**LAPORAN PRAKTIKUM DATA STRUCTURE**

**SIMPLE SORTING**

**Dosen Pengampu:**

**H. Fatchurrochman,M.Kom**

**Asisten Praktikum:**

**Fillah Anjany 230605110033**

****

**Oleh :**

|  |  |
| --- | --- |
| **240605110082** | **Muhammad Alif Mujaddid** |

**Jurusan Teknik Informatika Fakultas Sains dan Teknologi**

**UIN Maulana Malik Ibrahim Malang**

**2025**

**A. Pendahuluan**

1. Algoritma sorting yang paling sederhana adalah bubble sort. Langkah

sorting secara ascending menggunakan bubble sort adalah sebagai berikut:

a. Bandingkan dua item

b. Jika item pertama (sebelah kiri) lebih besar daripada item kedua (sebelah

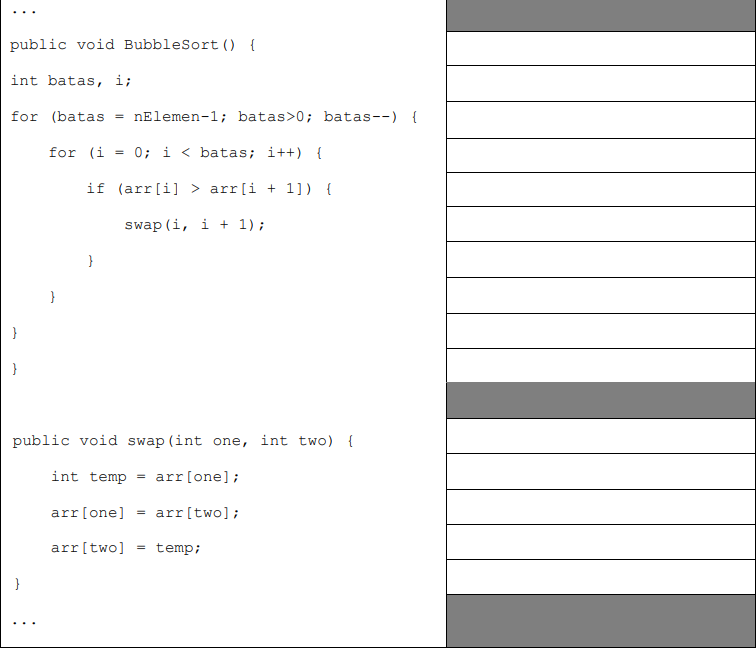
kanan), maka tukar kedua item tersebut

c. Pindah ke posisi kanan. Lakukan langkah a dan b hingga posisi sampai di

batas akhir

d. Ketika satu item telah berada sesuai urutan, ulangi langkah a-c hingga

semua item sesuai urutan.



Masukan temp ke arr[two]

Masukan arr[two] ke arr[one]

Buat variable sementara untuk menimpa variable yang akan diturkar

Deklarasi methode untuk menukar data 1 dg sebelahnya

Tukar array I dengan setelahnya

Jika arr ke I lebih besar dari setelahnya :

