

**LAPORAN PRAKTIKUM
STRUKTUR DATA PEMOGRAMAN**

**MODUL VIII
“ALGORITMA SEARCHING”**



Disusun Oleh:

NAMA : Trie Nabilla Farhah

NIM : 2311102071

Dosen Pengampu:

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

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

2024

A. DASAR TEORI

Pencarian (searching) merupakan proses yang sangat penting dalam pengolahan data. Proses pencarian adalah menemukan nilai (data) tertentu di dalam sekumpulan data yang bertipe sama (baik bertipe dasar atau bertipe bentukan). Sebagai contoh, untuk mengubah (update) data tertentu, langkah pertama yang harus dilakukan adalah mencari keberadaan data tersebut di dalam kumpulannya. Jika data yang dicari ditemukan, maka data tersebut dapat diubah nilainya dengan data yang baru.

Algoritma pencarian (searching algorithm) adalah algoritma yang menerima sebuah kata kunci dan dengan langkah-langkah tertentu akan mencari rekaman dengan kata kunci tersebut. Setelah proses pencarian dilaksanakan, akan diperoleh salah satu dari dua kemungkinan, yaitu data yang dicari ditemukan atau tidak ditemukan.

1. Pencarian Berurutan (*Sequential Searching*)

Sequential Search adalah teknik pencarian data dimana data dicari secara urut dari depan ke belakang atau dari awal sampai akhir. Berdasarkan key yang dicari Kelebihan dari proses pencarian secara sequential ini jika data yang dicari terletak didepan, maka data akan ditemukan dengan cepat. Tetapi dibalik kelebihan ini, teknik ini juga memiliki kekurangan jika data yang dicari terletak dibelakang atau paling akhir, maka akan membutuhkan waktu yang lama dalam proses pencariannya dan juga beban komputer akan semakin bertambah jika jumlah data dalam array sangat banyak.

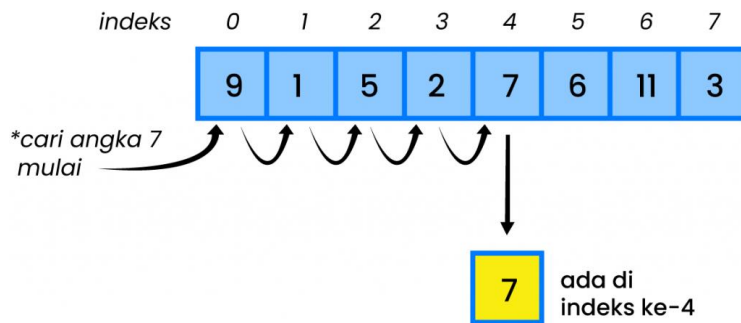
Pencarian berurutan sering disebut pencarian linear merupakan metode pencarian yang paling sederhana. Pencarian berurutan menggunakan prinsip sebagai berikut : data yang ada dibandingkan satu per satu secara berurutan dengan yang dicari sampai data tersebut ditemukan atau tidak ditemukan. Pada dasarnya, pencarian ini hanya melakukan pengulangan dari 1 sampai dengan jumlah data. Pada setiap pengulangan, dibandingkan data ke-i dengan yang dicari. Apabila sama, berarti data telah ditemukan. Sebaliknya apabila sampai akhir pengulangan tidak ada data yang sama, berarti data tidak ada. Pada kasus yang paling buruk, untuk N elemen data harus dilakukan pencarian sebanyak N kali pula.

Konsep Pencarian Sekuensial:

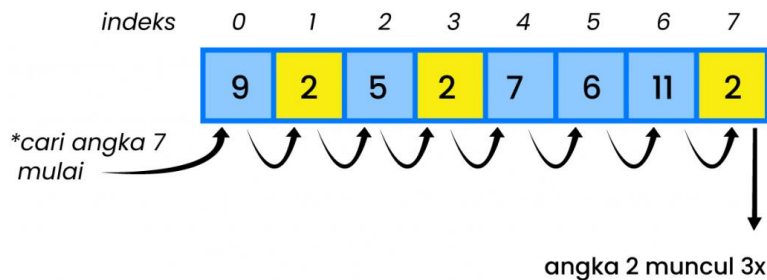
- Membandingkan setiap elemen pada array satu per satu secara berurut
- Proses pencarian dimulai dari indeks pertama hingga indeks terakhir
- Proses pencarian akan berhenti apabila data ditemukan. Jika hingga akhir array data masih juga tidak ditemukan, maka proses pencarian tetap akan dihentikan
- Proses pengulangan pada pencarian akan terjadi sebanyak jumlah N elemen pada array

Ilustrasi Sequential Search

Berikut adalah ilustrasi dari pencarian sekuensial:



Ilustrasi di atas menunjukkan bagaimana proses dari algoritma pencarian Sekuensial. Algoritma ini mencari angka 2 dengan mengecek setiap elemen pada array. Ketika sudah ditemukan maka proses pencarian dapat diakhiri.



