LAPORAN PRAKTIKUM STRUKTUR DATA DAN ALGORITMA

MODUL VIII
ALGORITMA SEARCHING

Disusun Oleh :

NAMA : FAISAL KHOIRUDDIN

NIM : 2311102046

Dosen

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

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

A. Dasar Teori

Pencarian (Searching) yaitu proses menemukan suatu nilai 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 di dalam sebuah array dengan cara mengecek satu persatu pada setiap index baris atau setiap index kolomnya dengan menggunakan teknik perulangan untuk melakukan pencarian data. Terdapat 2 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 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:

1. Membandingkan setiap elemen pada array satu per satu secara berurut.
2. Proses pencarian dimulai dari indeks pertama hingga indeks terakhir.
3. Proses pencarian akan berhenti apabila data ditemukan. Jika hingga akhir array data masih juga tidak ditemukan, maka proses pencarian tetap akan dihentikan.
4. 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 ← false

3) Selama (tidak ketemu) dan (i <= N) kerjakan baris 4

4) Jika (Data[i] = x) maka ketemu ← true, jika tidak i ← i + 1

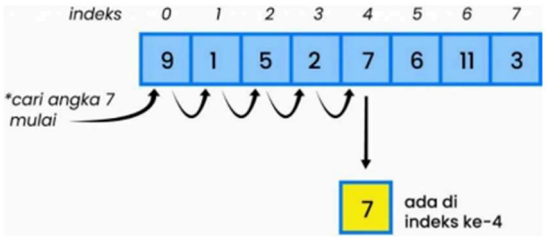
5) Jika (ketemu) maka i adalah indeks dari data yang dicari, jika tidak data tidak ditemukan.

Di bawah ini merupakan fungsi untuk mencari data menggunakan pencarian sekuensial.

```
int SequentialSearch (int x)
{
    int i = 0;
    bool ketemu = false;
    while ((!ketemu) && (i < Max)){
        if (Data[i] == x)
            ketemu = true;
       else
            i++;
    }
        if  (ketemu)
            return i;
        else
            return -1;
    }
```

Fungsi diatas akan mengembalikan indeks dari data yang dicari. Apabila data tidak ditemukan maka fungsi diatas akan mengembalikan nilai –1. Contoh dari Sequential Search, yaitu:

Int A[8] = {9,1,5,2,7,6,11,3}



Misalkan, dari data di atas angka yang akan dicari adalah angka 7 dalam array A, maka proses yang akan terjadi yaitu:

- Pencarian dimulai pada index ke-0 yaitu angka 9, kemudian dicocokkan dengan angka yang akan dicari, jika tidak sama maka pencarian akan dilanjutkan ke index selanjutnya.

- Pada index ke-1, yaitu angka 1, juga bukan angka yang dicari, maka Praktikum Struktur Data dan Algoritma 2 pencarian akan dilanjutkan pada index selanjutnya.
- Pada index ke-2 dan index ke-3 yaitu angka 5 dan 2, juga bukan angka yang dicari, sehingga pencarian dilanjutkan pada index selanjutnya.
- Pada index ke-4 yaitu angka 7 dan ternyata angka 7 merupakan angka yang dicari, sehingga pencarian akan dihentikan dan proses selesai.

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:

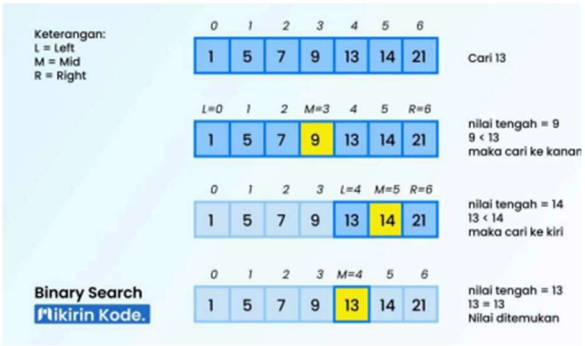
- Data diambil dari posisi 1 sampai posisi akhir N.
- Kemudian data akan dibagi menjadi dua untuk mendapatkan posisi data tengah.
- Selanjutnya data yang dicari akan dibandingkan dengan data yang berada di posisi tengah, apakah lebih besar atau lebih kecil.
- Apabila data yang dicari lebih besar dari 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.

