PRAKTIKUM STRUKTUR DATA
Tugas Jobsheet 9

Dosen Pengampu :
Randi Proska Sandra, M.Sc

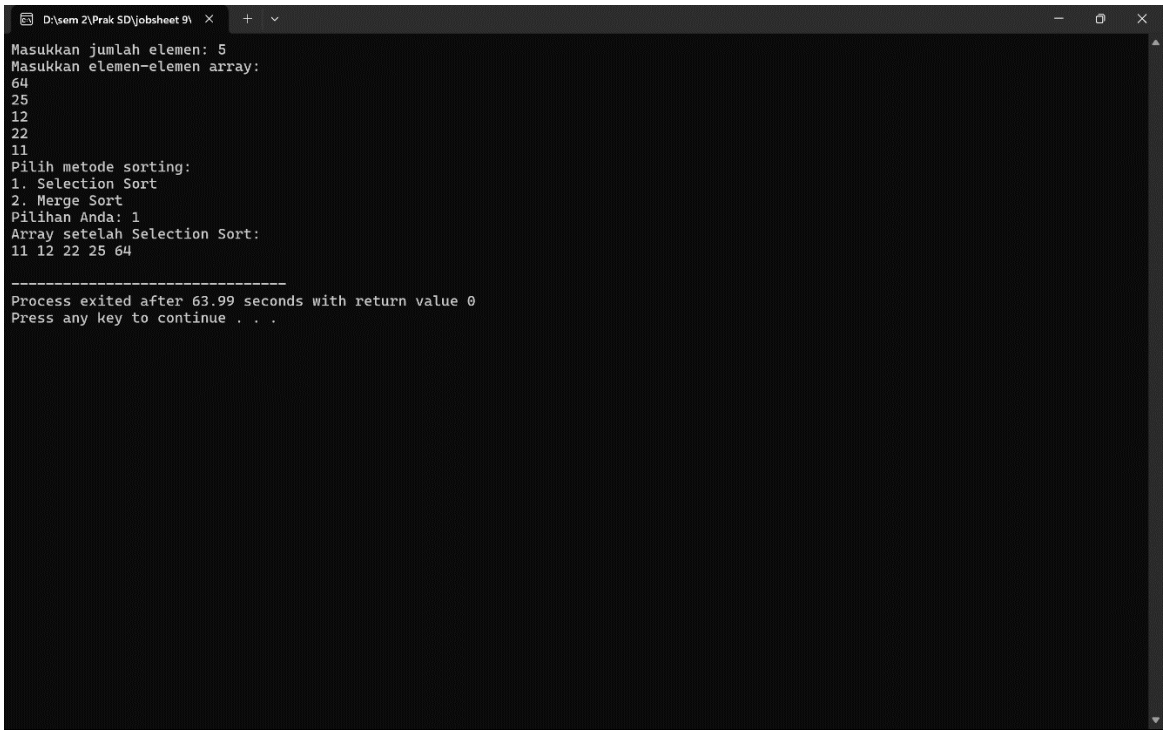
Disusun Oleh :