Inner loop dari 0 sampai batas

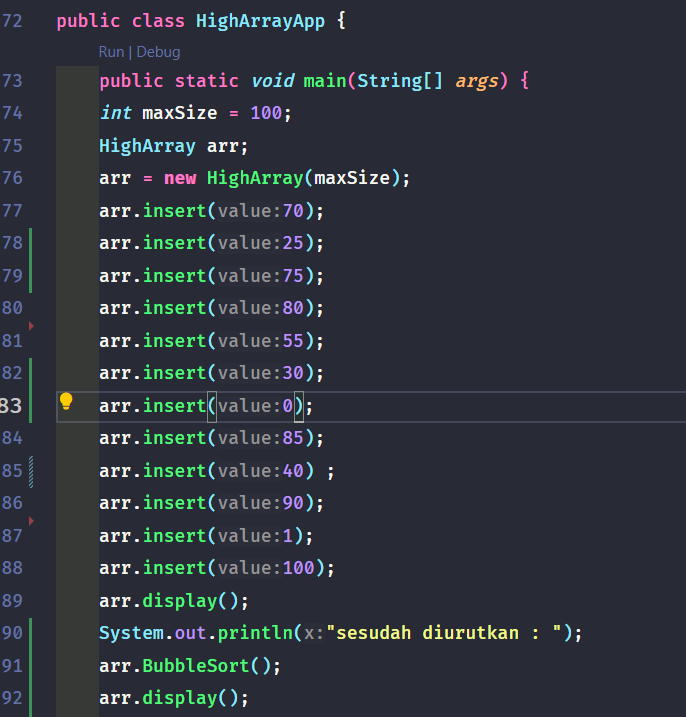
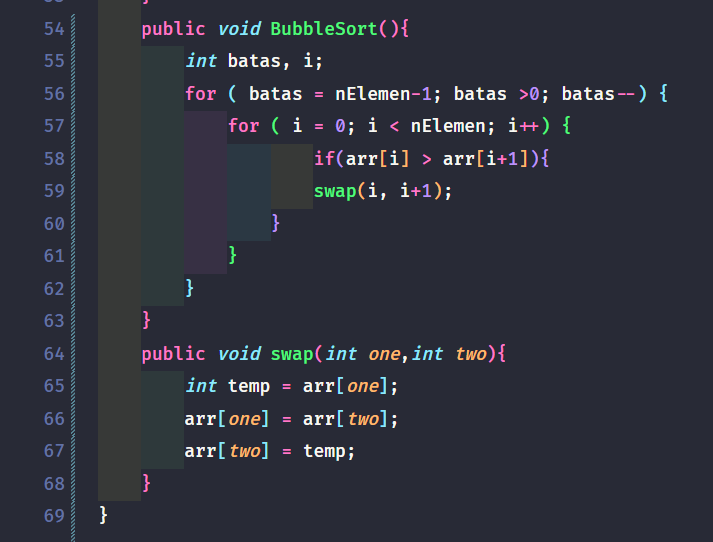
Deklarasi Methode

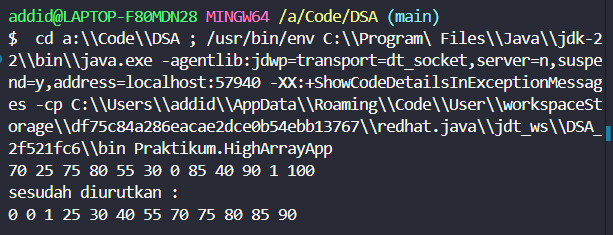
Membuat variable batas dan iterasi

Membuat itertasi dari Panjang data sampai 0 (dec)

Listing diatas adalah baris kode yang mengimplementasikan tiap langkah bubble sort. Implementasikan baris kode tersebut ke dalam sebuah program sehingga dapat digunakan untuk mengurutkan elemen array. Anda dapat menambahkan listing

tersebut pada class HighArray (listing 5 praktikum 1) yang dipanggil pada class HighArrayApp. Atau anda juga dapat menuliskan listing ini pada satu class secara terstruktur. Insert 6 item pada array program tersebut. Tampilkan isi array sebelum dilakukan pengurutan. Kemudian urutkan dan tampilkan isi array setelah pengurutan. Jalankan program yang telah Anda buat dan tuliskan output program tersebut.

Output :

