

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VIII
SEARCHING**



Disusun Oleh :

NAMA : GALIH TRISNA
NIM : 2311102050

Dosen

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

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFROMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

A. Dasar Teori

1. Sequential Search

Sequential search adalah teknik pencarian data yang dilakukan dengan cara membandingkan setiap elemen data satu per satu, mulai dari elemen pertama hingga elemen yang dicari ditemukan. Proses ini terus berlanjut hingga data ditemukan atau seluruh elemen telah diperiksa. Algoritma ini termasuk salah satu algoritma pencarian yang paling sederhana dan sering digunakan dalam situasi di mana data tidak memiliki struktur tertentu.

Fungsi Sequential Search

- Digunakan untuk mencari data dengan melakukan proses membandingkan setiap elemen larik secara beruntun satu persatu, mulai dari elemen pertama, sampai elemen yang dicari ditemukan, atau seluruh elemen sudah diperiksa .
- Dapat digunakan untuk pengelolaan data, seperti fungsi pencarian data barang pada master data barang .
- Dalam konteks pengujian kecepatan pencarian, Algoritma Sequential Search dapat digunakan dengan fungsi microtime untuk mengevaluasi kecepatan pencarian data

Cara Kerja Sequential Search

- Algoritma Sequential Search bekerja dengan melakukan perbandingan satu persatu secara beruntun dalam kumpulan data dengan data yang dicari sampai data tersebut ditemukan atau tidak ditemukan.

2. Binary Search

Binary search adalah teknik pencarian data yang dilakukan dengan cara membandingkan data yang dicari dengan elemen tengah dari daftar data yang sudah diurutkan. Jika data yang dicari sama dengan elemen tengah, pencarian selesai. Jika tidak, proses pencarian dilanjutkan pada separuh daftar yang relevan (kiri atau kanan) tergantung pada perbandingan tersebut. Proses ini terus berlanjut dengan membagi dua daftar hingga data ditemukan atau tidak ada lagi elemen yang tersisa untuk diperiksa. Algoritma ini lebih efisien daripada sequential search terutama pada data yang berukuran besar.

Fungsi Binary Search

- Digunakan untuk mencari data dalam daftar yang sudah diurutkan dengan membagi dua daftar secara berulang, dan membandingkan data yang dicari dengan elemen tengah, hingga data ditemukan atau seluruh elemen sudah diperiksa.

- Dapat digunakan dalam aplikasi yang membutuhkan pencarian cepat, seperti dalam basis data yang besar dan sudah diurutkan.
- Dalam konteks pengujian kecepatan pencarian, Algoritma Binary Search dapat digunakan dengan fungsi microtime untuk mengevaluasi kecepatan pencarian data.

Cara Kerja Binary Search

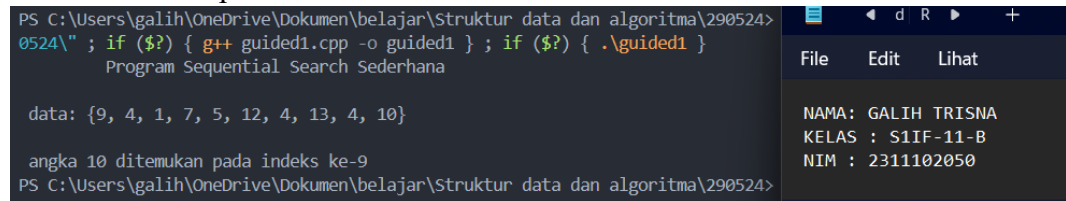
- Algoritma Binary Search bekerja dengan cara membandingkan data yang dicari dengan elemen tengah dari daftar yang sudah diurutkan.
- Jika data yang dicari lebih kecil dari elemen tengah, maka pencarian dilanjutkan pada separuh daftar sebelah kiri.
- Jika data yang dicari lebih besar dari elemen tengah, maka pencarian dilanjutkan pada separuh daftar sebelah kanan.
- Proses ini terus berulang dengan membagi dua daftar yang tersisa hingga data ditemukan atau tidak ada lagi elemen yang tersisa untuk diperiksa.

