

## **SORTING ALGORITHM**

disusun untuk memenuhi tugas mata kuliah  
Struktur Data dan Algoritma

Oleh:

**MAHARDIKA SHIDDIQ ANSHARI**

**2308107010032**



**JURUSAN INFORMATIKA  
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM  
UNIVERSITAS SYIAH KUALA  
DARUSSALAM, BANDA ACEH  
2025**

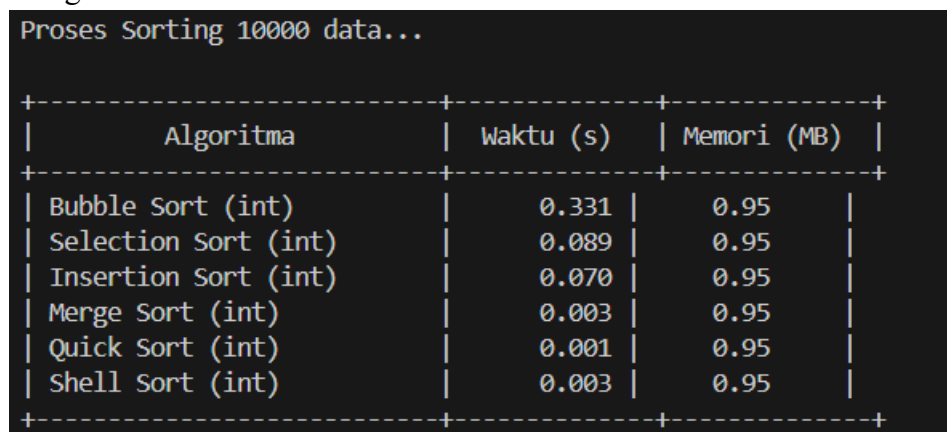
Sorting atau pengurutan data merupakan salah satu proses fundamental dalam bidang struktur data dan algoritma. Pengurutan bertujuan untuk menata data dalam urutan tertentu, baik secara menaik (ascending) maupun menurun (descending), sehingga memudahkan proses pencarian, pengambilan, dan pengolahan data lebih lanjut. Dalam dunia yang serba digital saat ini, kebutuhan akan pengelolaan data yang efisien semakin meningkat, terutama ketika harus menangani data dalam jumlah besar. Data yang telah terurut memungkinkan berbagai operasi lain seperti pencarian dan analisis dapat dilakukan dengan lebih cepat dan akurat. Oleh karena itu, pemahaman mengenai algoritma sorting sangat penting bagi siapa saja yang mempelajari struktur data dan algoritma.

Berbagai algoritma sorting telah dikembangkan, masing-masing dengan karakteristik, kelebihan, dan kekurangannya sendiri. Selection Sort, misalnya, bekerja dengan cara mencari elemen terkecil atau terbesar dari kumpulan data dan menempatkannya pada posisi yang sesuai secara bertahap hingga seluruh data terurut. Insertion Sort mengurutkan data dengan membangun subarray terurut satu per satu, dengan cara menyisipkan setiap elemen ke posisi yang tepat dalam subarray tersebut. Merge Sort menggunakan prinsip divide and conquer dengan membagi data menjadi dua bagian, mengurutkan masing-masing bagian, lalu menggabungkannya kembali secara rekursif. Quick Sort juga menggunakan prinsip divide and conquer, namun dengan memilih elemen pivot, mempartisi data berdasarkan pivot, lalu mengurutkan subarray secara rekursif. Sementara itu, Shell Sort merupakan pengembangan dari insertion sort yang membandingkan dan menukar elemen-elemen yang berjarak tertentu, lalu secara bertahap mengurangi jarak tersebut hingga data benar-benar terurut.

Setiap algoritma memiliki kompleksitas waktu dan ruang yang berbeda-beda, sehingga pemilihan algoritma yang tepat sangat bergantung pada karakteristik data dan kebutuhan aplikasi. Dengan memahami berbagai jenis algoritma sorting seperti Selection Sort, Insertion Sort, Merge Sort, Quick Sort, dan Shell Sort, kita dapat memilih metode pengurutan yang paling sesuai untuk mengelola data secara efisien dan optimal dalam berbagai situasi.

