Laporan Praktikum

Struktur Data

Disusun Oleh :

SASKIA ALIFAH (2411531002)

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

Departemen Informatika

Fakultas Teknologi Informasi

Universitas Andalas

Tahun 2025

SelectionSort & InsertionSort

A. TUJUAN PRAKTIKUM

1. Memahami konsep dan implementasi algoritma Selection Sort dan Insertion Sort pada array menggunakan bahasa pemrograman Java dan aplikasi GUI.
2. Menguasai pembuatan aplikasi GUI berbasis Java Swing untuk memvisualisasikan proses sorting secara interaktif dan langkah demi langkah.
3. Mempelajari teknik validasi input, penanganan error, serta logika traversal array dalam proses sorting menggunakan pendekatan visual.

B. PENDAHULUAN

Sorting atau pengurutan data adalah salah satu topik fundamental dalam struktur data dan algoritma. Dua algoritma sorting yang sering digunakan sebagai dasar pembelajaran adalah Selection Sort dan Insertion Sort. Selection Sort bekerja dengan cara mencari elemen terkecil dari bagian array yang belum terurut, lalu menukarnya dengan elemen di posisi awal bagian tersebut. Proses ini diulang hingga seluruh array terurut. Sementara itu, Insertion Sort menyisipkan setiap elemen ke posisi yang tepat pada bagian array yang sudah terurut dengan menggeser elemen-elemen yang lebih besar ke kanan. Kedua algoritma ini dikenal sederhana secara logika dan sangat cocok untuk dipelajari secara visual.

Pada praktikum ini, implementasi kedua algoritma dilakukan menggunakan bahasa pemrograman Java dan antarmuka grafis berbasis Java Swing. Java Swing menyediakan berbagai komponen GUI seperti JFrame, JPanel, JLabel, JTextField, JButton, dan JTextArea yang memungkinkan pembuatan aplikasi desktop interaktif. Dengan memanfaatkan komponen-komponen ini, proses pengurutan data dapat divisualisasikan secara real-time, sehingga setiap langkah algoritma dapat diamati dan dipahami dengan lebih mudah. Visualisasi ini sangat membantu dalam pembelajaran karena mahasiswa dapat langsung melihat perubahan array pada setiap langkah sorting.

Selain pemrograman algoritma sorting dan GUI, materi lain yang digunakan adalah event handling dan error handling. Event handling diperlukan agar aplikasi dapat merespons aksi pengguna, misalnya saat tombol "Set Array", "Langkah Selanjutnya", atau "Reset" ditekan. Sementara itu, error handling sangat penting untuk menangani input yang tidak valid, seperti karakter non-angka atau format input yang salah. Dengan adanya validasi dan penanganan error yang baik, aplikasi menjadi lebih ramah pengguna dan tahan terhadap kesalahan input, sehingga proses pembelajaran menjadi lebih efektif dan menyenangkan.

C. METODE PRAKTIKUM

1. Kelas InsertionSortGUI

a) Deskripsi Kelas

InsertionSortGUI adalah kelas utama untuk visualisasi algoritma Insertion Sort dalam bentuk aplikasi GUI. Kelas ini juga mewarisi JFrame dan memiliki struktur serta komponen GUI yang mirip dengan SelectionSortGUI. Bedanya, logika sorting yang diimplementasikan adalah Insertion Sort, yaitu menyisipkan elemen ke posisi yang tepat pada bagian array yang sudah terurut dengan cara menggeser elemen yang lebih besar ke kanan

b) Penjelasan Fungsi dan Cara Kerja

i. Package dan Import

Program diawali dengan mendeklarasikan package tempat file ini berada, yaitu Pekan7. Package digunakan untuk mengelompokkan kelas-kelas Java agar struktur proyek lebih terorganisir. Setelah itu, terdapat sejumlah *import statements* yang digunakan untuk mengakses berbagai komponen GUI. java.awt.* digunakan untuk elemen layout dan kontrol dasar seperti FlowLayout atau BorderLayout, sedangkan javax.swing.* mengimpor komponen GUI lanjutan seperti JFrame, JButton, dan JTextField. EmptyBorder sebenarnya tidak digunakan di kode ini, namun umumnya digunakan untuk memberikan jarak di dalam komponen GUI.

```
package Pekan7;
import java.awt.*;
import javax.swing.*;
import javax.swing.border.EmptyBorder;
```

ii. Deklarasi Kelas dan Variabel Atribut

Kelas InsertionSortGUI merupakan turunan dari JFrame, sehingga memiliki seluruh kemampuan dari jendela aplikasi Swing. Di dalamnya terdapat sejumlah atribut penting. array adalah array integer yang menyimpan data angka input dari pengguna. labelArray adalah array JLabel yang mewakili elemen-elemen array secara visual. Tiga buah tombol (stepButton, resetButton, dan setButton) mengatur interaksi pengguna. inputField adalah tempat pengguna mengetik input angka. panelArray akan menampilkan visualisasi array, sedangkan stepArea akan menampilkan log atau catatan langkah-langkah sorting. Variabel tambahan seperti i, j, sorting, dan stepCount digunakan untuk mengatur jalannya proses sorting per langkah.

```
public class InsertionSortGUI extends JFrame {
    private int[] array;
    private JLabel[] labelArray;
```

```
    private JButton stepButton, resetButton, setButton;
    private JTextField inputField;
    private JPanel panelArray;
    private JTextArea stepArea;

    private int i = 1, j;
    private boolean sorting = false;
    private int stepCount = 1;
```

iii. Method main()

Method main merupakan titik awal eksekusi program. Di sini digunakan EventQueue.invokeLater() untuk menjalankan GUI di thread khusus event-dispatching. Ini adalah praktik standar di Swing untuk menjaga konsistensi dan keamanan tampilan GUI. Di dalamnya, objek InsertionSortGUI dibuat dan ditampilkan menggunakan setVisible(true).

```
public static void main(String[] args) {
    EventQueue.invokeLater(() -> {
        try {
            InsertionSortGUI frame = new InsertionSortGUI();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    });
}
```

iv. Konstruktor InsertionSortGUI()

