LAPORAN PRAKTIKUM

PRAKTIKUM STRUKTUR DATA

PERTEMUAN 3

SORTING



Disusun oleh:

Nama : Aditya Lucky Zulkarnaen

NIM : 24/537764/SV/24449

Kelas : PLB1

Dosen Pengampu : Dr. Umar Taufiq, S.Kom., M.Cs.

PROGRAM STUDI D-IV TEKNOLOGI REKAYASA PERANGKAT LUNAK

DEPARTEMEN TEKNIK ELEKTRO DAN INFORMATIKA

SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

YOGYAKARTA

2025

**PERTEMUAN 3**

**SORTING**

1. **Tujuan Percobaan**
2. Memahami konsep dan cara kerja algoritma sorting, khususnya Selection Sort, Insertion Sort, dan Bubble Sort.
3. Mengimplementasikan algoritma sorting dalam program komputer untuk mengurutkan data secara ascending atau descending.
4. Menganalisis kompleksitas waktu dari setiap algoritma sorting yang digunakan.
5. **Dasar Teori**
6. Sorting

Algoritma sorting adalah metode atau teknik untuk mengurutkan data atau elemen-elemen dalam suatu struktur data secara teratur. Tujuannya adalah untuk mengubah data yang berantakan menjadi urutan yang teratur, misalnya dari data yang tidak terurut menjadi data yang terurut menaik (ascending) atau menurun (descending). (Annisa, 2025)

1. Selection Sort

Selection sort adalah algoritma sorting yang bekerja dengan cara mencari elemen terkecil di dalam array, lalu menempatkannya di posisi paling awal. Proses ini terus dilakukan untuk elemen berikutnya hingga semua elemen berada dalam urutan yang benar.

Selection sort memiliki kompleksitas waktu O(n2), sehingga kurang efisien untuk dataset yang besar. Namun, algoritma ini tetap sering digunakan karena mudah dipahami dan cocok untuk data kecil. Beberapa modifikasi pada selection sort, seperti teknik bidirectional, yang dapat mempercepat proses sorting. (Vilchez, 2020)

1. Insertion Sort

Insertion sort adalah algoritma yang bekerja dengan cara menyisipkan elemen ke posisi yang sesuai di bagian array yang sudah terurut. Insertion sort sangat cocok untuk dataset kecil atau data yang hampir terurut karena algoritma ini memiliki kompleksitas terbaik O(n) pada kasus tersebut. (Thabit & Bhawazir, 2020)

Insertion sort memiliki algoritma modifikasi yang disebut Enhanced Insertion Sort (EIS). Algoritma ini mengurangi jumlah perbandingan dengan menggunakan pendekatan hybrid. Hasilnya, EIS lebih cepat dibandingkan insertion sort biasa, terutama ketika digunakan pada dataset yang lebih besar. (CiteSeerX, 2023).

1. Bubble Sort

Bubble sort adalah salah satu algoritma sorting paling sederhana. Algoritma ini bekerja dengan cara membandingkan elemen berdekatan dan menukarnya jika urutannya salah. Bubble sort memiliki kompleksitas waktu O(n2), sehingga tidak efisien untuk dataset besar. Namun, bubble sort masih sering digunakan untuk dataset kecil karena mudah dipahami dan diimplementasikan. Meski bubble sort lambat dibandingkan algoritma modern seperti tim sort, bubble sort memiliki keunggulan dalam menjaga stabilitas sorting. Stabilitas berarti elemen dengan nilai yang sama tidak akan berubah urutannya setelah proses sorting selesai. Hal ini membuat bubble sort tetap relevan dalam situasi tertentu. (Hanafi et al., 2022)

1. **Hasil dan Pembahasan**

Studi kasus dengan sebuah Array berisikan 10 data angka acak:

Arr = [29, 10, 14, 37, 13, 7, 18, 25, 40, 22]

1. Selection Sort

* Iterasi 1: Cari nilai terkecil dan tukar dengan elemen pertama → [7, 10, 14, 37, 13, 29, 18, 25, 40, 22] (9 perbandingan, 1 pertukaran)
* Iterasi 2: Cari nilai terkecil di sisa array → [7, 10, 14, 37, 13, 29, 18, 25, 40, 22] (8 perbandingan, 0 pertukaran)
* Iterasi 3: Cari nilai terkecil → [7, 10, 13, 37, 14, 29, 18, 25, 40, 22] (7 perbandingan, 1 pertukaran)
* Iterasi 4: Cari nilai terkecil → [7, 10, 13, 14, 37, 29, 18, 25, 40, 22] (6 perbandingan, 1 pertukaran)
* Iterasi 5: Cari nilai terkecil → [7, 10, 13, 14, 18, 29, 37, 25, 40, 22] (5 perbandingan, 1 pertukaran)
* Iterasi 6: Cari nilai terkecil → [7, 10, 13, 14, 18, 22, 37, 25, 40, 29] (4 perbandingan, 1 pertukaran)
* Iterasi 7: Cari nilai terkecil → [7, 10, 13, 14, 18, 22, 25, 37, 40, 29] (3 perbandingan, 1 pertukaran)
* Iterasi 8: Cari nilai terkecil → [7, 10, 13, 14, 18, 22, 25, 29, 40, 37] (2 perbandingan, 1 pertukaran)
* Iterasi 9: Cari nilai terkecil → [7, 10, 13, 14, 18, 22, 25, 29, 37, 40] (1 perbandingan, 1 pertukaran)

Total: 45 perbandingan, 9 pertukaran.

