

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VIII
ALGORITMA SEARCHING**



Disusun Oleh :

NAMA : ANISA YASAROH

NIM : 2311102063

Dosen :

WAHYU ANDI SAPUTRA, S.Pd., M.Eng.

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

A. DASAR TEORI

Algoritma pencarian (*searching algorithm*) adalah algoritma yang menerima sebuah argumen kunci dan dengan langkah-langkah tertentu akan mencari rekaman dengan kunci tersebut. Setelah proses pencarian dilaksanakan, akan diperoleh salah satu dari dua kemungkinan, yaitu data yang dicari ditemukan (*successful*) atau tidak ditemukan (*unsuccessful*). Tujuannya adalah untuk menemukan posisi atau keberadaan elemen yang dicari. Dalam pemrograman, algoritma search menjadi salah satu teknik penting dalam menyelesaikan berbagai masalah.

Ada dua macam teknik pencarian yaitu pencarian sekuensial (*sequential searching*) dan pencarian biner (*binary searching*). Perbedaan dari dua teknik ini terletak pada keadaan data. Pencarian sekuensial digunakan apabila data dalam keadaan acak atau tidak terurut. Sebaliknya, pencarian biner digunakan pada data yang sudah dalam keadaan urut.

a. Sequential Search

Sequential Searching adalah salah satu algoritma pencarian data yang biasa digunakan untuk data yang berpola acak atau belum terurut. Algoritma ini akan mencari data sesuai kata kunci yang diberikan mulai dari elemen awal pada array hingga elemen akhir array. Kemungkinan terbaik (best case) ketika menggunakan algoritma ini adalah jika data yang dicari terletak di indeks awal array sehingga hanya membutuhkan sedikit waktu pencarian. Sedangkan kemungkinan terburuknya (worst case) adalah jika data yang dicari ternyata terletak dibagian akhir dari array sehingga pencarian data akan memakan waktu yang lama.

Algoritma pencarian berurutan dapat dituliskan sebagai berikut :

- 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.

b. Binary Search

Binary Searching adalah algoritma pencarian yang efisien digunakan untuk mencari elemen dalam kumpulan data yang sudah terurut. Algoritma ini membagi data menjadi dua bagian dan membandingkan elemen tengah dengan elemen yang dicari. Dalam kehidupan sehari-hari, sebenarnya kita juga sering menggunakan pencarian biner. Misalnya saat ingin mencari suatu kata dalam kamus.

Algoritma binary search :

- Data diambil dari posisi 1 sampai posisi akhir N
- Kemudian cari posisi data tengah dengan rumus: $(\text{posisi awal} + \text{posisi akhir})/2$
- Kemudian data yang dicari dibandingkan dengan data yang di tengah, apakah sama atau lebih kecil, atau lebih besar.
- Jika lebih besar, maka proses pencarian dicari dengan posisi awal adalah posisi tengah + 1
- Jika lebih kecil, maka proses pencarian dicari dengan posisi akhir adalah posisi tengah – 1
- Jika data sama, berarti ketemu.

1. 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;

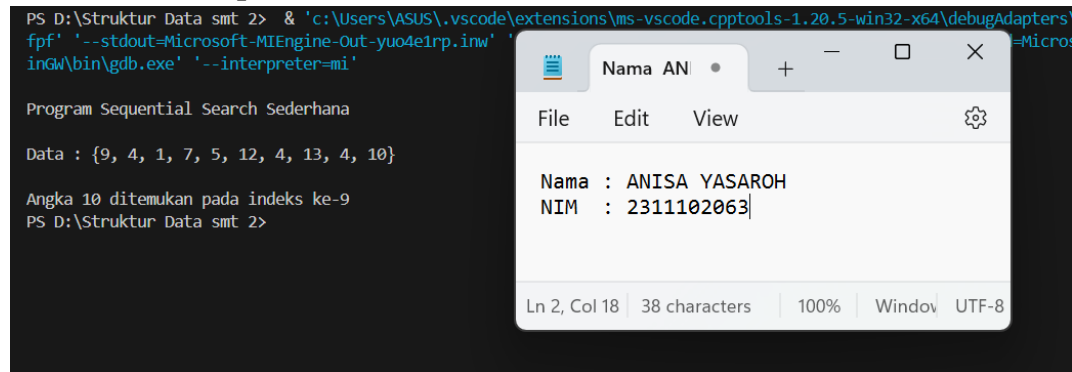
    //Algoritma Sequential Search
    for (i = 0; i < n; i++){
        if (data[i] == cari){
            ketemu = true;
            break;
        }
    }

