LAPORAN PRAKTIKUM
STRUKTUR DATA DAN ALGORITMA

MODUL 8
ALGORITMA SEARCHING

Dosen : Wahyu Andi Saputra, S.Pd., M.Eng.

Disusun oleh:

Falah Asbi Pangestu

(2311102045)

IF-11-B

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

BAB I

DASAR TEORI

1. Dasar Teori

Pencarian (Searching) yaitu proses menemukan suatu nilai tertentu pada kumpulan data. Hasil pencarian adalah salah satu dari tiga keadaan ini: data ditemukan, data ditemukan lebih dari satu, atau data tidak ditemukan. Searching juga dapat dianggap sebagai proses pencarian suatu data di dalam sebuah array dengan cara mengecek satu persatu pada setiap index baris atau setiap index kolomnya dengan menggunakan teknik perulangan untuk melakukan pencarian data. Terdapat 2 metode pada algoritma Searching, yaitu:

a. Sequential Search

Sequential Search merupakan salah satu algoritma pencarian data yang biasa digunakan untuk data yang berpola acak atau belum terurut. Sequential search juga merupakan teknik pencarian data dari array yang paling mudah, dimana data dalam array dibaca satu demi satu dan diurutkan dari index terkecil ke index terbesar, maupun sebaliknya. Konsep Sequential Search yaitu:

- Membandingkan setiap elemen pada array satu per satu secara berurut.
- Proses pencarian dimulai dari indeks pertama hingga indeks terakhir.
- Proses pencarian akan berhenti apabila data ditemukan. Jika hingga akhir array data masih juga tidak ditemukan, maka proses pencarian tetap akan dihentikan.
- Proses perulangan pada pencarian akan terjadi sebanyak jumlah N elemen pada array.

Algoritma pencarian berurutan dapat dituliskan sebagai berikut :

1) i ← 0

2) ketemu ← false

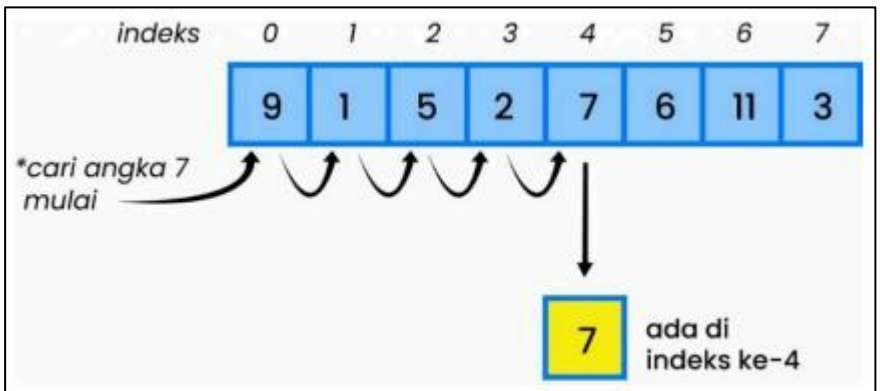
3) Selama (tidak ketemu) dan (i <= N) kerjakan baris 4

4) Jika (Data[i] = x) maka ketemu ← true, jika tidak i ← i + 1

5) Jika (ketemu) maka i adalah indeks dari data yang dicari, jika tidak data tidak ditemukan.

Fungsi diatas akan mengembalikan indeks dari data yang dicari. Apabila data tidak ditemukan maka fungsi diatas akan mengembalikan nilai –1. Contoh dari Sequential Search, yaitu:

Int A[8] = {9,1,5,2,7,6,11,3}



Gambar 1. Ilustrasi Sequential Search

Misalkan, dari data di atas angka yang akan dicari adalah angka 7 dalam array A, maka proses yang akan terjadi yaitu:

- Pencarian dimulai pada index ke-0 yaitu angka 9, kemudian dicocokkan dengan angka yang akan dicari, jika tidak sama maka pencarian akan dilanjutkan ke index selanjutnya.
- Pada index ke-1, yaitu angka 1, juga bukan angka yang dicari, makapencarian akan dilanjutkan pada index selanjutnya.
- Pada index ke-2 dan index ke-3 yaitu angka 5 dan 2, juga bukan angka yang dicari, sehingga pencarian dilanjutkan pada index selanjutnya.
- Pada index ke-4 yaitu angka 7 dan ternyata angka 7 merupakan angka yang dicari, sehingga pencarian akan dihentikan dan proses selesai.

b. Binary Search

Binary Search termasuk ke dalam interval search, dimana algoritma ini merupakan algoritma pencarian pada array/list dengan elemen terurut.

