Maliki Interdisciplinary Journal (MIJ) eISSN: 3024-8140
Volume 3, Issue April, 2025 pp. 681-687 http://urj.uin-malang.ac.id/index.php/mij/index

Penggunaan algoritma bubble sort pada Bahasa pemprograman Java

Hayyi Al Ghifari
Program Studi Teknik informatika, Universitas Islam Negeri Maulana Malik Ibrahim Malang
e-mail: hayyialghifari9@gmail.com

Kata Kunci:
Kompleksitas, data, teknologi, algoritma, bubble sort

Keywords:
Complexity, data, technology, algorithm,

A B S T R A K

Volume dan kompleksitas suatu data telah meningkat karena kemajuan teknologi yang sangat pesat di zaman yang serba digital ini, sehingga pengelolaan data menjadi semakin sulit. Maka penggunaan algoritma pengurutan adalah salah satu cara untuk membuat pemrosesan data menjadi lebih sederhana. Algoritma Bubble Sort adalah teknik pengurutan langsung yang dibahas dalam karya ini. Algoritma ini membandingkan dan mengganti elemen dalam suatu array hingga tidak diperlukan lagi pertukaran, Bubble Sort mudah dipahami dan digunakan, tetapi tidak terlalu efisien dalam hal waktu, terutama saat menangani kumpulan data yang besar. Meskipun demikian, algoritma ini bekerja dengan sangat baik dengan kumpulan data yang sangat kecil atau saat data hampir terurut. Penggunaan praktis algoritma ini termasuk mengurutkan evaluasi kinerja siswa dalam aplikasi seluler, yang meningkatkan efisiensi pemrosesan data. Temuan analisis menunjukkan bahwa Bubble Sort bekerja paling baik dalam skala kecil, tetapi algoritma lain mungkin bekerja lebih baik pada kumpulan data yang lebih besar.

A B S T R A C T

The volume and complexity of data increases due to the rapid advancement of technology in this digital era, so that data management becomes increasingly difficult. So the use of sorting algorithms is one way to make data processing simpler. The Bubble Sort algorithm is a direct sorting technique discussed in this work. This algorithm compares and replaces elements in an array until no more swapping is needed, Bubble Sort is easy to understand and use, but it is not very efficient in half the time, especially when dealing with large data sets. However, this algorithm works very well with very small data sets or when the data is almost sorted. Practical uses of this algorithm include sorting student performance evaluations in mobile applications, which increases the efficiency of data processing. Analysis of the findings shows that Bubble Sort works best on a small scale, but other algorithms may work better on larger data sets.

Pendahuluan

Algoritma Sorting (pengurutan) merupakan langkah langkah yang sistematis dan berurutan tujuan nya adalah untuk mendapatkan hasil yang sesuai dengan keinginan. Seorang programmer seharusnya selalu memiliki beberapa alternatif dalam penggunaan algoritma yang bisa digunakan untuk menyelesaikan suatu permasalahan sorting (tara et al., 2024). Sorting adalah pengurutan jumlah data berdasarkan nilai yang diberikan. Pengurutannya bisa dari nilai paling kecil ke nilai yang paling besar

(ascending) ataupun dari nilai paling besar ke nilai paling kecil (descending).

Sorting dibagi menjadi dua kategori: Comparison dan Non-Comparison.Comparison yaitu algoritma pengurutan yang bekerja dengan membandingkan pasangan elemen dalam daftar. Algoritma ini menentukan urutan elemen berdasarkan hasil perbandingan tersebut. Non-Comparasion yaitu algoritma pengurutan yang tidak bergantung pada perbandingan langsung antara elemen-elemen data, algoritma ini memanfaatkan karakteristik khusus dari data atau menggunakan struktur data tambahan untuk mencapai pengurutan.

