LAPORAN PRAKTIKUM

SORTING ALGORITHMS



Oleh :

MUHAMMAD GALID AVERO

NIM 2311532008

MATA KULIAH STRUKTUR DATA

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

ASISTEN LABORATORIUM : ZAKY ADIL HAKIM

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

PADANG, MEI 2024

1. **PEDAHULUAN**

Sorting algorithms adalah prosedur atau metode yang digunakan untuk mengatur elemen-elemen dalam suatu daftar atau array dalam urutan tertentu, biasanya dalam urutan menaik (ascending) atau menurun (descending). Sorting adalah salah satu operasi fundamental dalam ilmu komputer dan penting dalam berbagai aplikasi, seperti pengolahan data, algoritma pencarian, dan optimasi.

Berikut adalah penjelasan mengenai berbagai sorting algorithms:

**1. Bubble Sort**

**Deskripsi:** Bubble Sort adalah algoritma penyortiran sederhana yang berulang kali membandingkan elemen yang berdekatan dan menukarnya jika mereka berada dalam urutan yang salah.

**Langkah-langkah:**

1. Bandingkan elemen pertama dengan elemen kedua. Jika elemen pertama lebih besar dari elemen kedua, tukar mereka.
2. Lanjutkan proses ini untuk setiap pasangan elemen berikutnya hingga akhir array.
3. Ulangi langkah-langkah di atas untuk seluruh array hingga tidak ada lagi pertukaran yang diperlukan.

**Kompleksitas Waktu:**

* Worst-case: O(n2)O(n^2)O(n2)
* Best-case: O(n)O(n)O(n) (ketika array sudah tersortir)
* Average-case: O(n2)O(n^2)O(n2)

**Kelebihan:**

* Implementasi sederhana
* Cocok untuk daftar kecil

**Kekurangan:**

* Tidak efisien untuk daftar besar

**2. Insertion Sort**

**Deskripsi:** Insertion Sort adalah algoritma penyortiran yang membagi array menjadi subarray yang sudah tersortir dan subarray yang belum tersortir. Setiap elemen dari subarray yang belum tersortir diambil dan ditempatkan di posisi yang benar dalam subarray yang sudah tersortir.

**Langkah-langkah:**

1. Mulai dari elemen kedua, bandingkan dengan elemen di depannya dan tempatkan di posisi yang benar dalam subarray yang sudah tersortir.
2. Ulangi proses ini untuk setiap elemen hingga akhir array.

**Kompleksitas Waktu:**

* Worst-case: O(n2)O(n^2)O(n2)
* Best-case: O(n)O(n)O(n) (ketika array sudah tersortir)
* Average-case: O(n2)O(n^2)O(n2)

**Kelebihan:**

* Implementasi sederhana
* Efisien untuk daftar kecil atau hampir tersortir

**Kekurangan:**

* Tidak efisien untuk daftar besar

**3. Selection Sort**

**Deskripsi:** Selection Sort adalah algoritma penyortiran yang membagi array menjadi dua bagian: subarray yang sudah tersortir dan subarray yang belum tersortir. Algoritma ini berulang kali memilih elemen terkecil dari subarray yang belum tersortir dan menempatkannya di posisi yang benar dalam subarray yang sudah tersortir.

**Langkah-langkah:**

1. Temukan elemen terkecil di dalam array.
2. Tukar elemen terkecil tersebut dengan elemen pertama array.
3. Ulangi proses ini untuk elemen kedua, ketiga, dan seterusnya.