Algoritma pencarian biner dapat dituliskan sebagai berikut :

1) L □ 0
2) R □ N – 1
3) ketemu □ false
4) Selama (L <= R) dan (tidak ketemu) kerjakan baris 5 sampai dengan 8
5) m □ (L + R) / 2
6) Jika (Data[m] = x) maka ketemu □ true
7) Jika (x < Data[m]) maka R □ m – 1
8) Jika (x > Data[m]) maka L □ m + 1
9) Jika (ketemu) maka m adalah indeks dari data yang dicari, jika tidak data tidak ditemukan

Contoh dari Binary Search, yaitu:



- Terdapat sebuah array yang menampung 7 elemen seperti ilustrasi di atas. Nilai yang akan dicari pada array tersebut adalah 13.
- Jadi karena konsep dari binary search ini adalah membagi array menjadi dua bagian, maka pertama tama kita cari nilai tengahnya dulu, total elemen dibagi 2 yaitu 7/2 = 4.5 dan kita bulatkan jadi 4.
- Maka elemen ke empat pada array adalah nilai tengahnya, yaitu angka 9 pada indeks ke 3.
- Kemudian kita cek apakah 13 > 9 atau 13 < 9?
- 13 lebih besar dari 9, maka kemungkinan besar angka 13 berada setelah 9 atau di sebelah kanan. Selanjutnya kita cari ke kanan dan kita dapat mengabaikan elemen yang ada di kiri.
- Setelah itu kita cari lagi nilai tengahnya, didapatlah angka 14 sebagai nilai tengah. Lalu, kita bandingkan apakah 13 > 14 atau 13 < 14?
- Ternyata 13 lebih kecil dari 14, maka selanjutnya kita cari ke kiri.
- Karna tersisa 1 elemen saja, maka elemen tersebut adalah nilai tengahnya. Setelah dicek ternyata elemen pada indeks ke-4 adalah elemen yang dicari, maka telah selesai proses pencariannya.

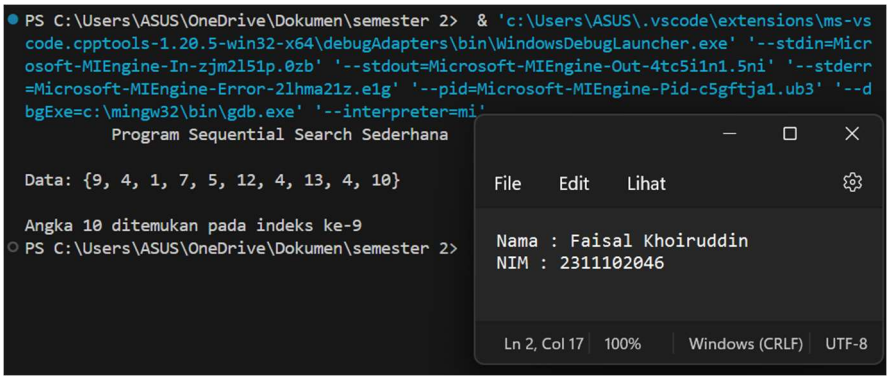
Guided

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 mencari
dari indeks ke 0 dari kiri
    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 << "\nAngka " << cari << "
ditemukan pada indeks ke-" << i << endl;
    } else {
        cout << "\nAngka " << cari << "
tidak dapat ditemukan pada data." << endl;
    }