Konstruktor ini bertanggung jawab untuk menyusun seluruh tampilan GUI. Judul jendela diatur menjadi "Insertion Sort Langkah per Langkah", ukuran ditetapkan, dan layout utama menggunakan BorderLayout. Panel input dibuat menggunakan FlowLayout, berisi label petunjuk, JTextField untuk input angka, dan tombol setButton untuk menetapkan array. panelArray disiapkan untuk menampilkan elemen array dalam bentuk JLabel. Tombol kontrol stepButton dan resetButton ditempatkan di panel bawah. stepButton awalnya dinonaktifkan karena proses sorting belum dimulai. Sebagai tambahan, stepArea adalah area teks dengan font monospaced untuk menampilkan log proses sorting, dilengkapi dengan JScrollPane agar dapat di-scroll.

```
    public InsertionSortGUI() {
        setTitle("Insertion Sort Langkah per Langkah");
        setSize(750, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout());

        JPanel inputPanel = new JPanel(new FlowLayout());
        inputField = new JTextField(30);
```

```
        setButton = new JButton("Set Array");
        inputPanel.add(new JLabel("Masukkan angka (pisahkan dengan
koma):"));
        inputPanel.add(inputField);
        inputPanel.add(setButton);

        panelArray = new JPanel(new FlowLayout());

        JPanel controlPanel = new JPanel(new FlowLayout());
        stepButton = new JButton("Langkah Selanjutnya");
        resetButton = new JButton("Reset");
        stepButton.setEnabled(false);
        controlPanel.add(stepButton);
        controlPanel.add(resetButton);

        stepArea = new JTextArea(8, 60);
        stepArea.setEditable(false);
        stepArea.setFont(new Font("Monospaced", Font.PLAIN, 14));
        JScrollPane scrollPane = new JScrollPane(stepArea);

        add(inputPanel, BorderLayout.NORTH);
        add(panelArray, BorderLayout.CENTER);
        add(controlPanel, BorderLayout.SOUTH);
        add(scrollPane, BorderLayout.EAST);

        setButton.addActionListener(e -> setArrayFromInput());
        stepButton.addActionListener(e -> performStep());
        resetButton.addActionListener(e -> reset());
    }
```

v. Method setArrayFromInput()

Method ini akan membaca input dari JTextField, memisahkan string berdasarkan koma, dan mengonversinya ke dalam array integer. Jika input kosong atau tidak valid, maka akan ditampilkan pesan kesalahan. Jika input valid, maka GUI di-reset dan panelArray akan menampilkan elemen array dalam bentuk label. Proses sorting kemudian dimulai, ditandai dengan mengaktifkan tombol stepButton.

```
private void setArrayFromInput() {
    String text = inputField.getText().trim();
    if (text.isEmpty()) return;

    try {
        String[] parts = text.split(",");
        array = new int[parts.length];
        for (int k = 0; k < parts.length; k++) {
            array[k] = Integer.parseInt(parts[k].trim());
        }
```

```
            i = 1;
            stepCount = 1;
            sorting = true;
            stepButton.setEnabled(true);
            stepArea.setText("");

            panelArray.removeAll();
            labelArray = new JLabel[array.length];
            for (int k = 0; k < array.length; k++) {
                labelArray[k] = new JLabel(String.valueOf(array[k]));
                labelArray[k].setPreferredSize(new Dimension(40, 40));
                labelArray[k].setHorizontalAlignment(SwingConstants.CENTER);

        labelArray[k].setBorder(BorderFactory.createLineBorder(Color.BLACK));
                panelArray.add(labelArray[k]);
            }
            panelArray.revalidate();
            panelArray.repaint();

        } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(this, "Masukkan hanya angka yang
        dipisahkan dengan koma.", "Input Tidak Valid",
        JOptionPane.ERROR_MESSAGE);
        }
    }
```

vi. Method performStep()

Method ini menjalankan satu langkah dari algoritma Insertion Sort. Elemen key dari indeks i diambil, lalu dibandingkan dengan elemen sebelumnya. Jika lebih kecil, maka pergeseran elemen dilakukan sampai menemukan posisi yang tepat. Hasil array setelah setiap langkah akan ditampilkan di stepArea, dan tampilan GUI diperbarui menggunakan updateLabels(). Jika proses sudah mencapai akhir array, tombol stepButton akan dinonaktifkan dan ditampilkan pesan bahwa sorting telah selesai.

```
    private void performStep() {
        if (i < array.length && sorting) {
            int key = array[i];
            int j = i - 1;

            StringBuilder stepLog = new StringBuilder();
            stepLog.append("Langkah ").append(stepCount)
                    .append(": Memasukkan ").append(key).append("\n");

            while (j >= 0 && array[j] > key) {
                array[j + 1] = array[j];
                j--;
            }
            array[j + 1] = key;
```

```
            updateLabels();

            stepLog.append("Hasil: ").append(arrayToString(array)).append("\n\n");
            stepArea.append(stepLog.toString());

            i++;
            stepCount++;

            if (i == array.length) {
                sorting = false;
                stepButton.setEnabled(false);
                JOptionPane.showMessageDialog(this, "Sorting selesai!");
            }
        }
    }
```

vii. Method reset()

Method reset digunakan untuk menghapus semua input, mereset array dan tampilan GUI, serta mengembalikan semua variabel kontrol ke kondisi awal. Hal ini memungkinkan pengguna untuk mencoba kembali proses sorting dari awal.

```
private void reset() {
    inputField.setText("");
    panelArray.removeAll();
    panelArray.revalidate();
    panelArray.repaint();
    stepArea.setText("");
    stepButton.setEnabled(false);
    sorting = false;
    i = 1;
    stepCount = 1;
}
```

viii. Method updateLabels()

Method ini memperbarui teks dari seluruh labelArray sesuai dengan isi terbaru dari array, sehingga tampilan visual selalu sinkron dengan data aktual.