Ilustrasi kedua mirip dengan sebelumnya, perbedaannya yaitu ilustrasi di atas menghitung berapa banyak angka yang dicari muncul pada data, sedangkan ilustrasi sebelumnya akan menghentikan pencarian ketika data yang dicari berhasil ditemukan.

2. Binary Searching

Salah satu syarat agar pencarian biner dapat dilakukan adalah data sudah dalam keadaan urut. Dengan kata lain, apabila data belum dalam keadaan urut, pencarian biner tidak dapat dilakukan. Dalam kehidupan sehari-hari, sebenarnya kita juga sering menggunakan pencarian biner. Misalnya saat ingin mencari suatu kata dalam kamus.

Prinsip dari pencarian biner dapat dijelaskan sebagai berikut : mula-mula diambil posisi awal 0 dan posisi akhir = $N - 1$, kemudian dicari posisi data tengah dengan rumus $(\text{posisi awal} + \text{posisi akhir}) / 2$. Kemudian data yang dicari dibandingkan dengan data tengah. Jika lebih kecil, proses dilakukan kembali tetapi posisi akhir dianggap sama dengan posisi tengah $- 1$. Jika lebih besar, proses dilakukan kembali tetapi posisi awal dianggap sama dengan posisi tengah $+ 1$. Demikian seterusnya sampai data tengah sama dengan yang dicari.

Algoritma pencarian biner dapat dituliskan sebagai berikut :

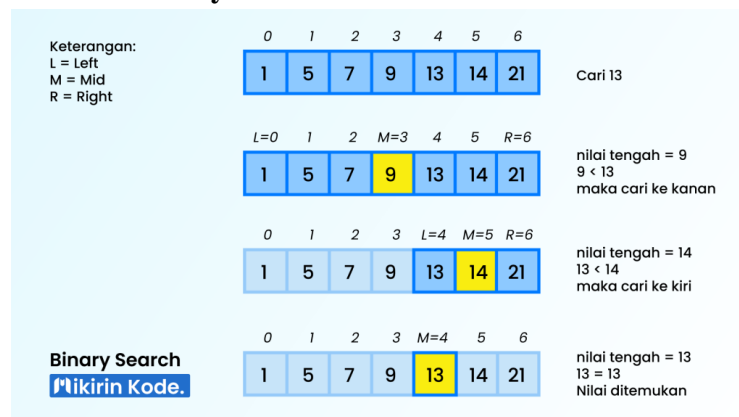
- $L \square 0$
- $R \square N - 1$
- ketemu \square false
- Selama $(L \leq R)$ dan (tidak ketemu) kerjakan baris 5 sampai dengan 8

- $m = (L + R) / 2$
- Jika $(Data[m] = x)$ maka $ketemu = true$
- Jika $(x < Data[m])$ maka $R = m - 1$
- Jika $(x > Data[m])$ maka $L = m + 1$
- Jika $(ketemu)$ maka m adalah indeks dari data yang dicari, jika tidak data tidak ditemukan

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

Ilustrasi Binary Search



- 1) Terdapat sebuah array yang menampung 7 elemen seperti ilustrasi di atas
- 2) Nilai yang akan dicari pada array tersebut adalah 13

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

B. GUIDED

GUIDED 1

Buatlah sebuah project dengan menggunakan sequential search sederhana untuk melakukan pencarian data.