    return 0;
}
```

Screenshots Output



Deskripsi:

Program tersebut merupakan program sequential search. Program tersebut menampilkan data berupa angka yang sudah disediakan. Program tersebut menampilkan output letak indeks dari angka yang dicari.

- #include <iostream> ➔ merupakan input output stream header yang digunakan sebagai standar input output operasi yang digunakan di c++
- using namespace std; ➔ digunakan untuk mendeklarasikan/ memberitahukan kepada compiler bahwa kita akan menggunakan semua fungsi/class/file yang terdapat dalam namespace std
- int main(){ ➔ merupakan fungsi utama
- int n = 10; ➔ deklarasi variabel n dengan nilai 10
- int data[n] = {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}; ➔
- int cari = 10; ➔ mencari data 10
- bool ketemu = false; ➔ deklarasi variabel Boolean ketemu dengan nilai false
- int i; ➔ deklarasi variabel i dengan tipe data integer
- for (i = 0; i < n; i++) { ➔ perulangan untuk setiap indeks dalam array
- if(data[i] == cari) { ➔ percabangan if jika i = cari
- ketemu = true; ➔ ketemu sama dengan true

- break; ➔ mengakhiri perulangan karena cari sudah ketemu
- cout << "\t Program Sequential Search Sederhana\n" << endl; ➔ cout menampilkan Program Sequential Search Sederhana
- cout << "Data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}"<< endl; ➔ cout menampilkan array Data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}
- if (ketemu){ ➔ percabangan if jika ketemu
- cout << "\nAngka " << cari << " ditemukan pada indeks ke-" << i << endl; ➔ cout menampilkakn angka ditemukan pada ideks ke dan memanggil variabel i
- } else { ➔ jika ketemu yaitu false
- cout << "\nAngka " << cari << " tidak dapat ditemukan pada data." << endl; ➔ cout menampilkan angka tidak ditemukan pada dan memanggil variabel cari
- return 0; ➔ program akan mengembalikan (return) nilai 0

2. Guided 2

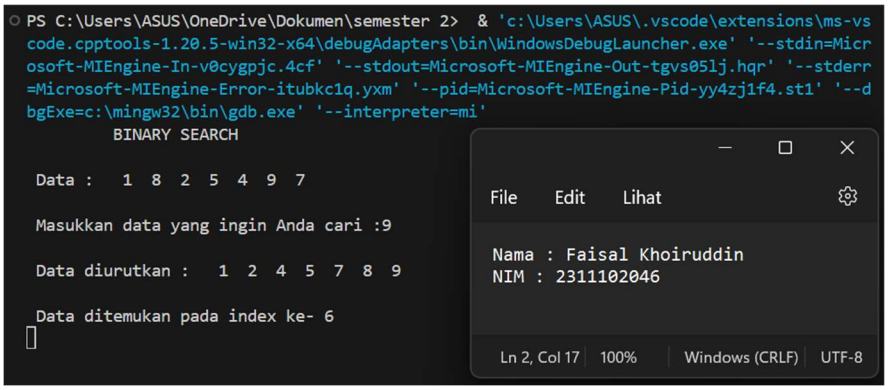
Source Code

```
#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;
        }
}

void binarysearch() //
{
        //searching
        int awal, akhir, tengah, b_flag =
0;
        awal = 0; // awal 0
        akhir = 7; // 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



Deskripsi:

Program tersebut merupakan program binary search. Program tersebut menampilkan data berupa angka yang sudah disediakan. Kemudian pengguna diminta menginputkan angka yang ingin dicari. Program tersebut menampilkan output data yang sudah diurutkan dan letak indeks dari angka yang dicari.

- #include <iostream> ➔ merupakan input output stream header yang digunakan sebagai standar input output operasi yang digunakan di c++

- #include <conio.h> ➔ Merupakan File Header yang berfungsi untuk menampilkan hasil antarmuka kepada pengguna

