

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VIII
SEARCHING**



Disusun oleh:
Alfin Ilham Berlianto
2311102047

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

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

MODUL 8 SEARCHING

A. Tujuan

1. Mampu memahami konsep stack pada struktur data dan algoritma.
2. Mampu mengimplementasikan operasi-operasi pada stack.
3. Mampu memecahkan permasalahan dengan sokusi stack.

B. Dasar Teori

Pengertian Searching

Searching merupakan kegiatan mencari data yang akan dibutuhkan. Searching dalam pemrograman dapat dilakukan untuk mencari data yang berada pada memory komputer. Dalam kehidupan sehari-hari kita sering melakukan Searching seperti pada saat mencari data maupun informasi yang ada pada internet. Terdapat macam-macam metode yang dapat digunakan dalam Searching, antara lain:

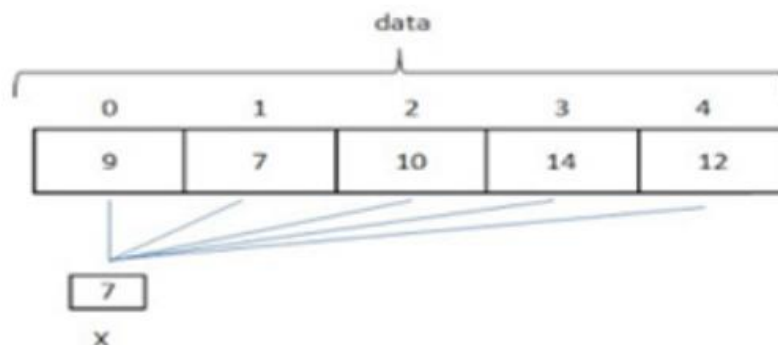
- Sequential Search
- Binary Search

1. Sequential Search

Sequential Search adalah sebuah metode pencarian data dalam array dengan cara membandingkan data yang akan dicari dengan data yang terdapat dalam array secara berurutan. Pencarian data dengan menggunakan metode Sequential Search lebih efektif untuk mencari data dalam posisi tidak teratur atau acak.

Proses Sequential Search dapat dijelaskan sebagai berikut:

1. Menentukan data yang akan dicari.
2. Membaca data array satu per satu secara ekuensial
3. Membaca data mulai dari data pertama sampai dengan data terakhir, kemudian data yang dicari akan dibandingkan dengan masing-masing data yang terdapat dalam array.
 - a. Apabila data yang dicari ditemukan maka kita dapat membuat pernyataan bahwa data telah ditemukan.
 - b. Jika data yang dicari tidak ditemukan maka kita dapat membuat pernyataan bahwa data tidak temuan.



Data yang dicari yaitu 7, lalu disimpan dalam variabel x, kemudian akan dibandingkan satu per

satu secara kuensial terhadap data yang terdapat dalam array. Jika ditemukan data pada array yang sama dengan data yang dicari maka data ditemukan.

Kelebihan dan Kekurangan Sequential Searching yaitu:

- Kelebihan Sequential Searching adalah Sequential Searching lebih mudah dalam proses implementasi dalam pemrograman.
- Kekurangan Sequential Searching adalah data yang terdapat dalam suatu array jumlahnya sangat banyak, maka waktu yang diperlukan dalam membandingkan data yang dicari dengan jumlah data dalam suatu array membutuhkan waktu yang lebih lama.

Salah Satu Contoh Studi Kasus:

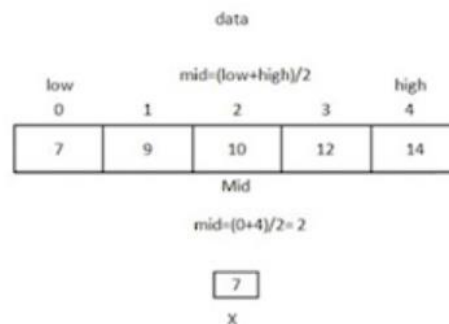
Menemukan data yang akan dicari dalam sebuah array 1 dimensi dimana di dalamnya terdapat data dengan menggunakan metode Sequential Searching. Jika data yang akan dicari dalam array ditemukan, kemudian ditampilkan letak dari indexnya.

