

**LAPORAN PRAKTIKUM STRUKTUR DATA DAN
PEMROGRAGAMAN**

**MODUL 8
ALGORITMA SEARCHING**



Disusun oleh :

Rafa Aldhino Fatin

2311102023

IF-11-A

Dosen Pengampu :

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

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

2023

BAB I

TUJUAN PRAKTIKUM

- a. Mahasiswa mampu memahami dan menunjukan beberapa algoritma dan pencarian.
- b. Mahasiswa mampu menunjukan bahwa pencarian merupakan suatu persoalan yang bisa diselesaikan dengan beberapa algoritma yang berbeda.
- c. Mahasiswa mampu memilih algoritma yang paling sesuai untuk menyelesaikan suatu permasalahan pemrograman.

BAB II

DASAR TEORI

SEARCHING

Pencarian (searching) yaitu proses menemukan nilai suatu 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 didalam sebuah array dengan cara mengecek satu per satu pada setiap index baris atau setiap index kolomnya dengan menggunakan teknik perulangan untuk melakukan pencarian data. Terdapat dua 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 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.

b. Binary search

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 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.

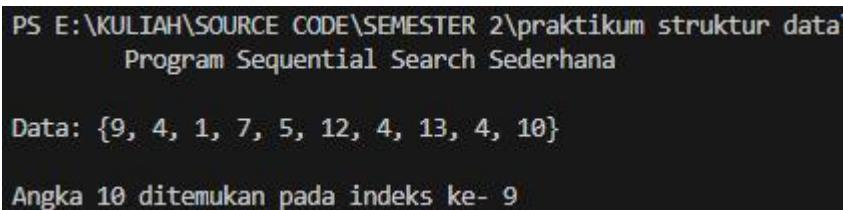
BAB III GUIDED

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 << "\tProgram 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 << cari << " tidak dapat ditemukan pada data." << endl;
    }
    return 0;
}
```

SCREENSHOOT PROGRAM



```
PS E:\KULIAH\SOURCE CODE\SEMESTER 2\praktikum struktur data
    Program Sequential Search Sederhana

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

Angka 10 ditemukan pada indeks ke- 9
```

DESKRIPSI PROGRAM

Program ini mencari angka tertentu dalam array dan melaporkan apakah angka tersebut ditemukan serta di indeks berapa. Program mendeklarasikan sebuah array dengan 10 elemen dan memeriksa apakah angka 10 ada di dalamnya. Jika angka tersebut ditemukan, program akan menampilkan indeks posisi angka tersebut; jika tidak, program akan menginformasikan bahwa angka tersebut tidak ditemukan.

Guided 2

Source code

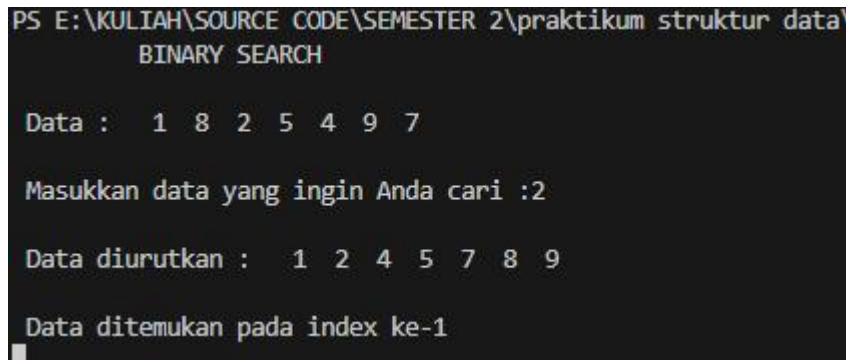
```
#include <iostream>
using namespace std;
#include <conio.h>
#include <iomanip>
int data1[7] = {1, 8, 2, 5, 4, 9, 7};
int cari;
void selection_sort()
{
    int temp, min, i, j;
    for (i = 0; i < 7; i++)
    {
        min = i;
        for (j = i + 1; j < 7; j++)
        {
            if (data1[j] < data1[min])
            {
                min = j;
            }
        }
        temp = data1[i];
        data1[i] = data1[min];
        data1[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 (data1[tengah] == cari)
        {
            b_flag = 1;
            break;
        }
        else if (data1[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 data1 awal
    for (int x = 0; x < 7; x++)
        cout << setw(3) << data1[x];
    cout << endl;
    cout << "\n Masukkan data yang ingin Anda cari :";
    cin >> cari;
    cout << "\n Data diurutkan : ";
    // urutkan data1 dengan selection sort
    selection_sort();
    // tampilkan data1 setelah diurutkan
    for (int x = 0; x < 7; x++)
        cout << setw(3) << data1[x];
    cout << endl;
    binarysearch();
    _getche();
    return EXIT_SUCCESS;
}

```

SCREENSHOOT PROGRAM