- #include <iomanip> ➔ Manipulator mengembalikan objek yang, ketika diekstrak dari aliran str
- using namespace std; ➔ digunakan untuk mendeklarasikan/ memberitahukan kepada compiler bahwa kita akan menggunakan semua fungsi/class/file yang terdapat dalam namespace std
- int data[7] = {1, 8, 2, 5, 4, 9, 7}; ➔ deklarasi array data dengan 7 angka
- int cari; ➔ deklarasi variabel cari tipe data integer
- void selection_sort() ➔ mendefinisikan fungdi selection sort
- int temp, min, i, j; ➔ deklarasi variabel temp, min, i, j dengan tipe data integer
- for(i=0; i<7;i++) ➔ perulangan for untuk setiap indeks array
- min = i; ➔ min sama dengan i
- for(j = i+1; j<7; j++) ➔ perulangan for untuk setiap indeks setelah indeks ke- i
- if(data[j]<data[min]) ➔ percabanangan if jika data indeks ke j kurang dari data indeks min
- min=j; ➔ min sama dengan j
- temp = data[i]; ➔ menyimpan adngka pada indeks ke i dalam temp
- data[i] = data[min]; ➔ menyalin angka minimum ke indeks i
- data[min] = temp; ➔ menyalin temp ke indeks elemen minimum
- void binarysearch() ➔ mendefinisikan fungsi binary search
- int awal, akhir, tengah, b_flag = 0; ➔ deklarasi variabel awal, akhir, tengah, b_flag menginialisasi dengan nilai 0
- awal = 0; ➔ awal sama dengan 0
- akhir = 7; ➔ akhir sama dengan 7
- while (b_flag == 0 && awal<=akhir) ➔ perulangan whileselama b_flag == 0 && awal<=akhir
- tengah = (awal + akhir)/2; ➔ menetapkan tengah sebagai indeks tengah antara awal dan akhir

- if(data[tengah] == cari) ➔ percabangan jika angka pada indeks tengah sama dengan cari
- b_flag = 1; ➔ mengubah b_flag menjadi 1
- break; ➔ mengakhiri perulangan karena cari sudah ditemukan
- else if(data[tengah]<cari) ➔ jika angka pada indeks ke tengah lebih kecil dari cari
- awal = tengah + 1; ➔ menetapkan sebagai tengah + 1
- else ➔ jika pada indeks ke- tengah lebih besar dari cari
- akhir = tengah -1; ➔ menetepkan akhir sebagai tengah - 1
- if(b_flag == 1) ➔ percabangan if jika b_flag sama dengan satu
- cout<<"\n Data ditemukan pada index ke- "<< tengah << endl; ➔ cout menampilkan Data ditemukan pada index ke- dan memanggil variabel tengah
- else ➔ jika b_flag sama dengan 0 atau tidak ditemukan
- cout<<"\n Data tidak ditemukan\n"; ➔ cout menampilkan data tidak ditemukan
- int main() ➔ merupakan fungsi utama
- cout << "\t BINARY SEARCH " << endl; ➔ cout menampilkan judul program
- cout << "\n Data : "; ➔ cout menampilkan statement data untuk memasukkan data
- for(int x = 0; x<7; x++) ➔ perulangan for untuk setiap indeks dalam array data
- cout << setw(3) << data[x]; ➔ cout menampilkan angka pada indeks ke- x dengan lebar tiga karakter
- cout << endl; ➔ cout menampilkan baris baru
- cout << "\n Masukkan data yang ingin Anda cari :"; ➔ cout menampilkan statement Masukkan data yang ingin Anda cari :
- cin >> cari; ➔ menerima inputan dari pengguna dan menyimpan dalam variabel cari
- cout << "\n Data diurutkan : "; ➔ cout untuk menampilkan data

diurutkan

- selection_sort(); ➔ memanggil selection_sort(); untuk mengurutkan data
- for(int x = 0; x < 7; x++) ➔ perulangan for setiap indeks dalam array data
- cout<<setw(3)<<data[x]; ➔ cout mencetak angka pada indeks ke- x yang telah diurutkan dengan lebar tiga karakter
- cout<<endl; ➔ cout mengakhiri baris
- binarysearch(); ➔ cout memanggil fungsi binarysearch() untuk mencari array dalam data
- getche(); ➔ menunggu pengguna menekan tombol sebelum program berakhir
- return EXIT_SUCCESS; ➔ mengakhiri fungsi main dan mengembalikan EXIT_SUCCESS

