LAPORAN PRAKTIKUM STRUKTUR DATA

PEKAN 7: IMPLEMENTASI DAN VISUALISASI ALGORITMA INSERTION SORT DAN SELECTION SORT MENGGUNAKAN GUI PADA JAVA

OLEH

Abdullah Al Ramadhani (2411533016)

DOSEN PENGAMPU

DR. Wahyudi, S.T, M.T

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

2025

A. Pendahuluan

Algoritma **insertion sort** dan **selection sort** merupakan dua metode pengurutan yang sering digunakan karena logikanya sederhana dan mudah dipahami. Keduanya cocok untuk data berukuran kecil atau sebagai pengantar dalam mempelajari sorting, meskipun memiliki pendekatan yang berbeda dalam menyusun elemen-elemen array.

Dalam program yang dibuat menggunakan **Java** dan ditampilkan melalui **GUI berbasis Swing**, kedua algoritma ini divisualisasikan agar pengguna dapat mengikuti proses sorting secara interaktif. **InsertionSortGUI** menampilkan langkah-langkah pengurutan dengan insertion sort disertai pembaruan visual dan log teks. Sementara itu, **SelectionSortGUI** menunjukkan proses selection sort dengan penyorotan elemen terkecil dan perpindahannya, sehingga logika pemilihan dan pertukaran elemen bisa lebih mudah dipahami.

B. Tujuan

Tujuan dari praktikum ini adalah:

1. Mengembangkan implementasi algoritma insertion sort dan selection sort menggunakan bahasa pemrograman Java.

2. Merancang dan membangun GUI interaktif yang mampu menampilkan proses pengurutan data secara visual.

3. Menyediakan tampilan informatif yang memudahkan pengguna dalam memahami setiap langkah kerja dari kedua algoritma pengurutan tersebut.

C. Langkah-langkah Pengerjaan

a) InsertionSortGUI

1. Langkah pertama adalah membuat kelas JFrame baru. Caranya, klik kanan pada bagian package, pilih Other, lalu buka folder WindowBuilder dan pilih JFrame. Setelah itu beri nama kelasnya "InsertionSortGUI". Di awal, impor beberapa kelas penting seperti JFrame, JButton, JTextField, dan komponen GUI lainnya yang dibutuhkan untuk membangun antarmuka pengguna dalam aplikasi

```
1 package Pekan7;
2
3 import java.awt.*;
4 import java.awt.event.*;
5 import javax.swing.*;
6 import javax.swing.border.Border;
```

2. Kelas InsertionSortGUI ini berperan sebagai kelas utama yang mewarisi JFrame, sehingga berfungsi sebagai jendela utama aplikasi. Di dalamnya, kita mendeklarasikan berbagai variabel, seperti array untuk menyimpan angka, labelArray untuk menampilkan angka-angka, serta tombol-tombol seperti stepButton, resetButton, dan setButton sebagai kontrol antarmuka.

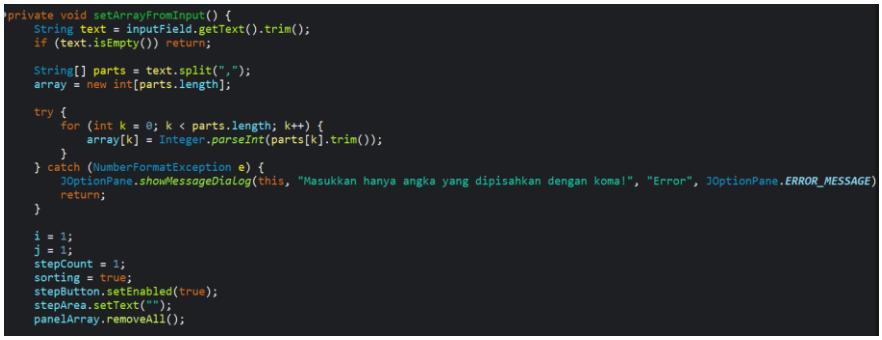
```
8  public class InsertionSortGUI extends JFrame {
9      private static final long serialVersionUID = 1L;
10     private int[] array;
11     private JLabel[] labelArray;
12     private JButton stepButton, resetButton, setButton;
13     private JTextField inputField;
14     private JPanel panelArray;
15     private JTextArea stepArea;
16
17     private int i = 1, j;
18     private boolean sorting = false;
19     private int stepCount = 1;
```

3. Desain antarmuka dibuat dengan menetapkan judul menggunakan setTitle dan menentukan ukuran jendela. Selanjutnya, dibuat beberapa panel, yaitu panel input, panel array, panel kontrol, dan area untuk menampilkan langkah-langkah proses sorting. Semua elemen dan panel tersebut disusun dan ditambahkan ke dalam frame menggunakan metode add(...).