2. Binary Search

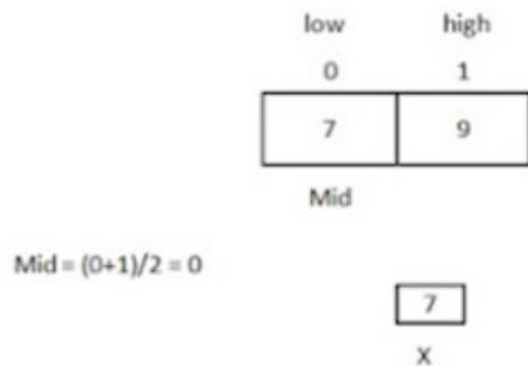
Metode Binary Search adalah suatu metode pencarian data dengan cara mengelompokkan array menjadi bagian-bagian. Binary Search hanya dapat diterapkan pada data yang telah terurut baik ascending maupun descending dalam suatu array.

Proses Binary Search dengan data ascending dan telah urut:

1. Membuat perulangan kemudian menentukan posisi low yaitu posisi yang menandakan index paling rendah kemudian menentukan posisi high, setelah itu mencari posisi mid atau $\text{tengah} = (\text{high} + \text{low})/2$
2. Membandingkan data yang dicari dengan nilai yang berada di posisi mid atau tengah.
3. Jika data yang dicari sama dengan nilai yang ada di posisi mid atau tengah berarti data ditemukan.
4. Jika data yang dicari lebih kecil dari nilai yang terdapat pada mid maka pencarian data akan dilakukan dibagian kiri mid dengan melakukan pembandingan. Dengan kondisi posisi high berubah yaitu $(\text{mid} - 1)$ dan posisi low tetap.
5. Jika data yang dicari lebih besar dari nilai yang terdapat pada mid maka pencarian data akan dilakukan di bagian kanan dari mid dengan posisi low yang berubah yaitu $(\text{mid} + 1)$ dan posisi high tetap.



Setelah menentukan low dan high lalu menentukan mid. perhitungan $\text{mid} = (\text{low} + \text{high})/2$ jadi $\text{mid} = (0+4)/2$, artinya mid berada pada data (2). Data yang dicari kemudian ditampung di variabel x lalu dibandingkan dengan data atau nilai yang berada di mid. data yang dicari adalah $7 < 10$ kemudian pencarian data dilakukan di bagian kiri mid.



Pencarian di bagian kiri mid masih dalam perulangan yang sama, namun indeksnya dipersempit. Karena data yang dicari lebih kecil dari data mid yang sebelumnya maka posisi high akan berubah yaitu $(mid - 1)$ yang sebelumnya dan low tetap pada posisi yang sama. Kemudian menentukan $mid = (0+1)/2 = 0$ artinya mid sekarang terletak di data (0). lalu data yang dicari dibandingkan dengan mid. $7=7$ artinya data telah ditemukan

Kelebihan dan Kekurangan Binary Search:

1. Kelebihan Binary Search adalah tidak perlu membandingkan data yang dicari dengan seluruh data array yang ada, hanya cukup melalui titik tengah kemudian kita bias menentukan ke mana selanjutnya mencari data yang akan dicari.
2. Kekurangan Binary Search adalah penerapannya agak sedikit lebih rumit dikarenakan tidak dapat digunakan pada data array yang masih acak, sehingga harus melakukan sorting terlebih dahulu dalam penerapannya.

C. 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 << "\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;
}

```

SCREENSHOT OUTPUT

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\sjsjs\Desktop\code c++\guided 8 1> cd "c:\Users\sjsjs\Desktop\code c++\guided 8 1\" ; if ($?) { gcc guided1.cpp -o guided1 } ; if ($?) { .\guided1 }
Program Sequential Search Sederhana

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

angka 10 ditemukan pada indeks ke-9
PS C:\Users\sjsjs\Desktop\code c++\guided 8 1>