Source Code

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

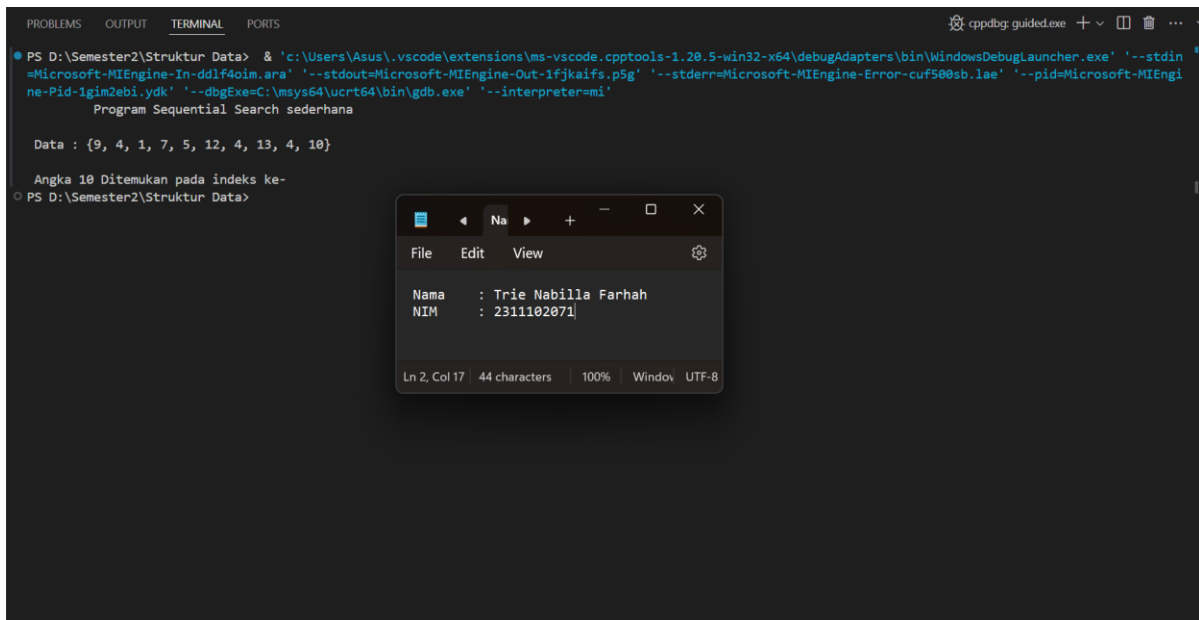
int main(){
    int n = 10;
    int data[n] = {9, 4, 1, 7, 5, 12, 4, 13, 4, 10};

    int cari = 10;
    bool ketemu = false;
    int i;

    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-" << endl;
    } else {
        cout << cari << "Tidak dapat ditemukan pada data."
<< endl;
    }
    return 0;
}
```

Output



