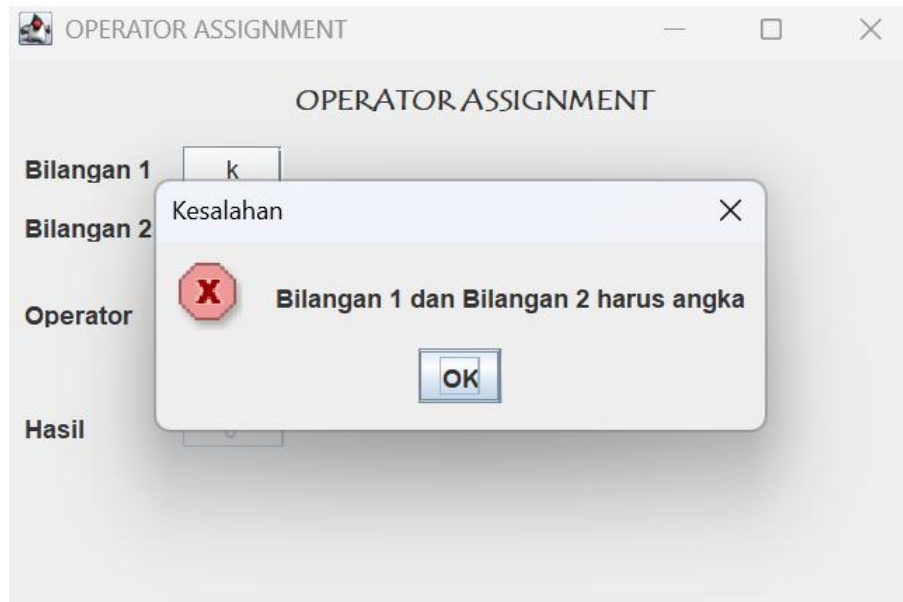
3.1.2 Output Peringatan



Gambar 3.2 (Hasil Praktikum Output Peringatan)

Output di atas menunjukkan bahwa program menampilkan pesan peringatan karena pengguna menekan tombol Proses tanpa mengisi nilai pada bagian Bilangan 1. Sistem otomatis mengecek apakah input sudah lengkap, dan ketika kolom Bilangan 1 kosong, muncul *popup* berisi pesan “Bilangan 1 harus diisi”. Peringatan ini membantu pengguna supaya tidak melakukan perhitungan dengan data yang belum lengkap, sehingga program bisa berjalan dengan benar dan tidak menimbulkan *error*.

3.1.3 Output Kesalahan



Gambar 3.3 (Hasil Praktikum Output Kesalahan)

Output di atas menunjukkan bahwa program menampilkan pesan kesalahan karena pengguna memasukkan nilai yang bukan angka pada salah satu kolom input, misalnya menulis huruf “k” pada Bilangan 1. Program otomatis memeriksa apakah kedua input benar-benar berupa angka sebelum melakukan perhitungan. Jika ada karakter lain selain angka, maka muncul *popup* dengan pesan “Bilangan 1 dan Bilangan 2 harus angka”. Peringatan ini memastikan supaya pengguna memasukkan data yang benar, sehingga proses perhitungan tidak error dan hasil yang ditampilkan tetap valid.

3.2 Saran Pengembangan

Berdasarkan hasil analisis dari kode program yang telah dijalankan, terdapat beberapa saran pengembangan agar program menjadi lebih interaktif, efisien, dan bermanfaat dalam penerapan nyata.

Untuk pengembangan ke depan, program operator assignment ini dapat ditambah dengan fitur operasi matematika yang lebih lengkap, seperti pangkat, akar kuadrat, persentase, atau operasi kombinasi. Dengan adanya variasi operasi yang lebih banyak, program akan menjadi lebih fleksibel dan dapat digunakan untuk kebutuhan perhitungan yang lebih kompleks. Selain itu, bagian validasi input juga dapat dibuat lebih detail, misalnya dengan menandai kolom mana yang salah agar pengguna lebih mudah memperbaiki kesalahan.

Pengembangan selanjutnya bisa dilakukan pada tampilan antarmuka (GUI). Desain aplikasi dapat dibuat lebih modern dengan menambahkan warna, ikon, atau tata letak yang lebih rapi agar terlihat profesional dan mudah dipahami oleh pengguna pemula. Selain itu, tombol-tombol dapat diberikan efek visual atau animasi ringan untuk meningkatkan pengalaman pengguna. Dengan tampilan yang lebih informatif dan menarik, aplikasi akan terasa lebih nyaman dan enak digunakan.

Terakhir, program ini dapat dikembangkan menjadi lebih fungsional dengan menambahkan fitur riwayat perhitungan, penyimpanan hasil ke file (seperti .txt atau .csv), dan fitur reset otomatis. Program juga dapat dikembangkan menjadi aplikasi berbasis web atau mobile agar lebih mudah diakses di berbagai perangkat. Dengan pengembangan seperti ini, program tidak hanya berguna untuk tugas praktikum, tetapi juga dapat menjadi aplikasi kecil yang bermanfaat dalam kehidupan sehari-hari.

DAFTAR PUSTAKA

- [1] P. J. Deitel and H. M. Deitel, *Java How to Program, Early Objects*, 10th ed. Pearson, 2015. [Online]. <https://openlibrary.telkomuniversity.ac.id/pustaka/117192/java-how-to-program-early-objects-global-edition-10-e>

- [2] H. Schildt, *Java: The Complete Reference*, 9th ed. McGraw-Hill, 2014. [Online]. <https://openlibrary.telkomuniversity.ac.id/pustaka/117218/java-the-complete-reference-9-e->

- [3] J. Nielsen, *Usability Engineering*. Morgan Kaufmann, 1994. [Online]. [https://openlibrary.org/books/OL7328356M/Usability_Engineering_%28Interactive Technologies%29](https://openlibrary.org/books/OL7328356M/Usability_Engineering_%28Interactive_Technologies%29)