```

Alfin Ilham Berlianto
2311102047
S1IF-11-B

Ln 3, Col 10 | 42 characters | 100% | Window UTF-8

DESKRIPSI PROGRAM

Program di atas adalah program yang mendemonstrasikan penggunaan sequential search. Demonstrasi ini dilakukan dengan mencari salah satu data dalam array berukuran 10 dengan tipe data integer (array berisi angka). Untuk proses pencarian dalam program ini dimulai dengan looping ke setiap index dalam array. Lalu, nilai array dalam indeks tersebut akan dibandingkan dengan nilai yang dicari. Jika nilai dalam indeks tersebut sama dengan nilai yang dicari (dalam konteks ini, nilai yang dicari adalah 10), maka program akan mengubah nilai variabel `ketemu` menjadi true dan looping akan dihentikan. Selain itu, maka program akan melanjutkan looping selagi menambahkan nilai `i` dengan 1 ($i + 1$). Setelah looping selesai, program akan menampilkan seluruh

elemen array dan pesan sesuai dengan kondisi apakah nilai ditemukan atau tidak (dari nilai variabel `ketemu`). Jika nilai ditemukan, maka program akan menampilkan pesan bahwa nilai ditemukan. Jika tidak, maka program akan menampilkan pesan bahwa nilai tidak ditemukan dalam array.

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)
{
    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;
}

```

SCREENSHOT OUTPUT

The screenshot shows a Windows command prompt window with the following text:

```

PS C:\Users\sjsjs\Desktop\code c++\guided 8 2> cd "c:\Users\sjsjs\Desktop\code c++\guided 8 2\" ; i
guided2 } ; if ($?) { .\guided2 }
    BINARY SEARCH

Data awal:   1  8  2  5  4  9  7

Masukkan data yang ingin Anda cari: 7

Data diurutkan:  1  2  4  5  7  8  9

Data ditemukan pada index ke-4
PS C:\Users\sjsjs\Desktop\code c++\guided 8 2>

```

Overlaid on the command prompt is a Notepad window titled "k You" containing the following text:

```

Alfin Ilham Berlianto
2311102047
S1IF-11-B

```

The Notepad window's status bar at the bottom indicates "Ln 3, Col 10 | 42 characters | 100% | Window UTF-8".

DESKRIPSI PROGRAM

Program di atas adalah program yang mendemonstrasikan penggunaan algoritma binary search. Dalam program ini, data disimpan dalam array dengan ukuran 7 elemen yang menyimpan data integer. Untuk proses running program dimulai dengan

menampilkan semua data dalam array. Setelah semua data ditampilkan, program akan meminta input dari pengguna berupa angka yang akan dikembalikan/tampilkan indeksinya. Selanjutnya, program akan mengurutkan data, lalu data baru akan dicari menggunakan binary search. Setelah itu, data ditemukan, maka program akan menampilkan pesan disertai indeks dari elemen yang dicari. Di samping itu, program ini memiliki 2 fungsi selain fungsi main, yaitu :

1. `selection_sort` : sebuah fungsi yang digunakan untuk menyortir/mengurutkan elemen array. Proses pengurutan dilakukan dengan cara memilih elemen dari array dengan nilai terkecil, lalu menempatkannya di indeks yang sesuai (semakin kecil nilai elemen, semakin kecil indeksinya, begitu pula sebaliknya).
2. `binarysearch` : berfungsi untuk mencari nilai dari array yang telah diurutkan menggunakan algoritma binary search.

D. Unguided

Unguided 1

SOURCE CODE

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

using namespace std;

string sentence;
char c;

void toLower() {
    string temp;
    for (int i = 0; i < sentence.length(); i++) {
        temp += tolower(sentence[i]);
    }
    sentence = temp;
}

void selection_sort()
{
    int min, i, j;
    char temp;
    toLower();
    for (i = 0; i < sentence.length(); i++)
    {
        min = i;
        for (j = i + 1; j < sentence.length(); j++)
        {
```

