LAPORAN PRAKTIKUM PEKAN 6 STRUKTUR DATA



NAMA : FIKHRI HANIF

NIM : 2311533007

DOSEN PENGAMPU : DR. WAHYUDI,S.T,MT.

DEPARTEMEN INFORMATIKA FAKULTAS TEKNOLOGI INFORMASI UNIVERSITAS ANDALAS

PADANG

1. Tujuan
   1. Memahami cara penggunaan Shell Sort, Merge Sort, dan Quick Sort.
   2. Memahami cara kerja Shell Sort, Merge Sort, dan Quick Sort pada data.
   3. Membuat program menggunakan Shell Sort, Merge Sort, dan Quick Sort.
2. Kajian Teori Shell Sort

Shell Sort adalah pengembangan dari Insertion Sort yang memperkenalkan konsep "gap" untuk mempercepat proses pengurutan. Algoritma ini dinamakan sesuai dengan penemunya, Donald Shell, yang pertama kali memperkenalkannya pada tahun 1959.

* Prinsip Kerja: Algoritma ini mengurutkan elemen-elemen yang berjarak tertentu (gap) satu sama lain, kemudian mengurangi gap secara bertahap hingga gap menjadi 1, di mana algoritma tersebut akan menjadi Insertion Sort. Penggunaan gap yang lebih besar pada tahap awal membantu memindahkan elemen-elemen ke posisi yang lebih mendekati posisi akhirnya lebih cepat.
* Kompleksitas Waktu: Kompleksitas waktu tergantung pada pilihan urutan gap yang digunakan. Dalam kasus terbaik, dengan gap yang dipilih secara optimal, kompleksitas bisa mendekati O(n log n). Namun, pada kasus terburuk, kompleksitas bisa mendekati O(n^2).
* Kelebihan dan Kekurangan:
  + Kelebihan: Lebih cepat daripada Insertion Sort untuk dataset yang lebih besar, mudah diimplementasikan.
  + Kekurangan: Kompleksitas tergantung pada urutan gap yang dipilih, yang bisa sulit untuk dioptimalkan.

Merge Sort

Merge Sort adalah algoritma pengurutan berbasis pembagian dan penaklukan (divide and conquer) yang sangat efisien. Algoritma ini diperkenalkan oleh John von Neumann pada tahun 1945.

* Prinsip Kerja: Algoritma ini membagi array menjadi dua bagian yang lebih kecil, kemudian mengurutkan kedua bagian tersebut secara rekursif, dan akhirnya menggabungkan kedua bagian yang sudah terurut menjadi satu array yang terurut.
* Kompleksitas Waktu: Merge Sort memiliki kompleksitas waktu O(n log n) pada semua kasus (terbaik, rata-rata, dan terburuk), yang membuatnya sangat efisien untuk dataset yang besar.
* Kelebihan dan Kekurangan:
  + Kelebihan: Konsisten dengan kompleksitas O(n log n), stabil, dan bekerja dengan baik untuk dataset yang besar.
  + Kekurangan: Memerlukan ruang tambahan sebesar O(n) untuk menyimpan array sementara selama proses penggabungan.

