

LAPORAN PRAKTIKUM
MODUL 8
ALGORITMA SEARCHING



Disusun oleh:
Rasyid Nafsyarie
NIM : 2311102011

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

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024

BAB I

TUJUAN PRAKTIKUM

1. Menunjukkan beberapa algoritma dalam Pencarian.
2. Menunjukkan bahwa pencarian merupakan suatu persoalan yang bisa diselesaikan dengan beberapa algoritma yang berbeda.
3. Dapat memilih algoritma yang paling sesuai untuk menyelesaikan suatu permasalahan pemrograman

BAB II

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 teratur. 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 berurutan. • 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.

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

GUIDED

1. GUIDED 1

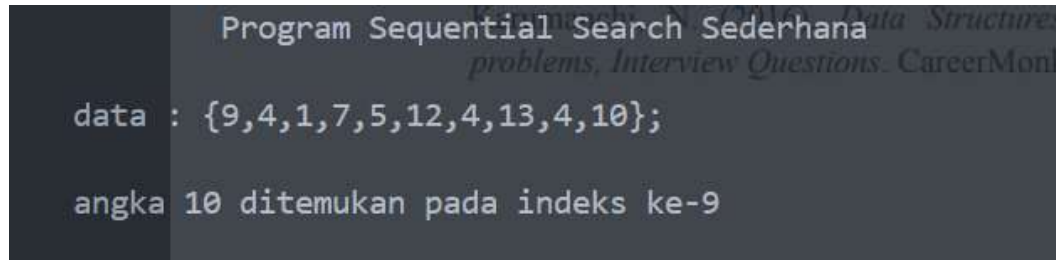
SOURCE CODE

```
#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;
    }
}
```

SCREENSHOOT PROGRAM



```
Program Sequential Search Sederhana  
data : {9,4,1,7,5,12,4,13,4,10};  
angka 10 ditemukan pada indeks ke-9
```

DESKRIPSI PROGRAM

Dalam contoh kode di atas, kita telah melihat implementasi algoritma Sequential Search dalam C++. Kode tersebut mencari nilai tertentu dalam array dengan cara memeriksa setiap elemen secara berurutan. Dengan demikian, kita dapat mengetahui apakah nilai yang dicari ada dalam array dan diindeks ke berapa jika ditemukan. Jika tidak ditemukan, pesan akan ditampilkan bahwa nilai tersebut tidak ada dalam data.

