

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VIII
ALGORITMA SEARCHING**



Disusun Oleh :

Nia Novela Ariandini
2311102057

Dosen

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

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

A. Dasar Teori

Sequential search dan binary search adalah dua jenis algoritma pencarian yang sering digunakan pada program untuk menelusuri suatu data yang dicari.

- Sequential Search

Algoritma ini membandingkan setiap elemen array satu per satu secara berurutan, mulai dari elemen pertama, sampai elemen yang dicari ditemukan atau sampai semua elemen diperiksa. Jika elemen ditemukan, ia mengembalikan indeksnya, jika tidak -1.

Contoh: Linear Search

- Binary Search

Algoritma ini hanya dapat digunakan pada data yang sudah terurut. Algoritma ini membagi data menjadi dua bagian setiap kali, dan memeriksa apakah elemen yang dicari berada di setengah kiri atau setengah kanan. Apabila ditemukan kecocokan nilai maka akan mengembalikan output, jika tidak pencarian akan terus berlanjut hingga akhir dari pembagian jumlah elemen tersebut.

Contoh: Binary Search.

Berikut adalah beberapa hal yang perlu dipertimbangkan saat menggunakan Algoritma Searching :

- Kapan menggunakan sequential search : Jika data tidak terurut atau dalam keadaan acak, Jumlah data yang dicari relatif kecil, Data yang dicari berada di awal atau tengah array.
- Kapan menggunakan binary search : Jika data sudah terurut, Jumlah data yang dicari relatif besar, Data yang dicari berada di akhir array atau tidak ada dalam array. Namun, perlu diingat bahwa binary search lebih efisien dari sisi waktu dibandingkan dengan sequential search karena binary search hanya memerlukan logaritma basis 2 dari jumlah data yang dicari, sedangkan sequential search memerlukan waktu yang lebih lama karena harus memeriksa setiap elemen satu per satu secara berurutan.

Perbedaan antara sequential search dan binary search :

- a) Sequential search membandingkan setiap elemen array satu per satu secara berurutan, sedangkan binary search membagi data menjadi dua bagian setiap kali dan memeriksa apakah elemen yang dicari berada di setengah kiri atau setengah kanan.
- b) Sequential search dapat digunakan pada data yang tidak terurut, sedangkan binary search hanya dapat digunakan pada data yang sudah terurut.

- c) Binary search lebih efisien dari sisi waktu dibandingkan dengan sequential search karena binary search hanya memerlukan logaritma basis 2 dari jumlah data yang dicari, sedangkan sequential search memerlukan waktu yang lebih lama karena harus memeriksa setiap elemen satu per satu secara berurutan. Namun, binary search lebih rumit daripada sequential search dan memerlukan data yang sudah terurut.

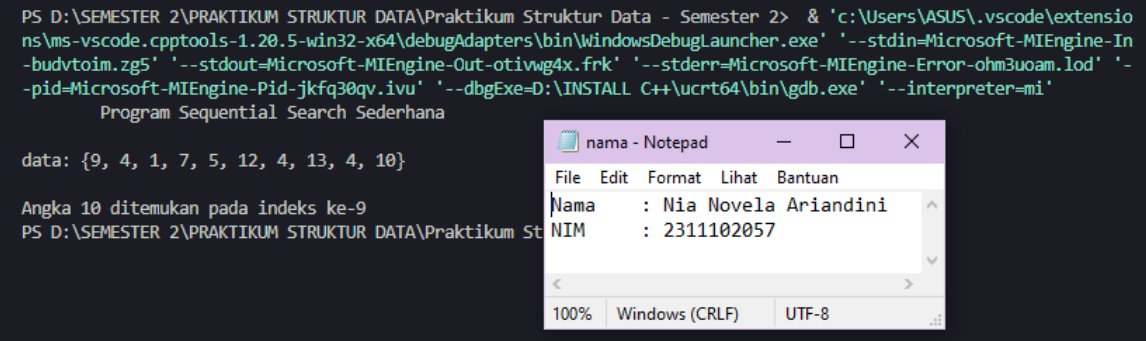
B. Guided

Guided 1

```
#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;
}
```

Screenshots Output :