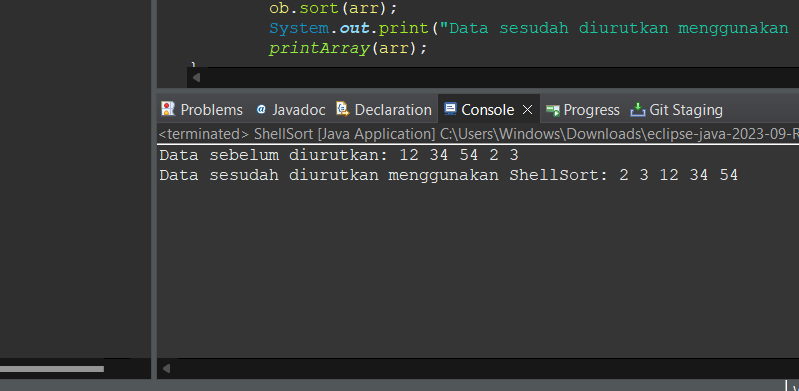
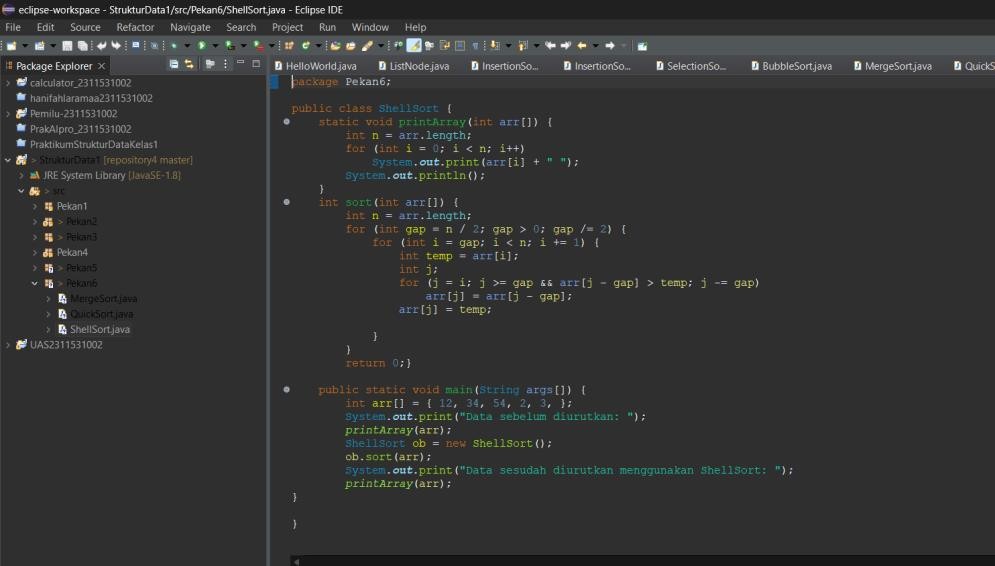
Quick Sort

Quick Sort adalah algoritma pengurutan berbasis pembagian dan penaklukan yang diperkenalkan oleh Tony Hoare pada tahun 1960. Algoritma ini terkenal karena kecepatan dan efisiensinya.

* Prinsip Kerja: Algoritma ini memilih sebuah elemen sebagai pivot dan membagi array menjadi dua subarray: elemen-elemen yang lebih kecil dari pivot dan

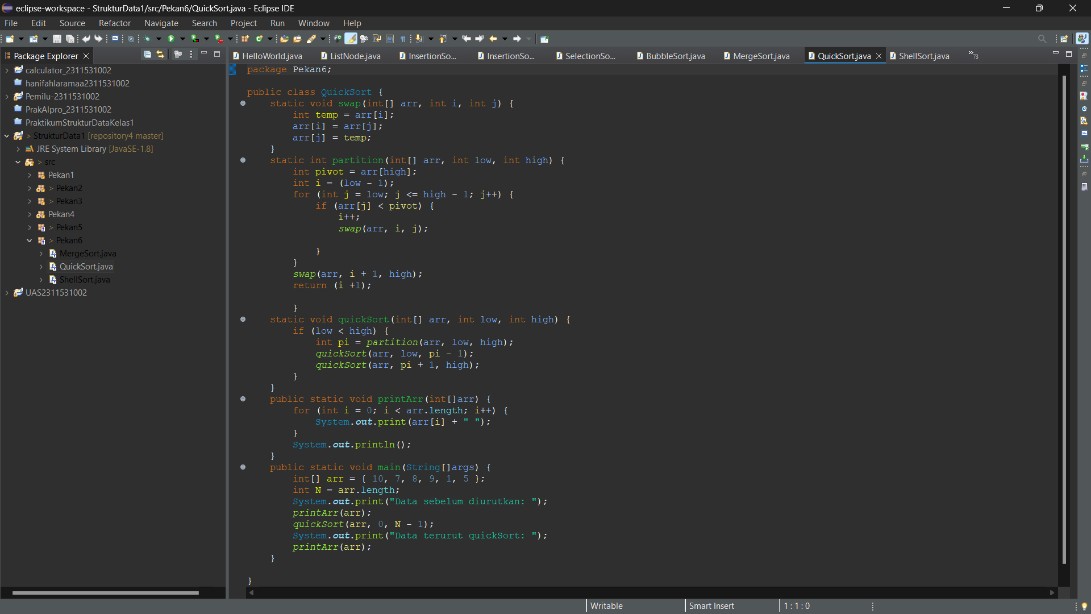
elemen-elemen yang lebih besar dari pivot. Proses ini diulang secara rekursif untuk kedua subarray.