2. GUIDED 1

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";
}

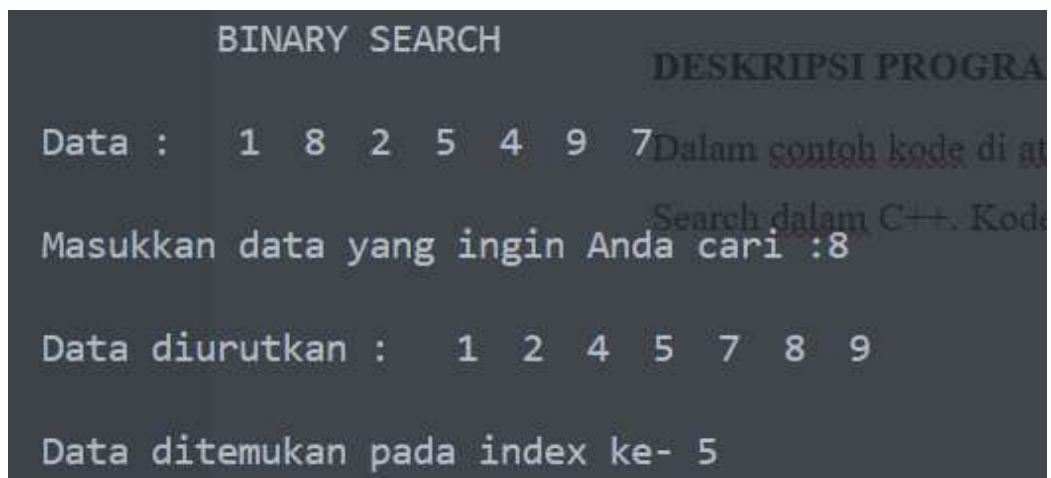
```

```

}
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



The screenshot shows the output of a C++ program titled "BINARY SEARCH". The output displays the initial data array, the user input for the search value, the sorted data array, and the index where the search value was found.

```

BINARY SEARCH
Data :    1  8  2  5  4  9  7
Masukkan data yang ingin Anda cari :8
Data diurutkan :    1  2  4  5  7  8  9
Data ditemukan pada index ke- 5

```


DESKRIPSI PROGRAM

Dalam kode di atas, kita menggunakan algoritma Selection Sort untuk mengurutkan array data secara ascending. Setelah itu, kita menggunakan algoritma Binary Search untuk mencari elemen yang diinginkan dalam array yang telah diurutkan. Jika elemen ditemukan, program akan menampilkan indeks elemen tersebut. Jika tidak ditemukan, program akan memberikan pesan bahwa data tidak ditemukan.

UNGUIDED

1. UNGUIDED 1

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

SOURCE CODE

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

int binarySearch(const string& sentence, char target) {
    int left = 0;
    int right = sentence.length() - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;

        if (sentence[mid] == target) {
            return mid;
        }

        if (sentence[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }

    return -1;
}
```

```
int main() {
    string sentence;
    char target;

    cout << "Masukkan kalimat: ";
    getline(cin, sentence);

    sort(sentence.begin(), sentence.end());

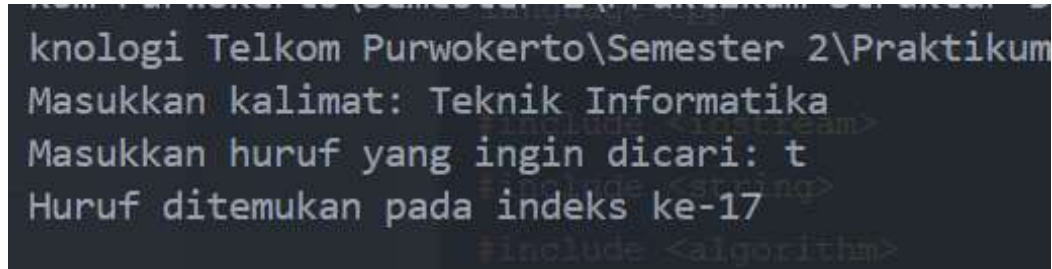
    cout << "Masukkan huruf yang ingin dicari: ";
    cin >> target;

    int index = binarySearch(sentence, target);

    if (index != -1) {
        cout << "Huruf ditemukan pada indeks ke-" << index <<
endl;
    } else {
        cout << "Huruf tidak ditemukan dalam kalimat." << endl;
    }

