

LAPORAN PRAKTIKUM STRUKTUR DATA

LAPORAN AWAL PRAKTIKUM

Pertemuan ke-07

Sorting



Disusun Oleh:

Nama Lengkap : Nova Ardiansyah

NIM : 211011401309

Kelas : 04-TPLE008

TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Raya Puspitek No. 11 Buaran, Serpong Telp. (021) 7412566, Fax. (021) 7412566
Tangerang Selatan - Banten

A. RANGKUMAN MATERI

Sorting adalah proses mengatur atau menyusun elemen-elemen dalam satu kumpulan data dengan aturan tertentu. Dalam program C++, terdapat beberapa algoritma sorting yang digunakan untuk mengurutkan array atau struktur data lainnya, seperti Bubble Sort, Selection Sort, dan Insertion Sort. Keuntungan menggunakan algoritma sorting yaitu menghemat waktu dan memudahkan dalam mencari nilai tertentu pada suatu array yang sudah terurut.

Algoritma Bubble Sort merupakan salah satu metode pengurutan sederhana dalam pemrograman C++. Pengurutan dilakukan dengan cara membandingkan 2 angka dalam sebuah array, jika angka pertama lebih besar dari angka kedua, maka mereka akan ditukar posisinya. Proses ini berlangsung terus-menerus hingga tidak ada nilai yang ditukar lagi. Algoritma Bubble Sort sangat mudah dipahami dan diimplementasikan oleh pemula, tetapi memiliki kompleksitas waktu yang tinggi saat jumlah data semakin banyak.

Selain itu, algoritma Selection Sort juga sering digunakan dalam pemrograman C++ untuk mengurutkan data. Cara kerja Selection Sort yaitu mencari angka terkecil dalam array dan menukar posisi angka tersebut dengan angka pertama dalam array. Kemudian proses diulang untuk subarray yang belum terurut sampai seluruh array tersusun dengan benar. Meskipun lebih cepat daripada Bubble Sort, Selection Sort masih sulit diimplementasikan untuk array yang sangat besar. Oleh karena itu, untuk data yang bersifat dinamis dan kompleks, diperlukan algoritma sorting yang lebih efisien seperti Merge Sort atau Quick Sort.

B. TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Sorting!

Jawab :

Sorting atau pengurutan merupakan proses mengatur atau menyusun elemen-elemen dalam suatu kumpulan data dengan aturan tertentu. Dalam konteks pemrograman, sorting umumnya digunakan untuk mengurutkan elemen-elemen dalam array atau struktur data lainnya sehingga memudahkan dalam mencari nilai tertentu atau membuat operasi lain pada data yang sudah terurut. Ada beberapa algoritma

sorting yang dapat digunakan, seperti Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, dan Quick Sort, masing-masing dengan kelebihan dan kekurangan yang berbeda-beda. Pemilihan algoritma sorting harus disesuaikan dengan kebutuhan dan karakteristik dari data yang akan diurutkan agar dapat menghasilkan waktu eksekusi yang optimal.

2. Jelaskan Perbedaan dari pengurutan Internal dan Pengurutan Eksternal!

Jawab :

Pengurutan internal adalah pengurutan elemen-elemen dalam struktur data di dalam memori komputer secara keseluruhan. Sedangkan pengurutan eksternal adalah pengurutan elemen-elemen dalam struktur data yang lebih besar dari kapasitas memori komputer, dengan melibatkan operasi baca-tulis pada media penyimpanan eksternal seperti hard disk atau CD-ROM.

Perbedaan mendasar antara keduanya adalah ketergantungan terhadap media penyimpanan data, dimana pengurutan internal hanya menggunakan memori komputer dan lebih cepat, sedangkan pengurutan eksternal melibatkan media penyimpanan eksternal dan lebih lambat tetapi cocok untuk dataset yang sangat besar.

3. Jelaskan perbedaan metode-metode Sorting seperti: Bubble Sort, Quick Sort, Selection Sort, Merge Sort, Tree Sort, Maximum Sort, Insertion Sort!

Jawab :

- **Bubble Sort:** Metode sorting sederhana di mana angka dibandingkan secara berpasangan dan kemudian ditukar jika perlu. Element-elemen besar akan bergeser ke akhir array, sehingga digunakan untuk pengurutan data terbatas.

- **Quick Sort:** Algoritma divide and conquer di mana elemen terakhir digunakan sebagai pivot point untuk membagi array menjadi dua bagian, satu dengan nilai lebih rendah dari pivot dan satunya lagi dengan nilainya lebih tinggi. Kemudian kedua bagian juga dicacah lagi menjadi bagian yang lebih kecil dan diurutkan secara rekursif.