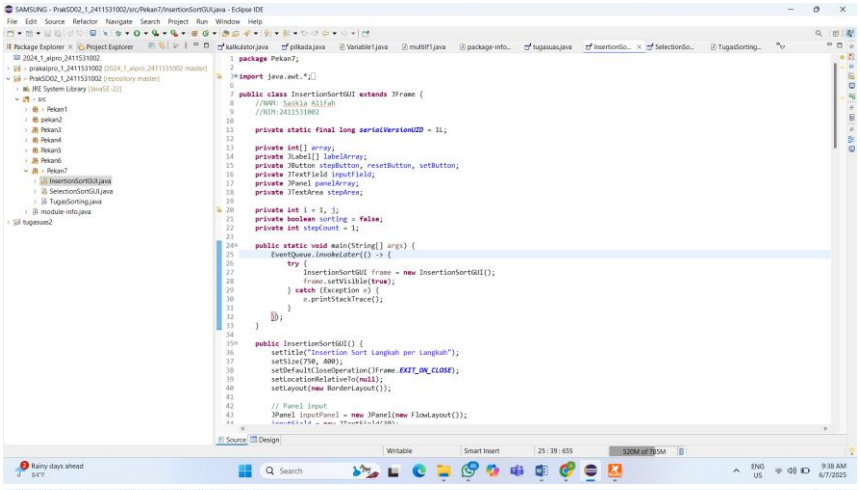
```
private void updateLabels() {
    for (int k = 0; k < array.length; k++) {
        labelArray[k].setText(String.valueOf(array[k]));
    }
}
```

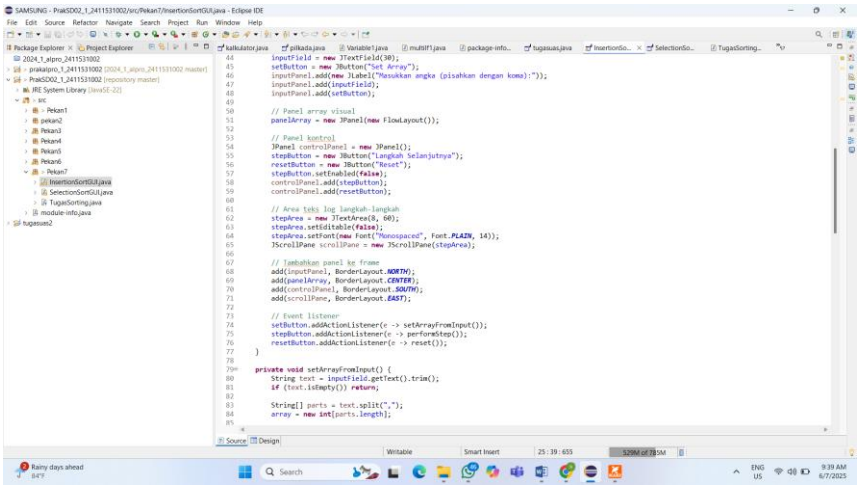
ix. Method arrayToString(int[] arr)

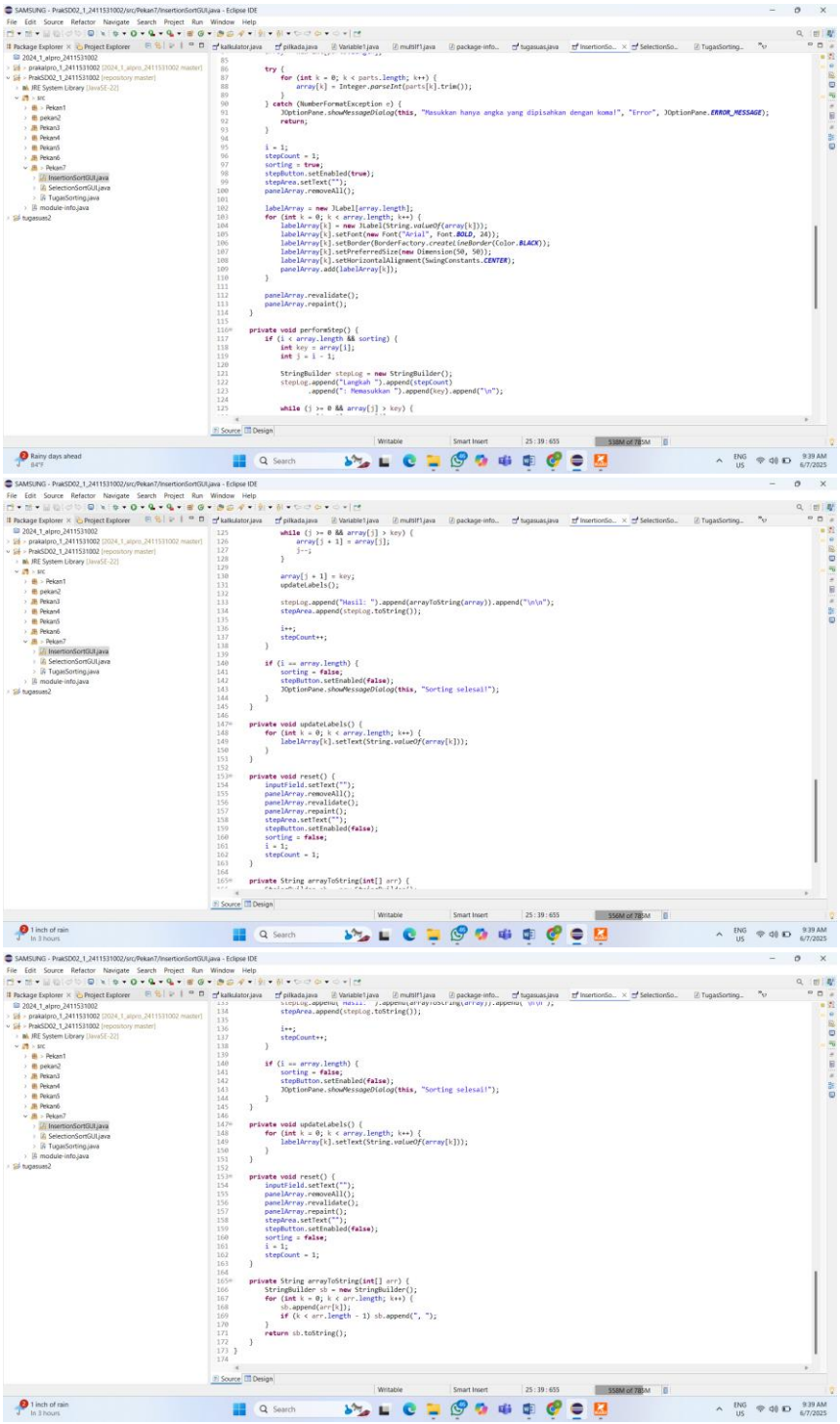
Method ini membantu mengubah array integer menjadi string yang dipisahkan koma, seperti 1, 2, 3. Fungsinya adalah untuk menampilkan hasil array di log langkah-langkah.

```
private String arrayToString(int[] arr) {
    StringBuilder sb = new StringBuilder();
    for (int k = 0; k < arr.length; k++) {
        sb.append(arr[k]);
        if (k < arr.length - 1) sb.append(", ");
    }
    return sb.toString();
}
```

c) Syntax







d) Diagram Alir

- Mulai
- i = 1 hingga n-1
- key = array[i]
- j = i-1
- Selama j >= 0 dan array[j] > key, array[j+1] = array[j], j--
- array[j+1] = key

- i++
- Selesai jika i = n

e) Error Handling

- **Input Tidak Valid**: Jika pengguna memasukkan karakter non-numerik atau format salah, JOptionPane akan menampilkan pesan kesalahan.
- **Tombol Salah Urutan**: Tombol "Langkah Selanjutnya" tidak dapat dijalankan sebelum array diset.
- **ArrayIndexOutOfBounds**: Diantisipasi dengan menonaktifkan tombol saat proses selesai.