Pembahasan

Informasi merupakan salah satu aspek penting dalam kehidupan manusia. Informasi bisa bersifat rahasia atau umum. Adapun informasi yang bersifat rahasia sangat penting untuk dijaga keamanan kerahasiaannya (Maghfiroh et al., 2023). Hal tersebut dapat dilakukan dengan pemrograman.

Dalam berbagai konteks pemrograman, algoritma adalah spesifikasi dari urutan langkah-langkah yang harus diikuti untuk menyelesaikan tugas tertentu. Pentingnya sebuah algoritma tidak hanya terletak pada kemampuannya untuk menghasilkan solusi yang benar, namun juga sejauh mana algoritma tersebut dapat menghasilkan output yang diinginkan berdasarkan serangkaian input yang diberikan (Mahrozi & Faisal, 2023).

Dalam algoritma terdapat bahasa enkripsi dan deskripsi. Enkripsi adalah proses memodifikasi informasi yang sebelumnya dapat dibaca dengan mudah kemudian diubah dengan menggunakan suatu algoritma sehingga tidak dapat dibaca oleh siapapun. Sementara dekripsi adalah teknik membuat data yang tidak dapat dibaca menjadi bentuk yang dapat dibaca, dekripsi adalah kebalikan dari enkripsi (Rafika Zahrotul Fauziah et al., 2024). Hal tersebut juga berlaku pada algoritma bubble sort.

Dalam hal pemahaman dan implementasi, algoritma bubble sort termasuk salah satu teknik pengurutan yang paling mudah. Ide dasar algoritma ini adalah membandingkan setiap elemen dalam array dan, jika urutannya salah maka akan ditukar hingga tidak diperlukan lagi pertukaran, prosedur perbandingan ini akan dilakukan lagi. Karena menggunakan perbandingan elemen dalam operasinya, metode ini diklasifikasikan sebagai algoritma pengurutan perbandingan.

Bubble sort memiliki keunggulan yaitu mekanisme yang sederhana sehingga mudah dipahami dan diimplementasikan dalam program aplikasi, kinerjanya juga cukup stabil, dan juga algoritma bubble sort dapat dijalankan tanpa memerlukan penambahan memori dalam komputer(Resiana et al., 2024)

Bubble sort melakukan pengurutan dengan cara yang cukup simple yaitu dengan inisialisasi, perbandingan, pertukaran, iterasi, pengulangan. Walaupun cukup simple tetapi algoritma ini menghasilkan hasil yang akurat. Cara kerja algoritma bubble sort

dapat ditemukan pada contoh di bawah berikut.

Terdapat array dengan elemen-elemen "300 100 400 200 500". Berikut adalah proses saat kita menggunakan algoritma bubble sort.

percobaan ke 1 :

(300 100 400 200 500) diubah menjadi (100 300 400 200 500)

(100 300 400 200 500) tetap (100 300 400 200 500)

(100 300 400 200 500) diubah menjadi (100 300 200 400 500)

(100 300 200 400 500) tetap (100 300 200 400 500)

percobaan Ke 2 :

(100 300 200 400 500) tetap (100 300 200 400 500)

(100 300 200 400 500) diubah menjadi (100 200 300 400 500)

(100 200 300 400 500) tetap (100 200 300 400 500)

(100 200 300 400 500) tetap (100 200 300 400 500)

percobaan Ke 3 :

(100 200 300 400 500) tetap (100 200 300 400 500)

(100 200 300 400 500) tetap (100 200 300 400 500)

(100 200 300 400 500) tetap (100 200 300 400 500)

(100 200 300 400 500) tetap (100 200 300 400 500)

Setelah tiga percobaan, array sudah terurut menjadi (100 200 300 400 500). Terlihat bahwa percobaan kedua, array sudah terurut. tapi, algoritma tetap dilanjut sampai percobaan kedua selesai. Percobaan tetap dilanjut karena definisi terurut dalam algoritma bubble sort adalah ketika tak ada lagi pertukaran yang terjadi dalam percobaan, maka dari itu, percobaan ke 3 diperlukan guna memastikan array tersebut sudah terurut.