```
PS D:\Semester2\Struktur Data> & 'c:\Users\Asus\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-dd1f40im.ara' '--stdout=Microsoft-MIEngine-Out-ifjkaifs.p5g' '--stderr=Microsoft-MIEngine-Error-cuf500sb.lae' '--pid=Microsoft-MIEngine-Pid-1gim2ebi.ydk' '--dbgExe=C:\msys64\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-
PS D:\Semester2\Struktur Data>
```

Na

File Edit View

Nama : Trie Nabilla Farhah
NIM : 2311102071

Ln 2, Col 17 44 characters 100% Window UTF-8

Deskripsi Program

Langkah-langkah kerja program ini adalah implementasi algoritma pencarian sequential search yang sederhana dalam bahasa C++:

- Program memulai dengan mendefinisikan ukuran array "n" dan menginisialisasikannya dengan nilai "{9, 4, 1, 7, 5, 12, 4, 13, 4, 10}". Selain itu, program mendefinisikan nilai target "cari" ke nilai "10."
- Variabel boolean "ketemu" diaktifkan oleh program ke "false". Fungsinya adalah untuk menentukan apakah nilai target ditemukan atau tidak.
- Selanjutnya, program memasuki perulangan "for" melalui setiap elemen dalam array.
- Program memeriksa nilai target "cari" untuk setiap elemen. Jika nilai itu sesuai, program menetapkan "ketemu" ke "true" dan menggunakan perintah "break" untuk keluar dari kesulitan.
- Program mencetak informasi tentang program dan data yang dicari setelah perulangan selesai.
- Program mencetak indeks di mana nilai target ditemukan jika ditemukan (yaitu, jika "ketemu" adalah "benar").
- Program mencetak pesan yang menyatakan bahwa nilai target tidak dapat ditemukan jika "ketemu" adalah "tidak benar".

Secara keseluruhan, program ini adalah implementasi sederhana dari algoritma pencarian sequential, yang memeriksa setiap elemen dalam array atau daftar secara berurutan hingga menemukan nilai target.

GUIDED 2

Buatlah sebuah project untuk melakukan pencarian data dengan menggunakan Binary Search.

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() {
    int awal, akhir, tengah, b_flag = 0;
    awal = 0;
    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 indeks ke-"<<tengah<<endl;
        else
            cout <<"\n Data tidak ditemukan\n";
    }
int main()
{
    cout<<"\t BINARY SEARCH "<<endl;
    cout<<"\n Data : ";

    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 : ";

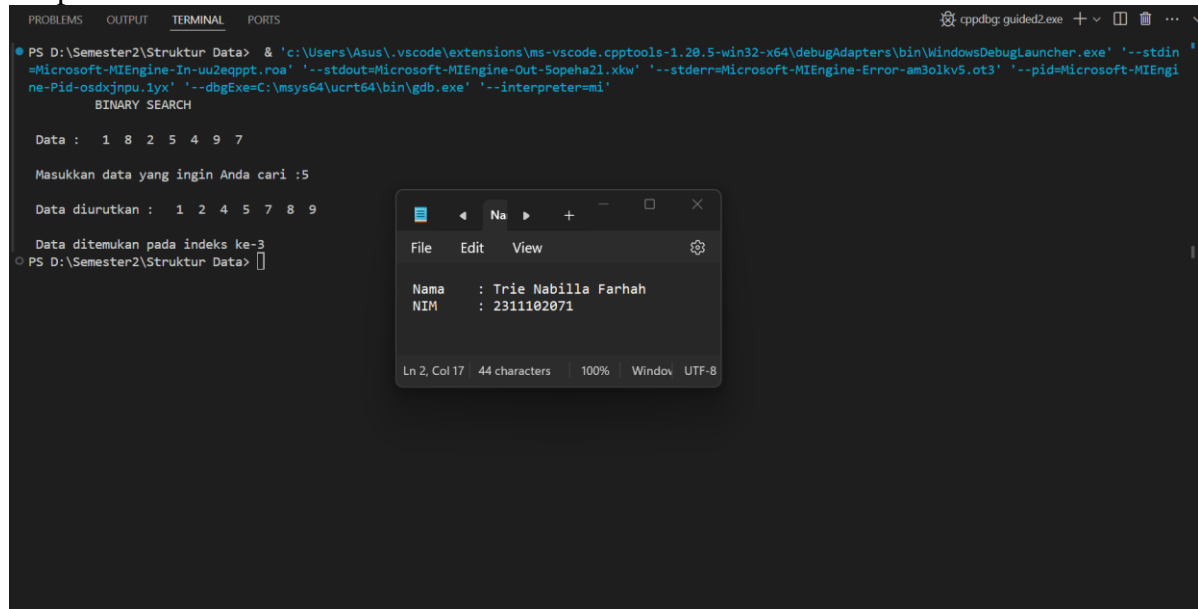
    selection_sort();
    for(int x = 0; x<7;x++)
        cout<<setw(3)<<Data[x];

    cout<<endl;

    binarysearch();

    _getche();
    return EXIT_SUCCESS;
}
```

Output



```
PS D:\Semester2\Struktur Data> & 'c:\Users\Asus\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin'
=Microsoft-MIEngine-In-uu2eqpt.roa' '--stdout=Microsoft-MIEngine-Out-5opeha21.xkw' '--stderr=Microsoft-MIEngine-Error-am3olkv5.ot3' '--pid=Microsoft-MIEngi
ne-Pid-osdxjnpu.1yx' '--dbgExe=C:\msys64\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 indeks ke-3
PS D:\Semester2\Struktur Data>
```

Na

File Edit View

Nama : Trie Nabilla Farhah
NIM : 2311102071

Ln 2, Col 17 44 characters 100% Window UTF-8

Deskripsi Program

Program C++ ini menggunakan algoritma pencarian nilai metode biner pada array data yang telah diurutkan.

