

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VIII
ALGORITMA SEARCHING**



Disusun Oleh :

NAMA : D'sharlendita Febianda Aurelia
NIM : 2311102069

Dosen :

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

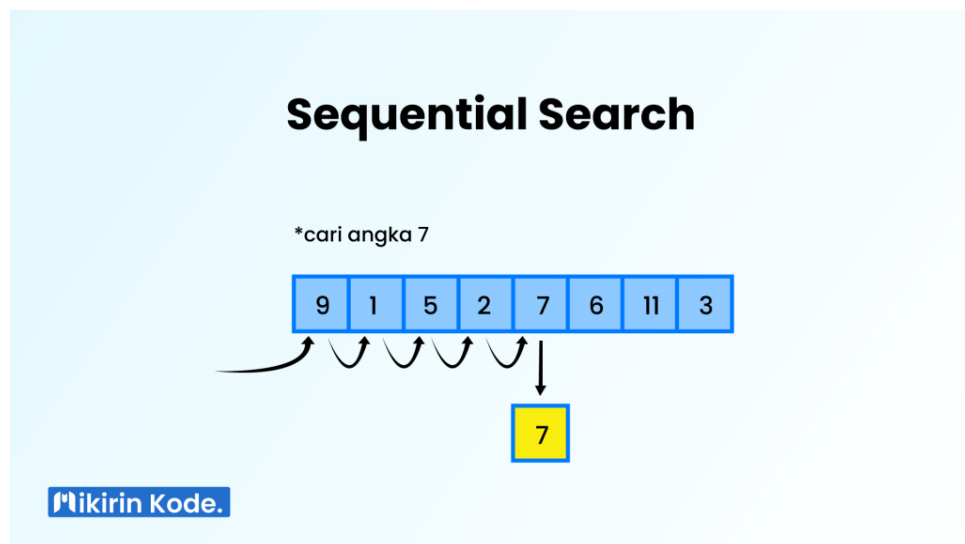
**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

A. Dasar Teori

Algoritma dan Struktur Data memiliki peran yang sangat penting dalam pembangunan sebuah perangkat lunak, baik dalam pembuatan desain serta implementasinya. Sebuah algoritma adalah satu set instruksi untuk melakukan tugas atau fungsi tertentu sehingga tercapai tujuan yang diinginkan dengan proses jalan menghabiskan waktu tertentu. Ada beberapa algoritma yang bisa digunakan untuk melakukan pencarian, salah satunya adalah Sequential Search dan Binary Search

1. Sequential Search

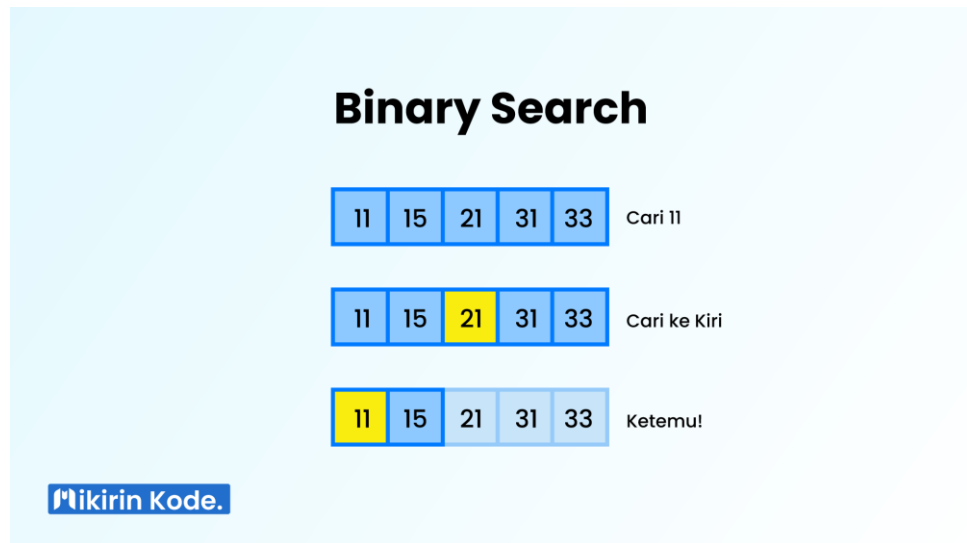
Sequential Search adalah cara untuk pencarian data dalam array 1 dimensi. Data yang akan dicari nanti akan ditelusuri dalam semua elemen-elemen array dari awal sampai akhir, dan data yang dicari tersebut tidak perlu diurutkan terlebih dahulu