Joel Wiseda Simanungkalit
23343071

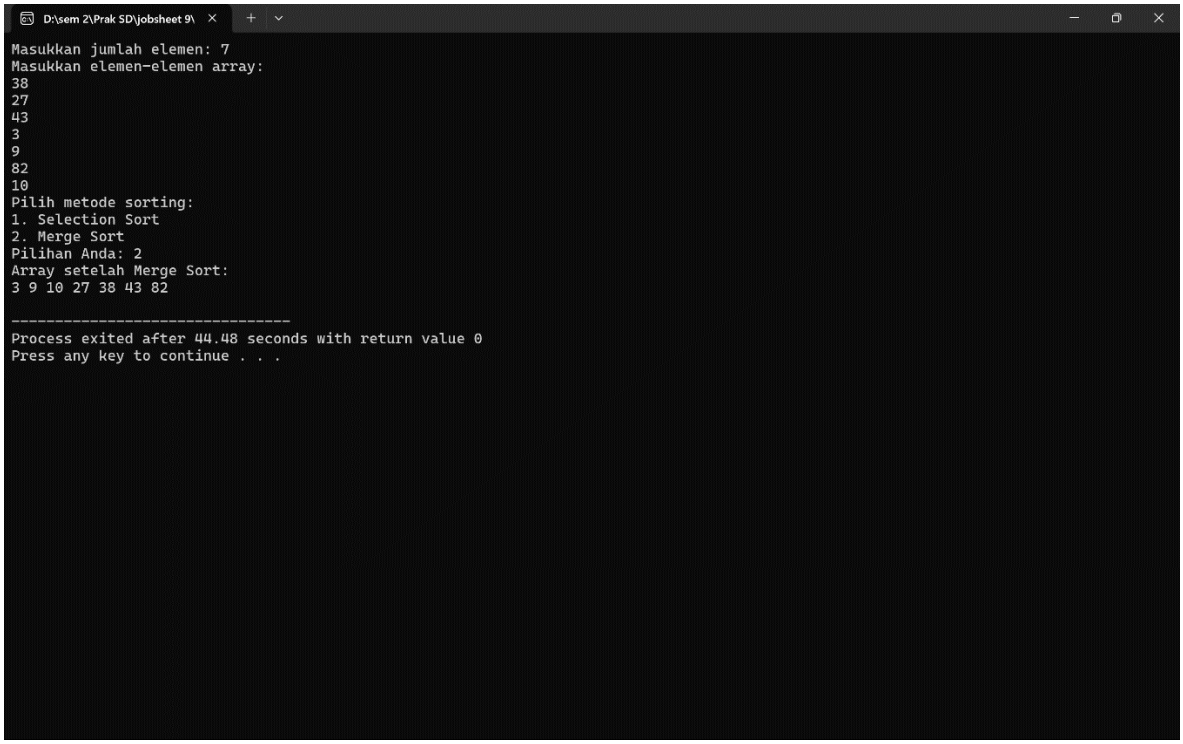
PROGRAM STUDI INFORMATIKA (NK)
FAKULTAS TEKNIK
UNIVERSITAS NEGERI PADANG
2024

A. Output

1. Selection Sort



2. Merge Sort



B. Penjelasan

Selection Sort

Selection Sort adalah algoritma sorting yang sederhana dan mudah dipahami. Prinsip kerja Selection Sort adalah sebagai berikut:

1. **Mencari Elemen Terkecil**: Algoritma ini berulang kali mencari elemen terkecil dalam bagian yang belum terurut dari array.
2. **Menempatkan Elemen Terkecil di Posisi yang Tepat**: Setelah menemukan elemen terkecil, algoritma menempatkannya di posisi awal dari bagian yang belum terurut, yang kemudian menjadi bagian dari bagian yang sudah terurut.
3. **Pengulangan hingga Terurut**: Proses ini diulangi untuk seluruh elemen dalam array sehingga seluruh elemen terurut.

Pada program di atas, fungsi `selectionSort` mengimplementasikan prinsip ini dengan menggunakan dua loop `for`. Loop pertama (`for (int i = 0; i < n-1; i++)`) mengontrol posisi saat ini di bagian yang terurut, dan loop kedua (`for (int j = i+1; j < n; j++)`) mencari elemen terkecil dalam bagian yang belum terurut. Variabel `min_idx` menyimpan indeks elemen terkecil yang ditemukan, dan elemen tersebut ditukar dengan elemen di posisi saat ini setelah loop kedua selesai.

Kode Selection Sort:

```
c
Copy code
void selectionSort(int arr[], int n) {
    int min_idx, temp;
    for (int i = 0; i < n-1; i++) {
        min_idx = i;
        for (int j = i+1; j < n; j++) {
            if (arr[j] < arr[min_idx]) {
                min_idx = j;
            }
        }
        temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}
```

Merge Sort

Merge Sort adalah algoritma sorting berbasis pembagian dan penaklukan yang lebih efisien untuk jumlah elemen yang besar. Prinsip kerja Merge Sort adalah sebagai berikut:

1. **Pembagian Array**: Algoritma ini membagi array menjadi dua bagian hingga setiap bagian hanya berisi satu elemen (yang secara default sudah terurut).
2. **Penggabungan Kembali**: Bagian-bagian yang sudah terurut tersebut kemudian digabungkan kembali secara berurutan untuk membentuk array yang terurut.

3. **Pengulangan hingga Terurut**: Proses pembagian dan penggabungan ini diulangi secara rekursif hingga seluruh elemen dalam array sudah terurut.