The screenshot displays a terminal window with the following output:

```
PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum Struktur Data - Semester 2> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-budvtoim.zg5' '--stdout=Microsoft-MIEngine-Out-otivwg4x.frk' '--stderr=Microsoft-MIEngine-Error-ohm3uoam.lod' '-pid=Microsoft-MIEngine-Pid-jkfq30qv.ivu' '--dbgExe=D:\INSTALL C++\ucrt64\bin\gdb.exe' '--interpreter=mi'
Program Sequential Search Sederhana

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

Angka 10 ditemukan pada indeks ke-9
PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum St
```

Overlaid on the terminal is a Notepad window titled 'nama - Notepad' with the following content:

	File	Edit	Format	Lihat	Bantuan
Nama	: Nia Novela Ariandini				
NIM	: 2311102057				

The Notepad window also shows a status bar at the bottom with '100%', 'Windows (CRLF)', and 'UTF-8'.

Deskripsi :

Script di atas adalah implementasi sederhana dari algoritma pencarian sequential dalam bahasa C++. Berikut adalah penjelasan singkat:

- Program ini dirancang untuk mencari suatu bilangan (`cari`) dalam sebuah array (`data`) dengan ukuran `n` (dalam kasus ini, `n` didefinisikan sebagai 10).
- Array `data` berisi bilangan {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}.
- Program menginisialisasi variabel boolean `ketemu` sebagai `false`, yang digunakan untuk melacak apakah bilangan target ditemukan.
- Algoritma iterasi melalui setiap elemen dalam array menggunakan perulangan for, membandingkan setiap elemen dengan bilangan target (`cari`).
- Jika ditemukan cocok, `ketemu` diatur sebagai `true` dan perulangan dihentikan menggunakan perintah `break`.
- Setelah perulangan, program memeriksa nilai `ketemu`. Jika `ketemu` adalah `true`, program mencetak pesan yang menunjukkan bahwa bilangan target ditemukan serta indeksnya dalam array. Jika `ketemu` adalah `false`, program mencetak pesan yang mengatakan bahwa bilangan target tidak ditemukan dalam data.

Inilah contoh dasar bagaimana algoritma sequential search bekerja, di mana algoritma memeriksa setiap elemen dalam array secara berurutan hingga mencapai akhir array atau menemukan bilangan target.

Guided 2

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

using namespace std;

int dataArray[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 (dataArray[j] < dataArray[min]) {
            min = j;
        }
    }
    temp = dataArray[i];
    dataArray[i] = dataArray[min];
    dataArray[min] = temp;
}
}

void binarysearch() {
    int awal, akhir, tengah;
    bool b_flag = false;
    awal = 0;
    akhir = 6; // Corrected to 6 to match array bounds
    while (!b_flag && awal <= akhir) {
        tengah = (awal + akhir) / 2;
        if (dataArray[tengah] == cari) {
            b_flag = true;
        } else if (dataArray[tengah] < cari) {
            awal = tengah + 1;
        } else {
            akhir = tengah - 1;
        }
    }
    if (b_flag) {
        cout << "\nData ditemukan pada index ke- " << tengah
<<endl;
    } else {
        cout << "\nData tidak ditemukan\n";
    }
}

int main() {
    cout << "\tBINARY SEARCH" << endl;
    cout << "\nData: ";

    // Tampilkan data awal
    for (int x = 0; x < 7; x++) {
        cout << setw(3) << dataArray[x];
    }
    cout << endl;

    cout << "\nMasukkan data yang ingin Anda cari: ";

```

```

    cin >> cari;

    cout << "\nData diurutkan: ";
    // Urutkan data dengan selection sort
    selection_sort();
    // Tampilkan data setelah diurutkan
    for (int x = 0; x < 7; x++) {
        cout << setw(3) << dataArray[x];
    }
    cout << endl;

    binarysearch();

    _getche();
    return 0;
}

```

Screenshots Output :

```

PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum Struktur Data - Semester 2> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-vh10ale3.1x4' '--stdout=Microsoft-MIEngine-Out-zjikagnq.zj3' '--stderr=Microsoft-MIEngine-Error-efggfejr.rcj' '--pid=Microsoft-MIEngine-Pid-gu24x40n.md2' '--dbgExe=D:\INSTALL C++\ucrt64\bin\gdb.exe' '--interpreter=mi'
BINARY SEARCH