- **Selection Sort:** Algoritma sorting sederhana yang bekerja dengan memilih elemen terkecil dari array dan menukar posisinya dengan elemen pertama. Selanjutnya, elemen kedua terkecil ditemukan di sisa array, dan ditukar dengan elemen kedua, dan seterusnya hingga seluruh array terurut.

- **Merge Sort:** Algoritma sorting dengan strategi divide and conquer di mana array dibagi menjadi dua bagian dan diurutkan secara independen, kemudian dua bagian tersebut digabungkan kembali menjadi satu array yang terurut.

- **Tree Sort:** Algoritma sorting mirip dengan binary search tree yaitu menyimpan setiap elemen sebagai simpul pada pohon biner, kemudian mengurutkan elemen-elemen tersebut dengan menjelajahi pohon secara in-order traversal.

- **Maximum Sort:** Metode sorting dalam pengembangan game di mana elemen-elemen diurutkan dengan cara memperbesar elemen-elemen terakhir sambil memindahkan elemen-elemen ke atas.

- **Insertion Sort:** Algoritma sorting sederhana yang mengambil element dari array dan memasukkannya pada tempat yang sudah terurut sebelumnya. Array dibagi menjadi dua bagian, satu bagian yang sudah diurutkan dan satunya yang belum diurutkan, lalu elemen-elemen baru ditempatkan pada posisi yang tepat di dalam array yang telah diurutkan secara increment.

Perbedaan utama antara metode-metode tersebut adalah pada strategi pengurutan yang digunakan, efisiensi waktu dan ruang, serta kompleksitas algoritmanya. Pemilihan metode sorting harus disesuaikan dengan karakteristik data yang akan diurutkan agar dapat menghasilkan waktu eksekusi yang optimal.

4. Buatlah contoh program sederhana menggunakan Sorting!

Jawab :

```
#include <iostream>
using namespace std;

void bubbleSort(int arr[], int n)
```

```

{
    for(int i = 0; i < n-1; i++)
    {
        for(int j = 0; j < n-i-1; j++)
        {
            if(arr[j] > arr[j+1])
            {
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}

int main()
{
    cout << "Nama \t: Nova Ardiansyah\n";
    cout << "NIM \t: 211011401309\n";
    cout << "=====\n\n";

    int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << "Nilai sebelum diurutkan: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }

    bubbleSort(arr, n);
    cout << "\nNilai setelah diurutkan: ";
    for(int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }

    return 0;
}

```

```

Nama      : Nova Ardiansyah
NIM       : 211011401309
=====

Nilai sebelum diurutkan: 64 25 12 22 11
Nilai setelah diurutkan: 11 12 22 25 64

```

C. TUGAS PRAKTIKUM

- Lat7_1

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    cout << "Nama \t: Nova Ardiansyah\n";
    cout << "NIM \t: 211011401309\n";
    cout << "=====\n\n";

    int Nilai[20];
    int i, j, k, N;
    int temp;
    bool tukar;

    cout << "Masukkan banyak data : ";
    cin >> N;

    for (i = 0; i < N; i++) {
        cout << "Masukkan data ke-" << i + 1 << " : ";
        cin >> Nilai[i];
    }

    cout << endl;
    cout << "Data sebelum diurutkan : ";
    for (i = 0; i < N; i++)
    {
        cout << Nilai[i] << " ";
    }

    i = 0;
    for (j = N - 1; j >= 0; j--)
    {
        tukar = false;
        for (k = 0; k < j; k++) {
            if (Nilai[k] > Nilai[k + 1]) {
                temp = Nilai[k];
                Nilai[k] = Nilai[k + 1];
                Nilai[k + 1] = temp;
                tukar = true;
            }
        }
    }

    if (tukar == false) {
        break;
    }

    cout << endl;
    i++;
}

cout << "Data setelah diurutkan : ";
for (i = 0; i < N; i++)
{
    cout << Nilai[i] << " ";
}
```

```
return 0;
}
```

```
Nama      : Nova Ardiansyah
NIM       : 211011401309
=====
```

```
Masukkan banyak data : 4
Masukkan data ke-1 : 21
Masukkan data ke-2 : 12
Masukkan data ke-3 : 10
Masukkan data ke-4 : 32
```