B. Unguided/Tugas

1. Unguided 1

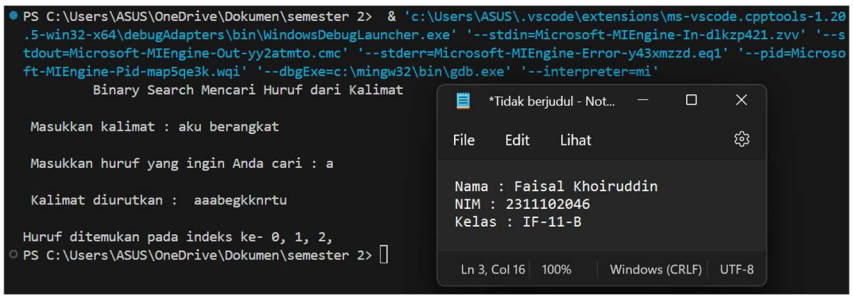
Buatlah sebuah program untuk mencari sebuah huruf pada sebuah kalimat yang sudah di input dengan menggunakan Binary Search!

```
#include <iostream>
#include <vector>
#include <string>

using namespace std;

string data;
char cari;

void selection_sort()
{
    int i, j, min;
    char temp;
    for (i = 0; i < data.length(); i++)
    {
```

```
        min = i;
        for (j = i + 1; j < data.length();
j++)
        {
            if (data[j] < data[min])
            {
                  min = j;
            }
    }
    temp = data[i];
    data[i] = data[min];
    data[min] = temp;
    }
}

void binarysearch(){
    vector<int> foundIndices;
    for(int i = 0; i < data.length();
i++){
        if(data[i] == cari){
            foundIndices.push_back(i);
        }
    }
    if(!foundIndices.empty()){
        cout << "\nHuruf ditemukan pada
indeks ke- ";
        for(int i = 0; i <
foundIndices.size(); i++){
            cout << foundIndices[i]-1 <<
", ";
        }
        cout << "\b\b";
    }
    else {
        cout << "\n Data tidak ditemukan\n
";
    }
}
int main(){
    cout << "\t Binary Search Mencari
Huruf dari Kalimat " << endl;
    cout << "\n Masukkan kalimat : ";
    getline(cin, data);
    cout << "\n Masukkan huruf yang ingin
Anda cari : ";
    cin >> cari;
```

```
    cout << "\n Kalimat diurutkan : ";
    selection_sort();
    cout << data << endl;
    binarysearch();
    return 0;
}
```

Screenshots Output



Deskripsi

Program tersebut merupakan binary search. Program tersebut pengguna diminta menginputkan kalimat. Kemudian pengguna diminta menginputkan huruf yang akan dicari. Setelah itu menampilkan output huruf ditemukan pada indeks.

- #include <iostream> ➔ merupakan input output stream header yang digunakan sebagai standar input output operasi yang digunakan di c++
- #include <vector> ➔ Vektor menyimpan elemen jenis tertentu dalam pengaturan linier, dan memungkinkan akses acak cepat ke elemen apa pun.
- #include <string> ➔ Fungsi ini dapat digunakan untuk mencari suatu nilai karakter yang berada dalam suatu nilai string.
- string data; ➔ deklarasi variabel data dengan tipe data string
- char cari; ➔ deklaraasi variabel cari dengan tipe data char
- void selection_sort() ➔ mendefinisikan fungsi selection sort untuk mengurutkan data
- int i, j, min; ➔ deklarasi variabel i, j, min tipe data integer
- char temp; ➔ deklarasi variabel temp dengan tipe data char
- for (i = 0; i < data.length(); i++) ➔ perulangan for untuk setiap karakter string data