Data:  1  8  2  5  4  9  7

Masukkan data yang ingin Anda cari: 5

Data diurutkan:  1  2  4  5  7  8  9

Data ditemukan pada index ke- 3

```

Deskripsi :

Script di atas adalah implementasi program C++ yang menggunakan algoritma binary search untuk mencari suatu bilangan (cari) dalam sebuah array (dataArray) yang telah diurutkan menggunakan algoritma selection sort. Berikut adalah penjelasan singkat :

- Program ini memuat array dataArray dengan 7 elemen: {1, 8, 2, 5, 4, 9, 7}.
- Program meminta pengguna untuk memasukkan bilangan yang ingin dicari (cari) melalui input.
- Program mengurutkan array menggunakan algoritma selection sort, yang mengurutkan array secara ascending.
- Setelah array diurutkan, program menggunakan algoritma binary search untuk mencari bilangan target (cari) dalam array.
- Algoritma binary search bekerja dengan cara membagi array menjadi dua bagian,

mencari bilangan target di bagian tengah, dan kemudian membagi bagian yang sesuai lagi hingga ditemukan bilangan target atau mencapai akhir array.

- Jika bilangan target ditemukan, program mencetak indeksinya dalam array. Jika tidak ditemukan, program mencetak pesan bahwa bilangan target tidak ditemukan.
- Program menggunakan fungsi `_getche()` untuk menghentikan program dan menampilkan tampilan "Tekan tombol apa pun untuk keluar...".

Dengan demikian, program ini menunjukkan bagaimana binary search dapat digunakan untuk mencari suatu bilangan dalam array yang telah diurutkan secara efisien.

C. Unguided/Tugas

Unguided 1

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

Source Code :

```
//Nia Novela Ariandini ( 2311102057 )
#include <iostream>
using namespace std;

void selectionSort(string &huruf, int n)
{
    int i, j, min057;
    for (i = 0; i < n - 1; i++)
    {
        min057 = i;
        for (j = i + 1; j < n; j++)
            if (huruf[j] < huruf[min057])
                min057 = j;
        char temp = huruf[i];
        huruf[i] = huruf[min057];
        huruf[min057] = temp;
    }
}

int binarySearch(string Huruf057, int kiri057, int kanan057,
```



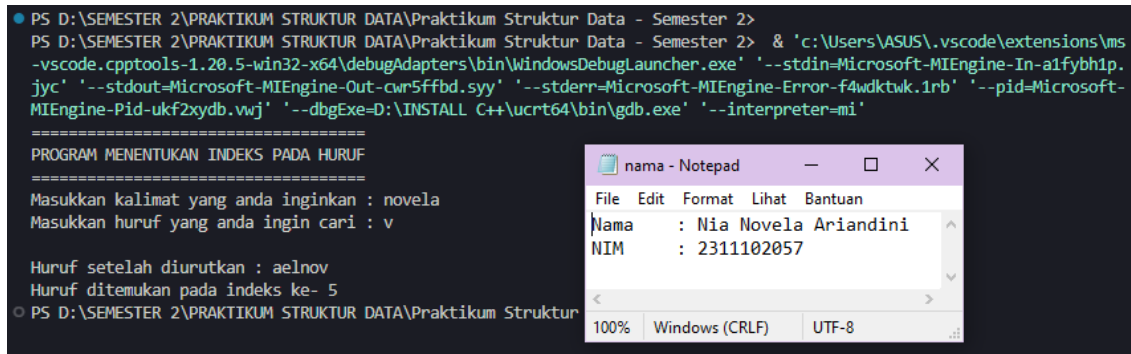
```

char target057)
{
    while (kiri057 <= kanan057)
    {
        int mid = kiri057 + (kanan057 - kiri057) / 2;
        if (Huruf057[mid] == target057)
            return mid;
        if (Huruf057[mid] < target057)
            kiri057 = mid + 1;
        else
            kanan057 = mid - 1;
    }
    return -1;
}