```

PS E:\KULIAH\SOURCE CODE\SEMESTER 2\praktikum struktur data
BINARY SEARCH

Data :  1  8  2  5  4  9  7

Masukkan data yang ingin Anda cari :2

Data diurutkan :  1  2  4  5  7  8  9

Data ditemukan pada index ke-1

```

DESKRIPSI PROGRAM

Program tersebut menggunakan algoritma pengurutan seleksi (selection sort) dan pencarian biner (binary search) untuk mencari nilai dalam sebuah array. Program dimulai dengan mendeklarasikan array berisi 7 elemen, lalu meminta pengguna memasukkan nilai yang ingin dicari. Array diurutkan terlebih dahulu dengan algoritma selection sort, kemudian algoritma binary search digunakan untuk mencari nilai yang dimasukkan oleh pengguna dalam array yang telah diurutkan. Program akan menginformasikan apakah nilai tersebut ditemukan dan di indeks berapa.

BAB III

UNGUIDED

UnGuided 1

Source code

```
#include <iostream>
#include <algorithm>
#include <string>
using namespace std;
// Fungsi binary search untuk mencari huruf dalam string
int binarySearch(const string &str, char target)
{
    int left = 0;
    int right = str.length() - 1;
    while (left <= right)
    {
        int mid = left + (right - left) / 2;
        if (str[mid] == target)
        {
            return mid; // huruf ditemukan
        }
        else if (str[mid] < target)
        {
            left = mid + 1;
        }
        else
        {
            right = mid - 1;
        }
    }
    return -1; // huruf tidak ditemukan
}

int main()
{
    string kalimat;
    char huruf;

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

    // Mengurutkan kalimat terlebih dahulu agar binary search dapat bekerja
    sort(kalimat.begin(), kalimat.end());

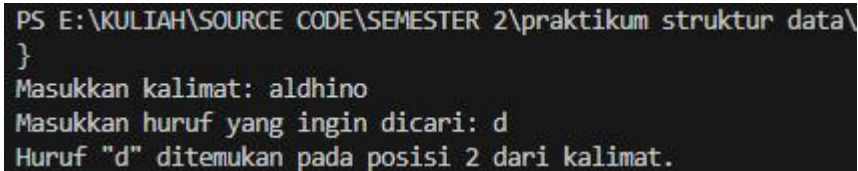
    // Input huruf yang dicari
    cout << "Masukkan huruf yang ingin dicari: ";
    cin >> huruf;

    // Melakukan binary search pada kalimat
    int posisi = binarySearch(kalimat, huruf);

    // Mengecek apakah huruf ditemukan atau tidak
    if (posisi != -1)
    {
        cout << "Huruf \"<\" << huruf << \"<\" ditemukan pada posisi \"<< posisi + 1 << \" dari kalimat.\" <<
endl;
    }
}
```

```
}  
else  
{  
cout << "Huruf \"" << huruf << "\"" tidak ditemukan  
dalam kalimat." << endl;  
}  
return 0;  
}
```

SCREENSHOOT PROGRAM



```
PS E:\KULIAH\SOURCE CODE\SEMESTER 2\praktikum struktur data\  
{  
Masukkan kalimat: aldhino  
Masukkan huruf yang ingin dicari: d  
Huruf "d" ditemukan pada posisi 2 dari kalimat.
```

DESKRIPSI PROGRAM

Kode ini merupakan implementasi fungsi binary search untuk mencari posisi suatu huruf dalam sebuah string (kalimat). Fungsi ini bekerja dengan cara membagi string menjadi dua bagian secara berulang, dan membandingkan huruf pada titik tengah dengan huruf yang dicari. Jika huruf ditemukan, posisinya dikembalikan. Jika huruf tidak ditemukan, fungsi ini menandakan bahwa huruf tidak ada dalam string.

UnGuided 2

Source code

```
#include <iostream>
#include <string>
#include <cctype> // Untuk fungsi isalpha dan tolower
using namespace std;
// Fungsi untuk menghitung jumlah huruf vokal dalam sebuah kalimat
int hitungVokal(const string &kalimat)
{
    int jumlahVokal = 0;
    for (char huruf : kalimat)
    { // Mengonversi huruf menjadi huruf kecil
        char lowerHuruf = tolower(huruf);
        // Memeriksa apakah huruf merupakan huruf vokal
        if (lowerHuruf == 'a' || lowerHuruf == 'e' || lowerHuruf == 'i' || lowerHuruf == 'o' || lowerHuruf
== 'u')
        {
            jumlahVokal++;
        }
    }
    return jumlahVokal;
}
int main()
{
    string kalimat;
    // Input kalimat dari pengguna
    cout << "Masukkan sebuah kalimat: ";
    getline(cin, kalimat);