2. Binary Search

Binary search adalah cara untuk pencarian data pada array yang sudah terurut. Karena salah satu syarat dalam binary search adalah data sudah dalam keadaan terurut, kemudian dibagi menjadi dua bagian. Awalnya

adalah membandingkan inputan dengan nilai tengah, selanjutnya dibandingkan ke kanan atau ke kiri sesuai dengan urutan listnya



B. Guided

Guided 1

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

Source Code:

```
#include <iostream>

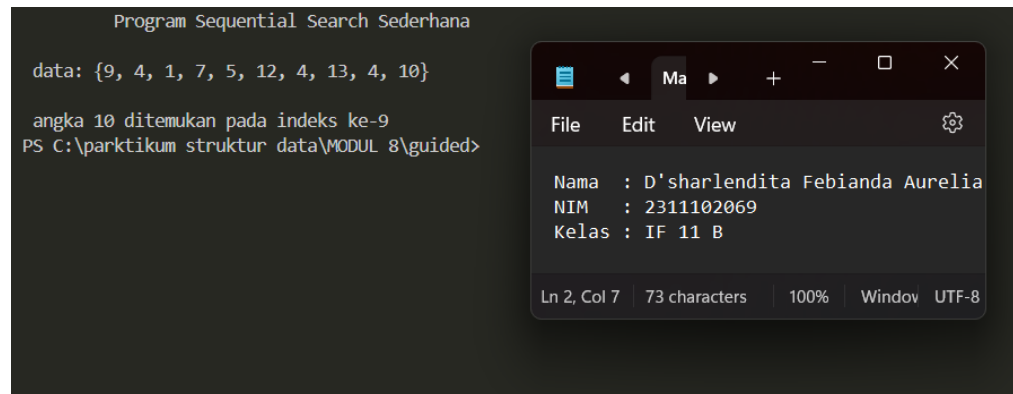
using namespace std;

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;
}
```

Screenshots Output :



The screenshot shows a terminal window titled "Program Sequential Search Sederhana" with the following output:

```
data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}
angka 10 ditemukan pada indeks ke-9
PS C:\parktikum struktur data\MODUL 8\guided>
```

Overlaid on the terminal is a Notepad window titled "Ma" containing the following text:

```
Nama : D'sharlendita Febianda Aurelia
NIM : 2311102069
Kelas : IF 11 B
```

The Notepad window's status bar shows "Ln 2, Col 7", "73 characters", "100%", "Window", and "UTF-8".

Deskripsi:

Program ini menggunakan algoritma Sequential Search untuk mencari angka 10 dalam array {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}. Program ini akan melakukan iterasi melalui elemen-elemen array dan menghentikan pencarian saat angka 10 ditemukan, kemudian mencetak indeksinya. Jika angka tidak ditemukan, program mencetak pesan bahwa angka tersebut tidak ada dalam array.

Guided 2

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

Source Code:

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

using namespace std;

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;
    }
}
```

```

    }
    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;
}

```

Screenshot Output:

The screenshot shows the output of a C++ program titled "BINARY SEARCH". The output displays an array of numbers: "Data : 1 8 2 5 4 9 7". It then prompts the user to enter a search value: "Masukkan data yang ingin Anda cari :7". The output shows the sorted array: "Data diurutkan : 1 2 4 5 7 8 9". Finally, it states: "Data ditemukan pada index ke-4".

Overlaid on the screenshot is a Notepad++ window titled "Ma". The window contains the following text:

```

Nama : D'sharlendita Febianda Aurelia
NIM  |: 2311102069
Kelas : IF 11 B

```

The Notepad++ status bar at the bottom indicates "Ln 2, Col 7", "73 characters", "100%", "Window", and "UTF-8".

Deksripsi:

Program ini menggunakan algoritma Binary Search. Array `data` berisi 7 elemen {1, 8, 2, 5, 4, 9, 7} akan diurutkan terlebih dahulu menggunakan Selection Sort. Setelah data diurutkan, pengguna diminta untuk memasukkan nilai yang dicari. Algoritma Binary Search akan mencari nilai tersebut. Jika nilai ditemukan, program mencetak indeksinya. Jika tidak, program mencetak pesan bahwa nilai tidak ditemukan.

C. Unguided

Unguided 1

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

Source Code :