    cout << "\nProgram Sequential Search Sederhana\n" <<
endl;
    cout << "Data : {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}" <<
endl;

    if (ketemu) {
        cout << "\nAngka " << cari << " ditemukan pada indeks
ke-" << i << endl;
    } else {
        cout << "\nAngka " << cari << " tidak dapat ditemukan
pada data." << endl;
    }

    return 0;
}
```

Screenshoot Output



```
PS D:\Struktur Data smt 2> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\
fpf' '--stdout=Microsoft-MIEngine-Out-yuo4e1rp.inw' '
ingW\bin\gdb.exe' '--interpreter=mi'

Program Sequential Search Sederhana

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

Angka 10 ditemukan pada indeks ke-9
PS D:\Struktur Data smt 2>
```

Nama : ANISA YASAROH
NIM : 2311102063

Deskripsi Program

Program di atas yaitu untuk menemukan angka tertentu dalam sebuah array. Program dimulai dengan mendeklarasikan array 'data' berisi 10 elemen dan menetapkan nilai yang akan dicari ('cari') sebesar 10. Variabel 'ketemu' digunakan sebagai penanda apakah nilai yang dicari ditemukan, dan variabel 'i' sebagai indeks untuk iterasi. Dalam loop for, program memeriksa setiap elemen dalam array. Jika elemen yang dicari ditemukan, variabel 'ketemu' diatur menjadi true dan loop dihentikan dengan perintah 'break'. Setelah loop selesai, program menampilkan pesan apakah angka yang dicari ditemukan atau tidak, beserta indeks di mana angka tersebut ditemukan jika ada. Output program termasuk pengumuman tentang dimulainya pencarian sekuensial, penampilan data array, dan hasil pencarian.

2. Guided 2

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++){
        if (data[j] < data [min]){
            min = j;
        }
        temp = data[i];
        data[i] = data[min];
        data[min] = temp;
    }
}

void binarysearch(){
```

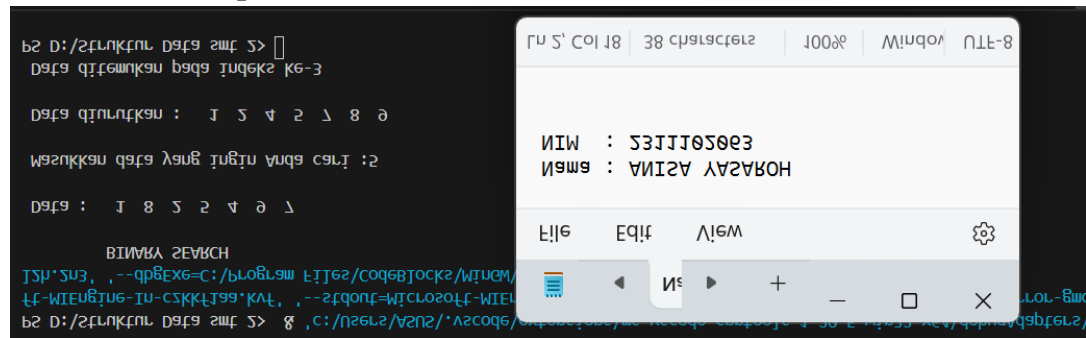
```

//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 : ";
    cout<<"\n Masukkan data yang ingin Anda cari : ";
    cin >>cari;
    cout<<"\n Data diurutkan : ";
    //urutkandatangenselectionsort
    selection_sort();
    //tampilkantadatelahdiurutkan
    for(int x = 0; x < 7; x++)
        cout<< setw(3)<<data[x];
    cout<<endl;
    binarysearch();
    _getche();
    return EXIT_SUCCESS;
}

```

Screenshot Output



```
bs D:/2ftruktur Data smf > []
Data diurutkan pada indeks ke-3

Data diurutkan : 1 5 4 2 1 8 0

Masukkan data yang ingin anda cari : 3

Data : 1 8 5 2 4 0 1

BINARY SEARCH
JSM'SU3, --qrgEx6=C:/Program Files/CodeBlocks/MinGW/
tC-WIN6406-10-CSKKT199.KAL, --gfont=WINCROSTC-WINEL
bs D:/2ftruktur Data smf > g ,c:/users/AN202/Ascode/
tC-WIN6406-10-CSKKT199.KAL, --gfont=WINCROSTC-WINEL
```