- min = i; ➔ min = i
- for (j = i + 1; j < data.length(); j++) ➔ perulangan for untuk setiap karakter setelah karakter ke - i
- if (data[j] < data[min]) ➔ percabangan if jika data indeks j kurang dari data indeks min
- min = j; ➔ j sebagai indeks minimum baru
- temp = data[i]; ➔ menyimpan I ke dalam temp
- data[i] = data[min]; ➔ menyalin karakter minimum ke posisi i
- data[min] = temp; ➔ menyalin temp ke posisi elemen minimum
- void binarysearch(){ ➔ mendefinisikan fungsi binary search untuk mencari karakter cari dalam string data
- vector<int> foundIndices; ➔ deklarasi vector foundIndices untuk menyimpan indeks tempat cari ditemukan
- for(int i = 0; i < data.length(); i++){ ➔ perulangan for untuk setiap karakter
- if(data[i] == cari){ ➔ percabangan if jikakarakter ke I sama dengan dari
- foundIndices.push_back(i); ➔ menambahkan indeks ke i ke foundindices
- if(!foundIndices.empty()){ ➔ percabangan if jika foundIndices tidak kosong
- cout << "\nHuruf ditemukan pada indeks ke- "; ➔ cout menampilkan huruf ditemukan pada indeks ke- i
- for(int i = 0; i < foundIndices.size(); i++){ ➔ perulangan untuk setiap indeks dalam foundIndices
- cout << foundIndices[i]-1 << ", "; ➔ cout mencetak indeks tempat cari ditemukan
- cout << "\b\b"; ➔ menghapus dua karakter terakhir
- else { ➔jika foundIndices kosong
- cout << "\n Data tidak ditemukan\n "; ➔ cout menampilkan data tidak ditemukan
- int main(){ ➔ merupakan fungsi utama
- cout << "\t Binary Search Mencari Huruf dari Kalimat " << endl; ➔ cout menampilkan judul program
- cout << "\n Masukkan kalimat : "; ➔ cout menampilkan masukkan kalimat
- getline(cin, data); ➔ membaca kalimat dari input pengguna

dan menyimpan dalam data

- cout << "\n Masukkan huruf yang ingin Anda cari : "; ➔ cout menampilkan masukkan huruf yang ingin anda cari
- cin >> cari; ➔ menerima inputan dari pengguna dan menyimpan dalam variabel cari
- cout << "\n Kalimat diurutkan : "; ➔ cout menampilkan kalimat diurutkan
- selection_sort(); ➔ memanggil fungsi selection sort
- cout << data << endl; ➔ cout dan memanggil variabel data
- binarysearch(); ➔ memanggil fungsi binary search
- return 0; ➔ program akan mengembalikan (return) nilai 0

2. Unguided 2

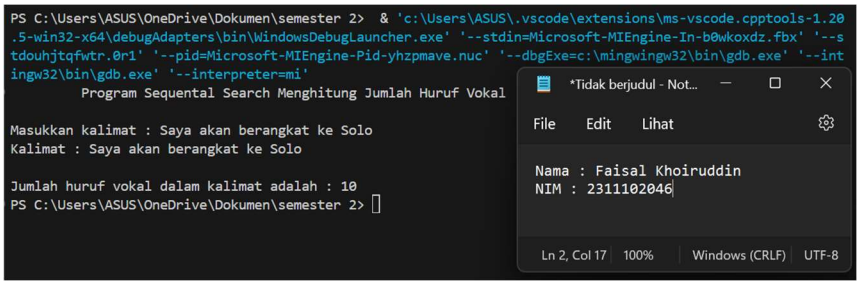
Buatlah sebuah program yang dapat menghitung banyaknya huruf vocal dalam sebuah kalimat!

Source Code

```
#include <iostream>
#include <string>

using namespace std;

int main (){
    string kalimat;
    cout << "\t Program Sequental Search
Menghitung Jumlah Huruf Vokal\n" << endl;
    cout << "Masukkan kalimat : ";

    getline(cin, kalimat);

    char vokal[5] = {'a', 'i', 'u', 'e',
'o'};
    int jumlahVokal = 0;

    for (int i = 0; i < kalimat.length();
i++){
        char huruf = tolower(kalimat[i]);
        for (int j = 0; j < 5; j++){
            if( huruf == vokal[j]){
                jumlahVokal++;
                break;
            }
```

```
        }
    }

    cout << "Kalimat : " << kalimat <<
endl;
    cout << "\nJumlah huruf vokal dalam
kalimat adalah : " << jumlahVokal << endl;

    return 0;
}
```

Screenshots Output



Deskripsi

Program tersebut merupakan sequential search. Program tersebut yaitu menghitung jumlah huruf vokal. Program tersebut meminta pengguna untuk menginputkan kalimat. Kemudian output program tersebut menapilkan kalimat yang telah diinputkan dan jumlah huruf vocal dari kalimat yang telah diinputkan.

