LAPORAN PRAKTIKUM

ALGORITMA DAN PEMOGRAMAN

“PEMOGRAMAN *GUI* KALKULATOR”

disusun Oleh:

NIDA TSABITA ARIBA

2511532025

Dosen Pengampu:

Dr. WAHYUDI, S.T., M.T.

Asisten Praktikum:

AUFAN TAUFIQURRAHMAN

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

2025

KATA PENGANTAR

Alhamdulillahi rabbil'alaim, segala puji dan syukur saya panjatkan kehadirat Allah SWT atas limpahan rahmat, nikmat, dan hidayah-Nya sehingga saya dapat menyelesaikan laporan praktikum mata kuliah Algoritma dan Pemograman dengan judul "Pemograman *GUI* Kalkultor".

Laporan ini disusun sebagai salah satu bentuk pemenuhan tugas pada mata kuliah Praktikum Algoritma dan Pemograman. Laporan ini saya harapkan dapat memperdalam wawasan saya tentang penerapan konsep dasar algoritma, logika pemograman, serta implementasi *GUI* (*Graphical User Interface*) menggunakan bahasa pemograman Java.

Saya menyampaikan terima kasih kepada Bapak Dr. Wahyudi, S.T., M.T. selaku dosen pengampu mata kuliah Algoritma dan Pemograman, serta asisten praktikum yang telah memberikan bimbingan dan arahan selama pelaksanaan kegiatan praktikum. Ucapan terima kasih juga saya sampaikan kepada teman-teman praktikan yang telah memberikan bantuan, dukungan, serta kesempatan berdiskusi dalam menyelesaikan tugas praktikum ini.

Saya menyadari bahwa laporan ini masih memiliki banyak kekurangan dan keterbatasan. Oleh karena itu saya mengharapkan kritik dan saran yang membangun demi perbaikan laporan praktikum dimasa mendatang. Meskipun demikian, penulis berharap laporan ini dapat memberikan manfaat bagi pembaca.

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam proses pembelajaran pemrograman, mahasiswa tidak hanya dituntut memahami logika dasar seperti percabangan, operator aritmatika, dan pengolahan data numerik, tetapi juga bagaimana menerapkan logika tersebut ke dalam aplikasi yang dapat digunakan secara langsung oleh pengguna. Setelah mempelajari pemrosesan data melalui program berbasis teks, mahasiswa mulai diarahkan untuk mengembangkan program yang memiliki tampilan program sehingga lebih mudah digunakan dan dipahami.

Pada praktikum ini, mahasiswa diperkenalkan dengan pembuatan aplikasi menggunakan *Java Swing* serta penggunaan *WindowBuilder* di *Eclipse*. Dengan bantuan *WindowBuilder*, proses pembuatan tampilan program secara visual lebih terstruktur dan membantu mahasiswa memahami susunan komponen grafis yang digunakan dalam aplikasi.

Melalui program *GUI* Kalkulator, mahasiswa mempelajari cara menghubungkan logika perhitungan dengan elemen-elemen *GUI*, mulai dari input angka melalui tombol, penggunaan tombol operasi seperti penjumlahan, pengurangan, perkalian, dan pembagian, hingga penampilan hasil pada display berbasis *TextField*. Praktikum ini juga membahas cara menangani berbagai kondisi penggunaan, seperti menghapus input dengan tombol C. Dengan demikian, praktikum ini membatu mahasiswa memahami prinsip pengembangan aplikasi *GUI* yang fungsional, interaktif, dan mampu menjalankan proses perhitungan secara tepat dan terstruktur.

1.2 Tujuan

Tujuan dari praktikum ini adalah sebagai berikut:

1. Memahami konsep dasar pembuatan tampilan program (*GUI*) menggunakan *Java Swing* sebagau media untuk membangun aplikasi interaktif.
2. Mempelajari penggunaan *WindowBuilder* di *Eclipse* untuk merancang tampilan program secara visual secara terstruktur.
3. Menerapkan operasi aritmetika dalam aplikasi *GUI*, seperti penjumlahan, pengurangan, perkalian, pembagian, dan sisa bagi memalui penggunaan tombol angka, tombol operasi, dan *display* hasil.
4. Menerapkan penanganan input dan alur penggunaan, seperti penggunaan tombol C untuk menghapus input, mamastikan perhitungan tetap konsisten, serta mencegah kesalahan saat tombol ditekan berurutan.
5. Melatih kemampuan mahasiswa dalam membangun aplikasi kalkulator berbasis *GUI* yang fungsional, mudah digunakan, dan mampu melakukan perhitungan secara tepat dan terstruktur.

1.3 Manfaat Praktikum

Manfaat yang diperoleh dari praktikum ini antara lain:

1. Menambah pemahaman mahasiswa dalam merancang tampilan program (*GUI*) menggunakan Java Strting dan *WindowBuilder* sebagai dasar pembuatan aplikasi interaktif.
2. Membiasakan mahasiswa menggunakan komponen-komponen *GUI* yang umum digunakan pada kalkulator, seperti tombol angka, tombol operasi, tombol hapus, serta *display* berbasis *text field* untuk menampilkan hasil.
3. Melatih kemampuan menghubungkan logika perhitungan dengan elemen *GUI*, sehingga proses penekanan tombol, pengolahan operasi aritmatika, dan output hasil dapat berjalan dengan benar.
4. Meningkatkan ketelitian dalam menangani alur input, seperti menghapus angka, mencegah kesalahan penekanan tombol, dan memastikan perhitungan tetap konsisten selama aplikasi digunakan.