- Variabel-variabel yang diperlukan dideklarasikan dan diinisialisasi untuk memulai program. Variabel cari adalah variabel yang akan digunakan untuk menyimpan nilai yang ingin dicari, dan variabel data adalah array yang terdiri dari tujuh elemen.
- Data awal yang terdiri dari tujuh elemen ditampilkan oleh program dengan menggunakan perulangan 'for'.
- Dalam program, pengguna diminta untuk mengetik nilai yang akan dicari dengan menggunakan cin; nilai ini akan disimpan dalam variabel cari.
- Fungsi selection_sort() digunakan oleh program untuk mengurutkan data dalam array Data; algoritma selection sort digunakan untuk mengurutkan data dari kecil ke besar.
- Dengan menggunakan perulangan for, program menampilkan data yang sudah diurutkan.
- Nilai yang diinginkan dalam array Data yang sudah diurutkan dapat ditemukan oleh program dengan menggunakan fungsi binarysearch(). Untuk menemukan nilai, fungsi ini menggunakan algoritma pencarian biner.
- Program akan menampilkan pesan yang menunjukkan bahwa nilai yang dicari ditemukan pada indeks ke-tengah. Jika nilai tidak ditemukan, pesan akan menunjukkan bahwa nilai tidak ditemukan.
- Sebelum program berakhir, program menggunakan _getche() untuk menunggu input user.

C. UNGUIDED

Unguided 1

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

```
#include <iostream>

using namespace std;
void selectionSort(string &huruf, int n)
{
    int x, y, min;
    for (x = 0; x < n - 1; x++)
    {
        min = x;
        for (y = x + 1; y < n; y++)
            if (huruf[y] < huruf[min])
                min = y;
        char temp = huruf[x];
        huruf[x] = huruf[min];
        huruf[min] = temp;
    }
}

int binarySearch(string huruf, int kiri, int kanan, char target)
{
    while (kiri <= kanan)
    {
        int mid = kiri + (kanan - kiri) / 2;
        if (huruf[mid] == target)
            return mid;
        if (huruf[mid] < target)
            kiri = mid + 1;
        else
            kanan = mid - 1;
    }
    return -1;
}

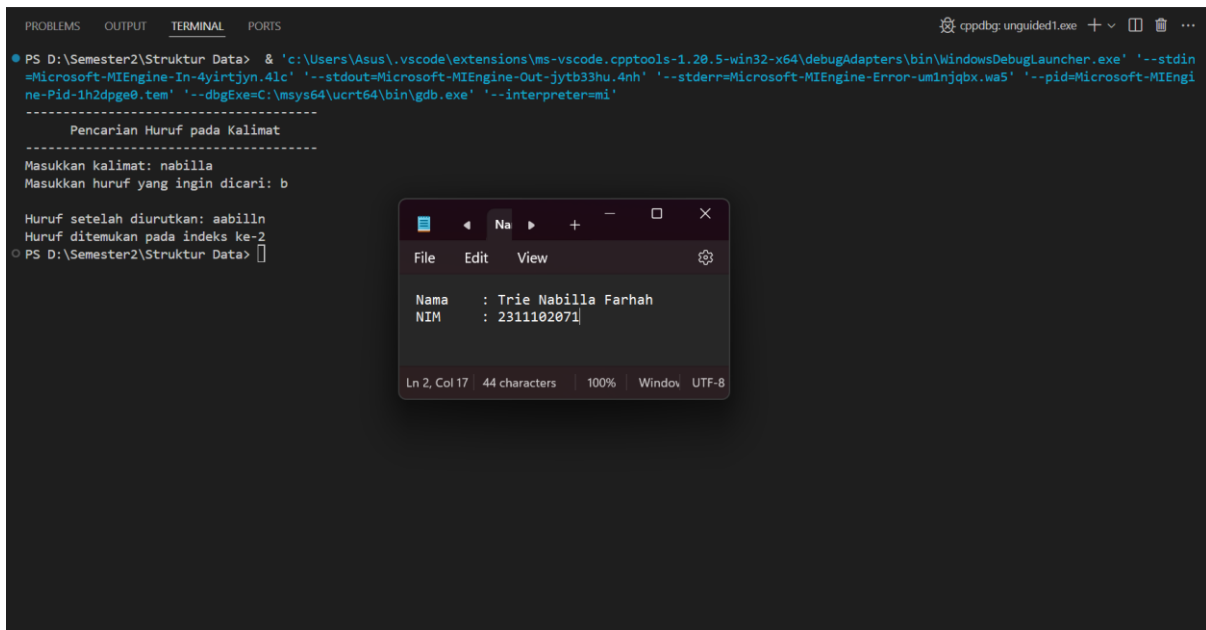
int main()
{
    string kalimat;
```

```

        char input;
        cout << "-----" <<
endl;
        cout << "          Pencarian Huruf pada Kalimat          " <<
endl;
        cout << "-----" <<
endl;
        cout << "Masukkan kalimat: ";
        getline(cin, kalimat);
        cout << "Masukkan huruf yang ingin dicari: ";
        cin >> input;
        cout << endl;
        selectionSort(kalimat, kalimat.size());
        int result = binarySearch(kalimat, 0, kalimat.size() -
1, input);
        if (result == -1)
        {
            cout << "Huruf yang Anda cari tidak ditemukan!" <<
endl;
        }
        else
        {
            cout << "Huruf setelah diurutkan: " << kalimat <<
endl;
            cout << "Huruf ditemukan pada indeks ke- " <<
result << endl;
        }
        return 0;
    }
}