    return 0;
}
```

SCREENSHOOT PROGRAM



```
knologi Telkom Purwokerto\Semester 2\Praktikum
Masukkan kalimat: Teknik Informatika
Masukkan huruf yang ingin dicari: t
Huruf ditemukan pada indeks ke-17
```

DESKRIPSI PROGRAM

Dengan menggunakan teknik pencarian binary, kita dapat mencari karakter tertentu dalam sebuah string yang sudah terurut dengan efisien. Melalui kode di atas, pengguna diminta untuk memasukkan sebuah kalimat, kemudian program akan mengurutkan kalimat tersebut, meminta karakter target, dan menampilkan apakah karakter tersebut ditemukan beserta indeksnya dalam kalimat. Dengan demikian, implementasi pencarian binary ini dapat membantu dalam menemukan elemen tertentu dalam data yang sudah terurut.

2. UNGUIDED 2

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

SOURCE CODE

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string kalimat;
    int jumlahVokal = 0;

    cout << "Masukkan kalimat: ";
    getline(cin, kalimat);
```

```

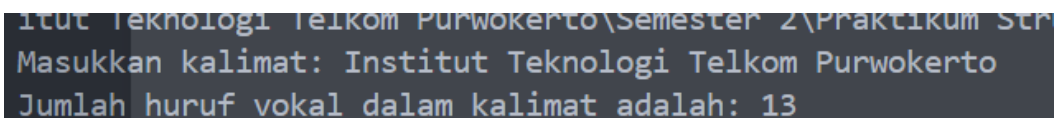
        for (char huruf : kalimat) {
            if (huruf == 'a' || huruf == 'i' || huruf == 'u' || huruf
== 'e' || huruf == 'o' ||
                huruf == 'A' || huruf == 'I' || huruf == 'U' || huruf
== 'E' || huruf == 'O') {
                jumlahVokal++;
            }
        }

        cout << "Jumlah huruf vokal dalam kalimat adalah: " <<
jumlahVokal << endl;

        return 0;
    }

```

SCREENSHOOT PROGRAM



```

Institut Teknologi Telkom Purwokerto\Semester 2\Praktikum Str
Masukkan kalimat: Institut Teknologi Telkom Purwokerto
Jumlah huruf vokal dalam kalimat adalah: 13

```

DESKRIPSI PROGRAM

Program ini memungkinkan pengguna untuk menghitung jumlah huruf vokal dalam sebuah kalimat yang dimasukkan. Dengan menggunakan loop dan kondisi if, program dapat mengidentifikasi huruf vokal dan menghitungnya secara akurat. Hal ini memperlihatkan bagaimana C++ dapat digunakan untuk membuat program sederhana yang dapat memproses input pengguna dengan efisien.

3. UNGUIDED 2

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>
#include <string>
using namespace std;
int main() {
    string kalimat;
    int jumlahVokal = 0;

    cout << "Masukkan kalimat: ";
    getline(cin, kalimat);

    for (char huruf : kalimat) {
        if (huruf == 'a' || huruf == 'i' || huruf == 'u' || huruf
== 'e' || huruf == 'o' ||
            huruf == 'A' || huruf == 'I' || huruf == 'U' || huruf
== 'E' || huruf == 'O') {
            jumlahVokal++;
        }
    }

    cout << "Jumlah huruf vokal dalam kalimat adalah: " <<
jumlahVokal << endl;

    return 0;
}
```

SCREENSHOOT PROGRAM



```
itut Teknologi Telkom Purwokerto\\Semest
Jumlah kemunculan angka 4: 4
```

DESKRIPSI PROGRAM

Dalam program kali ini, kita menggunakan fungsi `sequentialSearch` untuk mencari berapa kali angka 4 muncul dalam array data. Hasilnya akan dicetak ke layar. Pencarian berurutan adalah metode sederhana namun efektif untuk mencari nilai dalam array. Semoga penjelasan ini membantu Anda memahami konsep pencarian berurutan dalam C++.

BAB IV

KESIMPULAN

Setelah melakukan pembelajaran mengenai Queue di Bahasa Pemrograman C++ berikut poin utama yang telah dipelajari :

1. Algoritma pencarian digunakan dalam berbagai aplikasi seperti database querying, pencarian dalam aplikasi perangkat lunak, pemrosesan data, dan banyak lagi.
2. Dengan memahami dan mengimplementasikan berbagai algoritma searching, kita dapat memilih metode yang paling sesuai berdasarkan kebutuhan dan karakteristik data yang kita hadapi.
3. Metode pencarian yang efisien untuk array yang sudah diurutkan. Pencarian dilakukan dengan membagi array menjadi dua bagian, dan menentukan bagian mana yang mungkin mengandung elemen yang dicari.

DAFTAR PUSTAKA

Karumanchi, N. (2016). Data Structures and algorithms made easy: Concepts, problems, Interview Questions. CareerMonk Publications.