**Kompleksitas Waktu:**

* Worst-case: O(n2)O(n^2)O(n2)
* Best-case: O(n2)O(n^2)O(n2)
* Average-case: O(n2)O(n^2)O(n2)

**Kelebihan:**

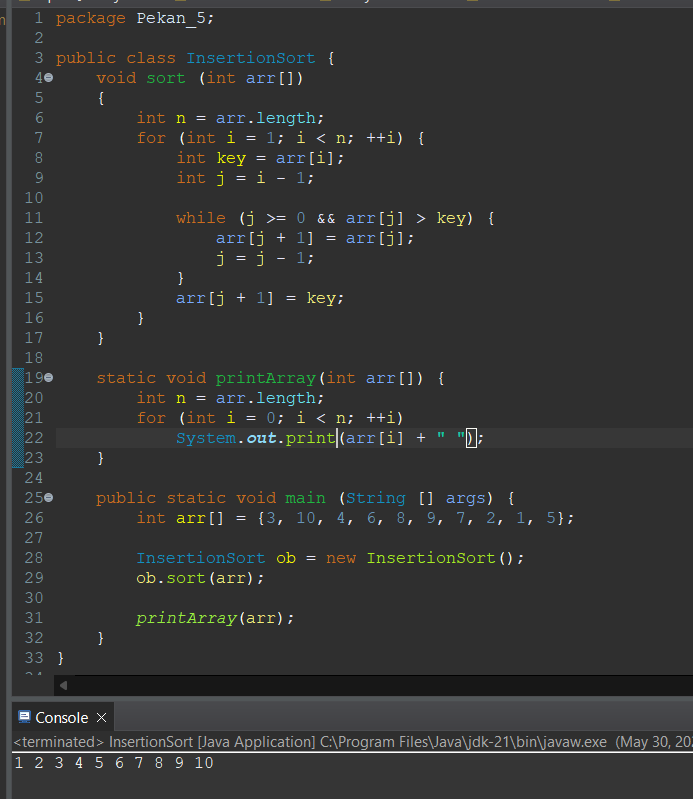
* Implementasi sederhana
* Jumlah pertukaran lebih sedikit dibandingkan Bubble Sort

**Kekurangan:**

* Tidak efisien untuk daftar besar

1. **TUJUAN PRAKTIKUM**
2. Mengetahui cara membuat InsertionSort
3. Mengetahui cara membuat SelectionSort
4. Mengetahui cara membuat BubbleSort
5. **LANGKAH PRAKTIKUM**

1. InsertionSort 1



Metode sort:

void sort(int arr[]): Metode ini menerima array integer arr sebagai input dan mengurutkannya menggunakan Insertion Sort.

int n = arr.length; : Menentukan panjang array arr dan menyimpannya ke dalam variabel n.

for (int i = 1; i < n; ++i) { ... }: Loop utama yang mengiterasi melalui elemen array dari indeks 1 hingga n - 1.

int key = arr[i];: Menyimpan elemen saat ini (indeks i) ke dalam variabel key.

int j = i - 1;: Inisialisasi variabel j dengan indeks sebelumnya dari elemen saat ini.

while (j >= 0 && arr[j] > key) { ... }: Loop yang membandingkan key dengan elemen sebelumnya (di indeks j) dan menggeser elemen-elemen yang lebih besar daripada key ke kanan.

arr[j + 1] = arr[j];: Menggeser elemen di indeks j ke indeks j + 1.

j = j - 1;: Mengurangi j untuk memeriksa elemen sebelumnya.

arr[j + 1] = key;: Memasukkan key (elemen saat ini) ke posisi yang benar di array.

Metode printArray:

static void printArray(int arr[]) { ... }: Metode ini menerima array integer arr sebagai input dan mencetaknya ke konsol.

int n = arr.length;: Menentukan panjang array arr.

for (int i = 0; i < n; ++i) { ... }: Loop yang mengiterasi melalui elemen array dari indeks 0 hingga n - 1.

System.out.print(arr[i] + " ");: Mencetak elemen di indeks i ke konsol, diikuti dengan spasi.

Metode main:

public static void main(String[] args) { ... }: Titik masuk program utama.