Pada metode ini, data harus diurutkan terlebih dahulu dengan cara data dibagi menjadi dua bagian (secara logika), untuk setiap tahap pencarian. Dalam penerapannya algoritma ini sering digabungkan dengan algoritma sorting karena data yang akan digunakan harus sudah terurut terlebih dahulu. Konsep Binary Search:

- Data diambil dari posisi 1 sampai posisi akhir N.
- Kemudian data akan dibagi menjadi dua untuk mendapatkan posisi data tengah.
- Selanjutnya data yang dicari akan dibandingkan dengan data yang berada di posisi tengah, apakah lebih besar atau lebih kecil.
- Apabila data yang dicari lebih besar dari data tengah, maka dapat dipastikan bahwa data yang dicari kemungkinan berada di sebelah kanan dari data tengah. Proses pencarian selanjutnya akan dilakukan pembagian data menjadi dua bagian pada bagian kanan dengan acuan posisi data tengah akan menjadi posisi awal untuk pembagian tersebut.
- Apabila data yang dicari lebih kecil dari data tengah, maka dapat dipastikan bahwa data yang dicari kemungkinan berada di sebelah kiri dari data tengah. Proses pencarian selanjutnya akan dilakukan pembagian data menjadi dua bagian pada bagian kiri. Dengan acuan posisi data tengah akan menjadi posisi akhir untuk pembagian selanjutnya.
- Apabila data belum ditemukan, maka pencarian akan dilanjutkan dengan kembali membagi data menjadi dua.
- Namun apabila data bernilai sama, maka data yang dicari langsung ditemukan dan pencarian dihentikan.

BAB II

GUIDED

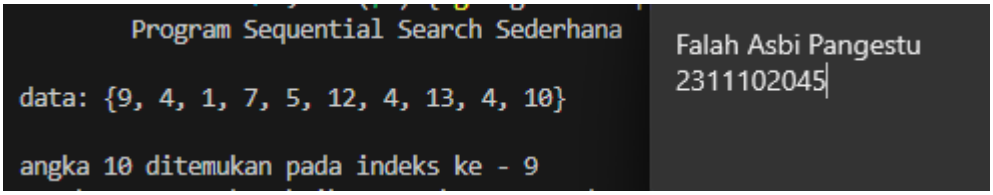
LATIHAN – GUIDED

1. Guided 1
   Buatlah sebuah project dengan menggunakan sequential search sederhana untuk melakukan pencarian data.

   Source Code

```
#include <iostream>
using namespace std;
int main()
{
    int n = 10;
    int data[n] = {9, 4, 1, 7, 5, 12, 4, 13, 4, 10};
    int cari = 10;
    bool ketemu = false;
    int i;
    // algoritma Sequential Search
    for (i = 0; i < n; i++)
    {
        if (data[i] == cari)
        {
            ketemu = true;
            break;
        }
    }
    cout << "\t Program Sequential Search Sederhana\n " << endl;
        cout
         << " data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}" << endl;
    if (ketemu)
    {
        cout << "\n angka " << cari << " ditemukan pada indeks ke - "
<< i << endl;
    }
    else
    {
        cout << cari << " tidak dapat ditemukan pada data."
             << endl;
    }
    return 0;
}
```

   Screenshoot program



Deskripsi program

Program ini memakai perulangan for untuk memeriksa setiap elemen dalam array data secara berurutan. Dalam setiap iterasi, program membandingkan nilai elemen yang sedang diperiksa dengan nilai yang dicari. Jika nilainya cocok, variabel ketemu diubah menjadi true, dan perulangan dihentikan menggunakan perintah break. Jika perulangan selesai tanpa menemukan nilai yang dicari, variabel ketemu tetap bernilai false.

2. Guided 2

Buatlah sebuah project untuk melakukan pencarian data dengan menggunakan Binary Search.