int main()
{
    string kalimat057;
    char input057;
    cout << "=====" << endl;
    cout << "PROGRAM MENENTUKAN INDEKS PADA HURUF" << endl;
    cout << "=====" << endl;
    cout << "Masukkan kalimat yang anda inginkan : ";
    getline(cin, kalimat057);
    cout << "Masukkan huruf yang anda ingin cari : ";
    cin >> input057;
    cout << endl;
    selectionSort(kalimat057, kalimat057.size());
    int result = binarySearch(kalimat057, 0, kalimat057.size()
- 1, input057);
    if (result == -1)
    {
        cout << "Huruf yang anda cari tidak ditemukan!" <<
endl;
    }
    else
    {
        cout << "Huruf setelah diurutkan : " << kalimat057 <<
endl;
        cout << "Huruf ditemukan pada indeks ke- " << result
<< endl;
    }
    return 0;
}

```

Screenshots Output :



The screenshot shows a terminal window with the following output:

```
PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum Struktur Data - Semester 2>
PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum Struktur Data - Semester 2> & 'c:\Users\ASUS\.vscode\extensions\ms
-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-a1fybh1p.
jyc' '--stdout=Microsoft-MIEngine-Out-cwr5ffbd.syy' '--stderr=Microsoft-MIEngine-Error-f4wdktwk.1rb' '--pid=Microsoft-
MIEngine-Pid-ukf2xydb.vwj' '--dbgExe=D:\INSTALL C++\ucrt64\bin\gdb.exe' '--interpreter=mi'

=====
PROGRAM MENENTUKAN INDEKS PADA HURUF
=====
Masukkan kalimat yang anda inginkan : novela
Masukkan huruf yang anda ingin cari : v

Huruf setelah diurutkan : aelnov
Huruf ditemukan pada indeks ke- 5
PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum Struktur
```

Overlaid on the terminal is a Notepad window titled 'nama - Notepad'. It contains the following text:

```
File Edit Format Lihat Bantuan
Nama : Nia Novela Ariandini
NIM : 2311102057
```

The Notepad window also shows a status bar at the bottom with '100%', 'Windows (CRLF)', and 'UTF-8'.

Deskripsi :

Script di atas adalah implementasi program C++ yang menggunakan algoritma selection sort dan binary search untuk mencari indeks suatu huruf dalam sebuah kalimat. Berikut adalah penjelasan singkat :

- Program ini meminta pengguna untuk memasukkan sebuah kalimat dan suatu huruf yang ingin dicari.
- Program menggunakan algoritma selection sort untuk mengurutkan huruf dalam kalimat secara ascending.
- Kemudian, program menggunakan algoritma binary search untuk mencari huruf yang ingin dicari dalam kalimat yang telah diurutkan.
- Algoritma binary search bekerja dengan cara membagi kalimat menjadi dua bagian, mencari huruf target di bagian tengah, dan kemudian membagi bagian yang sesuai lagi hingga ditemukan huruf target atau mencapai akhir kalimat.
- Jika huruf target ditemukan, program mencetak indeksnya dalam kalimat. Jika tidak ditemukan, program mencetak pesan bahwa huruf target tidak ditemukan.
- Program menggunakan fungsi getline untuk memasukkan kalimat dan fungsi cin untuk memasukkan huruf yang ingin dicari.

Dengan demikian, program ini menunjukkan bagaimana binary search dapat digunakan untuk mencari suatu huruf dalam sebuah kalimat yang telah diurutkan secara efisien.

Unguided 2

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

Source Code :

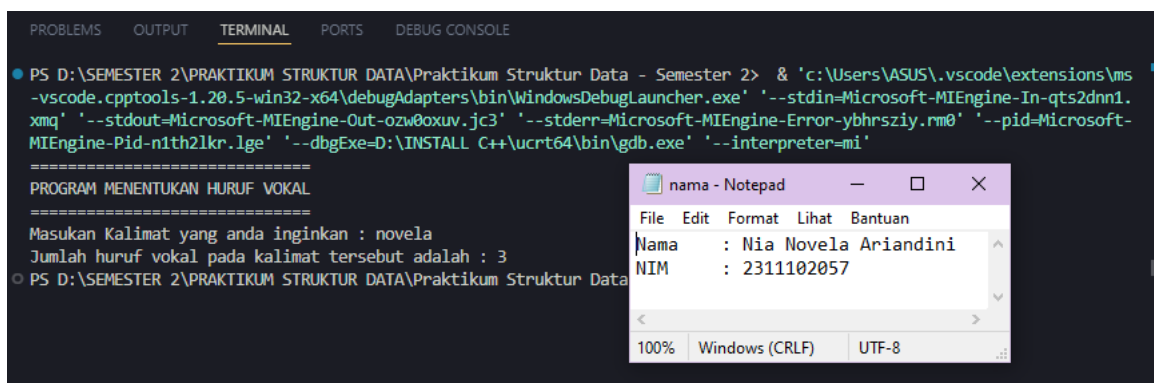
```
//Nia Novela Ariandini ( 2311102057 )
#include <iostream>
using namespace std;