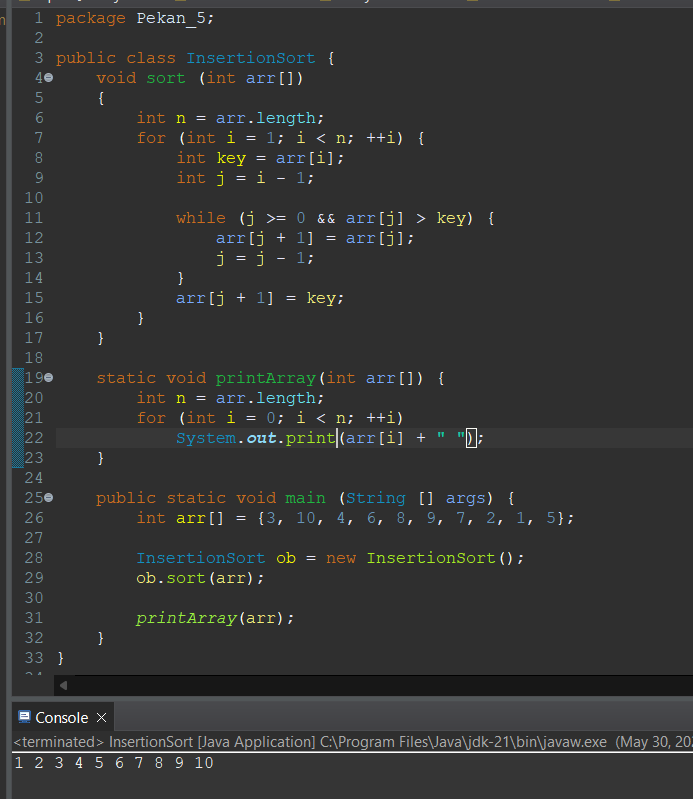
int arr[] = {3, 10, 4, 6, 8, 9, 7, 2, 1, 5};: Mendefinisikan array integer arr yang akan diurutkan.

InsertionSort ob = new InsertionSort();: Membuat objek ob dari kelas InsertionSort.

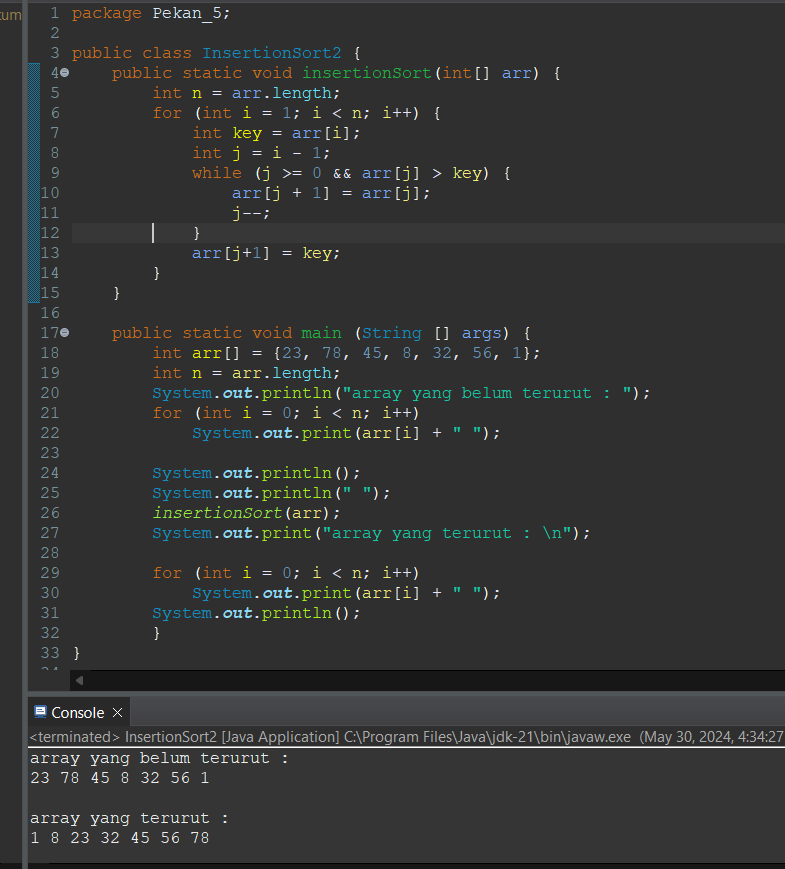
ob.sort(arr);: Memanggil metode sort pada objek ob untuk mengurutkan array arr.

printArray(arr);: Memanggil metode printArray untuk mencetak array arr yang sudah terurut ke konsol.