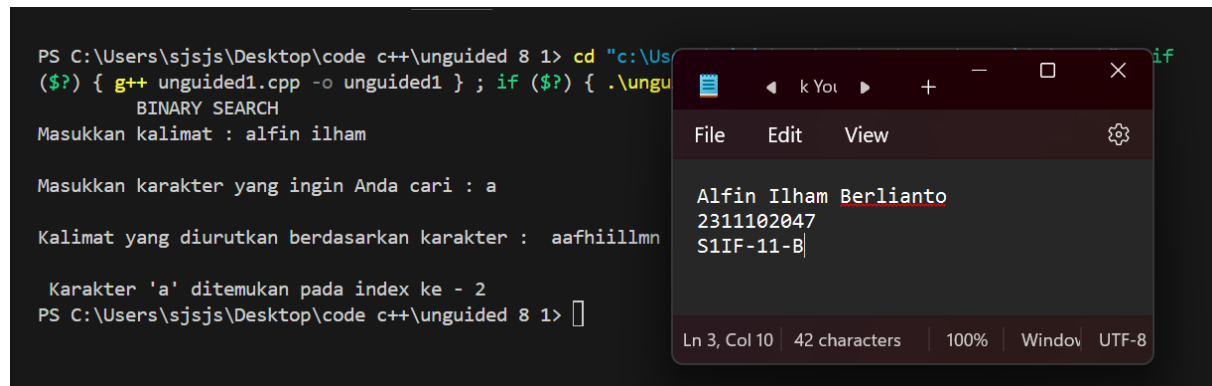
```

        if (sentence[j] < sentence[min])
        {
            min = j;
        }
    }
    temp = sentence[i];
    sentence[i] = sentence[min];
    sentence[min] = temp;
}
}
void binarysearch()
{
    int awal, akhir, tengah, b_flag = 0;
    awal = 0;
    akhir = sentence.length();
    while (b_flag == 0 && awal <= akhir)
    {
        tengah = (awal + akhir) / 2;
        if (sentence[tengah] == c)
        {
            b_flag = 1;
            break;
        }
        else if (sentence[tengah] < c)
            awal = tengah + 1;
        else
            akhir = tengah - 1;
    }
    if (b_flag == 1)
        cout << "\n Karakter '" << c << "' ditemukan pada index ke - " <<
tengah << endl;
    else
        cout << "\nData tidak ditemukan\n";
}
int main()
{
    cout << "\t BINARY SEARCH " << endl;
    cout << "Masukkan kalimat : ";
    getline(cin, sentence);
    cout << "\nMasukkan karakter yang ingin Anda cari : ";
    cin >> c;
    c = tolower(c);
    cout << "\nKalimat yang diurutkan berdasarkan karakter : ";
    selection_sort();
    for (int x = 0; x < sentence.length(); x++)
        cout << sentence[x];
    cout << endl;
    binarysearch();
}

```

```
    return 0;
}
```

SCREENSHOT OUTPUT



The screenshot shows a Windows terminal window with the following text:

```
PS C:\Users\sjsjs\Desktop\code c++\unguided 8 1> cd "c:\Us
($?) { g++ unguided1.cpp -o unguided1 } ; if ($?) { .\ungu
    BINARY SEARCH
Masukkan kalimat : alfin ilham
Masukkan karakter yang ingin Anda cari : a
Kalimat yang diurutkan berdasarkan karakter : aafhiillmn
Karakter 'a' ditemukan pada index ke - 2
PS C:\Users\sjsjs\Desktop\code c++\unguided 8 1> 
```

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

```
Alfin Ilham Berlianto
2311102047
S1IF-11-B
```

The Notepad window also shows a status bar at the bottom: "Ln 3, Col 10 | 42 characters | 100% | Window | UTF-8".

DESKRIPSI PROGRAM

Program di atas adalah sebuah program yang digunakan untuk mencari karakter di dalam suatu kalimat menggunakan binary search. Dalam program ini, pengguna akan memasukkan kalimat mereka, dilanjutkan dengan memasukkan karakter yang akan mereka cari indeksnya. Sebelum dicari, kalimat akan di-convert menjadi huruf kecil (lowercase). Setelah di-convert, program akan menyortir kalimat berdasarkan karakter terkecil ke terbesar (ascending) menggunakan algoritma selection sort (seperti program Guided 2). Program akan menampilkan kalimat yang telah di-convert dan diurutkan. Baru setelah itu, program akan mencari karakter yang diinginkan dan menampilkan posisi/indeksnya. Indeks dari karakter tadi ditentukan berdasarkan kalimat yang telah disortir (bukan kalimat asli, karena menggunakan binary search).