B. Guided

Guided 1

```
#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;
}
```

Screenshots Output



```
PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524>
0524\" ; if ($?) { g++ guided1.cpp -o guided1 } ; if ($?) { .\guided1 }
    Program Sequential Search Sederhana

data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}

angka 10 ditemukan pada indeks ke-9
PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524>
```

The screenshot shows a Windows PowerShell terminal window with a dark background. The command prompt is at the top, followed by the execution of a C++ program. The program outputs the data array and the index of the found element. On the right side of the terminal window, there is a sidebar with a menu (File, Edit, Lihat) and a text area containing the user's name, class, and NIM.

File	Edit	Lihat
NAMA: GALIH TRISNA		
KELAS : S1IF-11-B		
NIM : 2311102050		

Deskripsi:

Program di atas mengimplementasikan penggunaan sequential search untuk mencari elemen dalam array berukuran 10 dengan tipe data integer. Sequential search dilakukan dengan memeriksa setiap elemen dari awal hingga akhir. Proses pencarian dimulai dengan melakukan looping melalui setiap indeks dalam array. Jika nilai pada indeks tersebut sama dengan nilai yang dicari (misalnya, 10), maka variabel ketemu diubah menjadi true, dan looping dihentikan dengan break. Jika tidak, looping dilanjutkan hingga semua elemen diperiksa. Setelah looping selesai, program akan menampilkan seluruh elemen array dan mengecek nilai variabel ketemu. Jika ketemu bernilai true, program menampilkan posisi indeks elemen yang dicari. Jika ketemu bernilai false, program menampilkan pesan bahwa nilai tidak ditemukan dalam array.

Guided 2

```
#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;
}

```

Screenshots Output

```

PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524>
0524\" ; if ($?) { g++ guided2.cpp -o guided2 } ; if ($?) { .\guided2 }
BINARY SEARCH

Data :  1  8  2  5  4  9  7

Masukkan data yang ingin Anda cari : 10

```

File Edit Lihat

NAMA: GALIH TRISNA
 KELAS : S1IF-11-B
 NIM : 2311102050

Ln 3, Col 17 53 karakter 10

Deskripsi:

Program di atas adalah implementasi binary search untuk mencari nilai tertentu dalam array yang telah diurutkan secara ascending. Program ini terlebih dahulu menggunakan algoritma selection sort untuk mengurutkan array data secara ascending. Setelah itu, program meminta pengguna untuk memasukkan nilai yang ingin dicari dalam array. Kemudian, program akan mencari nilai tersebut menggunakan algoritma binary search. Algoritma ini membagi array menjadi dua bagian dan memeriksa nilai tengah. Jika nilai tengah sama dengan nilai yang dicari, maka program akan menampilkan indeks di mana nilai tersebut ditemukan. Jika tidak, pencarian akan dilanjutkan pada setengah array yang sesuai dengan nilai tengah. Proses ini akan berlanjut hingga nilai ditemukan atau batas pencarian berakhir. Hasil akhir dari program adalah pesan yang menunjukkan apakah nilai yang dicari ditemukan dalam array serta indeksnya, atau pesan bahwa nilai tidak ditemukan.

