

**LAPORAN PRAKTIKUM
STRUKTUR DATA DAN ALGORITMA
MODUL VIII
ALGORITMA SEARCHING**



**Disusun oleh:
Rizal Dwi Anggoro
2311102034**

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

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

BAB I

TUJUAN PRAKTIKUM

A. 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

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) $i \leftarrow 0$
- 2) $ketemu \leftarrow false$
- 3) Selama (tidak $ketemu$) dan ($i \leq N$) kerjakan baris 4
- 4) Jika ($Data[i] = x$) maka $ketemu \leftarrow true$, jika tidak $i \leftarrow i + 1$
- 5) Jika ($ketemu$) maka i adalah indeks dari data yang dicari, jika tidak data tidak ditemukan.

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 III

GUIDED

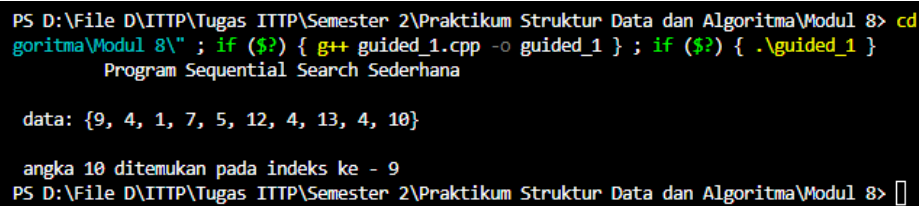
1. Guided 1

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;
```

```
}  
    return 0;  
}
```

Screenshoot program



```
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> cd  
goritma\Modul 8\" ; if ($?) { g++ guided_1.cpp -o guided_1 } ; if ($?) { .\guided_1 }  
Program Sequential Search Sederhana  
  
data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}  
  
angka 10 ditemukan pada indeks ke - 9  
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> █
```

Deskripsi program

Kode C++ sederhana ini melakukan pencarian sequential untuk menemukan angka 10 dalam array. Program mendeklarasikan array data dengan 10 elemen dan variabel cari dengan nilai 10. Menggunakan loop for, program memeriksa setiap elemen dalam array. Jika elemen yang dicari ditemukan, program menandai ketemu sebagai true dan menghentikan loop. Akhirnya, program mencetak hasil pencarian, menunjukkan apakah angka ditemukan dan di indeks berapa, atau menyatakan bahwa angka tidak ditemukan jika tidak ada yang cocok.

2. Guided 2

Source code

```
#include <iostream>  
#include <iomanip>  
using namespace std;  
  
// Deklarasi array dan variabel untuk pencarian  
int arrayData[7] = {1, 8, 2, 5, 4, 9, 7};  
int cari;
```

```

void selection_sort(int arr[], int n) //buat ngurutin datanya
{
    int temp, min;
    for (int i = 0; i < n - 1; i++)
    {
        min = i;
        for (int j = i + 1; j < n; j++)
        {
            if (arr[j] < arr[min])
            {
                min = j;
            }
        }
        // Tukar elemen
        temp = arr[i];
        arr[i] = arr[min];
        arr[min] = temp;
    }
}

void binary_search(int arr[], int n, int target)
{
    int awal = 0, akhir = n - 1, tengah, b_flag = 0;
    while (b_flag == 0 && awal <= akhir)
    {
        tengah = (awal + akhir) / 2;
        if (arr[tengah] == target)
        {
            b_flag = 1;
            break;
        }
        else if (arr[tengah] < target)
        {
            awal = tengah + 1;
        }
    }
}

```

```

        }
        else
        {
            akhir = tengah - 1;
        }
    }
    if (b_flag == 1)
        cout << "\nData ditemukan pada index ke-" << tengah <<
endl;
    else
        cout << "\nData tidak ditemukan\n";
}
int main()
{
    cout << "\tBINARY SEARCH" << endl;
    cout << "\nData awal: ";
    // Tampilkan data awal
    for (int x = 0; x < 7; x++)
    {
        cout << setw(3) << arrayData[x];
    }
    cout << endl;
    cout << "\nMasukkan data yang ingin Anda cari: ";
    cin >> cari;
    // Urutkan data dengan selection sort
    selection_sort(arrayData, 7);
    cout << "\nData diurutkan: ";
    // Tampilkan data setelah diurutkan
    for (int x = 0; x < 7; x++)
    {
        cout << setw(3) << arrayData[x];
    }
    cout << endl;
    // Lakukan binary search

```