Untuk hasil dari program adalah sebagai berikut :



2. InsertionSort 2



Method insertionSort

Method insertionSort menerima array bilangan bulat arr sebagai parameter. Method ini melakukan pengurutan array menggunakan algoritma Insertion Sort.

Di dalam method, variabel n menyimpan panjang array.

Loop pertama (for) iterasi dari elemen kedua array (i = 1) hingga elemen terakhir (i < n).

Pada setiap iterasi, variabel key menyimpan elemen array yang akan diurutkan.

Loop kedua (while) iterasi dari elemen sebelum key (j = i - 1) hingga nilai key lebih kecil dari elemen sebelumnya atau j mencapai batas bawah array (j >= 0).

Di dalam loop while, elemen array yang lebih besar dari key digeser ke kanan satu posisi (arr[j + 1] = arr[j]).

Setelah loop while selesai, elemen key ditempatkan pada posisi yang tepat.

Method main

Method main merupakan titik awal program.

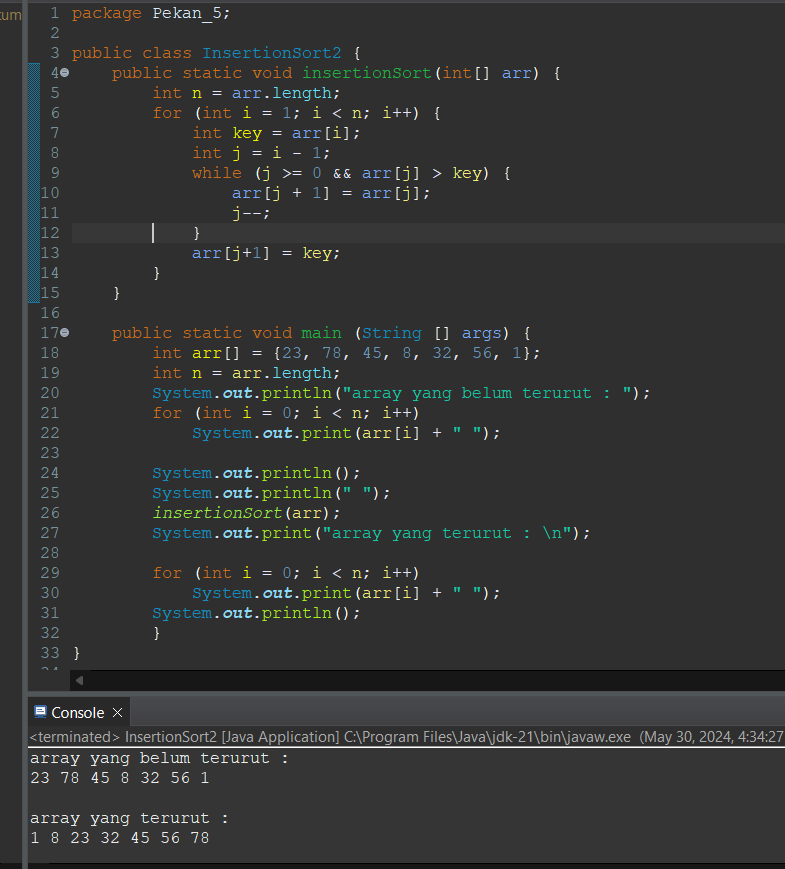
Di dalam method main, array bilangan bulat arr diinisialisasi dengan nilai [23, 78, 45, 8, 32, 56, 1].