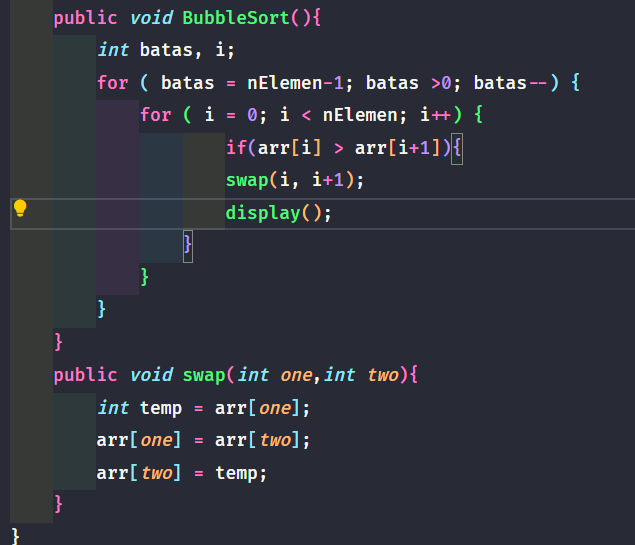
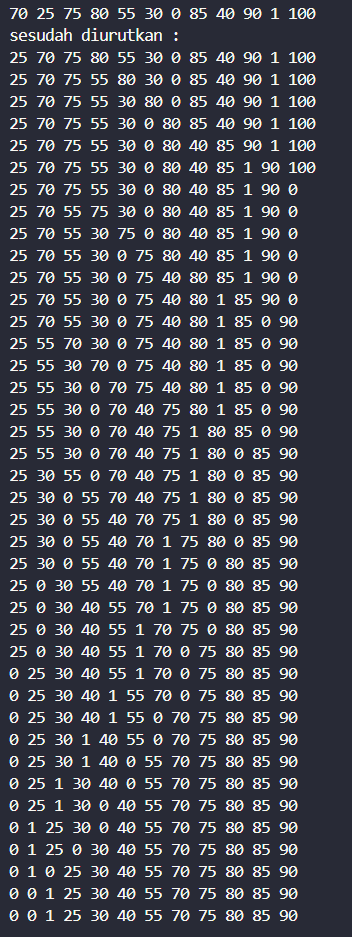


2. Untuk sorting array yang memiliki 6 item, berapa jumlah perbandingan item yang dilakukan hingga semua item sesuai urutan?

Jumlah Perbandingan=n(n−1) / 2

Jumlah Perbandingan=6×(6−1)/2​=6×5/2​=15

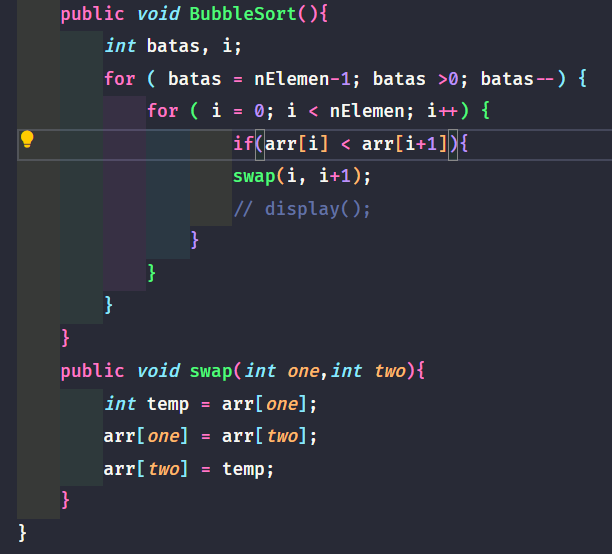
3. Pada program nomer 1, tambahkan kode untuk menampilkan isi array setelah baris kode pertukaran item (swap(i, i + 1);). Jalankan kembali dan amati bagaimana proses pengurutan pada tiap iterasi. Tuliskan isi array pada 10 perulangan pertama. Jelaskan!

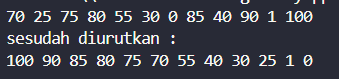
 ​

Bubble Sort itu kayak gelembung di air.  
Setiap kali gelembung bergerak, dia bakal naik ke atas pelan-pelan.  
Nah, di sini yang naik ke atas itu angka yang paling besar.

Caranya:

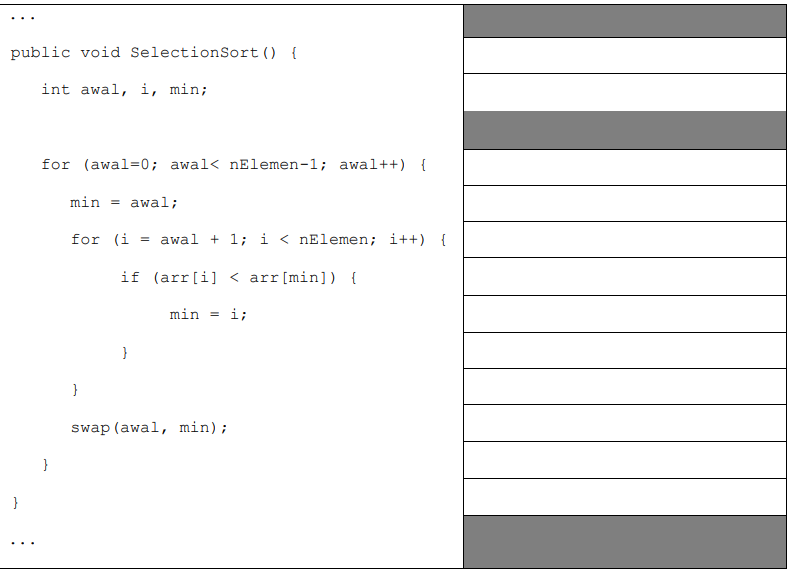
1. Bandingkan dua angka yang bersebelahan.
2. Kalau yang kiri lebih besar dari yang kanan => tukar tempat.
3. Ulangi terus sampai selesai 1 putaran => angka paling besar sudah ada di ujung kanan.
4. Lanjut lagi putaran berikutnya, sekarang angka terbesar kedua naik ke sampingnya.
5. Begitu terus sampai semua angka rapi.

4. Jika Anda ingin melakukan sorting menggunakan algoritma bubble sort secara descending, maka pada program nomer 1 bagian manakah yang harus diganti. Tuliskan code-nya dan jelaskan! Output :



Jadi yang diganti itu dipengecekanya jadi kurang dari, karena jika kita lihat semisal 70 < 80 maka true dan akan dilakukan swap yang mana 80 akan berada di kanan.

5. Algoritma sorting yang lain adalah selection sort. Langkah selection sort untuk pengurutan secara ascending yaitu: a. Cari item terkecil pada array b. Letakkan item terkecil sesuai urutannya dengan cara menukar item terkecil dengan item pada index awal c. Geser posisi awal pencarian ke kanan. d. Lakukan langkah a, b, dan c hingga semua item terurut. Berikut ini listing untuk algoritma selection sort. Lengkapi sebagaimana soal nomor 1, gunakan data array yang sama, jalankan dan jelaskan tiap baris code pengurutan berikut ini!



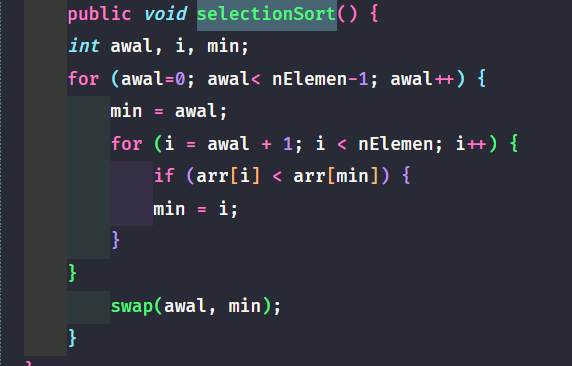
Tukar elemen di posisi awal dengan elemen terkecil yang ketemu (min). Jadi elemen terkecil pindah ke depan.

Anggap dulu nilai terkecil ada di posisi awal.

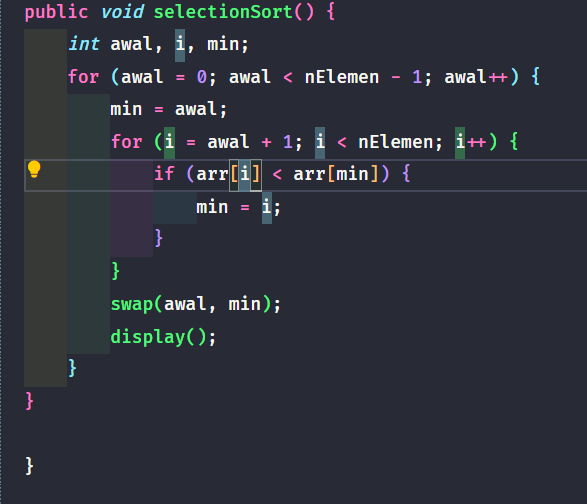
Bandingkan elemen setelah awal dengan elemen terkecil sementara (min). Kalau nemu elemen yang lebih kecil, ganti min jadi posisi elemen itu. Jadi sekarang min nunjuk ke elemen terkecil baru.