```
// D'sharlendita Febianda Aurelia
// 2311102069

#include <iostream>
#include <string>

using namespace std;

struct IndexedChar {
    char character;
    int index;
};

void selection_sort(IndexedChar* indexed_kalimat, int n) {
    int min, i, j;
    for (i = 0; i < n - 1; i++) {
        min = i;
        for (j = i + 1; j < n; j++) {
            if (indexed_kalimat[j].character <
indexed_kalimat[min].character) {
                min = j;
            }
        }
        swap(indexed_kalimat[i], indexed_kalimat[min]);
    }
}

void binarysearch(IndexedChar* indexed_kalimat, int n, char cari)
{
    int awal = 0, akhir = n - 1, tengah;
    int found_indices[100];
    int found_count = 0;
    bool found = false;

    while (awal <= akhir) {
```



```

        tengah = (awal + akhir) / 2;
        if (indexed_kalimat[tengah].character == cari) {
            found = true;
            found_indices[found_count++] =
indexed_kalimat[tengah].index;
            int left = tengah - 1;
            while (left >= awal &&
indexed_kalimat[left].character == cari) {
                found_indices[found_count++] =
indexed_kalimat[left].index;
                left--;
            }
            int right = tengah + 1;
            while (right <= akhir &&
indexed_kalimat[right].character == cari) {
                found_indices[found_count++] =
indexed_kalimat[right].index;
                right++;
            }
            break;
        } else if (indexed_kalimat[tengah].character < cari) {
            awal = tengah + 1;
        } else {
            akhir = tengah - 1;
        }
    }

    if (found) {
        for (int i = 0; i < found_count - 1; ++i) {
            for (int j = 0; j < found_count - i - 1; ++j) {
                if (found_indices[j] > found_indices[j + 1]) {
                    swap(found_indices[j], found_indices[j + 1]);
                }
            }
        }
    }

    cout << "\nHuruf ditemukan pada index ke- ";
    for (int i = 0; i < found_count; i++) {
        cout << found_indices[i];
        if (i < found_count - 1) {
            cout << ", ";
        }
    }
}

```

```

        cout << endl;
    } else {
        cout << "\nHuruf tidak ditemukan\n";
    }
}

int main() {
    cout << "\tBINARY SEARCH FOR SEARCHING ALPHABET" << endl;
    cout << "\nMasukkan sebuah kalimat: ";
    string kalimat;
    getline(cin, kalimat);

    cout << "Masukkan huruf yang ingin anda cari: ";
    char cari;
    cin >> cari;

    int n = kalimat.length();
    IndexedChar indexed_kalimat[100];

    for (int i = 0; i < n; ++i) {
        indexed_kalimat[i].character = kalimat[i];
        indexed_kalimat[i].index = i;
    }

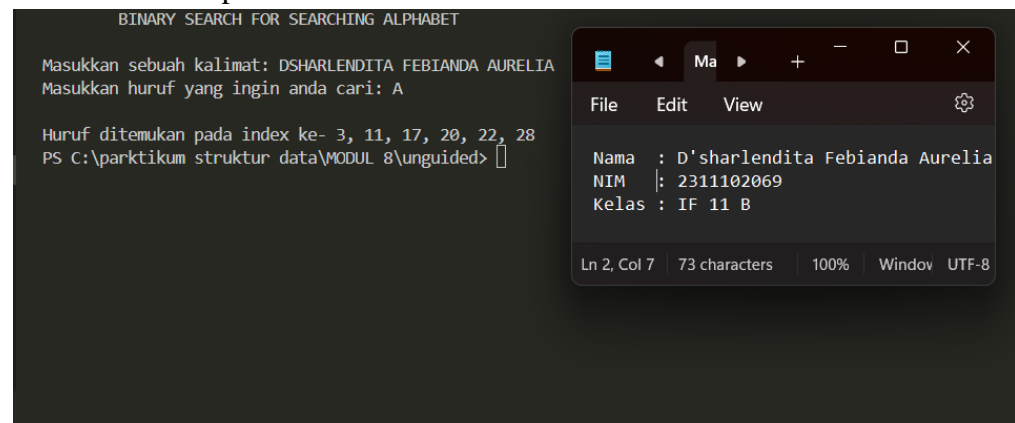
    selection_sort(indexed_kalimat, n);