## HASIL EKSEKUSI KODE PROGRAM:

### 1. Data Angka



```
Proses Sorting 10000 data...

+-----+-----+-----+
|           Algoritma           | Waktu (s) | Memori (MB) |
+-----+-----+-----+
| Bubble Sort (int)             | 0.331     | 0.95        |
| Selection Sort (int)           | 0.089     | 0.95        |
| Insertion Sort (int)           | 0.070     | 0.95        |
| Merge Sort (int)               | 0.003     | 0.95        |
| Quick Sort (int)               | 0.001     | 0.95        |
| Shell Sort (int)               | 0.003     | 0.95        |
+-----+-----+-----+
```

Gambar 1.1 Hasil Sorting untuk 10.000 data

Proses Sorting 50000 data...

Algoritma	Waktu (s)	Memori (MB)
Bubble Sort (int)	9.247	4.75
Selection Sort (int)	2.061	4.75
Insertion Sort (int)	1.344	4.75
Merge Sort (int)	0.019	4.75
Quick Sort (int)	0.012	4.75
Shell Sort (int)	0.025	4.75

Gambar 1.2 Hasil Sorting untuk 50.000 data

Proses Sorting 100000 data...

Algoritma	Waktu (s)	Memori (MB)
Bubble Sort (int)	37.399	9.50
Selection Sort (int)	8.004	9.50
Insertion Sort (int)	5.108	9.50
Merge Sort (int)	0.039	9.50
Quick Sort (int)	0.022	9.50
Shell Sort (int)	0.045	9.50

Gambar 1.3 Hasil Sorting untuk 100.000 data

Proses Sorting 250000 data...

Algoritma	Waktu (s)	Memori (MB)
Bubble Sort (int)	236.835	23.75
Selection Sort (int)	50.348	23.75
Insertion Sort (int)	32.520	23.75
Merge Sort (int)	0.093	23.75
Quick Sort (int)	0.049	23.75
Shell Sort (int)	0.102	23.75

Gambar 1.4 Hasil Sorting untuk 250.000 data

Proses Sorting 500000 data...

Algoritma	Waktu (s)	Memori (MB)
Bubble Sort (int)	955.395	47.50
Selection Sort (int)	203.973	47.50
Insertion Sort (int)	130.704	47.50
Merge Sort (int)	0.208	47.50
Quick Sort (int)	0.094	47.50
Shell Sort (int)	0.221	47.50

Gambar 1.5 Hasil Sorting untuk 500.000 data

Proses Sorting 1000000 data...

Algoritma	Waktu (s)	Memori (MB)
Bubble Sort (int)	3444.364	95.00
Selection Sort (int)	460.906	95.00
Insertion Sort (int)	454.329	95.00
Merge Sort (int)	0.295	95.00
Quick Sort (int)	0.162	95.00
Shell Sort (int)	0.312	95.00

*Gambar 1.6 Hasil Sorting untuk 1.000.000 data*

Proses Sorting 1500000 data...

Algoritma	Waktu (s)	Memori (MB)
Bubble Sort (int)	5104.406	142.50
Selection Sort (int)	766.564	142.50
Insertion Sort (int)	586.870	142.50
Merge Sort (int)	0.242	142.50
Quick Sort (int)	0.161	142.50
Shell Sort (int)	0.313	142.50

*Gambar 1.7 Hasil Sorting untuk 1.500.000 data*

Proses Sorting 2000000 data...

Algoritma	Waktu (s)	Memori (MB)
Bubble Sort (int)	6499.375	190.00
Selection Sort (int)	1367.513	190.00
Insertion Sort (int)	951.312	190.00
Merge Sort (int)	0.323	190.00
Quick Sort (int)	0.240	190.00
Shell Sort (int)	0.437	190.00

*Gambar 1.8 Hasil Sorting untuk 2.000.000 data*

## 2. Data Kata

Menjalankan sorting untuk 10000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	0.419	0.95
Selection Sort(abjad)	0.118	0.95
Insertion Sort(abjad)	0.050	0.95
Merge Sort (abjad)	0.003	0.95
Quick Sort (abjad)	0.001	0.95
Shell Sort (abjad)	0.003	0.95

Gambar 2.1 Hasil Sorting untuk 10.000 data

Menjalankan sorting untuk 50000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	12.928	4.75
Selection Sort(abjad)	4.475	4.75
Insertion Sort(abjad)	2.045	4.75
Merge Sort (abjad)	0.013	4.75
Quick Sort (abjad)	0.008	4.75
Shell Sort (abjad)	0.020	4.75