C. Unguided/Tugas

Unguided 1

```
#include <iostream>
#include <conio.h>
#include <iomanip>

using namespace std;

string kalimat;
char c;

void toLower() {
    string tmp;
    for (int i = 0; i < kalimat.length(); i++) {
        tmp += tolower(kalimat[i]);
    }
    kalimat = tmp;
}

void selection_sort()
{
    int min, i, j;
    char tmp;
    toLower();
    for (i = 0; i < kalimat.length(); i++)
    {
        min = i;
        for (j = i + 1; j < kalimat.length(); j++)
        {
            if (kalimat[j] < kalimat[min])
            {
                min = j;
            }
        }
        tmp = kalimat[i];
        kalimat[i] = kalimat[min];
        kalimat[min] = tmp;
    }
}

void binarysearch()
{

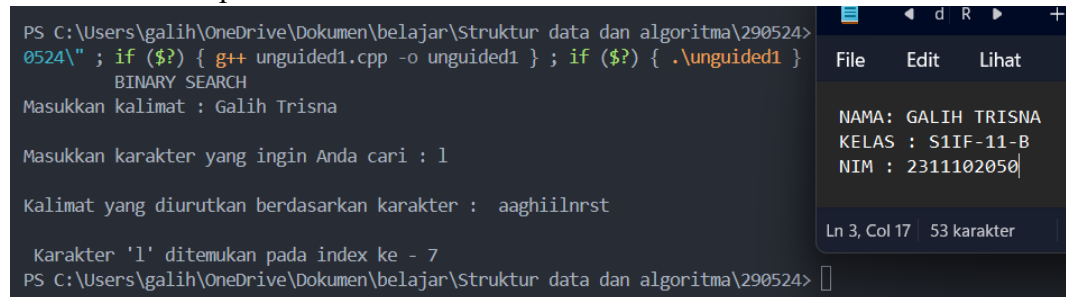
```

```

int awal, akhir, tengah, b_flag = 0;
awal = 0;
akhir = kalimat.length();
while (b_flag == 0 && awal <= akhir)
{
    tengah = (awal + akhir) / 2;
    if (kalimat[tengah] == c)
    {
        b_flag = 1;
        break;
    }
    else if (kalimat[tengah] < c)
        awal = tengah + 1;
    else
        akhir = tengah - 1;
}
if (b_flag == 1)
    cout << "\n Karakter '" << c << "' ditemukan pada index
ke - " << tengah << endl;
else
    cout << "\nData tidak ditemukan\n";
}
int main()
{
    cout << "\t BINARY SEARCH " << endl;
    cout << "Masukkan kalimat : ";
    getline(cin, kalimat);
    cout << "\nMasukkan karakter yang ingin Anda cari : ";
    cin >> c;
    c = tolower(c);
    cout << "\nKalimat yang diurutkan berdasarkan karakter : ";
    selection_sort();
    for (int x = 0; x < kalimat.length(); x++)
        cout << kalimat[x];
    cout << endl;
    binarysearch();
    return EXIT_SUCCESS;
}

```

Screenshots Output



```
PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524>
0524\" ; if ($?) { g++ unguided1.cpp -o unguided1 } ; if ($?) { .\unguided1 }
BINARY SEARCH
Masukkan kalimat : Galih Trisna
Masukkan karakter yang ingin Anda cari : l
Kalimat yang diurutkan berdasarkan karakter : aaghiilnrst
Karakter 'l' ditemukan pada index ke - 7
PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524>
```

File Edit Lihat

NAMA: GALIH TRISNA
KELAS : S1IF-11-B
NIM : 2311102050

Ln 3, Col 17 | 53 karakter

Deskripsi

Program ini digunakan untuk mencari karakter dalam sebuah kalimat menggunakan metode binary search. Pengguna akan memasukkan kalimat dan karakter yang ingin dicari indeksinya. Sebelum pencarian, kalimat diubah menjadi huruf kecil. Setelah itu, kalimat diurutkan dari karakter terkecil ke terbesar menggunakan algoritma selection sort. Program kemudian menampilkan kalimat yang telah diubah dan diurutkan. Selanjutnya, program mencari karakter yang diinginkan dan menampilkan posisi atau indeksinya berdasarkan kalimat yang sudah diurutkan, bukan kalimat asli, karena pencarian dilakukan menggunakan binary search.

Unguided 2

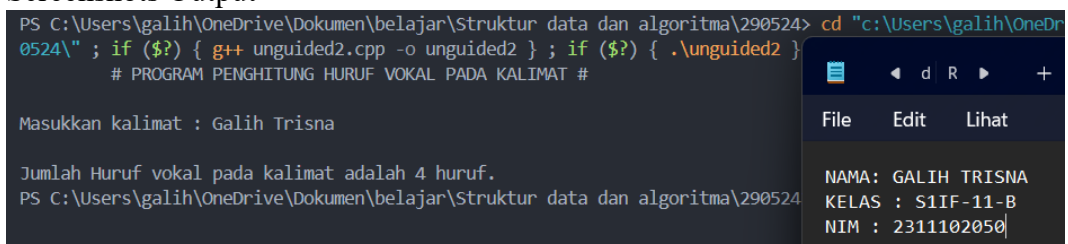
```
#include <iostream>
using namespace std;
int main()
{
    string kalimat;
    int jumlah = 0;

    cout << "\t # PROGRAM PENGHITUNG HURUF VOKAL PADA KALIMAT
#" << endl;
    cout << "\nMasukkan kalimat : ";
    getline(cin, kalimat);

    for (int i = 0; i < kalimat.length(); i++) {
        char huruf = tolower(kalimat[i]);
        if (huruf == 'a' || huruf == 'i' || huruf == 'u' ||
huruf == 'e' || huruf == 'o') jumlah++;
    }

    cout << "\nJumlah Huruf vokal pada kalimat adalah " <<
jumlah << " huruf.";
    return 0;
}
```