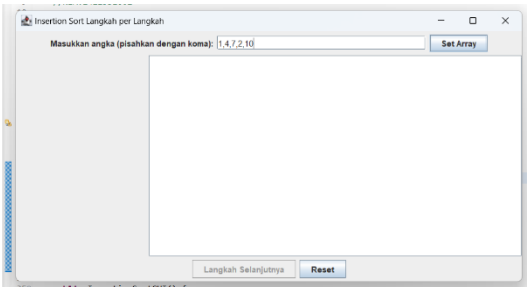
f) Alur Sintaks Program

1. Program dimulai dan GUI ditampilkan.
2. Pengguna memasukkan input angka.
3. Menekan tombol "Set Array" untuk menginisialisasi visualisasi.
4. Tombol "Langkah Selanjutnya" ditekan berulang untuk melakukan penyisipan satu elemen ke bagian array yang telah diurutkan.
5. Label diperbarui dan log dicatat di JTextArea.
6. Setelah sorting selesai, dialog pop-up ditampilkan.
7. Tombol "Reset" tersedia untuk memulai ulang proses.

g) Output

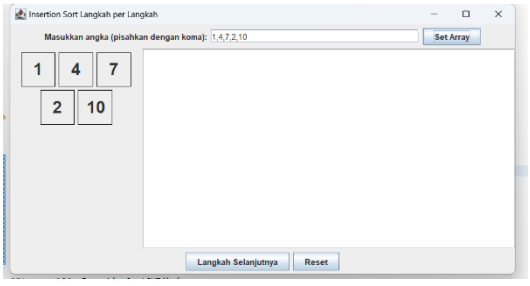
i. **Masukkan angka.**

masukkan angka yang ingin di urutkan 1,4,7,2,10.



ii. **Tekan “Set Array”.**

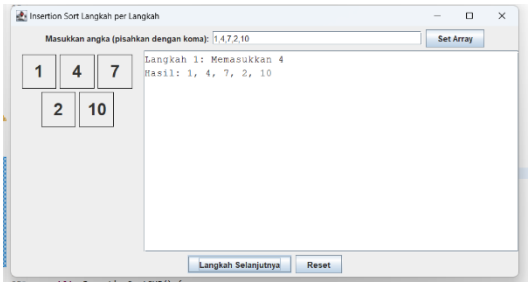
Setelah ditekan set array, maka dikiri akan muncul kotak kotak angka seperti ini.



iii. **Langkah 1 (i = 1):**

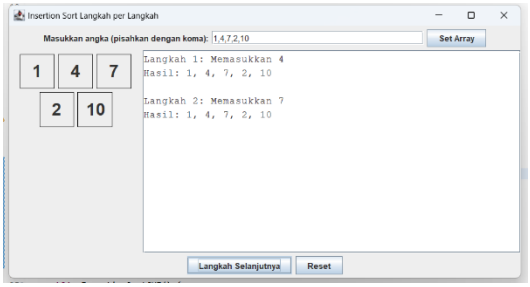
- Kartu yang ingin disisipkan: 4
- Dibandingkan dengan 1 (sudah lebih besar), tidak perlu geser.

- Array tetap: [1, 4, 7, 2, 10]



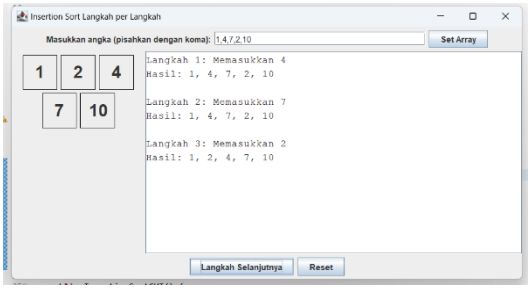
iv. Langkah 2 (i = 2):

- Kartu yang ingin disisipkan: 7
- Dibandingkan dengan 4 → tidak perlu geser.
- Array tetap: [1, 4, 7, 2, 10]



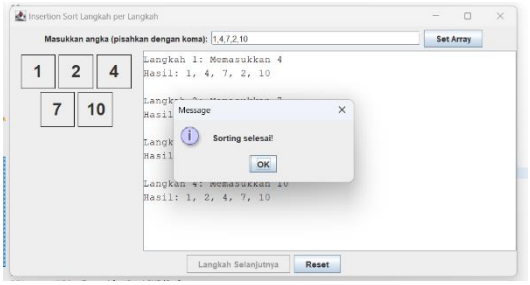
v. Langkah 3 (i = 3):

- Kartu yang ingin disisipkan: 2
- Dibandingkan dengan 7 → geser 7 ke kanan
- Dibandingkan dengan 4 → geser 4 ke kanan
- Dibandingkan dengan 1 → tidak perlu geser
- Sisipkan 2 di posisi 1
- Array menjadi: [1, 2, 4, 7, 10]



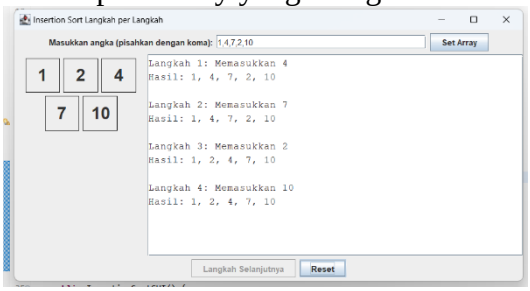
vi. Langkah 4 (i = 4):

- Kartu yang ingin disisipkan: 10
- Dibandingkan dengan 7 → tidak perlu geser
- Sisipkan 10 di posisi 4 (tetap)
- Array tetap: [1, 2, 4, 7, 10]



vii. Kesimpulan output

Setelah 4 langkah, array berhasil diurutkan secara ascending. Elemen yang benar-benar disisipkan ulang hanyalah angka 2, karena berada di posisi yang salah. Sisanya sudah dalam posisi tepat sejak awal, sehingga tidak banyak geseran yang terjadi. Inilah yang membuat Insertion Sort sangat efisien pada array yang sebagian besar sudah terurut.