1. Insertion Sort

* Iterasi 1: 10 lebih kecil dari 29, pindah → [10, 29, 14, 37, 13, 7, 18, 25, 40, 22] (1 perbandingan, 1 pertukaran)
* Iterasi 2: 14 disisipkan di tempat yang benar → [10, 14, 29, 37, 13, 7, 18, 25, 40, 22] (1 perbandingan, 1 pertukaran)
* Iterasi 3: 37 tetap di tempatnya → [10, 14, 29, 37, 13, 7, 18, 25, 40, 22] (1 perbandingan, 0 pertukaran)
* Iterasi 4: 13 disisipkan di tempat yang benar → [10, 13, 14, 29, 37, 7, 18, 25, 40, 22] (3 perbandingan, 3 pertukaran)
* Iterasi 5: 7 disisipkan di tempat yang benar → [7, 10, 13, 14, 29, 37, 18, 25, 40, 22] (5 perbandingan, 5 pertukaran)
* Iterasi 6: 18 disisipkan di tempat yang benar → [7, 10, 13, 14, 18, 29, 37, 25, 40, 22] (2 perbandingan, 2 pertukaran)
* Iterasi 7: 25 disisipkan di tempat yang benar → [7, 10, 13, 14, 18, 25, 29, 37, 40, 22] (2 perbandingan, 2 pertukaran)
* Iterasi 8: 40 tetap di tempatnya → [7, 10, 13, 14, 18, 25, 29, 37, 40, 22] (1 perbandingan, 0 pertukaran)
* Iterasi 9: 22 disisipkan di tempat yang benar → [7, 10, 13, 14, 18, 22, 25, 29, 37, 40] (4 perbandingan, 4 pertukaran)

Total: 19 perbandingan, 18 pertukaran.

1. Bubble Sort

* Iterasi 1: 9 pertukaran
* Iterasi 2: 8 pertukaran
* Iterasi 3: 7 pertukaran
* Iterasi 4: 6 pertukaran
* Iterasi 5: 5 pertukaran
* Iterasi 6: 4 pertukaran
* Iterasi 7: 3 pertukaran
* Iterasi 8: 2 pertukaran
* Iterasi 9: 1 pertukaran

Total: 45 perbandingan, 45 pertukaran.

1. **Kesimpulan**

Berdasarkan hasil praktikum, Selection Sort bekerja dengan mencari elemen terkecil dan menempatkannya di posisi yang benar secara iteratif. Meskipun mudah dipahami, algoritma ini kurang efisien untuk dataset besar karena memiliki kompleksitas waktu O(n^2). Insertion Sort, di sisi lain, menyisipkan elemen ke dalam bagian array yang sudah terurut. Algoritma ini lebih efisien dibandingkan Selection Sort, terutama pada dataset kecil atau hampir terurut, dengan kompleksitas terbaik O(n). Bubble Sort, yang bekerja dengan membandingkan dan menukar elemen berdekatan jika tidak dalam urutan yang benar, juga memiliki kompleksitas O(n^2), membuatnya kurang efisien untuk dataset besar meskipun mudah diimplementasikan.

Dari hasil implementasi, Insertion Sort terbukti lebih cepat dibandingkan Selection Sort dan Bubble Sort pada dataset kecil yang hampir terurut. Selection Sort unggul dalam kemudahan implementasi tetapi memiliki waktu eksekusi yang lebih lama dibandingkan Insertion Sort. Bubble Sort, meskipun stabil dalam mempertahankan urutan elemen yang bernilai sama, tetap kurang efisien dibandingkan algoritma lainnya. Oleh karena itu, untuk dataset yang lebih besar, penggunaan algoritma sorting yang lebih canggih seperti Tim Sort lebih direkomendasikan karena memiliki efisiensi yang lebih baik dibandingkan algoritma sorting dasar yang diuji dalam praktikum ini.

1. **Daftar Pustaka**

Annisa. (2025, Februari 27). *Pengertian Algoritma Sorting (Pengurutan) Dalam Pemrograman*. Retrieved from Fikti UMSU: https://fikti.umsu.ac.id/pengertian-algoritma-sorting-pengurutan-dalam-pemrograman/

Vilchez, R. N. (2020). Modified Selection Sort Algorithm Employing Boolean and Distinct Function in a Bidirectional Enhanced Selection Technique. International Journal of Machine Learning and Computing, 10(1), 93-100.

Thabit, K., & Bawazir, A. (2020). A Novel Approach of Selection Sort Algorithm with Parallel Computing and Dynamic Programming Concepts. King Abdulaziz University Research Papers.

Sitorus, Z., Prayogi, D., Rizko, M. A., Suteja, A. G., & Harahap, M. R. (2024). Implementation of the Insertion Sort Algorithm to Sort Positive Integers in Ascending Order Using Flowgorithm. Journal of Information Technology, Computer Science and Electrical Engineering, 1(3), 323-328.

CiteSeerX (2023). Enhanced Insertion Sort Algorithm.

Hanafi, M. R., Faadhilah, M. A., Pradeka, D., & Putra, M. T. D. (2022). Comparison Analysis of Bubble Sort Algorithm with Tim Sort Algorithm Sorting Against the Amount of Data. Journal of Computer Engineering, 1(1), 9-13.