Pada program di atas, fungsi `mergeSort` mengimplementasikan prinsip ini dengan menggunakan pendekatan rekursif. Fungsi `mergeSort` membagi array menjadi dua bagian (`l` ke `m` dan `m+1` ke `r`) dan memanggil dirinya sendiri untuk setiap bagian tersebut. Fungsi `merge` kemudian digunakan untuk menggabungkan dua bagian yang sudah terurut menjadi satu bagian yang terurut.

Kode Merge Sort:

```
c
Copy code
void mergeSort(int arr[], int l, int r) {
    if (l < r) {
        int m = l + (r - l) / 2;

        mergeSort(arr, l, m);
        mergeSort(arr, m + 1, r);

        merge(arr, l, m, r);
    }
}
```

Kode Penggabungan:

```
c
Copy code
void merge(int arr[], int l, int m, int r) {
    int n1 = m - l + 1;
    int n2 = r - m;

    int L[n1], R[n2];

    for (int i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (int j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];

    int i = 0, j = 0, k = l;
    while (i < n1 && j < n2) {
        if (L[i] <= R[j]) {
            arr[k] = L[i];
            i++;
        } else {
            arr[k] = R[j];
            j++;
        }
        k++;
    }

    while (i < n1) {
        arr[k] = L[i];
        i++;
        k++;
```

```
    }

    while (j < n2) {
        arr[k] = R[j];
        j++;
        k++;
    }
}
```

Ilustrasi Proses Selection Sort

Misalkan kita memiliki array `[64, 25, 12, 22, 11]`.

Iterasi 1:

- Temukan elemen terkecil dalam array `[64, 25, 12, 22, 11]` → 11
- Tukar elemen pertama dengan elemen terkecil → `[11, 25, 12, 22, 64]`

Iterasi 2:

- Temukan elemen terkecil dalam subarray `[25, 12, 22, 64]` → 12
- Tukar elemen kedua dengan elemen terkecil → `[11, 12, 25, 22, 64]`

Iterasi 3:

- Temukan elemen terkecil dalam subarray `[25, 22, 64]` → 22
- Tukar elemen ketiga dengan elemen terkecil → `[11, 12, 22, 25, 64]`

Iterasi 4:

- Temukan elemen terkecil dalam subarray `[25, 64]` → 25
- Elemen sudah pada posisi yang benar.

Array sudah terurut: `[11, 12, 22, 25, 64]`.

Ilustrasi Proses Merge Sort

Misalkan kita memiliki array `[38, 27, 43, 3, 9, 82, 10]`.

Pembagian:

- Bagian menjadi dua: `[38, 27, 43, 3]` dan `[9, 82, 10]`
- Bagian lagi `[38, 27]` dan `[43, 3]`, `[9, 82]` dan `[10]`
- Bagian lagi `[38]` dan `[27]`, `[43]` dan `[3]`, `[9]` dan `[82]`

Penggabungan:

- Gabungkan `[38]` dan `[27]` menjadi `[27, 38]`
- Gabungkan `[43]` dan `[3]` menjadi `[3, 43]`

- Gabungkan `[27, 38]` dan `[3, 43]` menjadi `[3, 27, 38, 43]`
- Gabungkan `[9]` dan `[82]` menjadi `[9, 82]`
- Gabungkan `[9, 82]` dan `[10]` menjadi `[9, 10, 82]`
- Gabungkan `[3, 27, 38, 43]` dan `[9, 10, 82]` menjadi `[3, 9, 10, 27, 38, 43, 82]`

Array sudah terurut: `[3, 9, 10, 27, 38, 43, 82]`.

Kesimpulan

Selection Sort dan Merge Sort adalah dua algoritma sorting dengan pendekatan yang berbeda. Selection Sort adalah algoritma yang sederhana dan mudah dipahami, tetapi tidak efisien untuk jumlah elemen yang besar karena memiliki kompleksitas waktu O(n^2). Merge Sort, di sisi lain, lebih efisien untuk jumlah elemen yang besar dengan kompleksitas waktu O(n log n) dan menggunakan pendekatan pembagian dan penaklukan. Merge Sort juga merupakan algoritma stabil, yang berarti elemen-elemen yang sama akan mempertahankan urutan relatifnya setelah sorting.