5. Memberikan dasar bagi mahasiswa untuk mengembangkan aplikasi *GUI* yang lebih kompleks, baik dalam rancangan tampilan maupun pengelolaan logika perhitungannya.

BAB II
PEMBAHASAN

2.1 Deskripsi Program

Program Calculator merupakan aplikasi kalkulator sederhana berbasis *GUI* (*Graphical User Interface*) yang dibuat menggunakan *Java Swing* dan dirancang melalui *WindowBuilder* pada *Eclipse*. Aplikasi ini memungkinkan pengguna melakukan operasi aritmatika dasar seperti penjumlahan, pengurangan, perkalian, pembagian, dan modulo dengan tampilan visual yang mudah digunakan.

Pada aplikasi ini, berbagai komponen *GUI* digunakan untuk membangun tampilan seperti *JFrame*, *JTextFiled*, dan *JButton*. Setiap tombol angka dan operator dibuat secara individual dan diberi *event listener* sehingga ketika tombol ditekan, nilainya akan ditampilkan atau diproses pada *textField* sebagai area input dan output.

Ketika pengguna memilih sebuah operator (seperti +, -, *, /, %), nilai pertama disimpan terlebih dahulu, kemudian pengguna memasukkan nilai kedua. Saat tombol "=" ditekan, program membaca operator yang digunakan, memproses kedua bilangan tersebut, lalu menampilkan hasilnya pada *textField* dalam format angka desimal.

Aplikasi ini juga dilengkapi beberapa fitur tambahan, seperti:

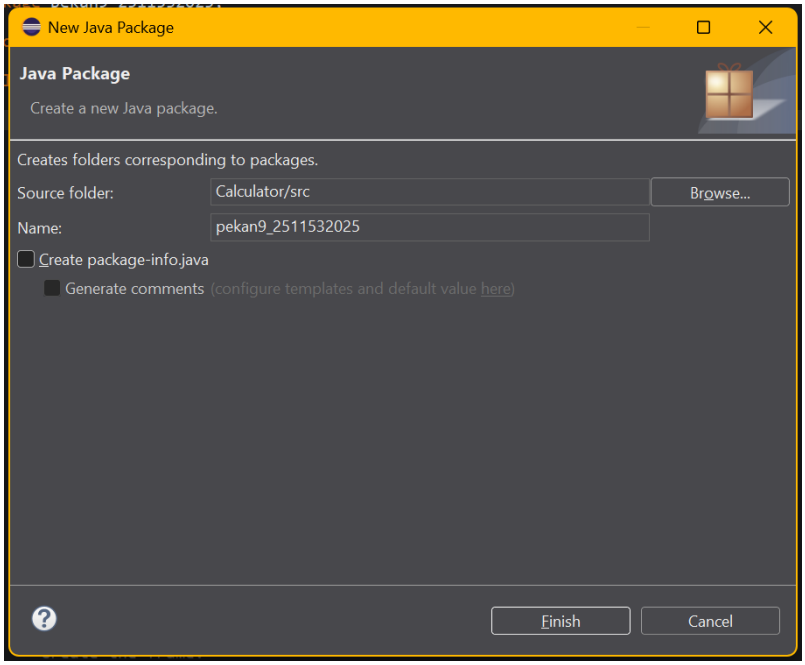
a. Tombol C untuk mengapus seluruh input.
b. Tombol *Backspace* untuk menghapus satu karakter terakhir.
c. Memungkinkan pengguna memasukkan angka desimal dengan tombol titik (.).

Melalui program ini, pengguna dapat melihat contoh penerapan perhitungan aritmatika dalam bentuk aplikasi *GUI* menggunakan *Java Swing* lengkap dengan pengaturan *event*, pengolahan angka, dan pengaturan tampilan komponen *GUI*.

2.2 Langkah Kerja Program

2.2.1 Membuat *Package* Baru

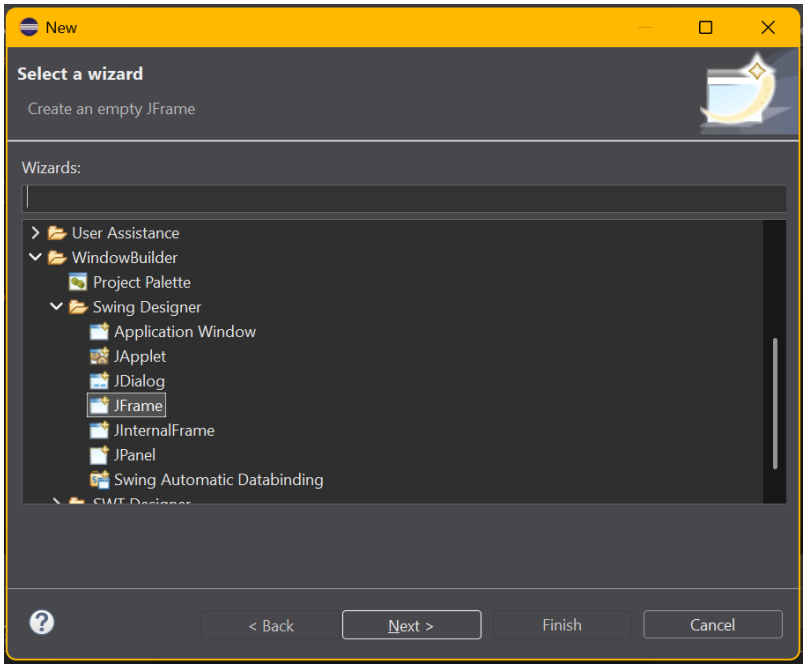
1. Buka aplikasi *Eclipse*
2. Klik kanan pada folder src > New > *Package*
3. Beri nama *Package* dengan "pekan9_2511532025", lalu klik *Finish*