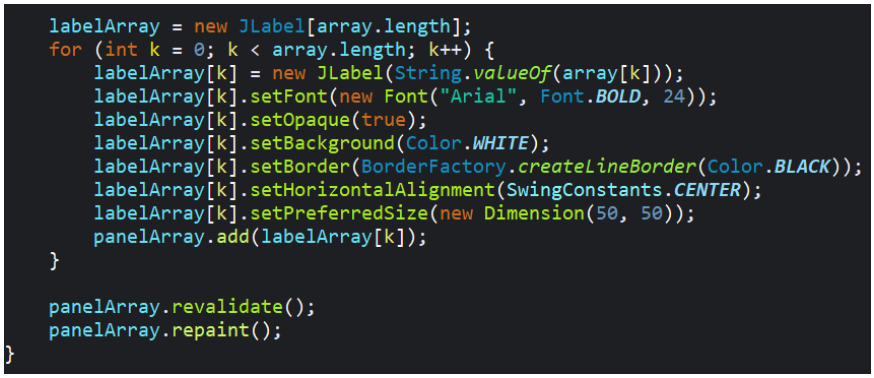
```
public InsertionSortGUI() {
    setTitle("Insertion Sort langkah per langkah");
    setSize(800, 400);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLocationRelativeTo(null);
    setLayout(new BorderLayout());

    // Panel input
    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);

    // Panel array visual
    panelArray = new JPanel();
    panelArray.setLayout(new FlowLayout());

    // Panel kontrol
    JPanel controlPanel = new JPanel();
    stepButton = new JButton("Langkah Selanjutnya");
    resetButton = new JButton("Reset");
    stepButton.setEnabled(false);
    controlPanel.add(stepButton);
    controlPanel.add(resetButton);
```

```
    // Area langkah
    stepArea = new JTextArea(10, 25);
    stepArea.setEditable(false);
    stepArea.setFont(new Font("Monospaced", Font.PLAIN, 14));
    JScrollPane scrollPane = new JScrollPane(stepArea);

    // Tambahkan ke frame
    add(inputPanel, BorderLayout.NORTH);
    add(panelArray, BorderLayout.CENTER);
    add(controlPanel, BorderLayout.SOUTH);
    add(scrollPane, BorderLayout.EAST);

    // Event set array
    setButton.addActionListener(e -> setArrayFromInput());

    // Event langkah selanjutnya
    stepButton.addActionListener(e -> performStep());

    // Event Reset
    resetButton.addActionListener(e -> reset());
}
```

4. Fungsi pengaturan array dijalankan ketika tombol "Set Array" diklik. Program akan mengambil input dari pengguna melalui inputField, lalu memisahkan setiap angka menggunakan koma. Angka-angka ini diubah ke dalam bentuk integer dan disimpan dalam array. Jika input tidak valid (misalnya huruf atau karakter selain angka), akan muncul pesan error. Setelah array berhasil dibuat, angka-angka akan ditampilkan di layar melalui label yang telah disiapkan.