Metode penelitian

Penelitian ini menggunakan Bahasa pemrograman java untuk mengurutkan elemen elemen di dalam array. Berikut adalah langkah langkah untuk mengimplementasikan bubble sort kedalam Bahasa pemrograman java.

1. Membuat Struktur Kelas

- Mulailah dengan mendefinisikan kelas Bubblesort. Kelas ini akan berisi metode untuk melakukan sorting dan mencetak array.

2. Membuat Metode bubbleSort

- Buat metode statis bubbleSort yang menerima sebuah array integer sebagai parameter

3. Menghitung Panjang Array

- Di dalam metode, tentukan panjang array dengan int n = arr.length;

4. Looping untuk Sorting

- Gunakan dua loop bersarang:
    - Loop luar (i) untuk iterasi melalui elemen array.
    - Loop dalam (j) untuk membandingkan elemen yang berdekatan.

5. Membandingkan dan Menukar Elemen

- Di dalam loop dalam, bandingkan elemen saat ini (arr[j]) dengan elemen berikutnya (arr[j + 1]).
- Jika elemen saat ini lebih besar, tukar posisi keduanya

6. Membuat Metode printArray

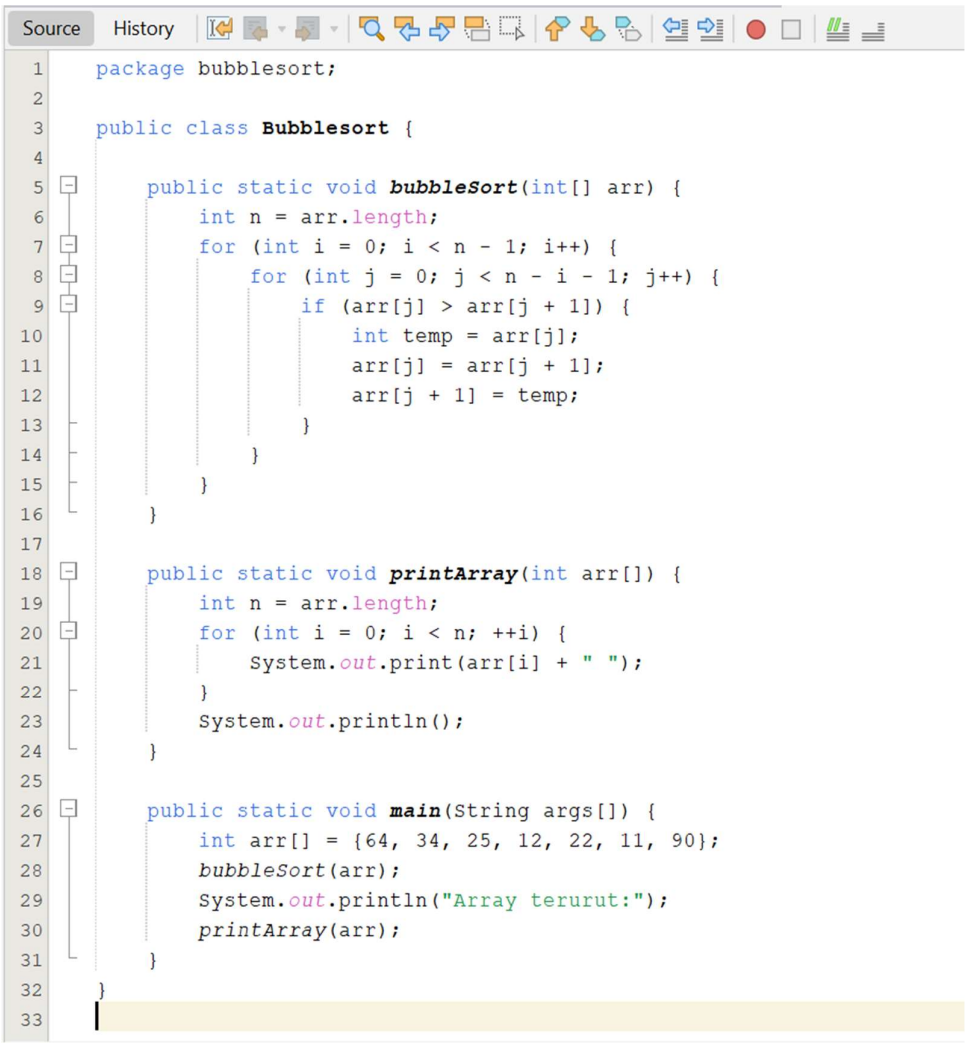
- Buat metode statis printArray untuk mencetak elemen-elemen dari array. Ini akan membantu untuk menampilkan hasil akhir