Gambar 2.2 Hasil Sorting untuk 50.000 data

Menjalankan sorting untuk 100000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	54.219	9.50
Selection Sort(abjad)	19.470	9.50
Insertion Sort(abjad)	9.345	9.50
Merge Sort (abjad)	0.030	9.50
Quick Sort (abjad)	0.019	9.50
Shell Sort (abjad)	0.056	9.50

Gambar 2.3 Hasil Sorting untuk 100.000 data

Menjalankan sorting untuk 250000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	543.129	23.75
Selection Sort(abjad)	143.806	23.75
Insertion Sort(abjad)	66.983	23.75
Merge Sort (abjad)	0.099	23.75
Quick Sort (abjad)	0.062	23.75
Shell Sort (abjad)	0.235	23.75

Gambar 2.4 Hasil Sorting untuk 250.000 data

Menjalankan sorting untuk 500000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	3002.107	47.50
Selection Sort(abjad)	1111.619	47.50
Insertion Sort(abjad)	449.712	47.50
Merge Sort (abjad)	0.217	47.50
Quick Sort (abjad)	0.142	47.50
Shell Sort (abjad)	0.525	47.50

Gambar2.5 Hasil Sorting untuk 500.000 data

Menjalankan sorting untuk 1000000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	12008.428	95.00
Selection Sort(abjad)	4446.476	95.00
Insertion Sort(abjad)	1798.848	95.00
Merge Sort (abjad)	0.432	95.00
Quick Sort (abjad)	0.283	95.00
Shell Sort (abjad)	1.052	95.00

Gambar 2.6 Hasil Sorting untuk 1.000.data

Menjalankan sorting untuk 1500000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	27018.963	142.50
Selection Sort(abjad)	10004.570	142.50
Insertion Sort(abjad)	4047.408	142.50
Merge Sort (abjad)	0.678	142.50
Quick Sort (abjad)	0.451	142.50
Shell Sort (abjad)	1.649	142.50

Gambar 2.7 Hasil Sorting untuk 1.500.000 data

Menjalankan sorting untuk 2000000 data...

Algoritma	Waktu(s)	Penyimpanan(MB)
Bubble Sort (abjad)	48033.712	190.00
Selection Sort(abjad)	17785.476	190.00
Insertion Sort(abjad)	7203.392	190.00
Merge Sort (abjad)	0.915	190.00
Quick Sort (abjad)	0.624	190.00
Shell Sort (abjad)	2.218	190.00

Gambar 2.8 Hasil Sorting untuk 2.000.000 data

## HASIL ANALISA KODE PROGRAM:

- **Tabel Hasil Eksperimen:**

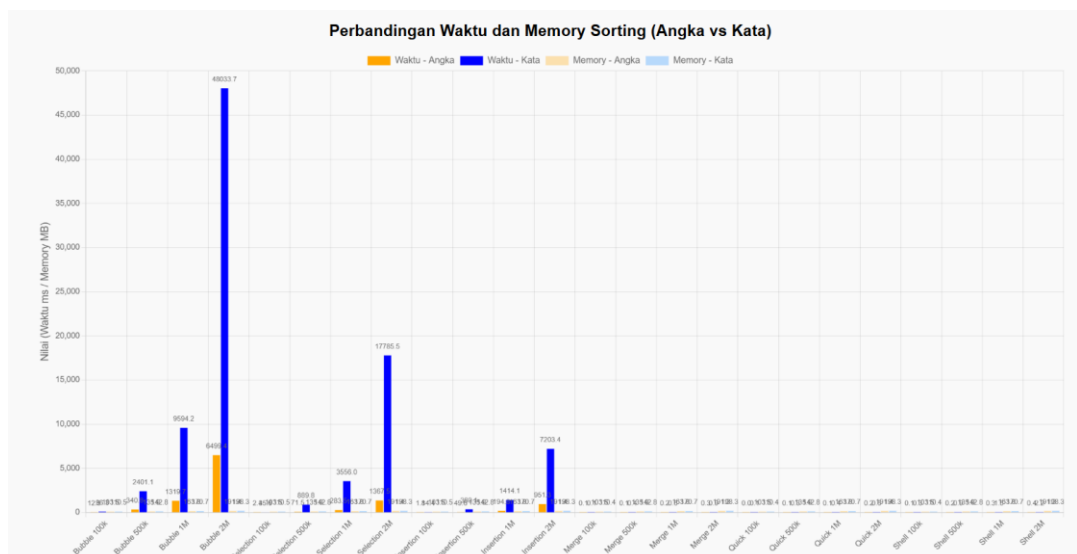
### 1. Data Angka:

Data	Bubble(S)	Selection(S)	Insertion(S)	Merge(S)	Quick(S)	Shell(S)	Memory avg (MB)
10.000	0.331	0.089	0.07	0.003	0.001	0.003	0.95
50.000	9.247	2.061	1.344	0.019	0.012	0.025	4.75
100.000	37.399	8.004	5.108	0.039	0.022	0.045	9.50
250.000	236.835	50.348	32.52	0.093	0.049	0.102	23.75
500.000	955.395	203.973	130.704	0.208	0.094	0.221	47.50
1.000.000	3444.364	460.906	454.329	0.295	0.162	0.312	95.00
1.500.000	5104.406	766.564	586.87	0.242	0.161	0.313	142.50
2.000.000	6499.375	1367.513	951.312	0.323	0.24	0.437	190.00

### 2. Data Kata:

Data	Bubble(S)	Selection(S)	Insertion(S)	Merge(S)	Quick(S)	Shell(S)	Memory avg (MB)
10.000	0.419	0.118	0.05	0.003	0.001	0.003	0.95
50.000	12.928	4.475	2.045	0.013	0.008	0.02	4.75
100.000	54.219	19.47	9.345	0.03	0.019	0.056	9.50
250.000	543.129	143.806	66.983	0.099	0.062	0.235	23.75
500.000	3002.107	1111.619	449.712	0.217	0.142	0.525	47.50
1.000.000	12008.428	4446.476	1798.848	0.432	0.283	1.052	95.00
1.500.000	27018.963	10004.57	4047.408	0.678	0.451	1.649	142.50
2.000.000	48033.712	17785.476	7203.392	0.915	0.624	2.218	190.00

## Hasil perbandingan :



## Analisa dan Kesimpulan

1. Perbedaan efisiensi algoritma: Untuk kedua jenis data, terdapat perbedaan signifikan dalam kecepatan eksekusi di mana Merge Sort, Quick Sort, dan Shell Sort jauh lebih efisien dibandingkan Bubble Sort, Selection Sort, dan Insertion Sort. Misalnya, untuk 2 juta data angka, Quick Sort (0,24 detik) sekitar 27.000 kali lebih cepat dibandingkan Bubble Sort (6499,375 detik).
2. Pengaruh jenis data: Algoritma pengurutan umumnya membutuhkan waktu lebih lama untuk mengurutkan data kata dibandingkan data angka. Contohnya untuk 2 juta data, Bubble Sort membutuhkan waktu 6499,375 detik untuk data angka namun meningkat drastis menjadi 48033,712 detik untuk data kata, menunjukkan bahwa kompleksitas perbandingan string lebih tinggi.
3. Skalabilitas algoritma: Algoritma seperti Merge Sort, Quick Sort, dan Shell Sort menunjukkan skalabilitas yang lebih baik saat ukuran data bertambah. Ketika data bertambah dari 1 juta menjadi 2 juta (dua kali lipat), waktu eksekusi algoritma efisien ini hanya meningkat sekitar 1,5-2 kali, sedangkan algoritma kurang efisien seperti Bubble Sort waktu eksekusinya hampir berlipat ganda.
4. Konsistensi penggunaan memori: Penggunaan memori untuk kedua jenis data (angka dan kata) relatif konsisten untuk jumlah data yang sama. Ini menunjukkan bahwa alokasi memori lebih dipengaruhi oleh jumlah elemen daripada jenis datanya, dengan penggunaan sekitar 190 MB untuk 2 juta data.

### Kesimpulan:

Data tersebut dengan jelas menunjukkan bahwa algoritma pengurutan memiliki perbedaan kinerja yang signifikan, di mana algoritma dengan kompleksitas waktu yang lebih rendah seperti Merge Sort, Quick Sort, dan Shell Sort mengungguli algoritma dengan kompleksitas waktu yang lebih tinggi seperti Bubble Sort. Jenis data yang diurutkan juga mempengaruhi kinerja, dengan data kata umumnya membutuhkan waktu pemrosesan yang lebih lama dibandingkan data angka. Meskipun penggunaan memori relatif konsisten antara kedua jenis data, perbedaan waktu eksekusi menjadi sangat signifikan ketika ukuran data bertambah, terutama untuk algoritma yang kurang efisien. Untuk aplikasi dengan dataset besar, pemilihan algoritma pengurutan yang tepat menjadi sangat krusial, dengan Quick Sort muncul sebagai algoritma tercepat di sebagian besar kasus yang diujikan.