    // Menghitung jumlah huruf vokal dalam kalimat
    int jumlahVokal = hitungVokal(kalimat);

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

    return 0;
}
```

SCREENSHOOT PROGRAM

```
PS E:\KULIAH\SOURCE CODE\SEMESTER 2\praktikum struktur data\
}
Masukkan sebuah kalimat: dhino
Jumlah huruf vokal dalam kalimat adalah: 2
```

DESKRIPSI PROGRAM

Kode ini menghitung jumlah huruf vokal dalam sebuah kalimat. Fungsi `hitungVokal` mengulangi setiap karakter dalam kalimat, mengubahnya menjadi huruf kecil, dan memeriksa apakah karakter tersebut merupakan huruf vokal ('a', 'e', 'i', 'o', 'u'). Jika ya, nilai penghitung vokal bertambah 1. Fungsi `main` meminta pengguna untuk memasukkan kalimat, menghitung jumlah vokal dengan fungsi `hitungVokal`, dan menampilkan hasilnya.

UnGuided 3

Source code

```
#include <iostream>
using namespace std;
// Fungsi untuk mencari jumlah kemunculan suatu angka dalam array dengan Sequential Search
int hitungAngka(const int data[], int ukuran, int angka)
{
    int jumlah = 0;
    for (int i = 0; i < ukuran; ++i)
    {
        if (data[i] == angka)
        {
            jumlah++;
        }
    }
    return jumlah;
}

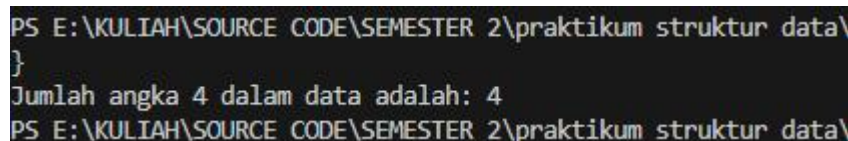
int main()
{
    const int ukuran = 10;
    int data[ukuran] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int angkaYangDicari = 4;

    // Menghitung jumlah kemunculan angka 4 dalam data menggunakan Sequential Search
    int jumlahAngka4 = hitungAngka(data, ukuran, angkaYangDicari);

    // Menampilkan hasil
    cout << "Jumlah angka 4 dalam data adalah: " << jumlahAngka4 << endl;

    return 0;
}
```

SCREENSHOOT PROGRAM



```
PS E:\KULIAH\SOURCE CODE\SEMESTER 2\praktikum struktur data\
}
Jumlah angka 4 dalam data adalah: 4
PS E:\KULIAH\SOURCE CODE\SEMESTER 2\praktikum struktur data\
```

DESKRIPSI PROGRAM

Kode ini menghitung kemunculan angka 4 dalam array menggunakan algoritma Sequential Search. Fungsi `hitungAngka` mengulangi setiap elemen dalam array, membandingkannya dengan 4, dan menambah penghitung jika sama. Fungsi `main` mendefinisikan array dan angka yang dicari, memanggil `hitungAngka`, dan menampilkan hasilnya.

BAB V

KESIMPULAN

Modul 8 tentang Algoritma Pencarian mencakup dua algoritma pencarian utama: pencarian urutan dan pencarian biner. Pencarian sekuensial adalah metode pencarian sederhana yang memeriksa setiap elemen satu per satu, dan cocok untuk data yang tidak diurutkan, namun kurang efisien untuk kumpulan data besar yang waktu pencariannya adalah $O(n)$. Di sisi lain, pencarian biner, yang memerlukan data yang diurutkan, bekerja dengan membagi data menjadi dua bagian, lebih efisien dengan waktu pencarian $O(\log n)$, dan lebih efektif untuk kumpulan data besar. Modul ini berisi contoh implementasi kedua algoritma dalam bahasa C++, mendemonstrasikan pencarian elemen dalam array dan menggunakan pengurutan pilihan sebelum pencarian biner.

DAFTAR PUSTAKA

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

<https://www.kodingakademi.id/pengenalan-searching-dalam-c/>

<https://www.duniailkom.com/latihan-kode-program-cpp-pencarian-data-array-searching/>