LAPORAN PRAKTIKUM PEKAN 7 STRUKTUR DATA
INSERTION SORT & SELECTION SORT

OLEH :

FADIL INSANUS SIDDIK

(2411532013)

MATA KULIAH : PRAKTIKUM STRUKTURR DATA

DOSEN PENGAMPU : Dr. Wahyudi, S.T, M.T.

FAKULTAS TEKNOLOGI INFORMASI

PROGRAM STUDI INFORMATIKA

UNIVERSITAS ANDALAS

PADANG, JUNI 2025

A. PENDAHULUAN

Dalam dunia pemrograman dan ilmu komputer, pengurutan data (sorting) merupakan proses dasar yang sangat penting untuk mempercepat pencarian, pengolahan, dan pengelompokan data. Algoritma sorting adalah salah satu topik utama dalam pembelajaran struktur data karena sering digunakan dalam berbagai aplikasi nyata, mulai dari pencarian data dalam database, pengurutan file, hingga pemrosesan data dalam sistem informasi.

Dua algoritma pengurutan yang umum digunakan dan dipelajari adalah **Insertion Sort** dan **Selection Sort**. Meskipun keduanya tergolong algoritma sederhana, mereka sangat efektif untuk dataset berukuran kecil dan juga sangat baik untuk dipahami dalam proses belajar dasar-dasar sorting.

- **Insertion Sort** bekerja dengan cara menyisipkan elemen ke posisi yang tepat satu per satu dari kiri ke kanan. Setiap elemen dibandingkan dan disisipkan ke dalam bagian array yang sudah terurut.
- **Selection Sort** bekerja dengan cara memilih elemen terkecil dari bagian yang belum terurut dan menukarnya dengan elemen pada posisi saat ini.

Pada praktikum ini, algoritma Insertion Sort dan Selection Sort diimplementasikan dalam bentuk aplikasi GUI berbasis Java untuk memvisualisasikan proses sorting secara interaktif dan langkah demi langkah. Pendekatan visual ini membantu dalam memahami logika dan alur algoritma secara lebih mendalam.

B. TUJUAN PRAKTIKUM

Tujuan dari praktikum ini adalah sebagai berikut:

1. Memahami konsep dasar algoritma Insertion Sort dan Selection Sort. Mahasiswa diharapkan mampu memahami cara kerja masing-masing algoritma melalui simulasi interaktif yang divisualisasikan dalam program GUI.
2. Mengimplementasikan algoritma sorting menggunakan bahasa pemrograman Java. Mahasiswa membuat program yang mampu mengurutkan array secara interaktif menggunakan struktur kontrol dan logika pemrograman Java.
3. Mengembangkan kemampuan membuat antarmuka pengguna (GUI) sederhana. Program menggunakan komponen seperti JFrame, JTextField, JButton, dan JLabel dari pustaka Swing Java untuk menampilkan data dan proses sorting secara real-time.

4. Melatih keterampilan debugging dan logika langkah demi langkah. Setiap langkah sorting ditampilkan dalam log teks sehingga mahasiswa dapat menganalisis proses sorting secara detail.

C. LANGKAH-LANGKAH

1. InsertionSort

```
public class InsertionSort extends JFrame {
```

- Mendefinisikan kelas public bernama InsertionSort. Ini adalah nama utama dari program.

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

- Menjalankan GUI dengan cara yang aman untuk event-driven programming di Swing.

```
public InsertionSort() {
    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();
    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(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());
```

- Menyusun tata letak GUI : input, control, tampilan array, log langkah.
- Menyambungkan tombol ke aksi :
  - setButton → memanggil setArrayFromInput()
  - stepButton → memanggil performStep()
  - resetButton → memanggil reset()

```
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;
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].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();
```

- Mengambil input teks dari inputField.
- Mem-parse angka menjadi array integer.
- Membuat JLabel untuk tiap elemen array.
- Menyiapkan state awal untuk sorting (i = 1, sorting = true, dll).

```
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!");
        }}
}
```

- Menampilkan log langkah di stepArea.
- Menampilkan array yang diperbarui di GUI.
- Jika sorting selesai (i == array.length), tombol spin dimatikan.

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

- Memperbaru isi teks setiap JLabel agar sesuai dengan array terbaru.

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

- Mengembalikan GUI dan log ke kondisi awal.

```
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();
}
}
```

- Fungsi bantu untuk mengubah array menjadi String, digunakan untuk log.

2. SelectionSort

```
public class SelectionSort extends JFrame {
```

- Mendeklarasikan jika class bernama SelectionSort.

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

- inputField : input teks (angka dipisahkan koma).
- stepArea : log proses.
- labelArray : label visual untuk angka-angka array.
- panelArray : tempat menampilkan array dalam GUI.

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

            }
        }
    });
}
```

- Proses disaat program mulai dijalankan.

```
public SelectionSort() {
    setTitle("Selection Sort Langkah per Langkah");
    setSize(750, 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 teks untuk log langkah-langkah
      stepArea = new JTextArea(8, 60);
      stepArea.setEditable(false);
      stepArea.setFont(new Font("Monospaced", Font.PLAIN, 14));
      JScrollPane scrollPane = new JScrollPane(stepArea);

      // tambahkan panel 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());

  }
```

- Tampilan untuk program yang dibuat.

```
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;
    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].setPreferredSize(new Dimension(50, 50));
        labelArray[k].setHorizontalAlignment(SwingConstants.CENTER);
        panelArray.add(labelArray[k]);
        }
```

```
 panelArray.revalidate();
 panelArray.repaint();
 j = i + 1;
 highlightMinIndex();

 }
```

- Menubah input String menjadi int[] array.
- Membuat label-label visual.
- Mengatur ulang state awal sorting (i = 0, j = 1, dst).

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

        // cari indeks minimum
        if (j < array.length) {
            if (array[j] < array[minIndex]) {
                minIndex = j;
            }
            j++;
        }
```

```
// Jika sudah selesai membandingkan
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();
```

```
      updateLabels();
      highlightMinIndex();

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

- Mencari elemen terkecil dari indeks i ke akhir.
- Setelah selesai, jika ada pertukaran, lakukan swap.
- Perbarui tampilan GUI dan log Langkah.
- Jika i mencapai akhir array, sorting selesai.

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

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

- Memberi warna kuning pada elemen terkecil yang sedang dibandingkan.
- Mengembalikan semua label ke warna putih.

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

- Mengganti nilai teks di setiap JLabel agar sesuai array terbaru.

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

- Mengatur ulang seluruh GUI dan variable ke keadaan awal.

D. KESIMPULAN

Berdasarkan praktikum yang telah dilakukan, dapat disimpulkan bahwa:

1. Insertion Sort dan Selection Sort merupakan algoritma sorting yang sederhana namun efektif untuk dataset berukuran kecil. Keduanya memiliki kompleksitas waktu $O(n^2)$ pada kasus terburuk, tetapi tetap relevan sebagai dasar pembelajaran algoritma pengurutan.
2. Implementasi dalam bentuk Java GUI membantu memvisualisasikan proses sorting secara lebih mudah dipahami. Melalui proses langkah demi langkah, mahasiswa dapat melihat perubahan yang terjadi dalam array setelah setiap iterasi.
3. Penerapan sorting dalam bentuk GUI juga mengasah kemampuan mahasiswa dalam pemrograman event-driven dan penggunaan komponen antarmuka pengguna.
4. Praktikum ini memberikan pemahaman yang lebih baik tentang bagaimana algoritma berjalan secara dinamis serta pentingnya logika dan struktur data dalam pengembangan program yang efisien.