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

    String[] parts = text.split(",");
    array = new int[parts.length];

    try {
        for (int k = 0; k < parts.length; k++) {
            array[k] = Integer.parseInt(parts[k].trim());
        }
    } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog(this, "Masukkan hanya angka yang dipisahkan dengan koma!", "Error", JOptionPane.ERROR_MESSAGE);
        return;
    }

    i = 1;
    j = 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].setFont(new Font("Arial", Font.BOLD, 24));
        labelArray[k].setOpaque(true);
        labelArray[k].setBackground(Color.WHITE);
        labelArray[k].setBorder(BorderFactory.createLineBorder(Color.BLACK));
        labelArray[k].setHorizontalAlignment(SwingConstants.CENTER);
        labelArray[k].setPreferredSize(new Dimension(50, 50));
        panelArray.add(labelArray[k]);
    }

    panelArray.revalidate();
    panelArray.repaint();
}
```

5. Proses insertion sort dilakukan saat tombol "Langkah Selanjutnya" ditekan. Algoritma akan mengambil elemen pada indeks tertentu sebagai key, lalu membandingkannya dengan elemen-elemen sebelumnya. Jika ada yang lebih besar dari key, elemen tersebut digeser ke kanan hingga posisi yang sesuai ditemukan untuk key. Setelah setiap langkah, tampilan array diperbarui dan area langkah (stepArea) akan mencatat proses yang berlangsung. Setelah seluruh data selesai diurutkan, tombol akan dinonaktifkan dan pesan "Sorting selesai!" akan muncul.

```
private void performStep() {
    if (i < array.length && sorting) {
        int key = array[i];
        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!");
        }
    }
}
```

6. Fungsi reset digunakan untuk mengembalikan aplikasi ke kondisi awal. Semua tampilan dan variabel akan dikosongkan, termasuk input dari pengguna, dan tombol "Langkah Selanjutnya" akan dimatikan agar pengguna bisa memulai kembali dari awal.

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

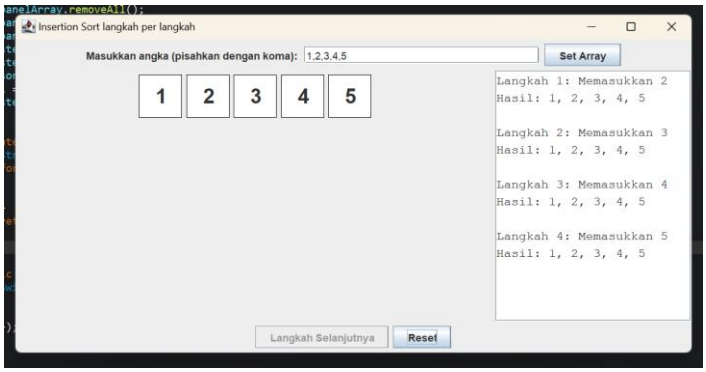
7. Fungsi konversi array ke string dibuat agar isi array dapat ditampilkan dalam bentuk teks di area langkah. Ini berguna untuk menampilkan penjelasan proses sorting dalam format yang lebih mudah dibaca.

```
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();

}
```

8. Buat method main sebagai titik awal program agar GUI bisa dijalankan.

```
public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        InsertionSortGUI gui = new InsertionSortGUI();
        gui.setVisible(true);
    });
}
```

9. Terakhir, jalankan program. Jika tampilan GUI muncul sesuai desain, berarti aplikasi telah berhasil dibuat dan berfungsi dengan baik.



b) SelectionSortGUI

1. Buatlah kelas baru bernama "SelectionSortGUI" yang merupakan turunan dari JFrame. Jangan lupa untuk mengimpor sejumlah kelas yang dibutuhkan untuk membangun antarmuka grafis, seperti JFrame untuk jendela utama, JButton untuk tombol-tombol, JTextField dan JTextArea untuk input dan area teks, serta JPanel dan FlowLayout untuk mengatur tata letak komponen dalam jendela aplikasi.

```
1 package Pekan7;
2
3 import java.awt.*;
5
```

2. Di dalam kelas utama SelectionSortGUI, deklarasikan berbagai variabel yang diperlukan, seperti array untuk menyimpan data yang akan diurutkan, labelArray untuk menampilkan angka-angka, dan tombol-tombol kontrol seperti stepButton, resetButton, serta setButton. Kelas ini akan bertugas mengatur seluruh jalannya aplikasi serta tampilan GUI-nya.