```
Data sebelum diurutkan : 21 12 10 32
```

```
Data setelah diurutkan : 10 12 21 32
```

- Lat7_2

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    cout << "Nama \t: Nova Ardiansyah\n";
    cout << "NIM \t: 211011401309\n";
    cout << "=====\n\n";

    int Nilai[20];
    int i, j, k, N;
    int temp;
    bool tukar;

    cout << "Masukkan banyak data : ";
    cin >> N;

    for (i = 0; i < N; i++) {
        cout << "Masukkan data ke-" << i + 1 << " : ";
        cin >> Nilai[i];
    }

    cout << endl;
    cout << "Data sebelum diurutkan : ";
    for (i = 0; i < N; i++)
    {
        cout << Nilai[i] << " ";
    }

    i = 0;
    for (j = N - 1; j >= 0; j--)
    {
        tukar = false;
        for (k = 0; k < j; k++) {
            if (Nilai[k] < Nilai[k + 1]) {
                temp = Nilai[k];
                Nilai[k] = Nilai[k + 1];
                Nilai[k + 1] = temp;
            }
        }
    }
}
```

```

        Nilai[k + 1] = temp;
        tukar = true;
    }
}

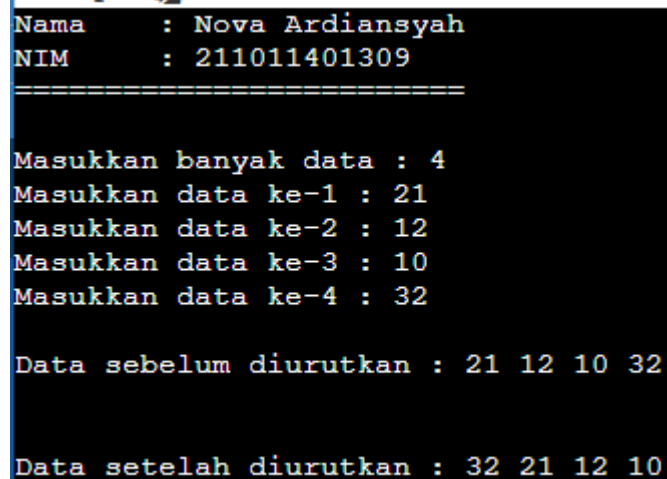
if (tukar == false) {
    break;
}

cout << endl;
i++;
}

cout << "Data setelah diurutkan : ";
for (i = 0; i < N; i++) {
    cout << Nilai[i] << " ";
}

return 0;
}

```



```

Nama      : Nova Ardiansyah
NIM       : 211011401309
=====

Masukkan banyak data : 4
Masukkan data ke-1 : 21
Masukkan data ke-2 : 12
Masukkan data ke-3 : 10
Masukkan data ke-4 : 32

Data sebelum diurutkan : 21 12 10 32

Data setelah diurutkan : 32 21 12 10

```

- Lat7_3

```

#include <iostream>
#include <string>
using namespace std;

void Cetak(int data[], int n)
{
    int i = 0;
    for (i = 0; i < n; i++)
    {
        cout << data[i] << " ";
    }
}

int partisi(int data[], int p, int r)
{
    int x, i, j, temp;
    x = data[p];
    i = p;
    j = r;

```



```

while (true)
{
    while (data[j] > x) {
        j--;
    }

    while (data[i] < x) {
        i++;
    }

    if (i < j) {
        temp = data[i];
        data[i] = data[j];
        data[j] = temp;
    } else {
        return j;
    }
}

void QuickSort(int data[], int p, int r)
{
    int q;
    if (p < r) {
        q = partisi(data, p, r);
        QuickSort(data, p, q);
        QuickSort(data, q + 1, r);
    }
}

int main()
{
    cout << "Nama \t: Nova Ardiansyah\n";
    cout << "NIM \t: 211011401309\n";
    cout << "=====\n\n";

    int Nilai[20];
    int i, N;

    cout << "Masukkan banyak data : ";
    cin >> N;

    for (i = 0; i < N; i++) {
        cout << "Masukkan data ke-" << i + 1 << " : ";
        cin >> Nilai[i];
    }

    cout << endl;
    cout << "Data sebelum diurutkan : ";
    for (i = 0; i < N; i++) {
        cout << Nilai[i] << " ";
    }

    QuickSort(Nilai, 0, N - 1);

    cout << "\nData setelah diurutkan : ";
    Cetak(Nilai, N);

    return 0;
}