Screenshots Output



```
PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524> cd "c:\Users\galih\OneDr
0524\" ; if ($?) { g++ unguided2.cpp -o unguided2 } ; if ($?) { .\unguided2 }
# PROGRAM PENGHITUNG HURUF VOKAL PADA KALIMAT #

Masukkan kalimat : Galih Trisna

Jumlah Huruf vokal pada kalimat adalah 4 huruf.
PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524>
```

Deskripsi

Program di atas adalah sebuah program sederhana untuk menghitung jumlah huruf vokal dalam sebuah kalimat yang dimasukkan oleh pengguna. Pertama, program meminta pengguna untuk memasukkan kalimat. Kemudian, program melakukan iterasi melalui setiap karakter dalam kalimat tersebut. Setiap karakter diubah menjadi huruf kecil menggunakan fungsi `tolower()` untuk memudahkan pengecekan. Jika karakter tersebut adalah salah satu huruf vokal (a, i, u, e, o), maka variabel jumlah akan ditambah satu. Setelah semua karakter diperiksa, program akan menampilkan jumlah huruf vokal yang ditemukan dalam kalimat.

Unguided 3

```
#include <iostream>

using namespace std;
int main()
{
    int n = 10;
    int data[n] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int cari = 4;
    int counter = 0;
    int i;
    for (i = 0; i < n; i++)
    {
        if (data[i] == cari)
        {
            counter++;
        }
    }

    cout << "Program Sequential mencari angka 4" << endl
         << endl;
    cout << " data: 9, 4, 1, 4, 7, 10, 5, 4, 12, 4" << endl;
    cout << "banyak angka 4 pada array adalah : " << counter <<
endl;
    return 0;
}
```

Screenshots Output

PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524> g++ unguided3.cpp -o unguided3 ; if (\$?) { .\unguided3 }	File	Edit	Lihat
Program Sequential mencari angka 4			
data: 9, 4, 1, 4, 7, 10, 5, 4, 12, 4			
banyak angka 4 pada array adalah : 4			
PS C:\Users\galih\OneDrive\Dokumen\belajar\Struktur data dan algoritma\290524>			
Ln 3, Col 17 53 karakter			10

Deskripsi

Program ini merupakan contoh sederhana dari sequential search yang digunakan untuk menghitung jumlah kemunculan suatu angka dalam sebuah array. Pada program ini, array data telah ditentukan dengan 10 elemen, dan angka yang akan dicari adalah 4. Variabel counter digunakan untuk menghitung jumlah kemunculan angka tersebut dalam array.

Program menggunakan loop for untuk memeriksa setiap elemen dalam array. Jika nilai elemen sama dengan angka yang dicari (cari), maka nilai counter akan ditambah satu. Setelah proses pengecekan selesai, program akan menampilkan pesan yang menyatakan jumlah kemunculan angka yang dicari dalam array.

D. Kesimpulan

Sequential Search adalah pilihan yang baik untuk daftar data yang kecil atau ketika data tidak terurut, karena kesederhanaannya dan kemudahan implementasi. Binary Search adalah pilihan yang lebih efisien untuk daftar data yang besar dan sudah diurutkan, karena waktu pencariannya yang lebih cepat. Namun, ini membutuhkan data yang terurut terlebih dahulu, yang mungkin memerlukan langkah tambahan.

E. Referensi

Asisten Praktikum. (2024). Modul VIII : Search

Annisa, (2023). Algoritma Sequential Search: Pengertian, Fungsi dan Cara Kerjanya. FIKTI UMSU. Diakses pada 3 Juni 2024, dari:

<https://fikti.umsu.ac.id/algoritma-sequential-search-pengertian-fungsi-dan-cara-kerjanya>

Parewa Labs .(2023). Binary Search. Diakses pada 3 Juni 2024, dari:

<https://www.programiz.com/dsa/binary-search>

Anshu, (2021). Binary Search and Its Powerful Applications. Medium. Diakses pada 3 Juni 2024, dari:

<https://medium.com/@imanshu822/binary-search-and-its-powerful-applications-39ae7d7bca69>