Deskripsi Program

Program diatas yaitu yang pertama, array 'data' yang berisi 7 elemen tidak terurut dan variabel 'cari' yang menyimpan nilai yang akan dicari dideklarasikan. Fungsi 'selection_sort' mengurutkan array 'data' dengan menemukan elemen terkecil dan menukarnya dengan elemen saat ini dalam setiap iterasi. Setelah array diurutkan, fungsi 'binarysearch' mencari nilai 'cari' dalam array yang telah diurutkan. Pencarian biner dilakukan dengan menentukan indeks awal dan akhir, lalu membagi array berulang kali untuk menemukan elemen yang dicari. Jika elemen ditemukan, program menampilkan indeksnya; jika tidak, program menampilkan pesan bahwa data tidak ditemukan. Fungsi main mengatur jalannya program, mulai dari menampilkan data awal, meminta input pengguna untuk nilai yang dicari, mengurutkan data, menjalankan pencarian biner, dan menampilkan hasilnya.

2. UNGUIDED

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

Source Code

```
#include <iostream>

using namespace std;
void selectionSort(string &huruf, int a_63)
{
    int x, y, min;
    for (x = 0; x < a_63 - 1; x++)
    {
        min = x;
        for (y = x + 1; y < a_63; y++)
            if (huruf[y] < huruf[min])
                min = y;
        char temp = huruf[x];
        huruf[x] = huruf[min];
        huruf[min] = temp;
    }
}
```

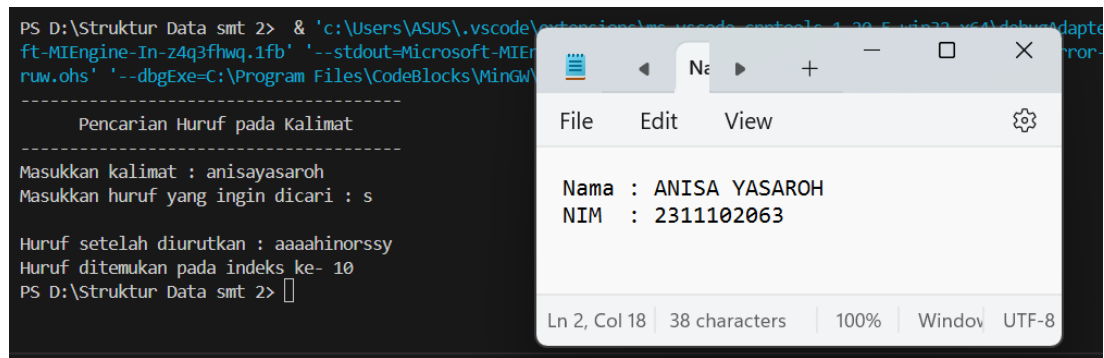
```

int binarySearch(string huruf, int kiri, int kanan, char
target)
{
    while (kiri <= kanan)
    {
        int mid = kiri + (kanan - kiri) / 2;
        if (huruf[mid] == target)
            return mid;
        if (huruf[mid] < target)
            kiri = mid + 1;
        else
            kanan = mid - 1;
    }
    return -1;
}

int main()
{
    string kalimat;
    char input;
    cout << "-----" <<
endl;
    cout << "          Pencarian Huruf pada Kalimat          " <<
endl;
    cout << "-----" <<
endl;
    cout << "Masukkan kalimat : ";
    getline(cin, kalimat);
    cout << "Masukkan huruf yang ingin dicari : ";
    cin >> input;
    cout << endl;
    selectionSort(kalimat, kalimat.size());
    int result = binarySearch(kalimat, 0, kalimat.size() -
1, input);
    if (result == -1)
    {
        cout << "Huruf yang Anda cari tidak ditemukan!" <<
endl;
    }
    else
    {
        cout << "Huruf setelah diurutkan : " << kalimat <<
endl;
        cout << "Huruf ditemukan pada indeks ke- " <<
result << endl;
    }
    return 0;
}