```

```
}  
Nama      : Nova Ardiansyah  
NIM       : 211011401309  
=====
```

```
Masukkan banyak data : 4  
Masukkan data ke-1 : 21  
Masukkan data ke-2 : 12  
Masukkan data ke-3 : 10  
Masukkan data ke-4 : 32
```

```
Data sebelum diurutkan : 21 12 10 32  
Data setelah diurutkan : 10 12 21 32
```

- Lat7_4

```
#include <iostream>  
#include <string>  
using namespace std;  
  
void Cetak(int data[], int n)  
{  
    int i = 0;  
    for (i = 0; i < n; i++)  
    {  
        cout << data[i] << " ";  
    }  
}  
  
int partisi(int data[], int p, int r)  
{  
    int x, i, j, temp;  
    x = data[p];  
    i = p;  
    j = r;  
  
    while (true)  
    {  
        while (data[j] > x) {  
            j--;  
        }  
  
        while (data[i] < x) {  
            i++;  
        }  
  
        if (i < j) {  
            temp = data[i];  
            data[i] = data[j];  
            data[j] = temp;  
        } else {  
            return j;  
        }  
    }  
}  
  
void QuickSort(int data[], int p, int r)
```

```

{
    int q;
    if (p < r) {
        q = partisi(data, p, r);
        QuickSort(data, p, q);
        QuickSort(data, q + 1, r);
    }
}

void pilihan1()
{
    int Nilai[20];
    int i, j, k, N;
    int temp;
    bool tukar;

    cout << "Masukkan banyak data : ";
    cin >> N;

    for (i = 0; i < N; i++) {
        cout << "Masukkan data ke-" << i + 1 << " : ";
        cin >> Nilai[i];
    }

    cout << endl;
    cout << "Data sebelum diurutkan : ";
    for (i = 0; i < N; i++)
    {
        cout << Nilai[i] << " ";
    }

    i = 0;
    for (j = N - 1; j >= 0; j--)
    {
        tukar = false;
        for (k = 0; k < j; k++) {
            if (Nilai[k] > Nilai[k + 1]) {
                temp = Nilai[k];
                Nilai[k] = Nilai[k + 1];
                Nilai[k + 1] = temp;
                tukar = true;
            }
        }
    }

    if (tukar == false) {
        break;
    }

    cout << endl;
    i++;
}

cout << "Data setelah diurutkan : ";
for (i = 0; i < N; i++)
{
    cout << Nilai[i] << " ";
}
}

void pilihan2()

```

```

{
    int Nilai[20];
    int i, j, k, N;
    int temp;
    bool tukar;

    cout << "Masukkan banyak data : ";
    cin >> N;

    for (i = 0; i < N; i++) {
        cout << "Masukkan data ke-" << i + 1 << " : ";
        cin >> Nilai[i];
    }

    cout << endl;
    cout << "Data sebelum diurutkan : ";
    for (i = 0; i < N; i++)
    {
        cout << Nilai[i] << " ";
    }

    i = 0;
    for (j = N - 1; j >= 0; j--)
    {
        tukar = false;
        for (k = 0; k < j; k++) {
            if (Nilai[k] < Nilai[k + 1]) {
                temp = Nilai[k];
                Nilai[k] = Nilai[k + 1];
                Nilai[k + 1] = temp;
                tukar = true;
            }
        }
    }

    if (tukar == false) {
        break;
    }

    i++;
}

cout << "\nData setelah diurutkan : ";
for (i = 0; i < N; i++) {
    cout << Nilai[i] << " ";
}
}

void pilihan3()
{
    int Nilai[20];
    int i, N;

    cout << "Masukkan banyak data : ";
    cin >> N;

    for (i = 0; i < N; i++) {
        cout << "Masukkan data ke-" << i + 1 << " : ";
        cin >> Nilai[i];
    }
}

```

```

    cout << endl;
    cout << "Data sebelum diurutkan : ";
    for (i = 0; i < N; i++) {
        cout << Nilai[i] << " ";
    }

    QuickSort(Nilai, 0, N - 1);

    cout << "\nData setelah diurutkan : ";
    Cetak(Nilai, N);
}

int main()
{
    cout << "Nama \t: Nova Ardiansyah\n";
    cout << "NIM \t: 211011401309\n";
    cout << "=====\n\n";

    int pilihan;

    cout << "Pilih metode pengurutan data: \n";
    cout << "1. Bubble Sort Menaik\n";
    cout << "2. Bubble Sort Menurun\n";
    cout << "3. Quick Sort Menaik\n";

    cout << "Pilihan: ";
    cin >> pilihan;

    switch (pilihan)
    {
    case 1:
        cout << "\n===== 1. Bubble Sort Menaik =====\n\n";
        pilihan1();
        break;
    case 2:
        cout << "\n===== 2. Bubble Sort Menurun =====\n\n";
        pilihan2();
        break;
    case 3:
        cout << "\n===== 3. Quick Sort Menaik =====\n\n";
        pilihan3();
        break;
    default:
        cout << "\n===== Pilihan tidak tersedia =====\n\n";
        break;
    }

    return 0;
}