    binarysearch(indexed_kalimat, n, cari);

    return EXIT_SUCCESS;
}

```

Screenshots Output:



Deskripsi:

Program ini berfungsi untuk mencari sebuah huruf dalam kalimat menggunakan binary search. Pengguna akan diminta untuk memasukkan kalimat dan huruf yang dicari. Setiap karakter dalam kalimat disimpan dalam struktur `IndexedChar` yang mencatat karakter beserta indeks aslinya. Setelah itu, binary search digunakan untuk menemukan huruf yang dicari. Jika ditemukan, indeks-indeks huruf tersebut akan ditampilkan. Jika tidak ditemukan, program akan menampilkan pesan bahwa huruf tidak ditemukan.

Unguided 2

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

Source Code:

```
// D'sharlendita Febianda Aurelia
// 2311102069

#include <iostream>
#include <string>

using namespace std;

bool is_vowel(char c) {
    c = tolower(c);
    return (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');
}

int count_vowels(const string& kalimat) {
    int count = 0;
    for (char c : kalimat) {
        if (is_vowel(c)) {
            count++;
        }
    }
    return count;
}

int main() {
    string kalimat;
```

```

cout << "\tMENGHITUNG HURUF VOKAL DALAM KALIMAT" << endl;

cout << "\nMasukkan sebuah kalimat: ";
getline(cin, kalimat);

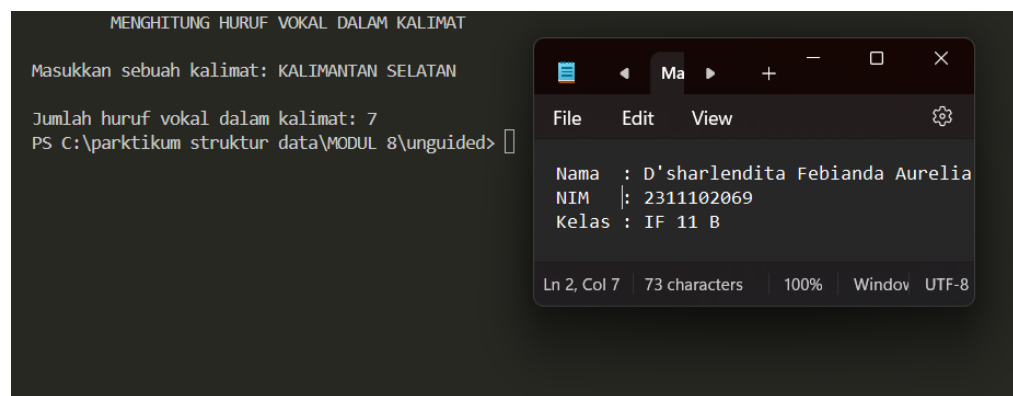
int vowel_count = count_vowels(kalimat);

cout << "\nJumlah huruf vokal dalam kalimat: " << vowel_count
<< endl;

return EXIT_SUCCESS;
}

```

Screenshot Output:



Deskripsi:

Program ini menghitung jumlah huruf vokal dalam kalimat yang dimasukkan oleh pengguna. Pertama, pengguna akan diminta memasukkan sebuah kalimat. Kemudian program akan menghitung jumlah huruf vokal (a, i, u, e, o) dalam kalimat tersebut.

Unguided 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:

```
// D'sharlendita Febianda Aurelia
// 2311102069

#include <iostream>
#include <cstdlib> // Untuk menggunakan konstanta EXIT_SUCCESS
                  // untuk mengakhiri program
#include <iomanip>

using namespace std;

int sequential_search_count(int data[], int size, int target) {
    int count = 0;
    for (int i = 0; i < size; i++) {
        if (data[i] == target) {
            count++;
        }
    }
    return count;
}