```
        binary_search(arrayData, 7, cari);  
        return 0;  
    }
```

Screenshoot program

```
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> cd  
goritma\Modul 8\" ; if ($?) { g++ guided_2.cpp -o guided_2 } ; if ($?) { .\guided_2 }  
        BINARY SEARCH  
  
Data awal:  1 8 2 5 4 9 7  
  
Masukkan data yang ingin Anda cari: 9  
  
Data diurutkan:  1 2 4 5 7 8 9  
  
Data ditemukan pada index ke-6  
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> █
```

Deskripsi program

Kode C++ sederhana ini mengurutkan array menggunakan selection sort dan kemudian mencari nilai tertentu menggunakan binary search. Array arrayData berisi 7 elemen, dan variabel cari menyimpan nilai yang akan dicari. Fungsi selection_sort mengurutkan array dengan mencari dan menukar elemen terkecil. Setelah array diurutkan, fungsi binary_search mencari nilai cari dengan membagi array dan mempersempit pencarian sampai nilai ditemukan atau tidak. Program menampilkan data awal, meminta input pengguna untuk nilai yang dicari, mengurutkan array, menampilkan array yang telah diurutkan, dan melakukan pencarian biner, kemudian mencetak hasil pencariannya.

LATIHAN KELAS - UNGUIDED

1. Unguided 1

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

Source code

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

// 2311102034_Rizal Dwi Anggoro_IF-11-A

bool binarySearch(const vector<char>& sortedChars, char target)
{
    int left = 0;
    int right = sortedChars.size() - 1;

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

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

    return false;
}
```

```

int main() {

    string kalimat = "sudah diinput dari sononya";
    char huruf;

    cout << "Kalimatnya : Sudah diinput dari sononya " << endl;
    cout << "Masukkan huruf yang ingin dicari: ";
    cin >> huruf;

    vector<char> chars;
    for (char c : kalimat) {
        // Abaikan spasi
        if (c != ' ') {
            chars.push_back(c);
        }
    }

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

    cout << "Huruf-huruf setelah diurutkan: ";
    for (char c : chars) {
        cout << c << " ";
    }
    cout << endl;

    bool found = binarySearch(chars, huruf);

    if (found) {
        cout << "Huruf '" << huruf << "' ditemukan dalam kalimat."
<< endl;
    } else {
        cout << "Huruf '" << huruf << "' tidak ditemukan dalam
kalimat." << endl;
    }
}

```

```
    return 0;
}
```

Screenshoot program

```
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> cd "%
$?) { g++ unguided1.cpp -o unguided1 } ; if ($?) { .\unguided1 }
Kalimatnya : Sudah diinput dari sononya
Masukkan huruf yang ingin dicari: r
Huruf-huruf setelah diurutkan: a a d d d h i i n n n o o p r s s t u u y
Huruf 'r' ditemukan dalam kalimat.
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> []
```

Deskripsi program

Kode C++ diatas yaitu mengimplementasikan pencarian biner untuk menemukan huruf dalam kalimat tetap "sudah diinput dari sononya". Pengguna diminta memasukkan huruf yang dicari, kemudian program membuat vector huruf dari kalimat, mengabaikan spasi. Setelah mengurutkan huruf-huruf dengan sort, program menggunakan binarySearch untuk mencari huruf tersebut. Hasil huruf yg dicari akan ditampilkan, menunjukkan apakah huruf ditemukan atau tidak dalam kalimat.