7. Membuat Metode main

- Di dalam metode main, buat array contoh yang ingin Anda urutkan.

8. Memanggil Metode Sorting dan Mencetak Hasil

Panggil metode bubbleSort untuk mengurutkan array dan gunakan printArray untuk menampilkan array yang sudah terurut.



```java
package bubblesort;

public class Bubblesort {

    public static void bubbleSort(int[] arr) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
    }

    public static void printArray(int arr[]) {
        int n = arr.length;
        for (int i = 0; i < n; ++i) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }

    public static void main(String args[]) {
        int arr[] = {64, 34, 25, 12, 22, 11, 90};
        bubbleSort(arr);
        System.out.println("Array terurut:");
        printArray(arr);
    }
}
```

9. Menutup Kelas

- Tutup kelas Bubblesort.
- berikut adalah kode lengkap setelah mengkuti langkah-langkah diatas:

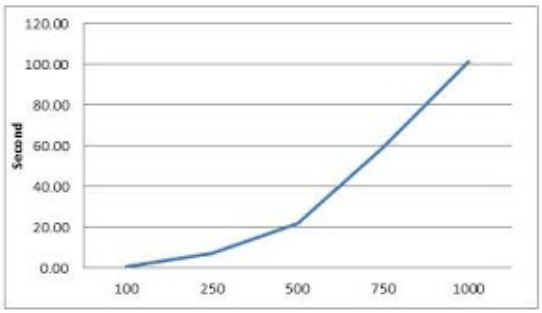
Hasil dan Pembahasan

Kompleksitas dari algoritma Bubble Sort bisa dianalisis berdasarkan tiga jenis kasus, yaitu kasus terburuk, kasus rata-rata, dan kasus terbaik.

Pada kondisi **terbaik**, Metode perhitungan cuma dilakukan (n-1) kali dalam satu lintasan ketika data telah diurutkan. Ini menunjukkan bahwa algoritma pengurutan gelembung bersifat linier dalam keadaan ini. Rumus berikut menentukan berapa banyak proses yang seharusnya ada dalam kondisi ini: **n - 1 adalah jumlah proses.**

Pada kondisi **terburuk**, Setiap kali proses dilakukan, elemen terkecil harus dipindahkan satu langkah lebih dekat ke awal array, di mana elemen tersebut berada di bagian akhir. Misalnya, diperlukan tiga proses ditambah satu proses tambahan untuk verifikasi guna menggeser elemen terkecil dari urutan keempat ke urutan pertama. Dalam kondisi ini, jumlah proses dapat dihitung menggunakan rumus berikut: **n2 + n adalah jumlah proses.**

Untuk kondisi **rata rata**, Jumlah pergeseran data ke kiri menentukan jumlah percobaan. Rumus berikut dapat digunakan untuk menentukan jumlah proses perhitungan: **X^2 + x adalah jumlah proses.** Kompleksitas algoritma ini bisa dijelaskan melalui grafik di bawah ini:



Semua algoritma memiliki kelebihan dan kekurangan tersendiri, termasuk bubble sort. Berikut kami sampaikan untuk kekurangan dan kelebihan pada penggunaan Algoritma Bubble Sort

- Kelebihan Algoritma Bubble Sort

1. **Sederhana** : Algoritma ini sangat mudah dipahami karena cuma menggunakan proses rekurensi dan perbandingan. Algoritma pengurutan lain umumnya lebih kompleks.
2. **Mudah diimplementasikan** : Karena kemudahannya, algoritma bubble sort memiliki kemungkinan kecil untuk menyebabkan kesalahan sintaks pada penulisan kode.
3. **Definisi terurutnya jelas** : Definisi terurut di dalam algoritma ini dinyatakan dengan jelas, yaitu tidak ada penukaran yang terjadi dalam satu percobaan. Tentu beda dengan algoritma lain, seperti Quick Sort.

4. **Efektif untuk data kecil yang sudah terurut :** Bubble sort punya kondisi terbaik (best-case) dengan kompleksitas O(n) saat data sudah terurut sebelumnya.

- Kekurangan algoritma bubble sort

1. **Tidak efisien untuk data besar :** Algoritma ini tidak efektif untuk pengurutan data dalam jumlah besar. Misalnya, untuk 1000 elemen, jumlah perbandingan yang dilakukan bisa mencapai lebih dari satu juta
2. **Proses pengurutan yang panjang :** Meskipun algoritma lain juga memiliki kompleksitas $O(n^2)$, algoritma seperti insertion sort biasanya lebih efisien dalam hal langkah pengurutan.

Kesimpulan dan Saran

Bubble Sort adalah metode pengurutan paling sederhana dan mudah diterapkan dalam berbagai program. Namun, bubble sort memiliki efisiensi waktu yang rendah dan bisa sangat lambat ketika digunakan untuk mengurutkan data dalam jumlah besar. Oleh karena itu, algoritma ini akan lebih cocok kalau dipergunakan dalam program dengan skala yang kecil dan jumlah data yang terbatas. Jika data yang diurutkan cukup banyak, waktu yang diperlukan untuk memprosesnya akan menjadi lebih lama. Bubble sort juga dapat digunakan dalam pembelajaran untuk mempelajari konsep dasar dalam pengurutan.

Daftar Pustaka

Maghfiroh, J., Turmudi, T., & Susanti, E. (2023). Pengamanan Pesan Menggunakan Algoritma One Time Pad dengan Linear Congruential Generator sebagai Pembangkit Kunci. *Jurnal Riset Mahasiswa Matematika*, *2*(3), 122–131. http://repository.uin-malang.ac.id/13042/2/13042.pdf

Mahrozi, N., & Faisal, M. (2023). *ANALISIS PERBANDINGAN KECEPATAN* ALGORITMA *SELECTION SORT DAN BUBBLE SORT*. http://repository.uin-malang.ac.id/17888/

Rafika Zahrotul Fauziah, Khudzaifah, M., & Herawati, E. (2024). Pengamanan Pesan Teks Menggunakan Affine Cipher dan Algoritma Goldbach Code. *Cyber Security dan Forensik Digital*, *7*(1), 1–6. https://doi.org/10.14421/csecurity.2024.7.1.4406 http://repository.uin-malang.ac.id/21274/

Resiana, C. D., Wijaya, Y. F., & Darusalam, U. (2024). *Penerapan Algoritma Bubble Sort Dalam Aplikasi Mobile Penentuan Nilai Prestasi Siswa*. *9*.

Tara, D. A., Ferdina, I., Sylvester, M. S., Firmansyah, M. F., Izza, M. S., Astuti, N. W., Amarta, R. P., & Fajariansyah, R. (2024). *Analisis Kompleksitas Waktu Menggunakan Sorting Algorithm pada Pengaplikasian Fitur Pengurutan Harga dari Terendah dan Tertinggi di Shopee*. *3*(1).