Kode kemudian menampilkan array sebelum diurutkan.

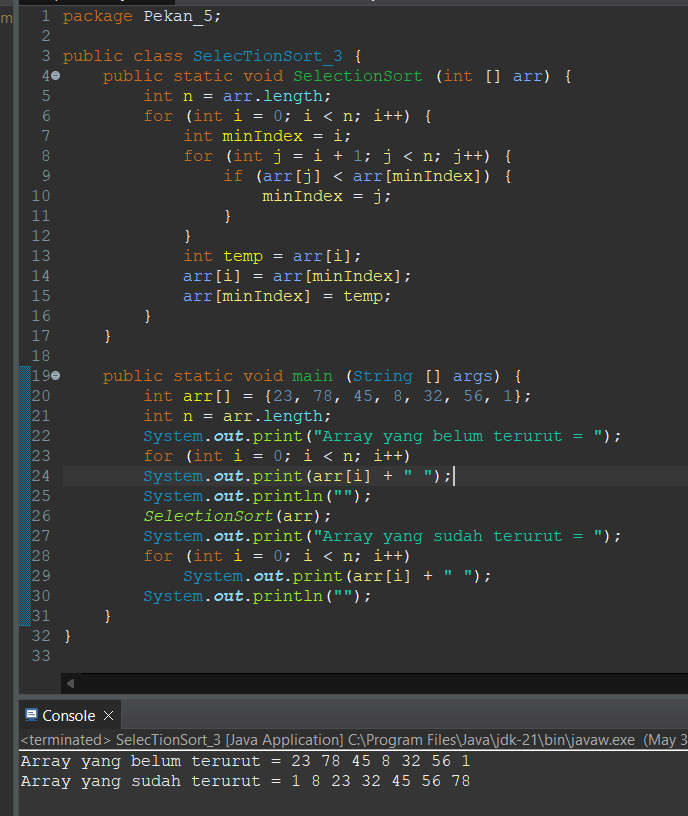
Selanjutnya, method insertionSort dipanggil untuk mengurutkan array.

Terakhir, kode menampilkan array yang telah diurutkan.

Untuk hasil dari kodenya adalah sebagai berikut :



3. SelectionSort

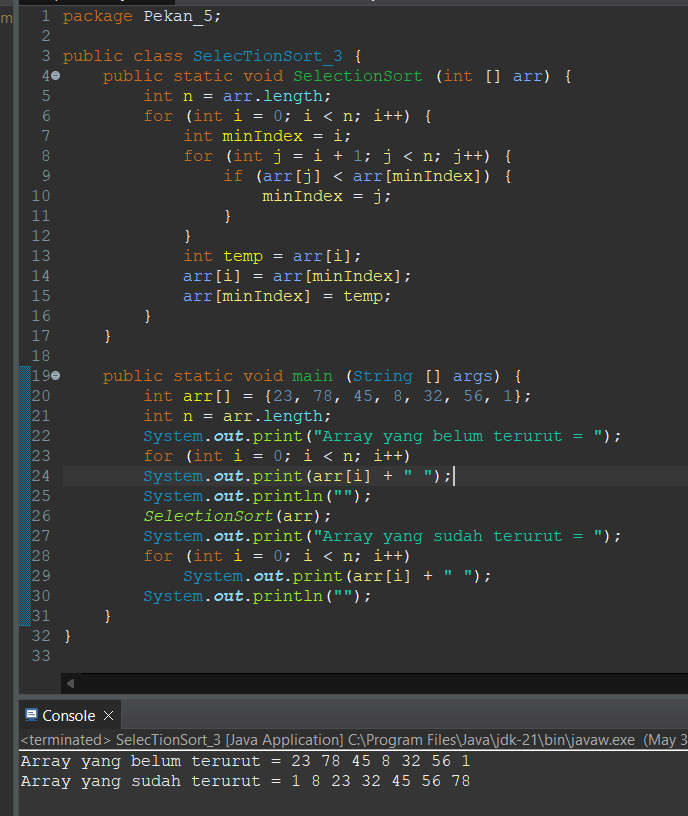


Metode sort menerima sebuah array bilangan bulat sebagai parameter dan mengurutkan array tersebut menggunakan algoritma Insertion Sort. Metode ini menggunakan looping untuk mengurutkan array dari elemen kedua hingga akhir. Setiap elemen diurutkan dengan cara mencari posisi yang tepat di dalam array yang sudah terurut sebelumnya.