```

Output



```
PS D:\Semester2\Struktur Data> & 'c:\Users\Asus\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-4yirtjyn.41c' '--stdout=Microsoft-MIEngine-Out-jytb33hu.4nh' '--stderr=Microsoft-MIEngine-Error-um1njqbx.wa5' '--pid=Microsoft-MIEngine-Pid-1h2dpge0.tem' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'

-----
Pencarian Huruf pada Kalimat
-----
Masukkan kalimat: nabilla
Masukkan huruf yang ingin dicari: b

Huruf setelah diurutkan: aabilln
Huruf ditemukan pada indeks ke-2
PS D:\Semester2\Struktur Data>
```

Deskripsi Program

Untuk menemukan huruf dalam kalimat yang dimasukkan oleh pengguna, program ini menggunakan algoritma Sorting Selection dan Binary Search.

- Variabel-variabel yang diperlukan dideklarasikan dan diinisialisasi untuk memulai program. Variabel string adalah variabel yang digunakan untuk menyimpan kalimat yang dimasukkan oleh pengguna, variabel char adalah variabel yang digunakan untuk menyimpan huruf yang dicari, dan variabel integer n adalah variabel yang digunakan untuk menyimpan panjang kalimat.
- Dalam program, pengguna diminta untuk memasukkan kalimat dan huruf yang diinginkan.
- Fungsi selectionSort() digunakan oleh program untuk mengurutkan kalimat dalam variabel kalimat. Fungsi ini menggunakan algoritma pengurutan untuk mengurutkan kalimat dari yang paling kecil ke yang paling besar.
- Setelah kalimat diurutkan, program akan menggunakan fungsi binarySearch untuk melakukan pencarian biner. Dengan menggunakan empat parameter, yaitu kalimat yang sudah diurutkan, batas kiri dan kanan, serta huruf target yang ingin dicari, fungsi ini berfungsi untuk menemukan posisi huruf target dalam kalimat yang sudah diurutkan.
- Program akan menampilkan pesan jika huruf yang dicari ditemukan pada indeks ke-result. Jika huruf tersebut tidak ditemukan, program akan menampilkan pesan yang menunjukkan bahwa huruf tersebut tidak ada dalam kalimat.

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;
    int jumlah = 0;

    cout << "-----" << endl;
    cout << "    Program Menghitung Huruf Vokal    " << endl;
    cout << "-----" << endl;