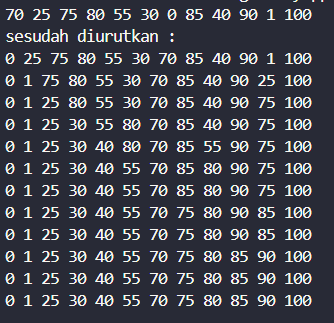
Mulai dari elemen pertama sampai elemen terakhir. Setiap langkah, kita anggap posisi awal itu yang mau ditaruh nilai terkecil.

Deklarasi variabel:awal = posisi elemen yang lagi dicek.i = buat jalan ngecek elemen lain. min = index nilai terkecil yang ketemu.



1. Pada listing nomor 5, tambahkan kode untuk menampilkan isi elemen pada array setelah kode pertukaran (swap(awal, min);). Jalankan program, amati dan tulis outputnya, kemudian jelaskan!





Jadi penjelasan gampangnya tuh si array yang paling kiri dibandingkan paling kanan, kalau ketemu angka yang lebih kecil, langsung ditandain dulu, terus di akhir tuker posisinya. Habis itu geser ke elemen berikutnya di kiri, terus ulang lagi sampai semua elemen diurutkan.

7. Sedikit berbeda dengan dua algoritma sebelumnya, pada insertion sort, aksi yang dilakukan adalah membandingkan dan meng-copy item. Berikut ini langkah insertion sort untuk pengurutan secara ascending:

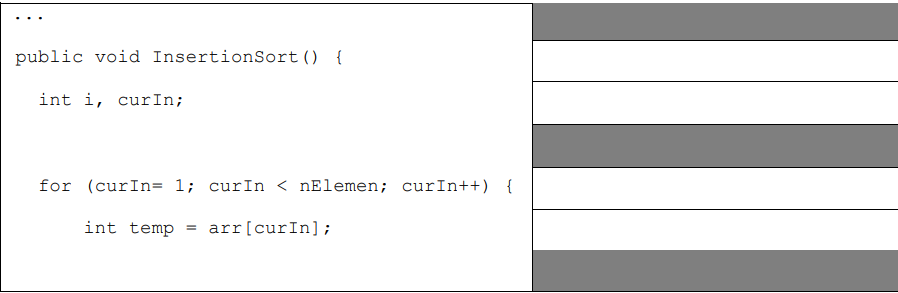
a. Tandai sebuah item sebagai batas antara partially sorted dan unsorted items.

b. Geser item pada partially sorted yang bernilai lebih besar dari pada item yang ditandai pada langkah a.

c. Sisipkan item tersebut pada posisi yang sesuai di bagian partially sorted.

d. Ulangi langkah a-c hingga semua unsorted items telah disisipkan (insert) ke sorted group. Berikut ini listing untuk algoritma insertion sort.

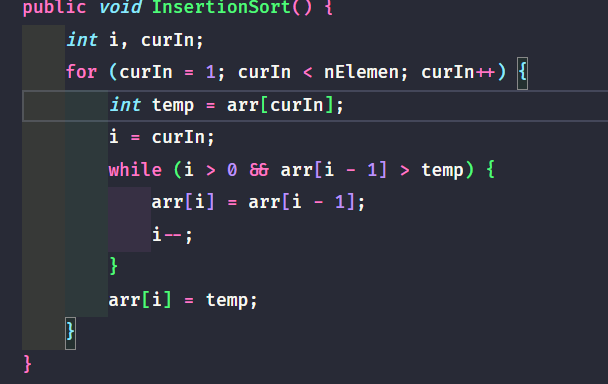
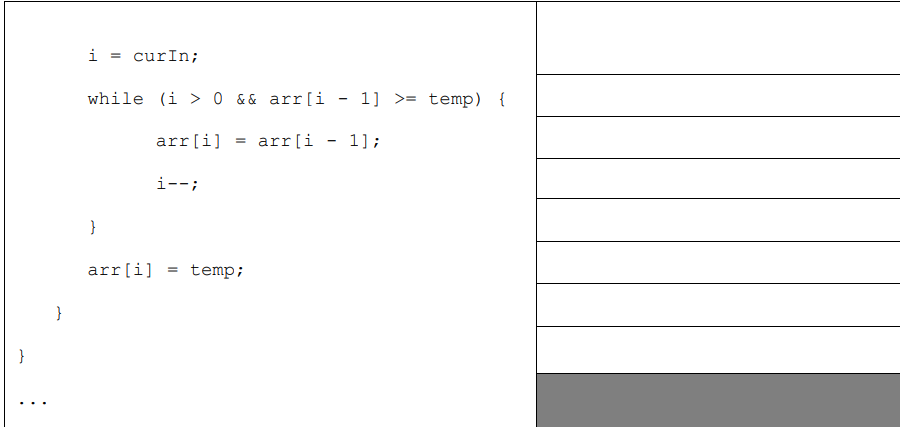
Lengkapi sebagaimana soal nomor 1, gunakan data array yang sama, jalankan dan jelaskan tiap baris code pengurutan berikut ini!