Gambar 2.1 Membuat *Package* baru

2.2.2 Membuat *Class* JFrame Menggunakan *WindowBuilder*

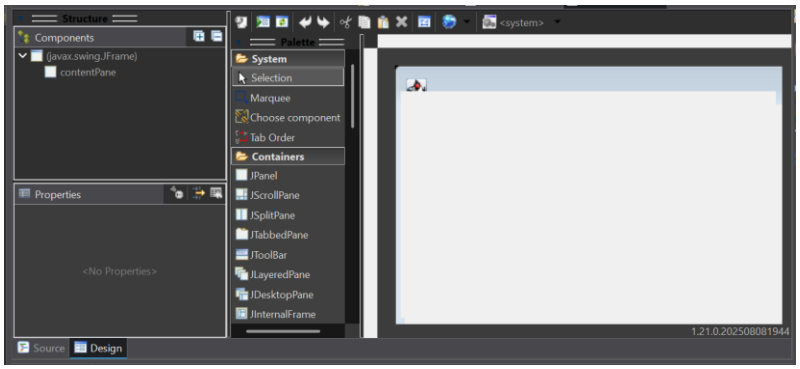
1. Klik kanan pada *folder src* > *New* > *Other*…
2. Pada jendela yang muncul, buka kategori *WindowBuilder*
3. Pilih *Swing Designer* > *JFRame*, lalu klik *Next*



Gambar 2.2 Tampilan pembuatan *Class* baru

4. Beri nama *class* “Calculator_2511532025”

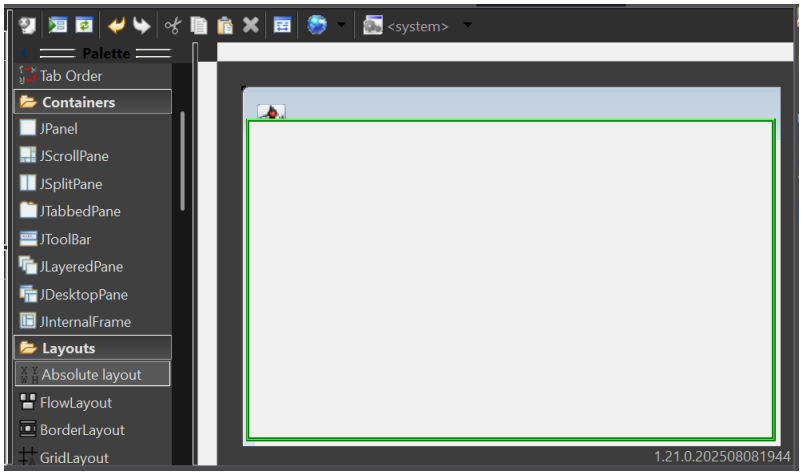
5. Klik *Finish*, dan Editor *WindowBuilder* akan terbuka otomatis.



Gambar 2.3 Tampilan awal *WindowBuilder*

2.2.3 Mengatur *Layout*

1. Mengatur *layout* awal form dengan *Absolute Layout* agar posisi setiap komponen dapat diatur dengan bebas dan presisi



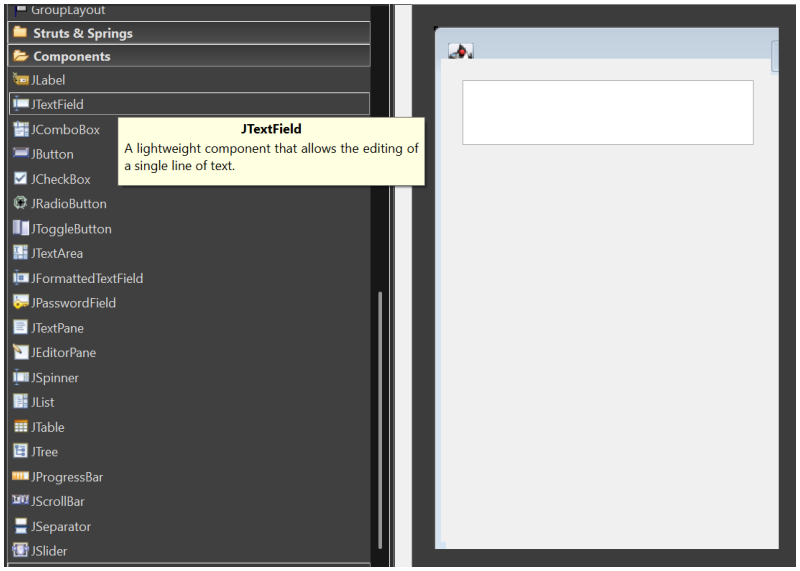
Gambar 2.4 Menambahkan *Absolute Layout*