```

```
Nama      : Nova Ardiansyah
NIM       : 211011401309
=====

Pilih metode pengurutan data:
1. Bubble Sort Menaik
2. Bubble Sort Menurun
3. Quick Sort Menaik
Pilihan: 2

===== 2. Bubble Sort Menurun =====

Masukkan banyak data : 4
Masukkan data ke-1 : 21
Masukkan data ke-2 : 12
Masukkan data ke-3 : 10
Masukkan data ke-4 : 32

Data sebelum diurutkan : 21 12 10 32
Data setelah diurutkan : 32 21 12 10
```

LAPORAN PRAKTIKUM STRUKTUR DATA

LAPORAN AKHIR PRAKTIKUM

Pertemuan ke-07

Sorting



Disusun Oleh:

Nama Lengkap : Nova Ardiansyah
NIM : 211011401309
Kelas : 04-TPLE008

TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Raya Puspitek No. 11 Buaran, Serpong Telp. (021) 7412566, Fax. (021) 7412566
Tangerang Selatan - Banten

A. TUGAS AKHIR

1. Buatlah program untuk mengurutkan sederetan data: 34, 12, 56, 78, 6, 43, 32, 29, 90, 50, 55, 75, 85, 95, 25!

Jawab :

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Nama \t: Nova Ardiansyah\n";
    cout << "NIM \t: 211011401309\n";
    cout << "=====\n\n";

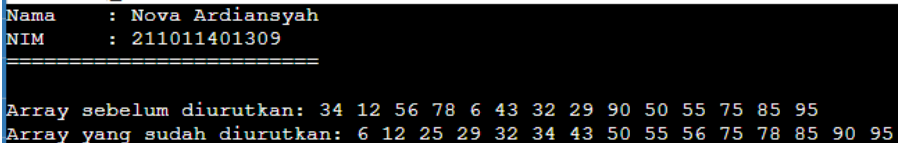
    int arr[] = {34, 12, 56, 78, 6, 43, 32, 29, 90, 50, 55, 75, 85, 95, 25};
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << "Array sebelum diurutkan: ";
    for (int i = 0; i < n - 1; i++) {
        cout << arr[i] << " ";
    }

    // * Bubble Sort
    for (int i = 0; i < n - 1; i++)
    {
        for (int j = 0; j < n - i - 1; j++)
        {
            if (arr[j] > arr[j + 1])
            {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }

    cout << "\nArray yang sudah diurutkan: ";
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    cout << endl;

    return 0;
}
```



The screenshot shows the output of the C++ program. It displays the user's name and NIM, followed by a separator line. Then, it shows the array before sorting and the array after sorting using bubble sort. The arrays are: [34, 12, 56, 78, 6, 43, 32, 29, 90, 50, 55, 75, 85, 95, 25] and [6, 12, 25, 29, 32, 34, 43, 50, 55, 56, 75, 78, 85, 90, 95].

```
Nama      : Nova Ardiansyah
NIM       : 211011401309
=====

Array sebelum diurutkan: 34 12 56 78 6 43 32 29 90 50 55 75 85 95
Array yang sudah diurutkan: 6 12 25 29 32 34 43 50 55 56 75 78 85 90 95
```


2. KESIMPULAN

Dalam pemrograman C++, sorting digunakan untuk mengurutkan elemen-elemen dalam array atau struktur data lainnya. Terdapat beberapa algoritma sorting yang dapat digunakan seperti Bubble Sort, Selection Sort, Merge Sort, dan Quick Sort. Meskipun algoritma sorting yang sederhana seperti Bubble Sort dan Selection Sort mudah dipahami dan diimplementasikan oleh pemula, tetapi memiliki kompleksitas waktu yang tinggi saat jumlah data semakin banyak. Sedangkan algoritma sorting yang lebih efisien seperti Merge Sort atau Quick Sort dapat digunakan untuk data yang bersifat dinamis dan kompleks. Oleh karena itu, penggunaan algoritma sorting harus disesuaikan dengan kebutuhan dan karakteristik dari data yang akan diurutkan.