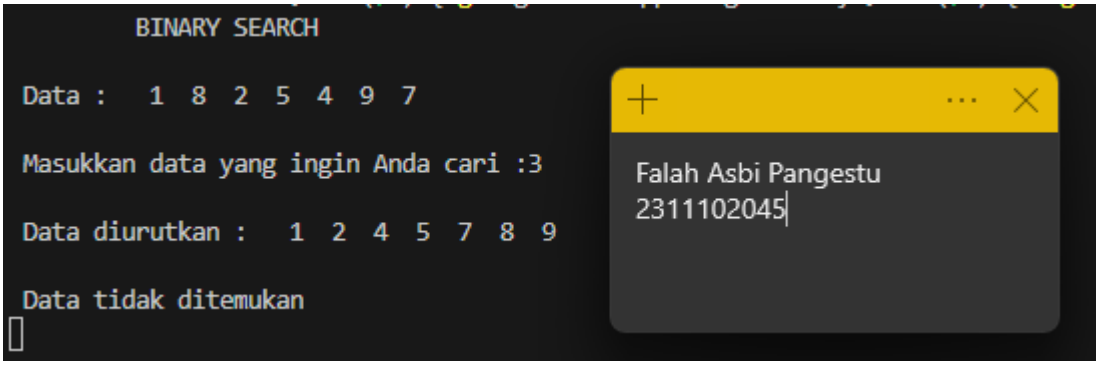
Source Code

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

#include <conio.h>
#include <iomanip>
int data[7] = {1, 8, 2, 5, 4, 9, 7};
int cari;
void selection_sort()
{
    int temp, min, i, j;
    for (i = 0; i < 7; i++)
    {
        min = i;
        for (j = i + 1; j < 7; j++)
        {
            if (data[j] < data[min])
            {
                min = j;
            }
        }
        temp = data[i];
        data[i] = data[min];
        data[min] = temp;
    }
}
void binarysearch()
{
    // searching
    int awal, akhir, tengah, b_flag = 0;
    awal = 0;
```

```
    akhir = 7;
    while (b_flag == 0 && awal <= akhir)
    {
        tengah = (awal + akhir) / 2;
        if (data[tengah] == cari)
        {
            b_flag = 1;
            break;
        }
        else if (data[tengah] < cari)
            awal = tengah + 1;
        else
            akhir = tengah - 1;
    }
    if (b_flag == 1)
        cout << "\n Data ditemukan pada index ke- "<<tengah<<endl;
            else cout
             << "\n Data tidak ditemukan\n";
}
int main()
{
    cout << "\t BINARY SEARCH " << endl;
    cout << "\n Data : ";
    // tampilkan data awal
    for (int x = 0; x < 7; x++)
        cout << setw(3) << data[x];
    cout << endl;
    cout << "\n Masukkan data yang ingin Anda cari :";
    cin >> cari;
    cout << "\n Data diurutkan : "; // urutkan data dengan selection
sort
    selection_sort(); // tampilkan data setelah diurutkan
    for (int x = 0; x < 7; x++)
        cout << setw(3) << data[x];
    cout << endl;
    binarysearch();
    _getche();
    return EXIT_SUCCESS;
}
```

Screenshoot program



Deskripsi program

Algoritma utama yang diterapkan dalam program ini adalah Selection Sort untuk mengurutkan array dan Binary Search untuk melakukan pencarian pada array yang sudah terurut. Algoritma Binary Search berfungsi dengan membagi array menjadi dua bagian di setiap iterasi, lalu mengabaikan setengah bagian yang tidak mungkin mengandung elemen yang dicari. Proses ini terus diulang hingga elemen ditemukan atau seluruh array sudah diperiksa.

BAB III

UNGUIDED

TUGAS – UNGUIDED

1. Unguided 1
   Buatlah sebuah program untuk mencari sebuah huruf pada sebuah kalimat yang sudah di input dengan menggunakan Binary Search!

   Source Code

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
// Fungsi untuk melakukan Binary Search
int binarySearch(const std::vector<char>& arr, char target) {
    int low = 0;
    int high = arr.size() - 1;

    while (low <= high) {
        int mid = low + (high - low) / 2;

        if (arr[mid] == target) {
            return mid;
        } else if (arr[mid] < target) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }

    return -1;
}

int main() {
    string kalimat;
    char hurufYangDicari;

    // Ambil input kalimat dan huruf yang dicari dari pengguna
    cout << "Masukkan kalimat: ";
    getline(std::cin, kalimat);

    cout << "Masukkan huruf yang ingin dicari: ";
    cin >> hurufYangDicari;

    //Ubah kalimat menjadi daftar huruf
    vector<char> daftarHuruf(kalimat.begin(), kalimat.end());
```