2. Mengatur ukuran area desain menjadi 320 x 485

2.2.4 Menambahkan Komponen *GUI*

Tambahkan komponen-komponen berikut ke area desain:

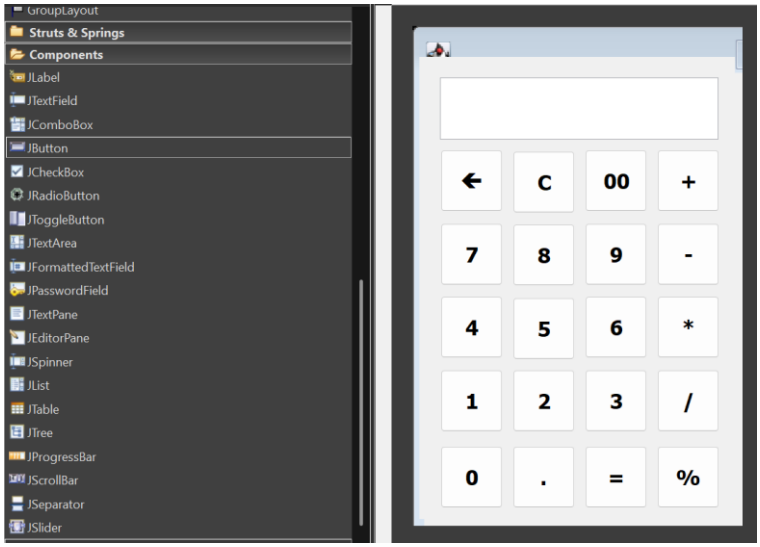
1. *JTextField*



Gambar 2.5 Menambahkan *JTextField*

- Untuk menampilkan angka dan hasil perhitungan
- Atur *font* menjadi *20pt Bold*

2. *JButton*



Gambar 2.6 Menambahkan *JButton*

- Penempatannya diatur menyerupai bentuk kalkulator
- Atur *font* tombol menjadi *20pt Bold*
- Sesuaikan nama *variable* dan *text* tiap tombol (misalnya btn1, btn2, btnplus, dll)

a. Tombol Angka

- Tambahkan tombol 1 sampai 9, termasuk 0 dan 00

- Setiap tombol berfungsi menambahkan angka ke dalam *JTextField*

b. Tombol Operator
   - Operator yang digunakan: +, -, *, / %
   - Setiap tombol menyimpan nilai pertama (*first*) dan jenis operasi

c. Tombol Fungsi
   - "C" untuk menghapus seluruh input
   - *Backspace* untuk menghapus satu karakter terakhir
   - "." untuk input desimal
   - "=" untuk menampilkan hasil perhitungan

2.2.5 Menambahkan Logika Program

Pada tahap ini, setiap tombol diberi *ActionListener* untuk menjalankan fungsi sesuai tugas masing-masing di dalam aplikasi kalkulator.

Deklarasi Variabel Utama

Tambahkan variabel tersebut pada *class* untuk menyimpan nilai dan jenis operasi:

```
double first;
double second;
double result;
String operation;
String answer;
```

Kode Program 2. 1 Deklarasi variabel utama

- Variabel *first* untuk menyimpan angka pertama yang dimasukkan pengguna
- Variabel *second* untuk menyimpan angka kedua yang dimasukkan pengguna
- Variabel *result* untuk menyimpan hasil perhitungan
- Variabel *operation* untuk menyimpan jenis operator yang dipilih
- Variabel *answer* untuk menyimpan tampilan hasil dalam bentuk teks yang akan ditampilkan kembali ke *textField*

Peran *ActionListener*

Setiap tombol pada kalkulator diberikan *ActionListener* agar dapar merespons saat pengguna menekannya. Ketika tombol ditekan, *method actionPerformed()* akan dijalankan untuk memproses perintah sesuai logika yang telah ditentukan.

1. Tombol Angka

   Berfungsi untuk memasukkan angka ke dalam *textField*.

   - Setiap tombol angka akan menambahkan karakter angka ke dalam *textField*
   - Digunakan saat pengguna memasukkan nilai input

   Contoh *Syntax*:

```
JButton btn1 = new JButton("1");
btn1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
String number = textField.getText()+btn1.getText();
textField.setText(number);
}
```

Kode Program 2.2 *Syntax* tombol angka

2. Tombol Fungsi

   a. Tombol C

   Berfungsi ntuk menghapus seluruh isi *textField*.

   Contoh *Syntax*:

```
JButton btn c = new JButton("C");
btnc.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
textField.setText(null);
}
});
```

Kode Program 2.3 *Syntax* tombol C

   b. Tombol *Backspace*

   Berfungsi untuk menghapus satu karakter terakhir pada *textField*

   Contoh *Syntax*:

```
JButton btnNewButton = new JButton("\uF0E7");
btnNewButton.addActionListener(new
ActionListener() {
public void actionPerformed(ActionEvent e) {
String backSpace = null;
if(textField.getText().length()>0);
{
StringBuilder str = new
StringBuilder(textField.getText());
str.deleteCharAt(textField.getText().length()-1);
backSpace = str.toString();
textField.setText(backSpace);
}
```

```
}
});
```