Ambil angka baru: mulai dari elemen ke-1 (karena elemen pertama dianggap sudah “rapi”), simpan di temp buat dibandingin.

Deklarasi variable I untuk geser dan curin untuk yg dicek

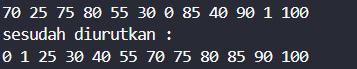
Deklarasi Method insert



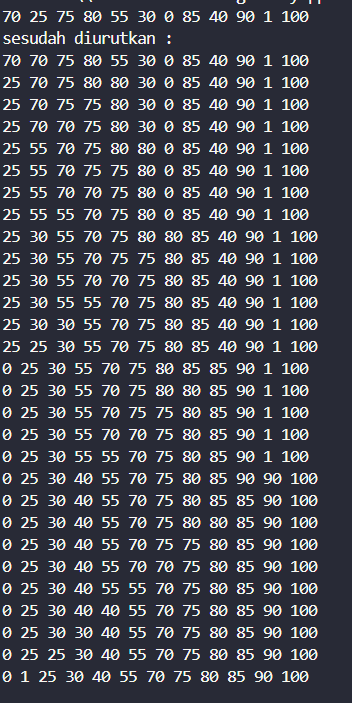
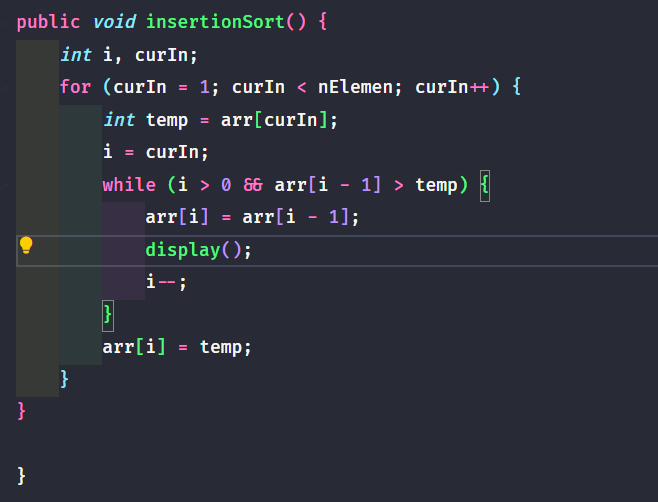
Taruh angka di posisi yang pas: setelah ketemu tempat yang tepat, masukkan angka yang tadi dipegang (temp).

Geser elemen ke kanan: selama angka di kiri (arr[i-1]) lebih besar dari angka yang lagi dipegang (temp), geser elemen itu ke kanan, terus mundur satu langkah.

Output :



8. Pada listing nomor 7, tambahkan kode untuk menampilkan isi elemen pada array setelah kode pergeseran item (arr[i] = arr[i - 1];) dan setelah kode insert item (arr[i] = temp;). Jalankan program, amati dan tulis outputnya hingga 4 kali tahap penyisipan, kemudian jelaskan!



Jadi penjelasan gampangnya tuh si array jalan dari kiri ke kanan. Setiap ambil angka baru, dia dibandingin sama angka-angka di sebelah kirinya. Kalau ada yang lebih besar, angka itu digeser ke kanan terus sampai ketemu posisi yang pas. Baru deh angka yang dipegang tadi ditaruh di situ. Ulang terus sampai semua angka rapi dari kecil ke besar.