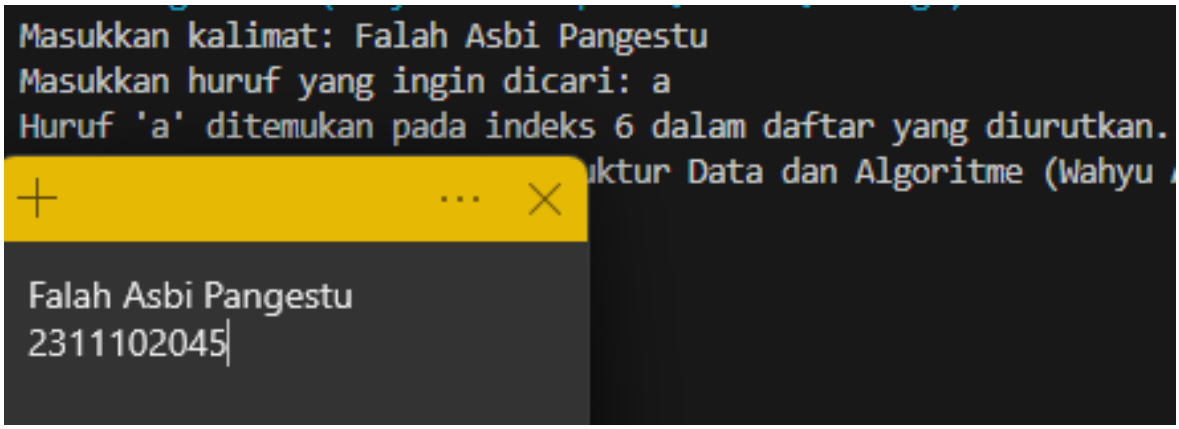
```
    //Urutkan daftar huruf
    sort(daftarHuruf.begin(), daftarHuruf.end());

    // Cari huruf menggunakan Binary Search
    int index = binarySearch(daftarHuruf, hurufYangDicari);

    // Tampilkan hasil
    if (index != -1) {
        cout << "Huruf '" << hurufYangDicari << "' ditemukan pada
indeks " << index << " dalam daftar yang diurutkan." << std::endl;
    } else {
        cout << "Huruf '" << hurufYangDicari << "' tidak ditemukan
dalam kalimat." << std::endl;
    }

    return 0;
}
```

Screenshot Program



Deskripsi program

Program menampilkan hasil pencarian kepada pengguna. Jika huruf ditemukan, program akan menampilkan indeks huruf tersebut dalam daftar terurut. Namun, jika huruf tidak ditemukan, program akan menampilkan pesan bahwa huruf tidak ada dalam kalimat yang dimasukkan sebelumnya.

2. Guided 2

Buatlah sebuah program yang dapat menghitung banyaknya huruf vocal dalam sebuah kalimat!

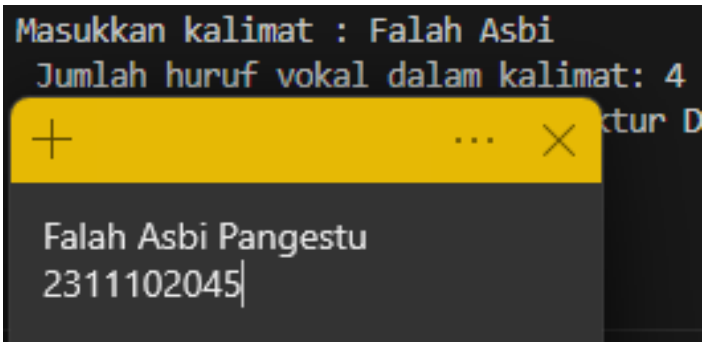
Source code

```
#include <iostream>
#include <string>
#include <algorithm>
```

```cpp
using namespace std;

int countVowels(string sentence)
{
    int count = 0;
    string vowels = "aeiou";
// Mengubah kalimat menjadi lowercase agar pencarian case insensitve
transform(sentence.begin(), sentence.end(),
sentence.begin(), ::tolower);
for (char c : sentence)
    {
        if (vowels.find(c) != string::npos)
        {
            count++;
        }
    }
    return count;
}
int main()
{
    string sentence;
    cout << "Masukkan kalimat : ";
    getline(cin, sentence);
    int vowelCount = countVowels(sentence);
    cout << " Jumlah huruf vokal dalam kalimat: " << vowelCount
         << endl;
    return 0;
}
```

Screenshoot program



Deskripsi program

Fungsi melakukan iterasi melalui setiap karakter dalam kalimat menggunakan perulangan for. Pada setiap iterasi, fungsi memeriksa apakah karakter tersebut merupakan huruf vokal atau bukan dengan menggunakan fungsi find dari kelas string.