2. Kelas SelectionSortGUI

a) Deskripsi Kelas

SelectionSortGUI adalah kelas utama untuk visualisasi algoritma Selection Sort dalam bentuk aplikasi GUI. Kelas ini mewarisi JFrame, sehingga mampu membangun window utama aplikasi. Komponen utamanya terdiri dari input field untuk menerima array dari pengguna, panel untuk menampilkan array secara visual, tombol kontrol (Set Array, Langkah Selanjutnya, Reset), dan area teks untuk menampilkan log langkah-langkah sorting. Proses sorting dilakukan secara bertahap setiap kali tombol "Langkah Selanjutnya" ditekan, sehingga pengguna dapat mengikuti proses sorting secara interaktif dan memahami mekanisme Selection Sort secara visual

b) Penjelasan Fungsi dan Cara Kerja

i. Package dan Import

Program ini berada dalam package Pekan7, yang menunjukkan bahwa file ini merupakan bagian dari modul atau folder Pekan7. Package membantu dalam mengelompokkan kelas Java agar lebih terorganisir. Import java.awt.* digunakan untuk komponen GUI dasar seperti BorderLayout dan FlowLayout, sedangkan javax.swing.* digunakan untuk komponen

GUI lanjutan seperti JFrame, JLabel, JButton, JTextField, dan JScrollPane. Meskipun EmptyBorder diimpor, dalam kode ini tidak digunakan secara eksplisit.

```
package Pekan7;

import java.awt.*;
import javax.swing.*;
import javax.swing.border.EmptyBorder;
```

ii. Deklarasi Kelas dan Variabel Atribut

Kelas SelectionSortGUI merupakan turunan dari JFrame, yang berarti kelas ini mewarisi semua kemampuan jendela aplikasi Java Swing. Di dalamnya terdapat sejumlah variabel penting untuk mendukung proses visualisasi Selection Sort. array menyimpan nilai-nilai yang dimasukkan oleh pengguna. labelArray adalah array dari JLabel yang merepresentasikan tiap elemen array secara visual. Terdapat tiga tombol utama: setButton untuk memulai array, stepButton untuk melakukan satu langkah sorting, dan resetButton untuk mengatur ulang tampilan. inputField digunakan untuk menerima input angka dari pengguna. Panel panelArray akan menampilkan array dalam bentuk label, sementara stepArea akan mencatat langkah-langkah sorting. Variabel i, j, dan minIndex digunakan dalam logika algoritma Selection Sort, serta stepCount untuk mencatat jumlah langkah yang telah dilakukan.

```
public class SelectionSortGUI extends JFrame {
    private int[] array;
    private JLabel[] labelArray;
    private JButton stepButton, resetButton, setButton;
    private JTextField inputField;
    private JPanel panelArray;
    private JTextArea stepArea;

    private int i = 0, j = 1, minIndex;
    private boolean sorting = false;
    private int stepCount = 1;
}
```

iii. Method main()

Method main() adalah titik masuk utama program Java. Untuk memastikan tampilan GUI berjalan dengan aman dan konsisten, kode GUI dijalankan di dalam EventQueue.invokeLater(), yang memastikan eksekusi dilakukan pada Event Dispatch Thread (EDT), yaitu thread khusus untuk memproses event Swing.

```
public static void main(String[] args) {
    EventQueue.invokeLater(() -> {
        try {
```

```
            SelectionSortGUI frame = new SelectionSortGUI();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    });
}
```

iv. Konstruktor SelectionSortGUI()

Konstruktor SelectionSortGUI bertugas menyusun seluruh komponen GUI. Judul jendela ditetapkan, ukuran dan lokasi jendela diatur, serta layout utama ditentukan sebagai BorderLayout. Panel input di bagian atas menggunakan FlowLayout, berisi JTextField dan tombol setButton. panelArray berada di tengah sebagai tempat menampilkan visualisasi array. Di bagian bawah terdapat controlPanel dengan tombol stepButton dan resetButton. Di sisi kanan, stepArea digunakan untuk mencatat dan menampilkan setiap langkah sorting, dibungkus dalam JScrollPane. Setiap tombol juga dihubungkan dengan *event listener* masing-masing.

```
public SelectionSortGUI() {
    setTitle("Selection Sort Langkah per Langkah");
    setSize(750, 400);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLocationRelativeTo(null);
    setLayout(new BorderLayout());

    JPanel inputPanel = new JPanel(new FlowLayout());
    inputField = new JTextField(30);
    setButton = new JButton("Set Array");
    inputPanel.add(new JLabel("Masukkan angka (pisahkan dengan
koma):"));
    inputPanel.add(inputField);
    inputPanel.add(setButton);

    panelArray = new JPanel(new FlowLayout());

    JPanel controlPanel = new JPanel(new FlowLayout());
    stepButton = new JButton("Langkah Selanjutnya");
    resetButton = new JButton("Reset");
    stepButton.setEnabled(false);
    controlPanel.add(stepButton);
    controlPanel.add(resetButton);

    stepArea = new JTextArea(8, 60);
    stepArea.setEditable(false);
    stepArea.setFont(new Font("Monospaced", Font.PLAIN, 14));
    JScrollPane scrollPane = new JScrollPane(stepArea);

    add(inputPanel, BorderLayout.NORTH);
```

```
    add(panelArray, BorderLayout.CENTER);
    add(controlPanel, BorderLayout.SOUTH);
    add(scrollPane, BorderLayout.EAST);

    setButton.addActionListener(e -> setArrayFromInput());
    stepButton.addActionListener(e -> performStep());
    resetButton.addActionListener(e -> reset());
}
```

v. Method setArrayFromInput()