int main()
{
    string kalimat057;
    int count=0;
    cout << "===== " << endl;
    cout << "PROGRAM MENENTUKAN HURUF VOKAL" << endl;
    cout << "===== " << endl;
    cout << "Masukan Kalimat yang anda inginkan : " ;
    cin  >> kalimat057;

    for (int i=0; i<kalimat057.length(); i++)
    {
        if(kalimat057[i]=='a' || kalimat057[i]=='i' ||
kalimat057[i]=='u' || kalimat057[i]=='e' || kalimat057[i]=='o')
        {
            count++;
        }
    }
    cout << "Jumlah huruf vokal pada kalimat tersebut adalah :
" << count;

}
```

Screenshots Output :



```
PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE
PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum Struktur Data - Semester 2> & 'c:\Users\ASUS\.vscode\extensions\ms
-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-qts2dnn1.
xmq' '--stdout=Microsoft-MIEngine-Out-ozw0oxuv.jc3' '--stderr=Microsoft-MIEngine-Error-ybhrsziy.rm0' '--pid=Microsoft-
MIEngine-Pid-n1th2lkr.lge' '--dbgExe=D:\INSTALL C++\ucrt64\bin\gdb.exe' '--interpreter=mi'
=====
PROGRAM MENENTUKAN HURUF VOKAL
=====
Masukan Kalimat yang anda inginkan : novela
Jumlah huruf vokal pada kalimat tersebut adalah : 3
PS D:\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\Praktikum Struktur Data
```

nama - Notepad

File	Edit	Format	Lihat	Bantuan
Nama	:	Nia Novela Ariandini		
NIM	:	2311102057		

100% Windows (CRLF) UTF-8

Deskripsi :

Script di atas mengimplementasikan program C++ yang digunakan untuk menghitung jumlah huruf vokal dalam sebuah kalimat. Berikut adalah penjelasan singkat :

- Program ini meminta pengguna untuk memasukkan sebuah kalimat melalui input.
- Program kemudian menggunakan perulangan untuk memeriksa setiap huruf dalam kalimat.
- Dalam perulangan, program memeriksa apakah huruf tersebut adalah huruf vokal ('a', 'i', 'u', 'e', atau 'o') menggunakan operator logika '||' untuk membandingkan huruf dengan huruf vokal.
- Jika huruf tersebut adalah huruf vokal, program meningkatkan nilai variabel count untuk menghitung jumlah huruf vokal.
- Setelah perulangan selesai, program mencetak jumlah huruf vokal yang ditemukan dalam kalimat.

Dengan demikian, program ini menunjukkan bagaimana cara menghitung jumlah huruf vokal dalam sebuah kalimat menggunakan C++.

Unguided 3

Diketahui data = 9, 4, 1, 4, 7, 10, 5, 4, 12, 4. Hitunglah berapa banyak angka 4 dengan menggunakan algoritma Sequential Search!sebuah program yang dapat menghitung banyaknya huruf vocal dalam sebuah kalimat!

Source Code :

```
//Nia Novela Ariandini ( 2311102057 )
#include <iostream>
using namespace std;

int HitungAngka057( const int array[], int size057, int
target057) {
    int count = 0;

    for (int i = 0; i < size057; i++) {
        if (array[i] == target057) {
            count++;
        }
    }
}
```

```

        return count;
    }

    int main() {
        const int size057 = 10;
        int array[size057] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
        int target057 = 4;