Jika karakter tersebut merupakan huruf vokal, maka variabel count akan diinkrementasi.

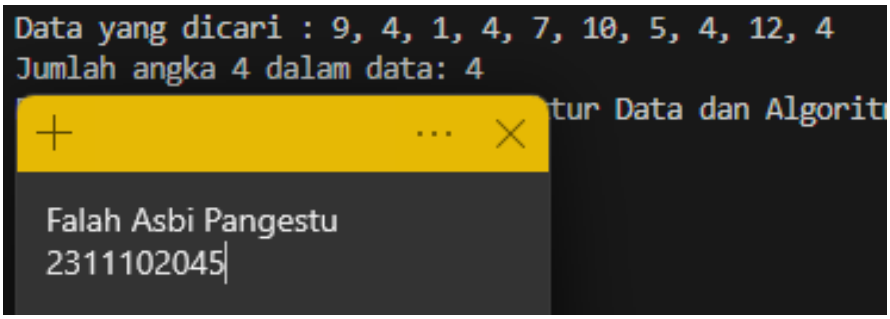
3. Guided 3
Diketahui data = 9, 4, 1, 4, 7, 10, 5, 4, 12, 4. Hitunglah berapa banyak angka 4 dengan menggunakan algoritma Sequential Search!

Source code

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

int main()
{
    int data[] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int size = sizeof(data) / sizeof(data[0]); //Mendapatkan ukuran
array
    int target = 4;
    int count = 0; // Variabel untuk menghitung jumlahkemunculan
angka 4
    for (int i = 0; i < size; i++)
    {
        if (data[i] == target)
        {
            count++;
        }
    }
    cout << "Data yang dicari : 9, 4, 1, 4, 7, 10, 5, 4, 12, 4"<<
endl;
    cout << "Jumlah angka 4 dalam data: " << count << endl;
    return 0;
}
```

Screenshoot program



Deskripsi program
program mendefinisikan sebuah array bernama data yang berisi sepuluh elemen berupa angka-angka bulat. Kemudian, program menghitung ukuran array tersebut dengan

menggunakan ekspresi sizeof(data) / sizeof(data[0]). Hasil perhitungan ini disimpan dalam variabel size. Selanjutnya, program mendefinisikan sebuah variabel target yang berisi angka 4, serta sebuah variabel count yang diinisialisasi dengan nilai 0. Variabel count ini akan digunakan untuk menghitung jumlah kemunculan angka 4 dalam array.

BAB IV

KESIMPULAN

Dalam mempelajari praktikum Algoritma Searching, terdapat dua algoritma yang telah dipelajari yaitu Sequential Search (Pencarian Linear) dan Binary Search (Pencarian Biner). Sequential Search adalah algoritma pencarian paling sederhana yang bekerja dengan mencari data secara berurutan dari awal hingga akhir dalam sebuah kumpulan data. Algoritma ini cocok untuk data yang tidak terurut dan memiliki kompleksitas waktu terburuk O(n), di mana n adalah jumlah elemen dalam kumpulan data.

Sebaliknya, Binary Search adalah algoritma pencarian yang lebih efisien untuk data yang telah terurut. Cara kerjanya adalah dengan membagi data menjadi dua bagian dan mencari di bagian yang sesuai dengan nilai yang dicari. Algoritma ini memiliki kompleksitas waktu O(log n), yang berarti lebih cepat dibandingkan Sequential Search untuk data besar yang terurut. Dalam praktikum, implementasi kedua algoritma ini dalam bahasa pemrograman seperti Python, Java, atau C++ memberikan pengalaman praktis untuk memahami kinerja dan karakteristik masing-masing algoritma melalui berbagai kasus uji dengan data yang berbeda-beda.

DAFTAR PUSTAKA

[1] Asisten Praktikum, *"Modul 8 Searching Seaching"*, Learning Management System, 2024.