- #include <iostream> ➔ merupakan input output stream header yang digunakan sebagai standar input output operasi yang digunakan di c++
- using namespace std; ➔ digunakan untuk mendeklarasikan/ memberitahukan kepada compiler bahwa kita akan menggunakan semua fungsi/class/file yang terdapat dalam namespace std
- int main (){ ➔ merupakan fungsi utama
- string kalimat; ➔ deklarasi variabel kalimat dengan tipr data string
- cout << "\t Program Sequental Search Menghitung Jumlah

Huruf Vokal\n" << endl; ➔ cout menampilkan Sequental Search Menghitung Jumlah Huruf Vokal

- cout << "Masukkan kalimat : "; ➔ cout menampilkan Masukkan kalimat
- getline(cin, kalimat); ➔ membaca kalimat dari input pengguna dan menyimpannya dalam kalimat
- char vokal[5] = {'a', 'i', 'u', 'e', 'o'}; ➔ deklarasi array huruf vocal yang berisi huruf vokal
- int jumlahVokal = 0; ➔ jumlah huruf vokal diatur ke 0
- for (int i = 0; i < kalimat.length(); i++){ ➔ perulangan for untuk setiap karakter dalam kalimat
- char huruf = tolower(kalimat[i]); ➔ memngubah karakter ke- i menjadi huruf kecil dan menyimpan dalam huruf
- for (int j = 0; j < 5; j++){ ➔ perulangam for untuk setiap huruf vocal dalam array vokal
- if(huruf == vokal[j]){ ➔ percabangan if jika huruf sama dengan vocal indeks i
- jumlahVokal++; ➔ jumlahvokal++ (increment)
- break; ➔ mengakhiri perulangan karena huruf sudah ditemhkan sebagai huruf vokal
- cout << "Kalimat : " << kalimat << endl; ➔ cout menampilkan kalimat: dan memanggil variabel kalimat
- cout << "\nJumlah huruf vokal dalam kalimat adalah : " << jumlahVokal << endl; ➔ cout menampilkan nJumlah huruf vokal dalam kalimat adalah
- return 0; ➔ program akan mengembalikan (return) nilai 0

3. Unguided 2

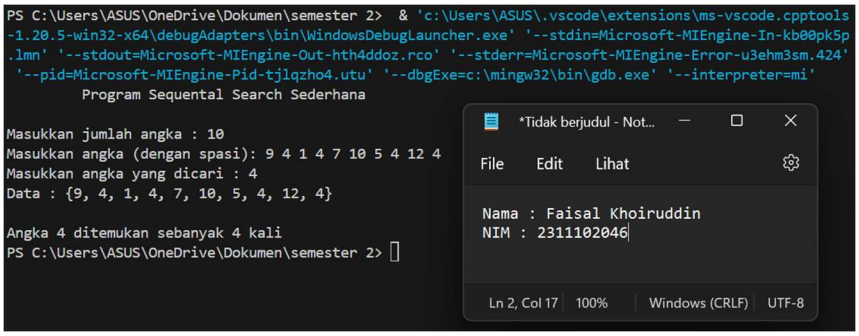
Diketahui data = 9, 4, 1, 4, 7, 10, 5, 4, 12, 4. Hitunglah berapa banyak angka 4 dengan menggunakan algoritma Sequential Search!

Source Code

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

int main(){
    int n;
    cout << "\t Program Sequental Search
```

```
Sederhana\n" << endl;
    cout << "Masukkan jumlah angka : ";
    cin >> n;

    int data[n];
    cout << "Masukkan angka (dengan
spasi): ";
    for (int i = 0; i < n; i++){
        cin >> data[i];
    }

    int cari;
    cout << "Masukkan angka yang dicari :
";
    cin >> cari;

    int jumlah = 0;