```
public class SelectionSortGUI extends JFrame {

    private static final long serialVersionUID = 1L;
    private int[] array;
    private JLabel[] labelArray;
    private JButton stepButton, resetButton, setButton;
    private JTextField inputField;
    private JPanel panelArray;
    private JTextArea stepArea;
    private int minIndex;


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

3. Rancang tampilan antarmuka pengguna dengan menetapkan judul jendela menjadi "Selection Sort Langkah per Langkah" dan atur ukuran jendelanya menjadi 750x400 piksel. Buat beberapa panel seperti panel input, panel array, panel kontrol, dan area langkah, untuk mengorganisir komponen GUI seperti pada program sebelumnya.

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

    JPanel textPanel = new JPanel(new FlowLayout());
    inputField = new JTextField(30);
    setButton = new JButton("Set array");
    textPanel.add(new JLabel("Input array (pisahkan dengan koma):"));
    textPanel.add(inputField);
    textPanel.add(setButton);

    panelArray = new JPanel();
    panelArray.setLayout(new FlowLayout());

    JPanel controlPanel = new JPanel();
    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(textPanel, 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());
}
```

4. Fungsi pengaturan array dipanggil ketika tombol "Set Array" ditekan. Program akan mengambil teks dari inputField, memisahkannya berdasarkan koma, lalu mengubahnya menjadi array bertipe integer. Bila ditemukan input yang tidak valid (seperti huruf atau simbol), akan ditampilkan pesan error. Setelah array berhasil diatur, variabel i, j, dan stepCount diinisialisasi untuk memulai proses sorting, lalu setiap elemen array ditampilkan sebagai label di layar.

```
private void setArrayFromInput() {
    String text = inputField.getText().trim();
    if (text.isEmpty()) return;
    String[] parts = text.split(",");
    array = new int[parts.length];

    try {
        for (int k = 0; k < parts.length; k++) {
            array[k] = Integer.parseInt(parts[k].trim());
        }
    } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog(this,
                "Masukkan hanya angka yang dipisahkan dengan koma!", "error",
                JOptionPane.ERROR_MESSAGE);
        return;
    }

    i = 0;
    j = i + 1;
    stepCount = 1;
    minIndex = i;
    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].setFont(new Font("Arial", Font.BOLD, 24));
        labelArray[k].setOpaque(true);
        labelArray[k].setBackground(Color.WHITE);
        labelArray[k].setBorder(BorderFactory.createLineBorder(Color.BLACK));
        labelArray[k].setPreferredSize(new Dimension(50, 50));
        labelArray[k].setHorizontalAlignment(SwingConstants.CENTER);
        panelArray.add(labelArray[k]);
    }

    panelArray.revalidate();
    panelArray.repaint();
    highlightMinIndex();
}
```

5. Saat tombol "Langkah Selanjutnya" diklik, program akan menjalankan proses selection sort. Mulai dari indeks i, program mencari elemen terkecil dengan membandingkan nilai di posisi i dengan elemen-elemen di sebelah kanannya (j). Jika ditemukan elemen yang lebih kecil, nilai minIndex akan diperbarui. Setelah pencarian selesai, elemen pada indeks i akan ditukar dengan elemen terkecil tersebut (jika perlu). Bila tidak ada pertukaran, maka dicatat bahwa elemen sudah berada di tempat yang benar. Penjelasan tiap langkah ditampilkan di stepArea.

```
private void performStep() {
    if (i < array.length - 1 && sorting) {
        StringBuilder stepLog = new StringBuilder();
        if (j == i + 1) {
            minIndex = i;
        }

        if (j < array.length) {
            if (array[j] < array[minIndex]) {
                minIndex = j;
            }
            j++;
        }

        if (j == array.length) {
            if (minIndex != i) {
                int temp = array[i];
                array[i] = array[minIndex];
                array[minIndex] = temp;
                stepLog.append("Langkah ").append(stepCount).append(": Menukar elemen ke- ")
                .append(i).append(" (").append(array[minIndex]).append(")")
                .append(" dengan elemen ke- ").append(minIndex)
                .append(" (").append(array[i]).append(")\n");
            } else {
                stepLog.append("Langkah ").append(stepCount)
                .append(": Tidak ada pertukaran (elemen ke- ")
                .append(i).append(" sudah minimum)\n");
            }
            stepLog.append("hasil: ").append(arrayToString(array)).append("\n\n");
            stepArea.append(stepLog.toString());
```

```
            i++;
            j = i + 1;
            stepCount++;
        }
        updateLabels();
        highlightMinIndex();

        if (i >= array.length - 1) {
            sorting = false;
            stepButton.setEnabled(false);
            resetHighlights();
            JOptionPane.showMessageDialog(this, "Sorting selesai!");
        }
    }

}
```

6. Fungsi khusus digunakan untuk menandai elemen terkecil yang ditemukan selama proses pencarian agar pengguna bisa melihat proses dengan lebih jelas secara visual.