Method ini mengambil input dari inputField, memisahkannya dengan koma, dan mengonversi tiap bagian menjadi angka integer untuk disimpan dalam array. Jika terdapat karakter non-angka atau format salah, akan muncul dialog kesalahan. Setelah array diset, elemen array ditampilkan secara visual dalam bentuk JLabel pada panelArray. Tombol "Langkah Selanjutnya" diaktifkan sebagai tanda bahwa sorting bisa dimulai.

```
private void setArrayFromInput() {
    String text = inputField.getText().trim();
    if (text.isEmpty()) return;

    try {
        String[] parts = text.split(",");
        array = new int[parts.length];
        for (int k = 0; k < parts.length; k++) {
            array[k] = Integer.parseInt(parts[k].trim());
        }

        i = 0;
        j = i + 1;
        stepCount = 1;
        sorting = true;
        stepButton.setEnabled(true);
        stepArea.setText("");

        panelArray.removeAll();
        labelArray = new JLabel[array.length];
        for (int k = 0; k < array.length; k++) {
            labelArray[k] = new JLabel(String.valueOf(array[k]));
            labelArray[k].setPreferredSize(new Dimension(40, 40));
            labelArray[k].setHorizontalAlignment(SwingConstants.CENTER);

labelArray[k].setBorder(BorderFactory.createLineBorder(Color.BLACK))
;
            panelArray.add(labelArray[k]);
        }
        panelArray.revalidate();
        panelArray.repaint();
```

```
            minIndex = i;

        } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(this, "Masukkan hanya angka yang
dipisahkan dengan koma.", "Input Tidak Valid",
JOptionPane.ERROR_MESSAGE);
        }
    }
```

vi. Method performStep()

Method ini menjalankan satu langkah dari algoritma Selection Sort. Pada setiap iterasi, elemen dari indeks i dibandingkan dengan seluruh elemen sesudahnya (j). Jika ditemukan elemen yang lebih kecil, minIndex akan diperbarui. Setelah semua perbandingan selesai, elemen ke-i akan ditukar dengan elemen minIndex jika memang berbeda. Perubahan ini kemudian dicatat dalam stepArea dan tampilan GUI diperbarui menggunakan updateLabels(). Ketika iterasi telah mencapai akhir array, sorting dihentikan dan tombol langkah dinonaktifkan.

```
private void performStep() {
    if (i < array.length - 1 && sorting) {
        if (j < array.length) {
            if (array[j] < array[minIndex]) {
                minIndex = j;
            }
            j++;
        } else {
            int temp = array[i];
            array[i] = array[minIndex];
            array[minIndex] = temp;

            updateLabels();

            StringBuilder stepLog = new StringBuilder();
            stepLog.append("Langkah ").append(stepCount)
                    .append(": Menukar indeks ").append(i)
                    .append(" dengan indeks ").append(minIndex).append("\n");
            stepLog.append("Hasil:
").append(arrayToString(array)).append("\n\n");
            stepArea.append(stepLog.toString());

            i++;
            minIndex = i;
            j = i + 1;
            stepCount++;

            if (i >= array.length - 1) {
                sorting = false;
                stepButton.setEnabled(false);
```

```
                JOptionPane.showMessageDialog(this, "Sorting selesai!");
            }
        }
    }
}
```

vii. Method reset()

Method reset() digunakan untuk mengatur ulang seluruh elemen GUI dan variabel logika agar pengguna dapat mengulang proses sorting dari awal. Input, tampilan array, serta area langkah akan dikosongkan. Sorting akan dihentikan dan indeks direset.

```
private void reset() {
    inputField.setText("");
    panelArray.removeAll();
    panelArray.revalidate();
    panelArray.repaint();
    stepArea.setText("");
    stepButton.setEnabled(false);
    sorting = false;
    i = 0;
    j = 1;
    stepCount = 1;
}
```

viii. Method updateLabels()

Method ini bertanggung jawab memperbarui nilai-nilai labelArray untuk merefleksikan isi terbaru dari array integer. Ini memastikan bahwa visualisasi array tetap sinkron dengan perubahan nilai aktual array setelah pertukaran.

```
private void updateLabels() {
    for (int k = 0; k < array.length; k++) {
        labelArray[k].setText(String.valueOf(array[k]));
    }
}
```

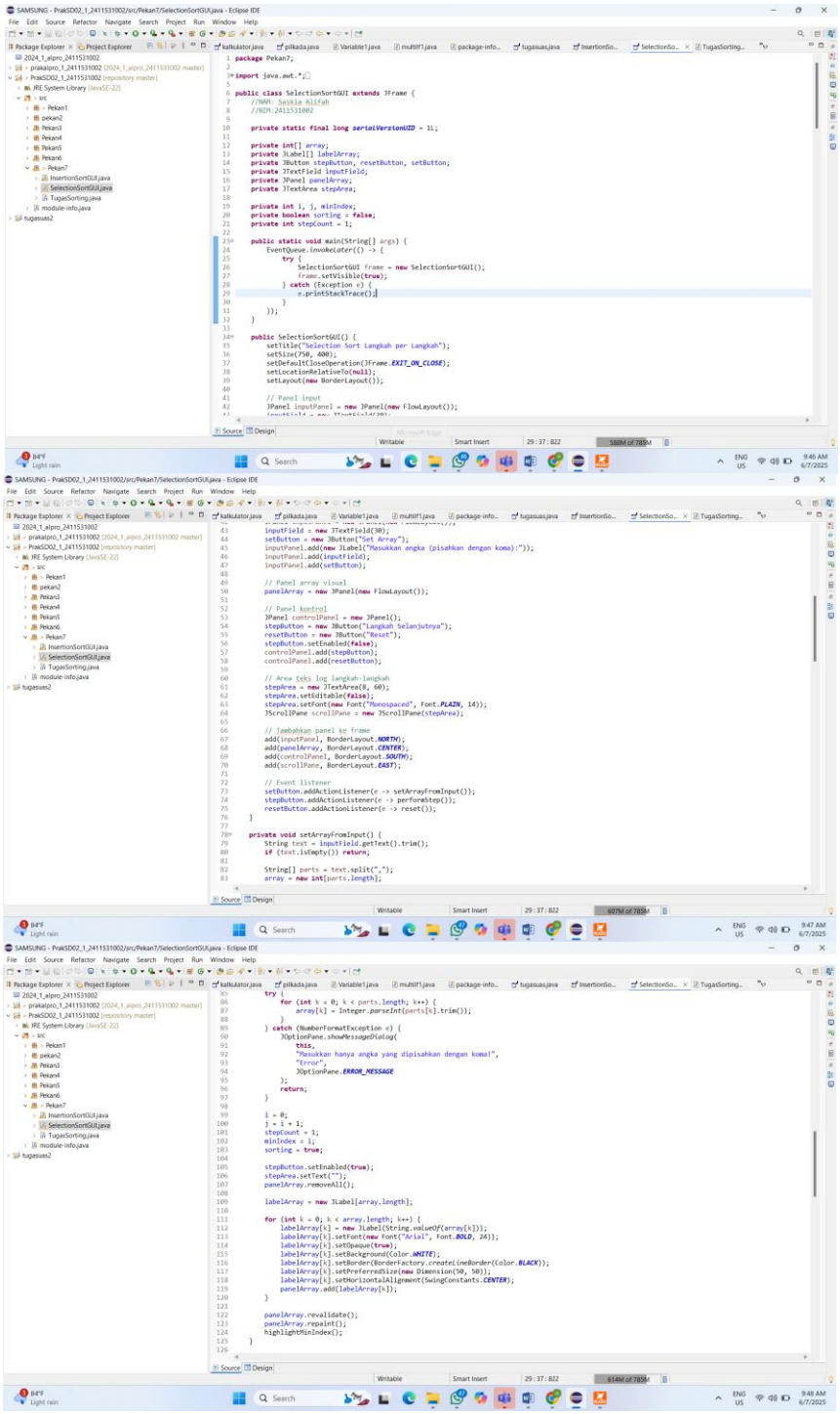
ix. Method arrayToString(int[] arr)