Kode Program 2.4 *Syntax* tombol Backspace

c. Tombol "." (Desimal)
Menambahkan titik desimal dalam angka
Contoh *Syntax*:

```
JButton btndot = new JButton(".");
btndot.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
String number =
textField.getText()+btndot.getText();
textField.setText(number);
}
});
```

Kode Program 2.5 *Syntax* tombol Desimal

3. Tombol Operator
Berfungsi untuk menyimpan angka pertama dan operasi yang dipilih.
Langkah kerjanya:
   1. Mengambil nilai dari *textField* lalu menyimpannya sebagai *first*
   Contoh *Syntax*:

```
first = Double.parseDouble(textField.getText());
```

Kode Program 2.6 *Syntax* penyimpanan nilai first

   2. Menyimpan jenis operator ke dalam variabel operation
   Contoh *Syntax*:

```
      operation = "+";
```

Kode Program 2.7 *Syntax* penyimpanan jenis operator

   3. Mengosongkan *textField* agar siap menerima angka kedua
   Contoh *Syntax*:

```
textField.setText("");
```

Kode Program 2.8 *Syntax* mengosongkan textField

Contoh *Syntax* lengkap:

```
JButton btnplus = new JButton("+");
  btnplus.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent e) {
  first = Double.parseDouble(textField.getText());
  textField.setText("");
```

```
operation = "+";
}
});
```

Kode Program 2.9 *Syntax* lengkap tombol operator

4. Tombol "="

   Menjalankan perhitungan berdasarkan operator yang telah dipilih.

   Langkah kerjanya:

   1. Menyimpan angka kedua (*second*)

      Contoh *Syntax*:

```
second = Double.parseDouble(textField.getText());
```

Kode Program 2.10 *Syntax* menyimpan nilai second

   2. Melakukan operasi sesuai nilai operation

      Contoh *Syntax*:

```
if(operation=="+") {
result = first + second;
}
else if(operation=="-") {
result = first - second;
}
```

Kode Program 2.11 *Syntax* proses operasi

   3. Menampilkan hasil pada *textField*

      Contoh *Syntax*:

```
String answer;
result = first + second;
answer = String.format("%.2f", result);
textField.setText(answer);
}
```

Kode Program 2.12 *Syntax* penampilan hasil

Contoh *Syntax* lengkap:

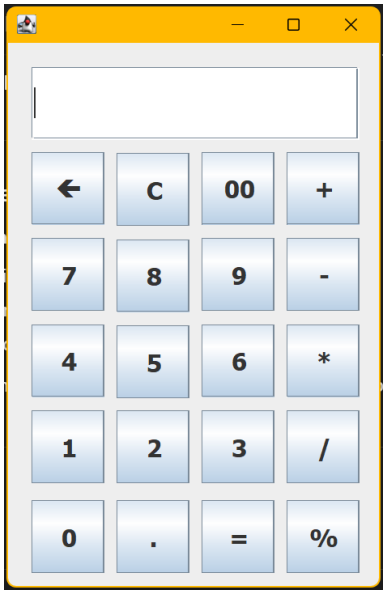
```
JButton btnequal = new JButton("=");
btnequal.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
String answer;
second = Double.parseDouble(textField.getText());
if(operation=="+") {
result = first + second;
answer = String.format("%.2f", result);
textField.setText(answer);
}
else if(operation=="-") {
```

```
result = first - second;
answer = String.format("%.2f", result);
textField.setText(answer);
}
else if(operation=="*") {
result = first * second;
answer = String.format("%.2f", result);
textField.setText(answer);
}
else if(operation=="/") {
result = first / second;
answer = String.format("%.2f", result);
textField.setText(answer);
}
else if(operation=="%") {
result = first % second;
answer = String.format("%.2f", result);
textField.setText(answer);
}
}
});
```

Kode Program 2.13 *Syntax* lengkap tombol "="

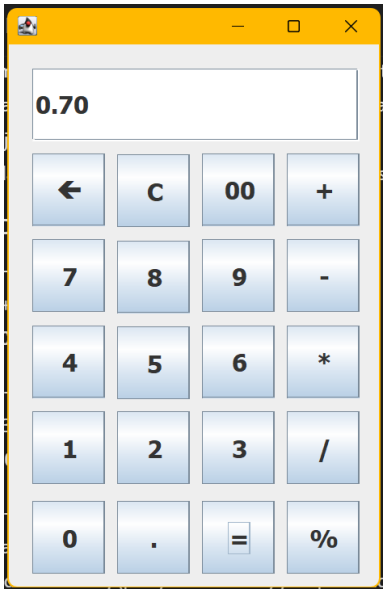
2.2.6 Menjalankan dan Menguji Program

1. Jalankan program melalui tombol *Run*



Gambar 2.7 Tampilan saat kalkulator dijalankan

2. Uji semua tombol: angka, operator, *backspace*, dan hasil perhitungan



Gambar 2.8 Contoh hasil perhitungan program

3. Pastikan output sesuai dan tidak ada error

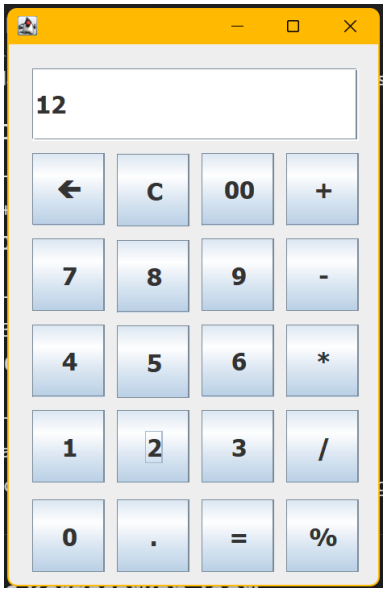
2.3 Analisis Hasil Program

Setelah program dijalankan dan diuji menggunakan berbagai kombinasi input, fitur-fitur utama kalkulator bekerja sesuai logika arimatika dasar. Setiap tombol memicu *event* melalui mekanisme *ActionListener*, sehingga proses kalkulasi berlangsung setelah pengguna menekan tombol yang sesuai.