```
private void highlightMinIndex() {
    resetHighlights();
    if (minIndex >= 0 && minIndex < labelArray.length) {
        labelArray[minIndex].setBackground(Color.YELLOW);
    }
}
```

7. Untuk menghapus sorotan elemen yang sebelumnya ditandai, digunakan fungsi resetHighlights() sehingga tampilan array kembali normal sebelum langkah berikutnya dijalankan.

```
private void resetHighlights() {
    for (JLabel label : labelArray) {
        label.setBackground(Color.WHITE);
    }
}
```

8. Fungsi reset akan dipanggil saat tombol "Reset" diklik. Program akan menghapus seluruh elemen yang tampil di layar, mengembalikan semua variabel ke kondisi awal, serta menonaktifkan tombol "Langkah Selanjutnya", sehingga pengguna dapat mengulang proses dari awal.

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

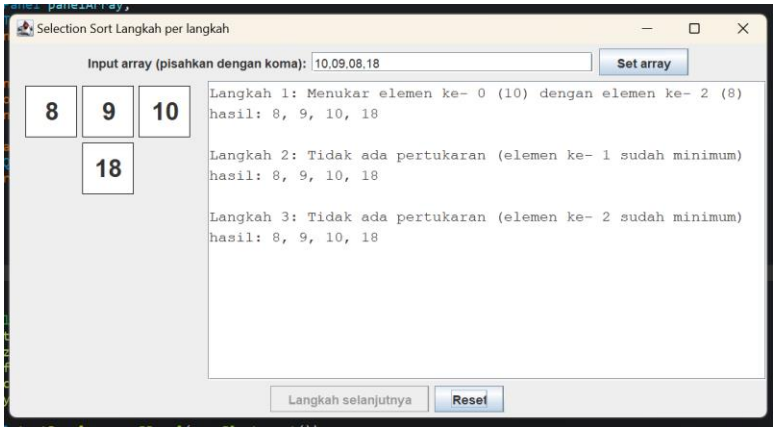
9. Sebuah fungsi disediakan untuk mengubah isi array menjadi string, yang kemudian akan ditampilkan dalam area langkah sebagai penjelasan yang lebih mudah dibaca.

```
    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();
    }
}
```

10. Tambahkan method main() sebagai titik awal eksekusi agar aplikasi dapat dijalankan.

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

11. Terakhir, jalankan program. Jika jendela GUI muncul seperti yang diharapkan dan fitur-fitur bekerja dengan baik, berarti aplikasi telah berhasil dijalankan dengan benar.



D. Kesimpulan

Berdasarkan implementasi dan hasil pengujian, dapat disimpulkan bahwa algoritma insertion sort dan selection sort dapat dijalankan dengan baik menggunakan bahasa Java, khususnya dengan dukungan antarmuka grafis (GUI) untuk memvisualisasikan proses pengurutan. Melalui GUI yang interaktif, pengguna bisa memahami setiap tahapan sorting secara lebih mudah dan terstruktur.

Program ini berhasil memperlihatkan perbedaan mendasar antara dua algoritma tersebut: insertion sort bekerja dengan cara menyisipkan elemen ke posisi yang tepat, sedangkan selection sort memilih elemen terkecil dan menempatkannya di posisi awal. Pendekatan visual semacam ini sangat efektif dalam membantu pengguna memahami logika dasar dari masing-masing metode.

Oleh karena itu, aplikasi ini tidak hanya berguna untuk mengurutkan data, tetapi juga sangat bermanfaat sebagai sarana pembelajaran interaktif yang memperkuat pemahaman tentang konsep algoritma pengurutan secara praktis.