```


Screenshoot Output



```
PS D:\Struktur Data smt 2> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.39.5-win32-x64\debugAdapte
ft-MIEngine-In-z4q3fhwq.1fb' '--stdout=Microsoft-MIEngine-1.39.5-win32-x64\bin\miengine.exe' '--dbgExe=C:\Program Files\CodeBlocks\MinGW\bin\g++.exe'
-----
Pencarian Huruf pada Kalimat
-----
Masukkan kalimat : anisayasarah
Masukkan huruf yang ingin dicari : s

Huruf setelah diurutkan : aaaahinorssy
Huruf ditemukan pada indeks ke- 10
PS D:\Struktur Data smt 2>

Nama : ANISA YASAROH
NIM : 2311102063

Ln 2, Col 18 | 38 characters | 100% | Window | UTF-8
```

Deskripsi Program

Program di atas yaitu mengimplementasikan pencarian huruf dalam sebuah kalimat dengan menggunakan algoritma selection sort untuk pengurutan dan binary search untuk pencarian. Pertama, pengguna diminta memasukkan sebuah kalimat dan huruf yang ingin dicari. Fungsi 'selectionSort' mengurutkan karakter-karakter dalam kalimat secara alfabetis dengan cara menemukan karakter terkecil dan menukarnya dengan karakter saat ini dalam setiap iterasi. Setelah kalimat diurutkan, fungsi 'binarySearch' mencari huruf yang diinginkan dengan membagi kalimat yang sudah diurutkan menjadi dua bagian secara berulang sampai huruf tersebut ditemukan atau sampai semua kemungkinan telah diperiksa. Jika huruf ditemukan, program menampilkan kalimat yang sudah diurutkan dan indeks tempat huruf tersebut ditemukan. Jika tidak, program menampilkan pesan bahwa huruf yang dicari tidak ditemukan. Fungsi main mengatur jalannya program ini, mulai dari input pengguna, pemanggilan fungsi pengurutan dan pencarian, hingga menampilkan hasilnya.

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 jumlah = 0;

    cout << "-----"
    << endl;
    cout << "        Program Menghitung Huruf Vokal        "
    << endl;
    cout << "-----"
    << endl;

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

```

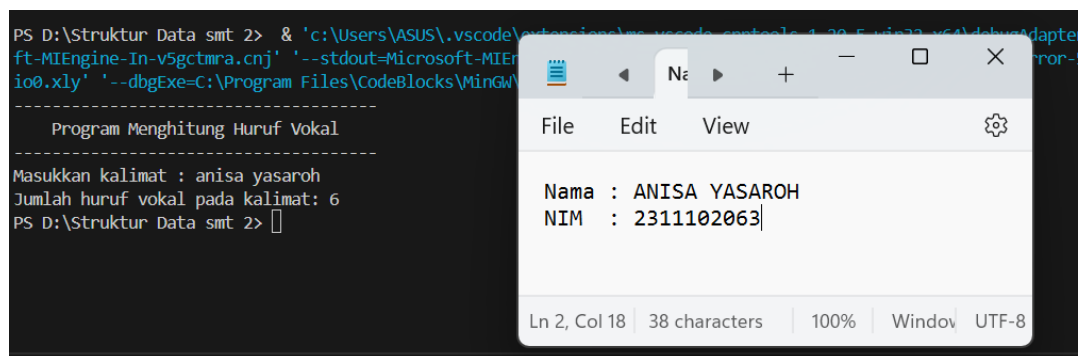
        for (int n_63 = 0; n_63 < kalimat.length();
n_63++) {
            char a = kalimat[n_63];
            if (a == 'a' || a == 'i' || a == 'u' || a ==
'e' || a == 'o' ||
                a == 'A' || a == 'I' || a == 'U' || a ==
'E' || a == 'O') {
                jumlah++;
            }
        }

        cout << "Jumlah huruf vokal pada kalimat: " <<
jumlah << endl;

        return 0;
}

```

Screenshoot Program



Deskripsi Program

Program di atas yaitu untuk menghitung jumlah huruf vokal dalam sebuah kalimat. Pertama, program mendeklarasikan variabel 'kalimat' untuk menyimpan kalimat yang diinput oleh pengguna dan variabel 'jumlah' untuk menghitung jumlah huruf vokal. Setelah menampilkan judul program dan meminta pengguna memasukkan kalimat, program membaca input tersebut menggunakan 'getline'. Program kemudian menggunakan loop 'for' untuk memeriksa setiap karakter dalam kalimat. Dalam loop, setiap karakter diperiksa apakah merupakan huruf vokal (baik huruf kecil maupun besar) dengan menggunakan pernyataan 'if'. Jika karakter tersebut adalah huruf vokal, variabel 'jumlah' akan ditambah satu. Setelah loop selesai, program menampilkan jumlah total huruf vokal yang ditemukan dalam kalimat. Program diakhiri dengan mengembalikan nilai 0 untuk menunjukkan bahwa eksekusi program berhasil.