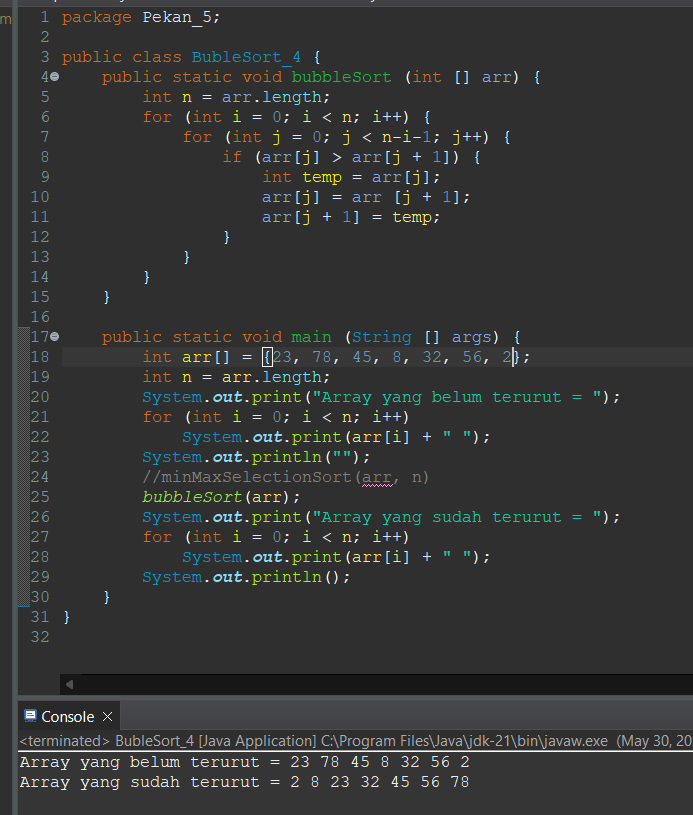
Metode printArray menerima sebuah array bilangan bulat sebagai parameter dan mencetak semua elemen array tersebut ke layar.

Dalam metode main, program membuat sebuah array bilangan bulat dan memanggil metode sort untuk mengurutkan array tersebut. Setelah itu, program memanggil metode printArray untuk mencetak array yang sudah terurut ke layar.

Berikut adalah hasil dari program tersebut



4. BubbleSort



Deklarasi Method

public static void bubbleSort(int[] arr) {: Deklarasi method bubbleSort dengan tipe static yang menerima parameter array integer arr dan tidak mengembalikan nilai (void). Method ini berisi logika algoritma bubble sort.

Inisialisasi Variabel

int n = arr.length;: Deklarasi variabel n yang menyimpan panjang array arr yang akan diurutkan.

Looping Utama untuk Perbandingan dan Penukaran

for (int i = 0; i < n; i++) {: Looping luar yang akan berulang n kali, dimana i adalah counter iterasi yang dimulai dari 0.

for (int j = 0; j < n - i - 1; j++) {: Looping dalam yang akan berulang n - i - 1 kali, dimana j adalah counter iterasi yang dimulai dari 0. Looping dalam ini bertujuan untuk membandingkan setiap elemen array dengan elemen selanjutnya.

if (arr[j] > arr[j + 1]) {: Perbandingan antara elemen ke-j dengan elemen ke-j+1. Jika elemen ke-j lebih besar dari elemen ke-j+1, maka dilakukan penukaran.

int temp = arr[j];: Deklarasi variabel temp yang menyimpan nilai elemen ke-j sebelum ditukar.

arr[j] = arr[j + 1];: Nilai elemen ke-j digantikan dengan nilai elemen ke-j+1.

arr[j + 1] = temp;: Nilai elemen ke-j+1 digantikan dengan nilai yang tersimpan di temp.