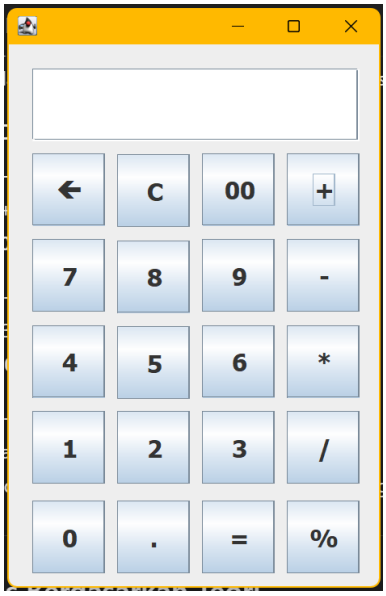
Program menampilkan hasil perhitungan dalam format dua desimal menggunakan *String.format()*, sehingga output selalu konsisten. Tombol angka berjalan baik dalam membangun input, tombol operator menyimpan nilai pertama dan jenis operasi, serta tombol "=" memproses hasil sesuai operator.

2.3.1 Contoh Output Program

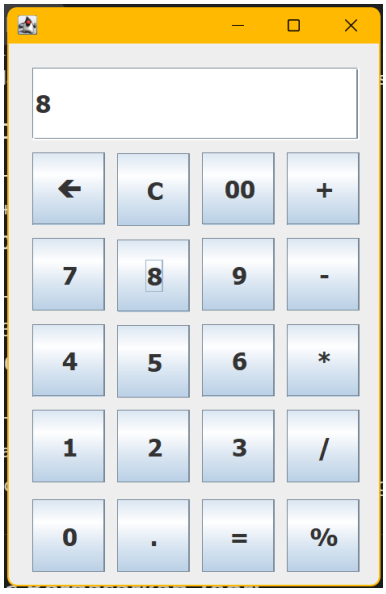
Contoh 1:



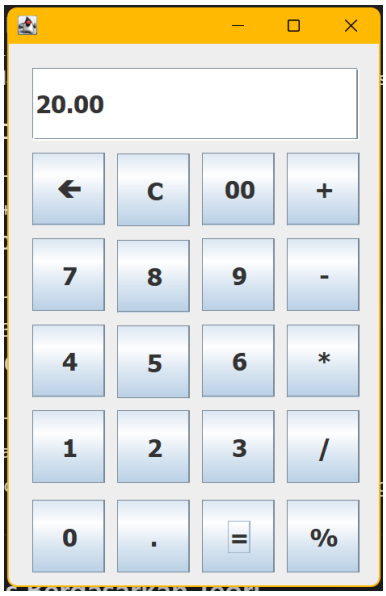
Gambar 3.1 Pengguna memasukkan angka pertama



Gambar 3.2 Pengguna menekan operator “+”

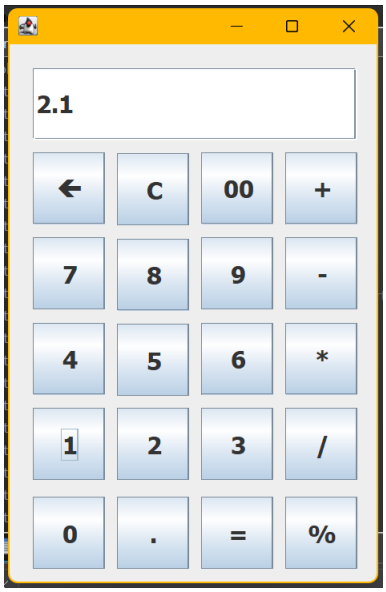


Gambar 3.3 Pengguna memasukkan angka kedua

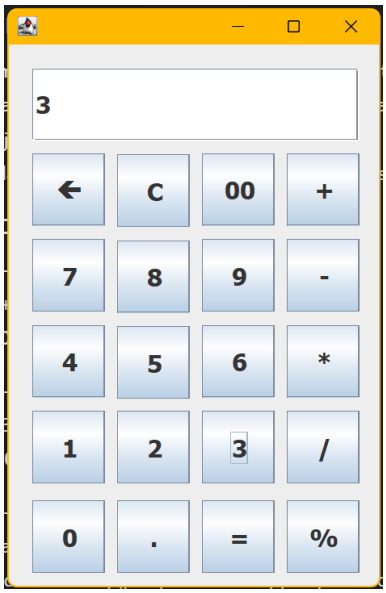


Gambar 3.4 Pengguna menekan tombol “=” dan program menampilkan hasil operasi

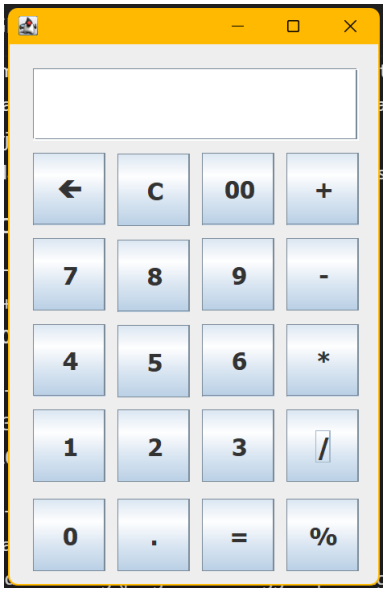
Contoh 2:



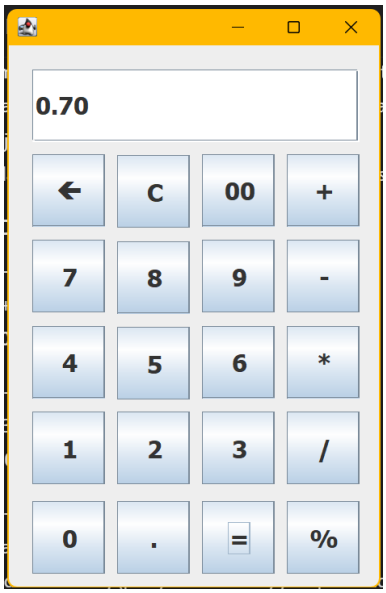
Gambar 3.4 Pengguna memasukkan angka pertama



Gambar 3.5 Pengguna memasukkan angka kedua

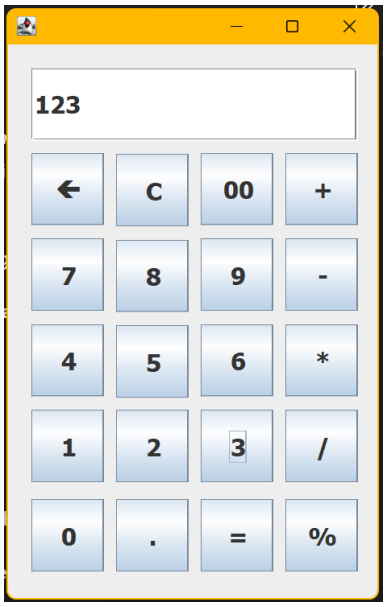


Gambar 3.6 Pengguna menekan operator “/”

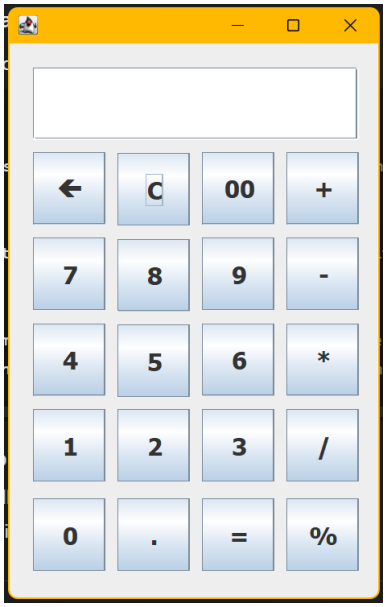


Gambar 3.8 Pengguna menekan tombol “=” dan program menampilkan hasil operasi

Contoh 3:

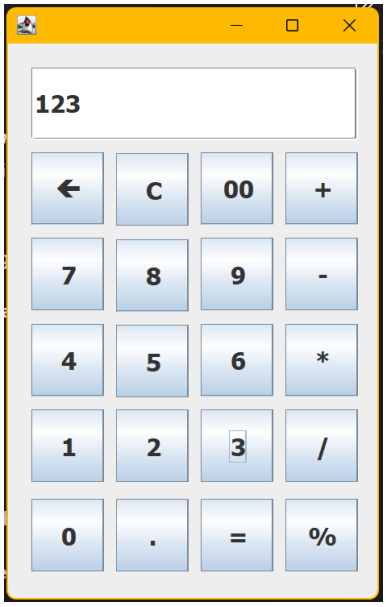


Gambar 3.6 Pengguna memasukkan angka

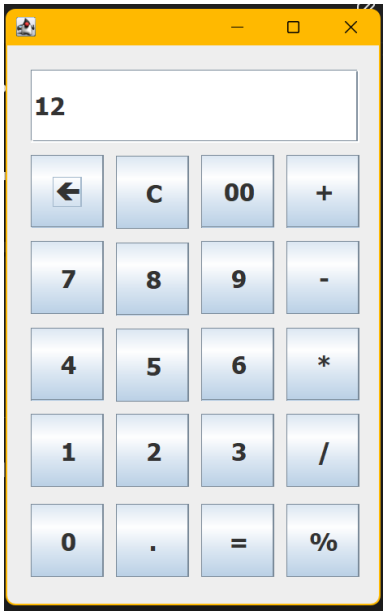


Gambar 3.10 Pengguna menekan tombol fungsi “C” sehingga *textField* kosong

Contoh 4:



Gambar 3.11 Pengguna memasukkan angka



Gambar 3.12 Pengguna menekan tombol *Backspace*, program mengapus karakter terakhir pada *textField*

2.3.2 Analisis Berdasarkan Teori

Analisis hasil program ini berdasar pada prinsip-prinsip dasar *Java Swing*, pemograman berbasis *event*, serta mekanisme pemrosesan aritmatika dalam Java.