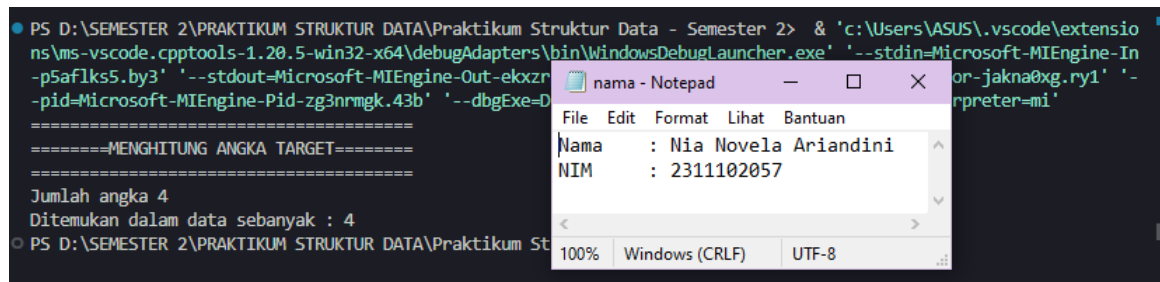
        int count = HitungAngka057(array, size057, target057);

        cout << "===== " << endl;
        cout << "=====MENGHITUNG ANGKA TARGET===== " << endl;
        cout << "===== " << endl;
        cout << "Jumlah angka " << target057 << endl;
        cout << "Ditemukan dalam data sebanyak : " << count <<
endl;

        return 0;
    }

```

Screenshots Output :



Deskripsi :

Script di atas mengimplementasikan program C++ yang digunakan untuk menghitung jumlah angka yang sama dengan suatu target dalam sebuah array. Berikut adalah penjelasan singkat :

- Program ini memiliki fungsi `HitungAngka057` yang menerima tiga parameter: array, ukuran array, dan target angka yang ingin dicari.
- Fungsi ini menggunakan perulangan untuk memeriksa setiap angka dalam array. Jika angka tersebut sama dengan target, fungsi meningkatkan nilai variabel `count` untuk menghitung jumlah angka yang sama dengan target.
- Setelah perulangan selesai, fungsi mengembalikan nilai `count` yang menunjukkan jumlah angka yang sama dengan target.

- Dalam fungsi main, program memasukkan array dengan ukuran 10 dan memasukkan target angka yang ingin dicari, yaitu 4.
- Program kemudian memanggil fungsi HitungAngka057 dengan parameter array, ukuran array, dan target, dan mengembalikan nilai count yang menunjukkan jumlah angka 4 dalam array.
- Program mencetak hasilnya, yaitu jumlah angka 4 yang ditemukan dalam data.

Dengan demikian, program ini menunjukkan bagaimana cara menghitung jumlah angka yang sama dengan suatu target dalam sebuah array menggunakan C++.

D. Kesimpulan

Setelah mempelajari algoritma searching sequential search dan binary search, dapat diketahui bahwa sequential search adalah metode pencarian data yang membandingkan setiap elemen dengan elemen yang dicari secara berurutan, mulai dari elemen pertama hingga elemen yang dicari ditemukan. Metode ini direkomendasikan untuk digunakan pada data yang relatif kecil. Sementara binary search adalah metode pencarian data yang memerlukan data dalam keadaan terurut, baik naik atau turun. Proses pencarian binary search hanya dapat dilakukan pada data yang telah diurutkan terlebih dahulu. Algoritma ini biasanya digunakan pada program dengan jumlah data yang besar, dengan kompleksitas $O(\log n)$, di mana n adalah jumlah item. Binary search lebih efisien dari sisi waktu jika dibandingkan dengan sequential search, namun memerlukan data yang telah diurutkan terlebih dahulu. Algoritma binary search juga lebih kompleks daripada sequential search. Dalam menggunakan binary search, data dalam array harus diurutkan terlebih dahulu menggunakan teknik sorting seperti bubble sort.

E. Referensi

- [1] Y Rahmanto, J Alfian, D Damayanti - Jurnal Buana Informatika, 2021 [2] Y. Rahmanto, M. F. Randhika, F. Ulum, and B. Priyopradono, "Aplikasi pembelajaran audit sistem informasi dan tata kelola teknologi informasi berbasis Mobile," J. TEKNOKOMPAK, vol. 14, no. 2, pp. 62–67, Aug. 2020, doi:10.33365/jtk.v14i2.723
- [2] Asisten Praktikum. "Modul VIII Algoritma Searching", Learning Management System, 2024.