* Kompleksitas Waktu: Kompleksitas waktu rata-rata adalah O(n log n). Namun, pada kasus terburuk (ketika pivot selalu elemen terkecil atau terbesar), kompleksitas waktu bisa menjadi O(n^2). Penggunaan teknik seperti randomisasi pivot atau median-of-three dapat mengurangi kemungkinan terjadinya kasus terburuk.
* Kelebihan dan Kekurangan:
  + Kelebihan: Sangat cepat untuk data yang besar, menggunakan in-place sorting sehingga tidak memerlukan ruang tambahan yang signifikan.
  + Kekurangan: Tidak stabil, performa sangat bergantung pada pemilihan pivot yang baik.

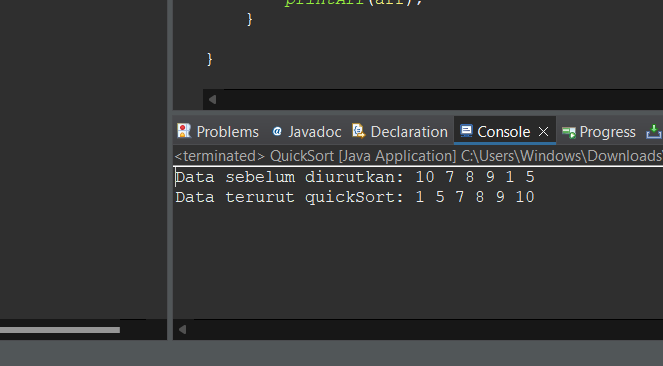
1. Langkah Kerja
   1. ShellSort Input:

Output:

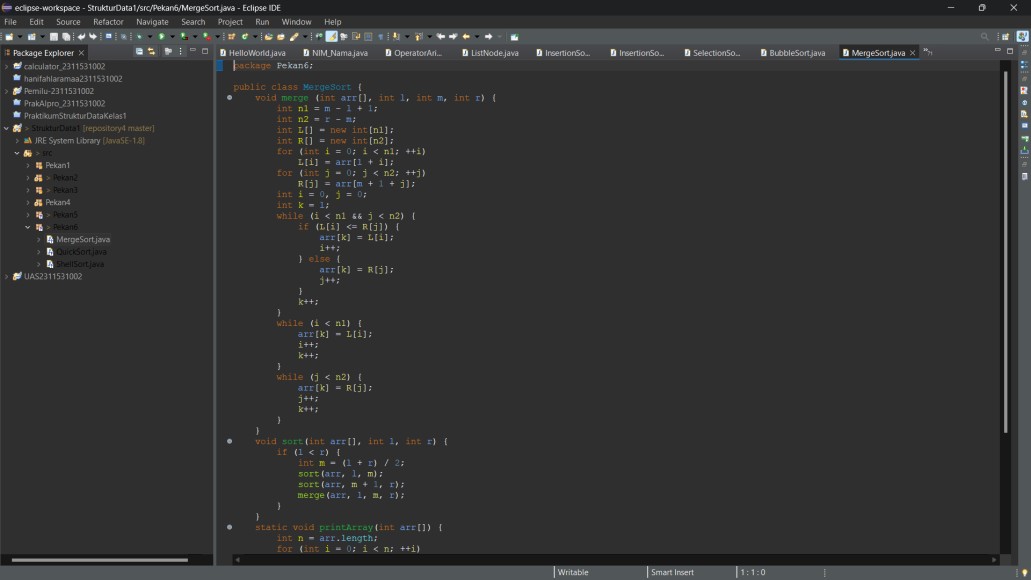
* 1. QuickSort Input:

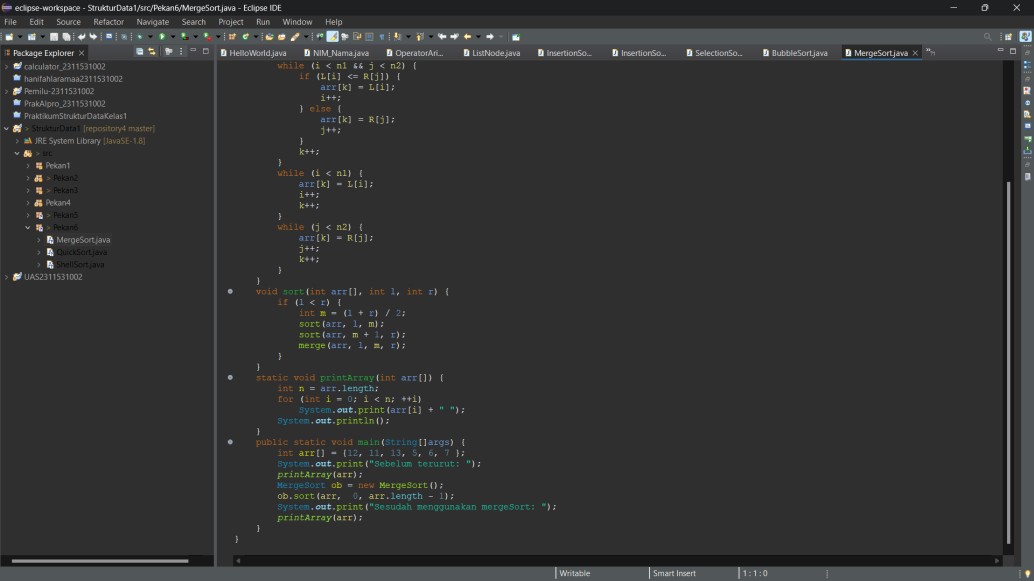


Output:

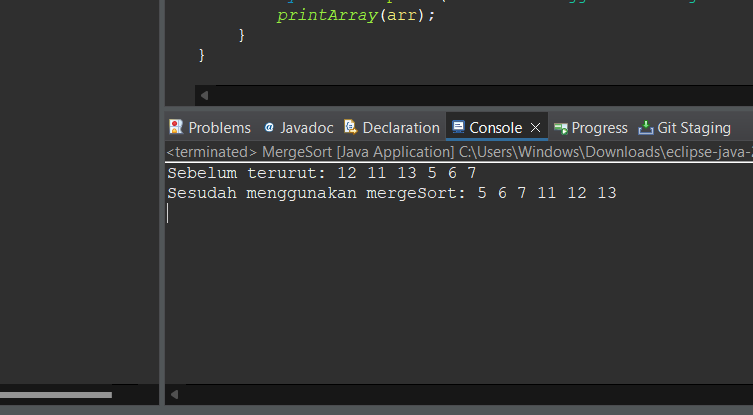


* 1. MergeSort Input:





Output:



1. Kesimpulan

Dalam praktikum ini, kami telah mempelajari dan mengimplementasikan tiga algoritma pengurutan yang berbeda: Shell Sort, Merge Sort, dan Quick Sort. Setiap algoritma memiliki karakteristik dan performa yang unik tergantung pada jenis data dan kondisi pengurutan yang dihadapi.

Secara keseluruhan, melalui praktikum ini, kita memahami bahwa pemilihan algoritma pengurutan harus mempertimbangkan karakteristik data dan kebutuhan aplikasi. Shell Sort, Merge Sort, dan Quick Sort masing-masing memiliki keunggulan dan kelemahan yang membuat mereka lebih cocok untuk situasi tertentu. Penguasaan terhadap berbagai algoritma ini memberikan dasar yang kuat untuk menerapkan solusi pengurutan yang optimal dalam berbagai konteks aplikasi praktis.

Dengan demikian, praktikum ini tidak hanya meningkatkan pemahaman kami tentang algoritma-algoritma pengurutan tetapi juga memberikan wawasan mengenai pentingnya analisis dan evaluasi performa dalam memilih algoritma yang tepat untuk masalah pengurutan yang dihadapi.