2.3.2.1 Teori *Event-Driven Programming*

Kalkulator bekerja dengan pendekatan *event-driven*, di mana alur program bergerak mengkuti peristiwa yang dipicu pengguna, seperti menekan tombol. Karena dalam pemograman *event-driven*, objek *GUI* menjadi sumber *event*, sementara *listener* bertugas menangani aksi tersebut.

Pada program ini tombol bertindak sebagai *event source* dan *ActionListener* berperan sebagai *event handler*. Setiap kali tombol ditekan, *metode actionPerformed()* dipanggil untuk menjalankan logika perhitungan. Dengan pola ini, interaksi berjalan responsif dan terstruktur.

2.3.2.2 Teori Parsing dan Konversi Tipe Data

Program menggunakan *Double.parseDouble()* untuk mengubah input berbentuk teks menjadi nilai numerik. Dalam teori pemograman, proses ini disebut parsing, yakni penerjemahan teks menjadi tipe numerik untuk digunakan dalam operasi komputasi. Dengan demikian, kalkulator dapat menerima input dari pengguna lalu mengolahnya sebagai angka yang valid.

2.3.2.3 Teori Operasi Aritmatika

Operator seperti =, - , *, /, dan % mengikuti aturan aritmatika yang berlaku da Java. Operator aritmatika menjalankan perhitungan berdasarkan aturan matematis standar, diproses sesuai alur yang ditentukan oleh program. Karena kalkulator memproses satu operasi setiap kali tombol hasil ditekan, urutan eksekusinya jelas dan tidak menimbulkan tumpang tindih operasi.

2.3.2.4 Teori *GUI (Graphical User Interface)*

Penggunaan komponen seperti *JFrame*, *JTextField*, dan *JButton* mengikuti prinsip dasar *GUI*, yaitu menyediakan media interaksi visual yang dapat merespons tindakan pengguna. Hal inilah yang membuat antarmuka kalkulator terasa mudah digunakan dan konsisten dalam responnya.

2.3.3 Kesimpulan Analisis

Berdasarkan hasil program dan teori yang mendasari:

- Program berjalan sesuai konsep *event-driven*, hanya memproses logika ketika sebuah aksi terjadi.
- Perhitungan dilakukan secara akurat menggunakan tipe data double dan operator aritmatika standar Java.
- Komponen *GUI* berfungsi harmonis melalui mekanisme listener, sehingga interaksi pengguna berlangsung stabil dan dapat diprediksi.
- Input yang belum lengkap tidak langsung diproses, sehingga mencegah kesalahan perhitungan.

Jadi secara keseluruhan, aplikasi kalkulator ini telah memenuhi prinsip-prinsip penting dalam pemrograman *GUI*, pengolahan event, dan operasi aritmatika pada Java.

BAB III
KESIMPULAN

3.1 Kesimpulan

Praktikum ini menghasilkan sebuah aplikasi kalkulator sederhana berbasis *Java Swing* yang mampu menjalankan operasi aritmatika dasar, seperti penjumlahan, pengurangan, perkalian, pembagian, dan modulus. Seluruh komponen *GUI* termasuk tombol angka, tombol operator, backspace, dan tombol hasil yang berfungsi sesuai rancangan.

Dari proses perancangan hingga pengujian, mahasiswa memperoleh pemahaman mengenai konsep *event-driven programming*, pengelolaan komponen *GUI*, serta cara menghubungkan input teks dengan proses perhitungan numerik melalui konversi tipe data. Pengujian menunjukkan bahwa program mampu menampilkan hasil perhitungan dengan benar dan menangani kondisi input yang belum lengkap.

Secara keseluruhan, tujuan praktikum untuk mengenalkan pembuatan tampilan grafis dan logika perhitungan pada Java dapat dicapai dengan baik.

3.2 Saran

Aplikasi ini dapat dikembangkan lebih lanjut dengan beberapa perbaikan, antara lain:

1. Menambahkan fitur riwayat perhitungan, agar pengguna dapat melihat operasi sebelumnya.
2. Menyertakan validasi input yang lebih lengkap, misalnya mencegah dua operator berurutan atau angka kosong.
3. Memperluas operasi matematika, seperti akar, pangkat, atau trigonometri.
4. Meningkatkan tampilan *GUI*, misalnya melalui pengaturan warna, ukuran tombol, atau tata letak agar lebih nyaman digunakan.
5. Mengoptimalkan struktur kode, seperti memanfaatkan *method* terpisah untuk mengurangi pengulangan kode.

Dengan pengembangan tersebut, aplikasi kalkulator dapat menjadi lebih fleksibel, lebih informatif, dan lebih mudah digunakan.

DAFTAR PUSTAKA

[1] Oracle, "The Java Tutorials," 2024 [Daring]. Tersedia pada: https://docs.oracle.com/javase/tutorial/uiswing/events/. [Diakses: 27-Nov-2025].

[2] Oracle, "Java Platform, Standard Edition" 2025 [Daring]. Tersedia pada: https://docs.oracle.com/javase/8/docs/api/java/lang/Double.html.]Diakses: 27-Nov-2025].