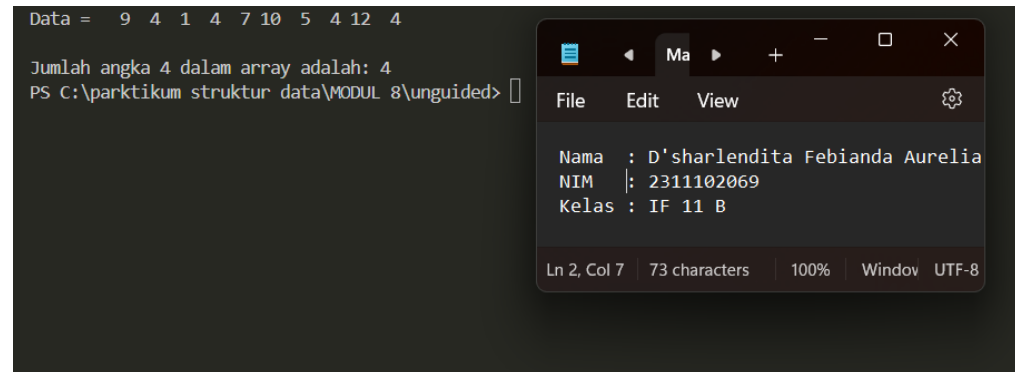
int main() {
    int data[] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int size = sizeof(data) / sizeof(data[0]);
    int target = 4;

    int count = sequential_search_count(data, size, target);

    cout << "Data = ";
    for(int x = 0; x<10; x++)
        cout<<setw(3)<<data[x];
    cout << endl << endl;
    cout << "Jumlah angka " << target << " dalam array adalah: "
    << count << endl;

    return EXIT_SUCCESS;
}
```

Screenshot Output:



The screenshot shows a terminal window on the left and a text editor window on the right. The terminal window displays the following text:

```
Data = 9 4 1 4 7 10 5 4 12 4  
Jumlah angka 4 dalam array adalah: 4  
PS C:\parktikum struktur data\MODUL 8\unguided>
```

The text editor window, titled 'Ma', shows the following text:

```
Nama : D'sharlendita Febianda Aurelia  
NIM  : 2311102069  
Kelas : IF 11 B
```

The text editor window also shows a status bar at the bottom: 'Ln 2, Col 7 | 73 characters | 100% | Window | UTF-8'.

Deskripsi:

Program ini adalah program yang melakukan pencarian menggunakan Sequential Search pada array untuk mencari berapa kali suatu nilai tertentu muncul di dalamnya.

D. Kesimpulan

Praktikum Modul Algoritma Searching memberikan pemahaman mendalam mengenai Sequential Search dan Binary Search serta penggunaannya dalam pemrograman. Melalui praktikum ini, saya memahami konsep dasar dari kedua algoritma pencarian ini.

Sequential Search adalah metode pencarian yang sederhana dan efektif untuk data yang tidak terurut atau berukuran kecil. Algoritma ini bekerja dengan memeriksa setiap elemen dalam urutan hingga nilai yang dicari ditemukan.

Sebaliknya, Binary Search adalah metode pencarian yang efisien untuk data yang sudah terurut. Algoritma ini menggunakan pendekatan "divide and conquer", dengan membagi data menjadi dua bagian dan membandingkan nilai tengah untuk menentukan arah pencarian berikutnya. Hal ini memungkinkan pencarian dilakukan lebih cepat dibandingkan dengan Sequential Search.

E. Referensi

- Asisten Praktikum. (2024, 29 Mei). “Modul 8 Algoritma Searching”. Diakses pada 29 Mei 2024, dari Learning Management System. 2024
- Religia, Yoga. (2019, 17 Juli). Analisis Algoritma Sequential Search dan Binary Search Pada Big Data. Diakses pada 04 Juni 2024, dari <https://jurnal.pelitabangsa.ac.id/index.php/pelitatekno/article/view/232/184>
- Yanti, F., Eriana, E, S. (2024). Algoritma Sorting dan Searching. Diakses pada 04 Juni 2024, dari <https://repository.penerbiteureka.com/media/publications/568014-algoritma-sorting-dan-searching-3ace29ad.pdf>
- Wafa, Muhammad. (2021, 7 Desember). Binary Search. Diakses pada 04 Juni 2024, dari <https://mikirinkode.com/binary-search/>
- Wafa, Muhammad. (2021, 6 Desember). Sequential Search – Algoritma Pencarian. Diakses pada 04 Juni 2024, dari <https://mikirinkode.com/sequential-search/>