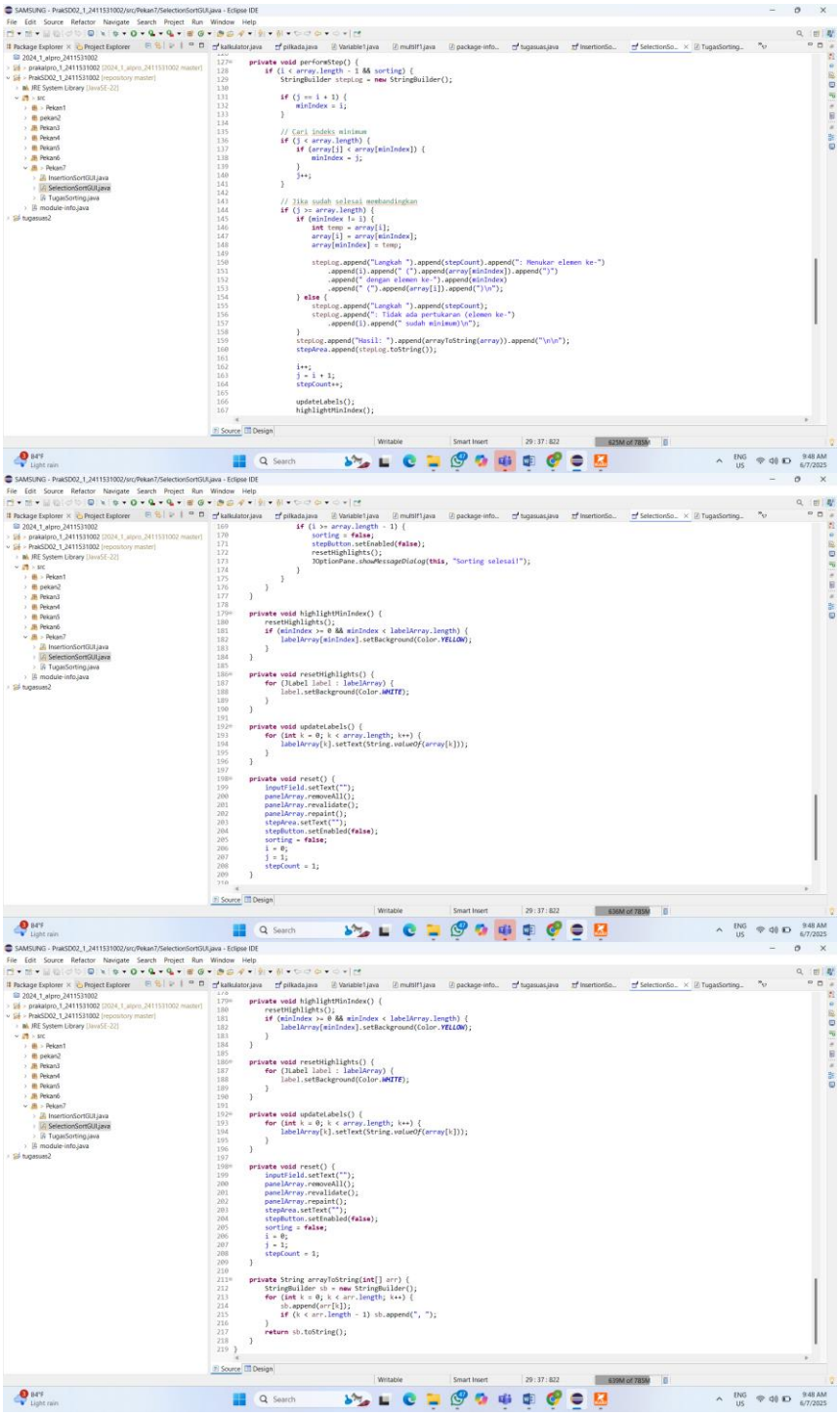
Method ini mengubah array integer menjadi format string dengan elemen-elemen yang dipisahkan koma, sehingga dapat dicetak dengan mudah di stepArea.

```
private String arrayToString(int[] arr) {
    StringBuilder sb = new StringBuilder();
    for (int k = 0; k < arr.length; k++) {
        sb.append(arr[k]);
        if (k < arr.length - 1) sb.append(", ");
```

```
        }
        return sb.toString();
    }
```

c) Syntax:





d) Diargram alur

- Mulai
- i = 0 hingga n-2
- minIndex = i
- j = i+1 hingga n-1
- Jika array[j] < array[minIndex], minIndex = j
- Tukar array[i] dan array[minIndex]
- i++

- Selesai jika i = n-1

e) Error Handling

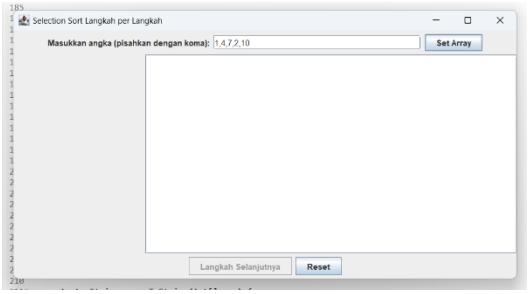
- Validasi input menggunakan blok try-catch untuk menangani NumberFormatException.
- Tombol tidak dapat digunakan saat proses belum dimulai atau sudah selesai.
- Pop-up interaktif untuk memberi tahu pengguna jika input salah atau sorting selesai.