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 hitungAngka(const int array[], int size, int target)

```

```

{
    int jumlah = 0;
    for (int a_63 = 0; a_63 < size; a_63++)
    {
        if (array[a_63] == target)
        {
            jumlah++;
        }
    }
    return jumlah;
}

int main()
{
    const int size = 10;
    int array[size] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int target = 4;
    int jumlah = hitungAngka(array, size, target);
    cout << "-----" <<
endl;
    cout << "                Menghitung Jumlah Angka                " <<
endl;
    cout << "-----" <<
endl;
    cout << "Data : ";
    for (int element : array)
    {
        cout << element << " ";
    }
    cout << "\nAngka yang dicari: " << target << endl;
    cout << "Ditemukan angka " << target << " dalam data
sebanyak: " << jumlah << endl;
    return 0;
}

```

Screenshoot Output

```

PS D:\Struktur Data smt 2> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapt
ft-MIEngine-In-ydhavc5n.gdj' '--stdout=Microsoft-MIEngine-110.k45' '--dbgExe=C:\Program Files\CodeBlocks\MinGW
-----
                Menghitung Jumlah Angka                -----
Data : 9 4 1 4 7 10 5 4 12 4
Angka yang dicari: 4
Ditemukan angka 4 dalam data sebanyak: 4
PS D:\Struktur Data smt 2>

```

Notepad window content:

```

Nama : ANISA YASAROH
NIM : 2311102063

```

Notepad status bar: Ln 2, Col 18 | 38 characters | 100% | Window | UTF-8

Deskripsi Program

Program diatas yaitu untuk menghitung frekuensi kemunculan suatu angka dalam sebuah array. Program dimulai dengan mendeklarasikan fungsi 'hitungAngka' yang menerima sebuah array, ukuran array, dan angka target yang akan dihitung frekuensinya. Fungsi ini menggunakan loop 'for' untuk memeriksa setiap elemen dalam array dan menambahkan nilai variabel 'jumlah' setiap kali elemen tersebut

sama dengan angka target. Fungsi ini kemudian mengembalikan nilai 'jumlah'. Dalam fungsi main, sebuah array dengan 10 elemen dideklarasikan dan diinisialisasi dengan beberapa angka. Angka target yang akan dihitung adalah 4. Program kemudian memanggil fungsi 'hitungAngka' untuk menghitung jumlah kemunculan angka 4 dalam array, dan hasilnya ditampilkan di layar. Program menampilkan isi array, angka yang dicari, dan jumlah kemunculan angka tersebut. Akhirnya, program mengembalikan nilai 0 untuk menunjukkan bahwa eksekusi program telah berhasil.

3. KESIMPULAN

Berdasarkan praktikum yang telah dilakukan, dapat disimpulkan bahwa :

1. Algoritma searching digunakan untuk mencari elemen tertentu.
2. Sequential Search adalah algoritma sederhana yang bekerja dengan memeriksa setiap elemen dalam urutan sekuensial. Cocok digunakan untuk kumpulan data yang belum terurut atau berukuran kecil.
3. Binary Search adalah algoritma efisien yang hanya dapat digunakan pada kumpulan data yang sudah terurut secara menaik atau menurun. Beroperasi dengan membagi kumpulan data menjadi dua bagian secara berulang.

4. REFERENSI

Goodrich, M. T., Tamassia, R., & Mount, D. M. (2020). Data structures and algorithms in C++. John Wiley & Sons.

Lin, A. (2019). Binary search algorithm. WikiJournal of Science, 2(1), 1-13.

Malik, D. S. (2023). C++ programming. Cengage Learning, EMEA.

Monteiro, F. R., Gadelha, M. R., & Cordeiro, L. C. (2022). Model checking C++ programs. Software Testing, Verification and Reliability, 32(1), e1793.

Wittenberg, L. (2019). Data Structures and Algorithms in C++: Pocket Primer. Mercury Learning and Information.