Unguided 2

SOURCE CODE

```
#include <iostream>

using namespace std;

string kalimat;
int banyak = 0;
char huruf;
int main()
{
    cout << "masukan kalimat: ";
    getline(cin,kalimat);

    for(int i = 0; i < kalimat.length(); i++) {
        huruf = kalimat[i];
```

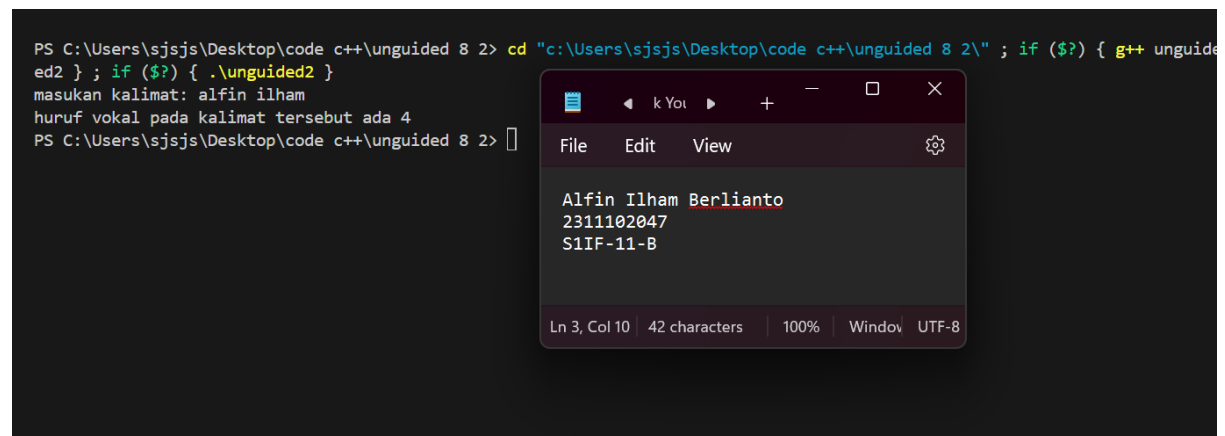
```

        if(huruf == 'a' || huruf == 'i' || huruf == 'u' || huruf == 'e' ||
huruf == 'o') {
            banyak++;
        }
    }

    cout << "huruf vokal pada kalimat tersebut ada " << banyak;
    return 0;
}

```

SCREENSHOT OUTPUT



The screenshot shows a Windows command prompt window with the following text:

```

PS C:\Users\sjsjs\Desktop\code c++\unguided 8 2> cd "c:\Users\sjsjs\Desktop\code c++\unguided 8 2\" ; if ($?) { g++ unguided2 } ; if ($?) { .\unguided2 }
masukan kalimat: alfin ilham
huruf vokal pada kalimat tersebut ada 4
PS C:\Users\sjsjs\Desktop\code c++\unguided 8 2>

```

Overlaid on the command prompt is a Notepad++ window titled "k You". The window contains the text:

```

Alfin Ilham Berlianto
2311102047
S1IF-11-B

```

The Notepad++ status bar at the bottom indicates "Ln 3, Col 10 | 42 characters | 100% | Window | UTF-8".

DESKRIPSI PROGRAM

Program tersebut merupakan sebuah program untuk menghitung banyaknya huruf vokal dalam suatu kalimat. Dalam program ini, pengguna perlu memasukkan kalimat mereka. Lalu, program akan memulai proses dengan mengonversi setiap karakter dari awal sampai akhir secara berurutan menjadi lowercase dan dicek apakah karakter tersebut termasuk dari salah satu huruf vokal (a, i, u, e, o). Jika iya, maka program akan menambahkan jumlah count dengan 1. Jika tidak, maka program akan lanjut ke karakter selanjutnya. Setelah itu, program akan menampilkan pesan yang disertai dengan jumlah huruf vokal dalam kalimat yang dimasukkan tadi.