f) Alur Sintaks Program

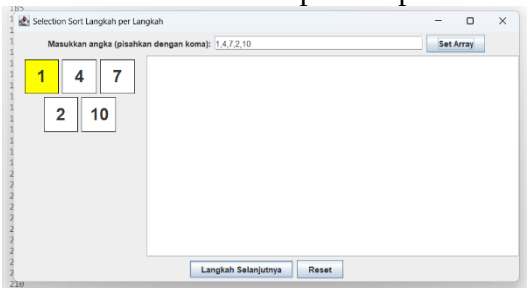
1. GUI dibuat dan ditampilkan.
2. Pengguna mengetik input array dan menekan "Set Array".
3. Visualisasi array muncul sebagai label.
4. Tombol "Langkah Selanjutnya" digunakan untuk menjalankan proses Selection Sort satu per satu.
5. Setelah proses selesai, muncul dialog notifikasi.
6. Tombol "Reset" digunakan untuk memulai ulang proses.

g) Output Program

i. **Masukkan angka**
masukkanlah yang ingin di urutkan, disini saya memasukkan 1,4,7,2,10.



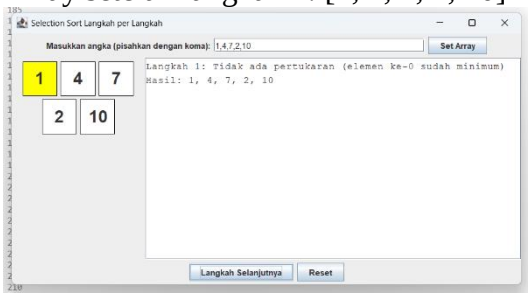
ii. **Pencet set array**
maka akan muncul tampilan seperti kotak kotak di sebelah kiri.



iii. **Iterasi 1 (currentStep = 0)**

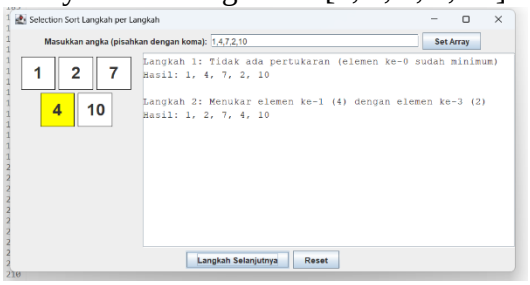
- Cari nilai minimum dari indeks 0 sampai akhir (0–4):
  - Elemen 1 (indeks 0) sebagai minimum awal.
  - Bandingkan dengan 4, 7, 2, 10 → nilai minimum tetap 1.

- Tukar array[0] dengan array[0] (tidak ada perubahan).
- Array setelah langkah 1: [1, 4, 7, 2, 10]



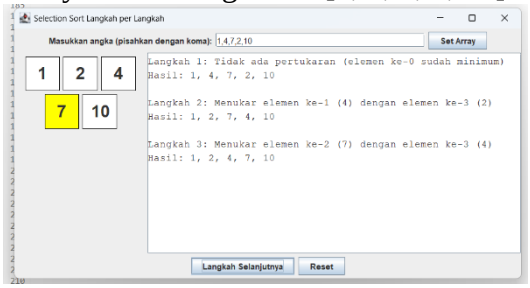
iv. Iterasi 2 (currentStep = 1)

- Cari nilai minimum dari indeks 1 sampai akhir (1–4):
  - Mulai dari 4 (indeks 1).
  - Bandingkan dengan 7, 2, 10 → nilai minimum adalah 2 (indeks 3).
- Tukar array[1] dengan array[3].
- Array setelah langkah 2: [1, 2, 7, 4, 10]



v. Iterasi 3 (currentStep = 2)

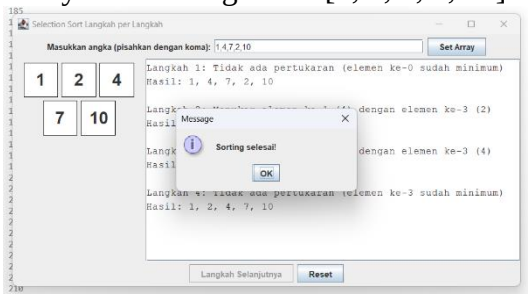
- Cari nilai minimum dari indeks 2 sampai akhir (2–4):
  - Mulai dari 7 (indeks 2).
  - Bandingkan dengan 4, 10 → nilai minimum adalah 4 (indeks 3).
- Tukar array[2] dengan array[3].
- Array setelah langkah 3: [1, 2, 4, 7, 10]



vi. Iterasi 4 (currentStep = 3)

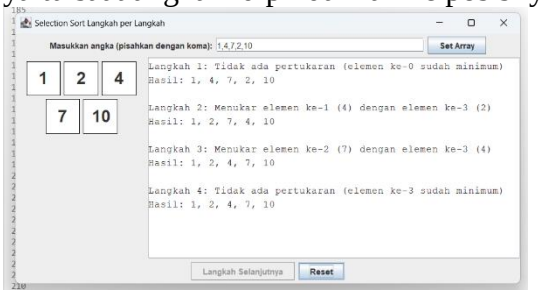
- Cari nilai minimum dari indeks 3 sampai akhir (3–4):
  - Mulai dari 7 (indeks 3).
  - Bandingkan dengan 10 → nilai minimum tetap 7.
- Tukar array[3] dengan array[3] (tidak ada perubahan).

- Array setelah langkah 4: [1, 2, 4, 7, 10]



vii. **Kesimpulan Output:**

Pada **Selection Sort**, langkah-langkah ditentukan dengan mencari elemen terkecil dari sisa array dan menukarnya dengan elemen pada posisi saat ini. Terlihat bahwa hanya terjadi satu pertukaran signifikan pada langkah ke-3, yaitu saat angka 2 dipindahkan ke posisi yang benar.



D. KESIMPULAN

Menunjukkan bahwa pemahaman algoritma sorting dapat meningkat signifikan dengan bantuan visualisasi berbasis GUI. Melalui aplikasi yang dikembangkan menggunakan Java Swing, setiap langkah proses Selection Sort dan Insertion Sort dapat diamati secara langsung, mulai dari input data, proses perbandingan, pergeseran, hingga hasil akhir array yang terurut. Visualisasi ini tidak hanya memudahkan dalam memahami logika traversal dan operasi setiap algoritma, tetapi juga membantu dalam mengidentifikasi kelebihan dan kekurangan masing-masing metode, seperti efisiensi Insertion Sort pada data yang hampir terurut dan kesederhanaan Selection Sort dalam implementasi.