Method Main untuk Eksekusi Program

public static void main(String[] args) {: Deklarasi method main yang merupakan titik awal eksekusi program.

int arr[] = {23, 78, 45, 8, 32, 56, 2};: Deklarasi array arr yang berisi data bilangan bulat yang akan diurutkan.

int n = arr.length;: Deklarasi variabel n yang menyimpan panjang array arr.

System.out.print("Array yang belum terurut ");: Menampilkan teks "Array yang belum terurut " ke console.

for (int i = 0; i < n; i++): Looping untuk menampilkan elemen array sebelum diurutkan.

System.out.print(arr[i] + " ");: Menampilkan nilai setiap elemen array ke console.

System.out.println("");: Menampilkan baris baru ke console.

bubbleSort(arr);: Memanggil method bubbleSort untuk mengurutkan array arr.

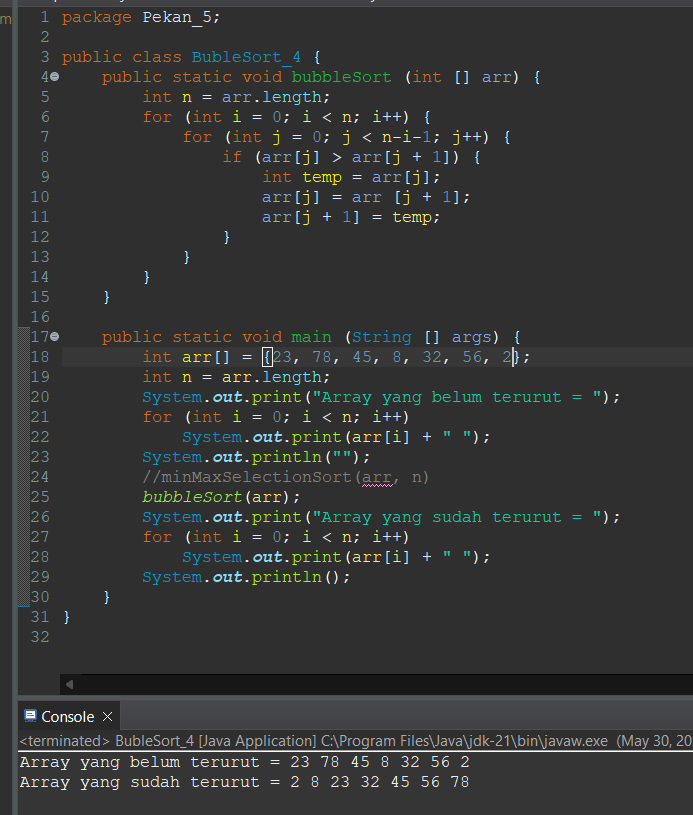
System.out.print("Array yang sudah terurut ");: Menampilkan teks "Array yang sudah terurut " ke console.

for (int i = 0; i < n; i++): Looping untuk menampilkan elemen array setelah diurutkan.

System.out.print(arr[i] + " ");: Menampilkan nilai setiap elemen array ke console.

System.out.println();: Menampilkan baris baru ke console.

Untuk hasil dari kode tersebut adalah sebagai berikut :



1. **KESIMPULAN**

Sorting algorithms berguna dalam proses pengurutan data yang dilakukan oleh seorang programmer untuk membuat data yang lebih rapi dan lebih teratur, Algoritma Sorting seperti Insertion Sort, Selection Sort, dan Bubble Sort adalah metode dasar untuk mengurutkan data. Memilih algoritma yang tepat tergantung pada kebutuhan spesifik dan karakteristik data yang akan diurutkan.