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

    cout << "Data : {";
    for (int i = 0; i < n; i++){
        cout << data[i];
        if (i < n - 1){
            cout << ", ";
        }
    }
    cout << "}" << endl;

    if (jumlah > 0){
        cout << "\nAngka " << cari << "
ditemukan sebanyak " << jumlah << " kali"
<< endl;
    } else {
        cout << "\nAngka " << cari << "
tidak ditemukan " << endl;
    }
    return 0;
}
```

Screenshots Output



Deskripsi:

Program tersebut merupakan program sequential search. Program tersebut yaitu menghitung barapa banyak angka dari inputan. Pada program tersebut yaitu pengguna menginputkan angka. Kemudian menginputkan angka yang dicari. Setelah itu output program yaitu menampilkan angka yang telah diinputkan dan angka menghitung angka yang dicari.

- #include <iostream> ➔ merupakan input output stream header yang digunakan sebagai standar input output operasi yang digunakan di c++
- using namespace std; ➔ digunakan untuk mendeklarasikan/ memberitahukan kepada compiler bahwa kita akan menggunakan semua fungsi/class/file yang terdapat dalam namespace std
- int main(){ ➔ merupakan fungdi utama
- int n; ➔ deklarasi variabel n dengan tipe data integer
- cout << "\t Program Sequental Search Sederhana\n" << endl; ➔ cout menampilkan judul program
- cout << "Masukkan jumlah angka : "; ➔ cout menampilkan jumlah angka
- cin >> n; ➔ menemrima inputan dari pengguna dan menyimpan pada variabel n
- int data[n]; ➔ deklarasi array data ukuran n tipe data integer
- cout << "Masukkan angka (dengan spasi): "; ➔ cout menampilkan masukkan angka dengan spasi

- for (int i = 0; i < n; i++){ ➔ perulangan for untuk setiap indeks dalam array data
- cin >> data[i]; ➔ menerima inputan dari pengguna dan menyimpan dalam data indeks ke- i
- int cari; ➔ deklarasi variabel cari dengan nama tipe data integer
- cout << "Masukkan angka yang dicari : "; ➔ cout menampilkan masukkan angka yang dicari
- cin >> cari; ➔ menerima inputan dari pengguna dan menyimpan dalam variabel cari
- int jumlah = 0; ➔ mengatur nilai jumlah sama dengan 0 dengan nama tipe data integer
- for (int i = 0; i < n; i++){ ➔ perulangan untuk setiap indeks dalam array data
- if(data[i] == cari){ ➔ percabangan if jika data indeks ke- i sama dengan cari
- jumlah++; ➔ jumlah++(increment)
- cout << "Data : {"; =➔ cout menampilkan data
- for (int i = 0; i < n; i++){ ➔ perulangan for untuk setiap indeks dalam array
- cout << data[i]; ➔ cout menampilkan angka indeks ke- i
- if (i < n - 1){ ➔ percabangan if jika indeks bukan terakhir
- cout << ", "; ➔ cout menampilkan koma dan spasi setelah angka
- cout << "}" << endl; ➔ cout menampilkan akhir array data
- if (jumlah > 0){ ➔ percabangan jika jumlah lebih dari 0
- cout << "\nAngka " << cari << " ditemukan sebanyak " << jumlah << " kali" << endl; ➔ cout menampilkan angka ditemukan sebanyak dan kali serta memanggil cari dan jumlah
- } else { ➔ jika jumlah sama dengan 0
- cout << "\nAngka " << cari << " tidak ditemukan " << endl; ➔

cout menampilkan angka tidak ditemukan dan memanggil variabel angka

- return 0; ➔ program akan mengembalikan (return) nilai 0 ke

C. Kesimpulan

Pencarian (Searching) yaitu proses menemukan suatu nilai tertentu pada kumpulan data. Hasil pencarian adalah salah satu dari tiga keadaan ini: data ditemukan, data ditemukan lebih dari satu, atau data tidak ditemukan. Terdapat 2 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 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.

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.

D. Referensi

[1] Asisten Praktikum, Struktur Data “QUEUE", WhatsApp, 2024.

[2] TylerMSFT. (n.d.) <iomanip>. diakses dari

https://learn.microsoft.com/id-id/cpp/standard-library/iomanip-functions?view=msvc-170

[3] TylerMSFT. (n.d.) <vector>. diakses dari

https://learn.microsoft.com/id-id/cpp/standard-library/vector-class?view=msvc-170