2. Unguided 2

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

Source code

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

// 2311102034_Rizal Dwi Anggoro_IF-11-A
```

```
bool vokal(char huruf) {
    huruf = tolower(huruf);
    return (huruf == 'a' || huruf == 'e' || huruf == 'i' || huruf
== 'o' || huruf == 'u');
}

int itungvokal(const string &kalimat) {
    int count = 0;
    for (char huruf : kalimat) {
        if (vokal(huruf)) {
            count++;
        }
    }
    return count;
}

int main() {
    string sentence;
    cout << "Input Kalimat: ";
    getline(cin, sentence);

    int vokal2 = itungvokal(sentence);
    cout << "Huruf Vokal Yang Ada Pada Kalimat " << sentence << "
: "<< vokal2 << endl;

    return 0;
}
```

Screenshoot program

```
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> cd
goritma\Modul 8\" ; if ($?) { g++ unguided2.cpp -o unguided2 } ; if ($?) { .\unguided2 }
Input Kalimat: Kalimasada
Huruf Vokal Yang Ada Pada Kalimat Kalimasada : 5
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> █
```

Deskripsi program

Kode C++ ini bertujuan untuk menghitung jumlah huruf vokal dalam sebuah kalimat yang diinput oleh pengguna. Fungsi vokal memeriksa apakah sebuah karakter adalah huruf vokal (**a, e, i, o, u**), dengan mengubahnya menjadi huruf kecil terlebih dahulu. Fungsi **itungvokal** menghitung jumlah huruf vokal dalam sebuah string dengan menggunakan fungsi **vokal** untuk setiap karakter dalam string tersebut. Di dalam fungsi **main**, program meminta pengguna untuk memasukkan sebuah kalimat, lalu menghitung dan menampilkan jumlah huruf vokal dalam kalimat tersebut.

3. Unguided 3

Soal : 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;

//2311102034_Rizal Dwi Anggoro_IF-11-A

int main(){
    int n = 10;
    int data[n] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int cari = 4;
    int ketemu = 0;
    int i;
```

```

    for (i = 0; i < n; i++){
        if(data[i] == cari){
            ketemu++;
        }
    }

    cout << "Sequential Search\n";
    cout << "DATA : {9, 4, 1, 4, 7, 10, 5, 4, 12, 4}\n";

    if (ketemu){
        cout << "Angka " << cari << " Ada Sebanyak " << ketemu
    << endl;
    }else{
        cout << "data tidak ditemukan\n" ;
    }

    return 0;
}

```

Screenshoot program

```

PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8> cd
goritma\Modul 8\ ; if ($?) { g++ unguided3.cpp -o unguided3 } ; if ($?) { .\unguided3 }
Sequential Search
DATA : {9, 4, 1, 4, 7, 10, 5, 4, 12, 4}
Angka 4 Ada Sebanyak 4
PS D:\File D\ITTP\Tugas ITTP\Semester 2\Praktikum Struktur Data dan Algoritma\Modul 8>

```

Deskripsi program

Kode C++ ini melakukan pencarian sequential untuk menghitung berapa kali angka tertentu muncul dalam sebuah array. Dalam program ini, terdapat sebuah array data yang berisi 10 angka dan variabel cari yang diset ke 4. Program menggunakan loop for untuk memeriksa setiap elemen dalam array. Jika elemen

tersebut sama dengan nilai cari, variabel ketemu akan ditambahkan 1. Setelah loop selesai, program mencetak jumlah kemunculan angka yang dicari dalam array. Jika angka ditemukan, program menampilkan berapa kali angka tersebut muncul.

BAB IV

KESIMPULAN

Melalui praktikum mata kuliah Struktur Data dan Algoritma pada materi Algoritma Searching, saya mampu memahami dan mengimplementasikan berbagai metode pencarian data, seperti Sequential Search dan Binary Search. Praktikum ini membantu dalam mengevaluasi efisiensi dan kompleksitas masing-masing algoritma, serta aplikasi praktisnya dalam pemecahan masalah nyata.

DAFTAR PUSTAKA

Assisten Praktikum, “Modul VIII Algoritma Searching”, Gropup WhatsApp, 2024.