Unguided 3

SOURCE CODE

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

int arr[] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
int arrLength = sizeof(arr)/sizeof(arr[0]);
int cari;

int seqSearch() {
    int count = 0;
    for (int i = 0; i < arrLength; i++)
    {
        if (arr[i] == cari)
        {
            count++;
        }
    }
    return count;
}

int main()
{
    // algoritma Sequential Search

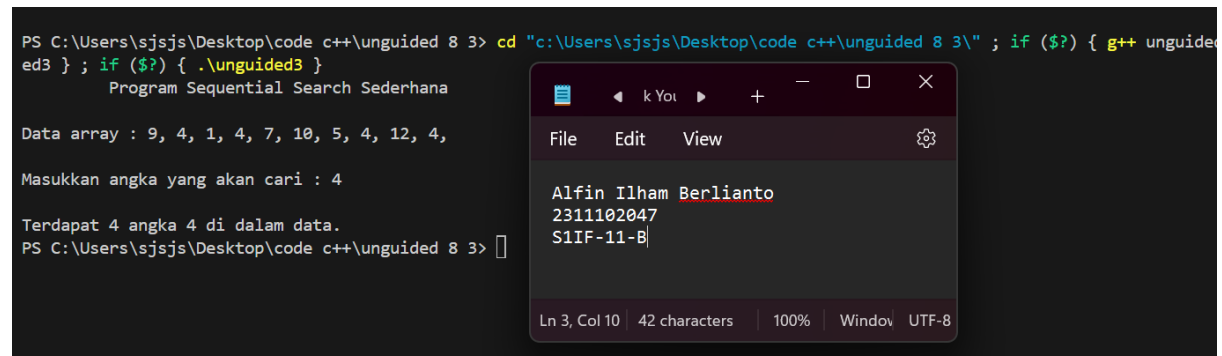
    cout << "\t Program Sequential Search Sederhana\n" << endl;
    cout << "Data array : ";
    for (int x : arr) {
        cout << x << ", ";
    }
    cout << "\n\nMasukkan angka yang akan cari : ";
    cin >> cari;
```

```

    cout << "\nTerdapat " << seqSearch() << " angka " << cari << " di dalam
data.";
    return 0;
}

```

SCREENSHOT OUTPUT



The screenshot shows a Windows command prompt window with the following text:

```

PS C:\Users\sjsjs\Desktop\code c++\unguided 8 3> cd "c:\Users\sjsjs\Desktop\code c++\unguided 8 3\" ; if ($?) { g++ unguided
ed3 } ; if ($?) { .\unguided3 }
Program Sequential Search Sederhana

Data array : 9, 4, 1, 4, 7, 10, 5, 4, 12, 4,

Masukkan angka yang akan cari : 4

Terdapat 4 angka 4 di dalam data.
PS C:\Users\sjsjs\Desktop\code c++\unguided 8 3>

```

Overlaid on the command prompt is a Notepad window titled "k You" containing the following text:

```

Alfin Ilham Berlianto
2311102047
S1IF-11-B

```

The Notepad window's status bar at the bottom indicates "Ln 3, Col 10 | 42 characters | 100% | Window | UTF-8".

DESKRIPSI PROGRAM

Program di atas adalah sebuah program sequential search sederhana yang digunakan untuk menghitung banyaknya angka yang dicari dalam array. Dalam program ini, data sudah di-set atau ditentukan di dalam kode program, sehingga pengguna hanya bisa memasukkan angka yang akan dicari. Setelah program menampilkan semua data array dan mendapatkan input dari pengguna, program akan looping dari awal sampai akhir array untuk mencari angka yang dicari sembari menghitung banyaknya angka tersebut. Setelah looping selesai, program akan menampilkan pesan berupa banyaknya angka yang dicari dalam data array.

KESIMPULAN

Sequential Search 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. Proses pencarian dimulai dari indeks pertama hingga indeks terakhir, dan berhenti apabila data ditemukan. Jika hingga akhir array data masih juga tidak ditemukan, maka proses pencarian tetap akan dihentikan. Sedangkan Binary Search membagi array menjadi dua bagian dan mencari data dengan membandingkan kata kunci dengan elemen tengah array. Algoritma ini memerlukan data yang sudah terurut dengan baik, baik secara ascending maupun descending. Jadi, metode binary search membutuhkan metode sorting.

REFERENSI

1. Blogger (2019, 1 April) Struktur Searching dalam Bahasa Pemrograman C++. Diakses pada 4 Juni 2024, dari <http://sinualpro.blogspot.com/2018/12/searching-dalam-bahasa-pemrograman-c.html>
- 2, Koding Akademi, (2022,4 Februaril) Pengenalan Searching dalamC++. Diakses pada 17 Juni 2024, dari <https://www.kodingakademi.id/pengenalan-searching-dalam-c/>