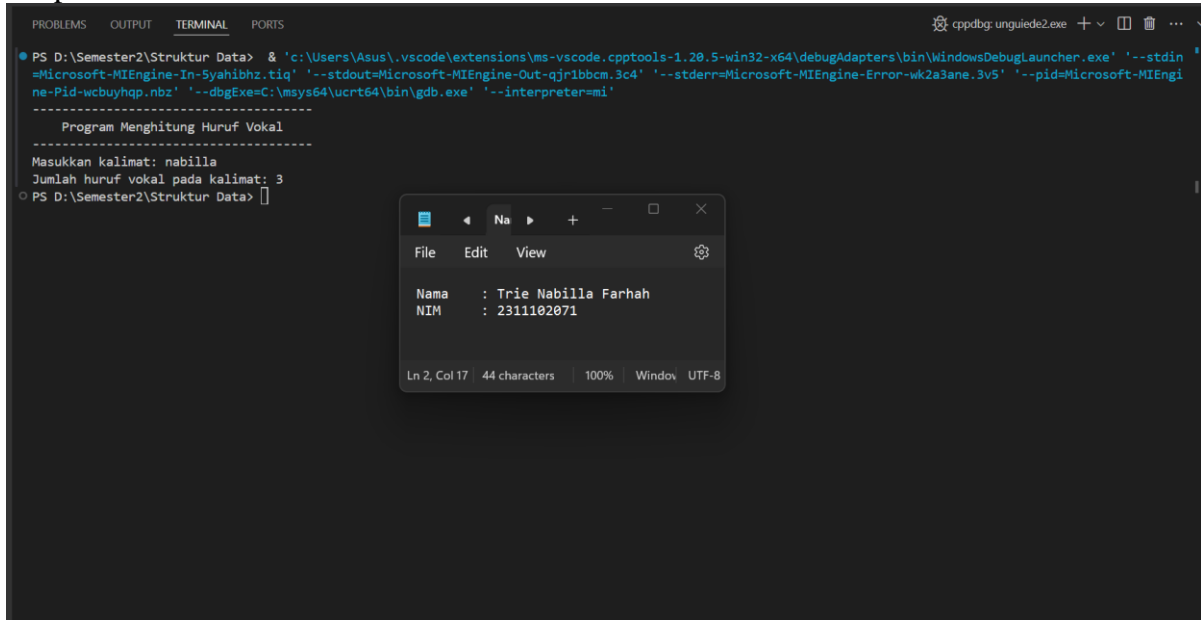
    cout << "Masukkan kalimat: ";
    getline(cin, kalimat);

    for (int i = 0; i < kalimat.length(); i++) {
        char a = kalimat[i];
        if (a == 'a' || a == 'i' || a == 'u' || a == 'e' || a
== 'o' ||
        a == 'A' || a == 'I' || a == 'U' || a == 'E' || a
== 'O') {
            jumlah++;
        }
    }

    cout << "Jumlah huruf vokal pada kalimat: " << jumlah <<
endl;

    return 0;
}
```

Output



```
PS D:\Semester2\Struktur Data> & 'c:\Users\Asus\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin
=Microsoft-MIEngine-In-5yahibhz.tiq' '--stdout=Microsoft-MIEngine-Out-qjr1bbcm.3c4' '--stderr=Microsoft-MIEngine-Error-wk2a3ane.3v5' '--pid=Microsoft-MIEngi
ne-Pid-wcbuyhqp.nbz' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
-----
Program Menghitung Huruf Vokal
-----
Masukkan kalimat: nabilla
Jumlah huruf vokal pada kalimat: 3
PS D:\Semester2\Struktur Data> []
```

Na

File Edit View

Nama : Trie Nabilla Farhah
NIM : 2311102071

Ln 2, Col 17 44 characters 100% Window UTF-8

Deskripsi Program

Program ini digunakan untuk menghitung jumlah huruf vokal dalam kalimat yang dimasukkan oleh user dan menampilkan hasilnya.

- Variabel-variabel yang diperlukan dideklarasikan dan diinisialisasi untuk memulai program. Variabel jumlah adalah variabel integer yang digunakan untuk menyimpan jumlah huruf vokal dalam kalimat, sedangkan variabel kalimat adalah variabel string yang digunakan untuk menyimpan kalimat yang dimasukkan oleh pengguna.
- Fungsi getline(cin, kalimat) digunakan oleh pengguna dalam program untuk memasukkan kalimat. Fungsi ini membaca input pengguna dan menyimpannya dalam variabel kalimat.
- Untuk menghitung jumlah huruf vokal dalam kalimat, program menggunakan loop for, yang akan berjalan sepanjang panjang kalimat, yang disimpan dalam variabel kalimat.length().
- Dengan menggunakan sintaks kalimat[i], program menggunakan setiap iterasi loop untuk mengambil karakter pada indeks i dari kalimat. Kemudian, menggunakan kondisi if untuk mengetahui apakah karakter tersebut adalah huruf vokal, seperti huruf ('a', 'i', 'u', 'e', 'o', atau huruf vokal besar), dan jika karakter tersebut adalah huruf vokal, program akan menambahkan satu ke variabel jumlah.
- Setelah loop selesai, program akan menggunakan sintaks cout untuk menampilkan jumlah huruf vokal dalam kalimat.

UNGUIDED 3

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

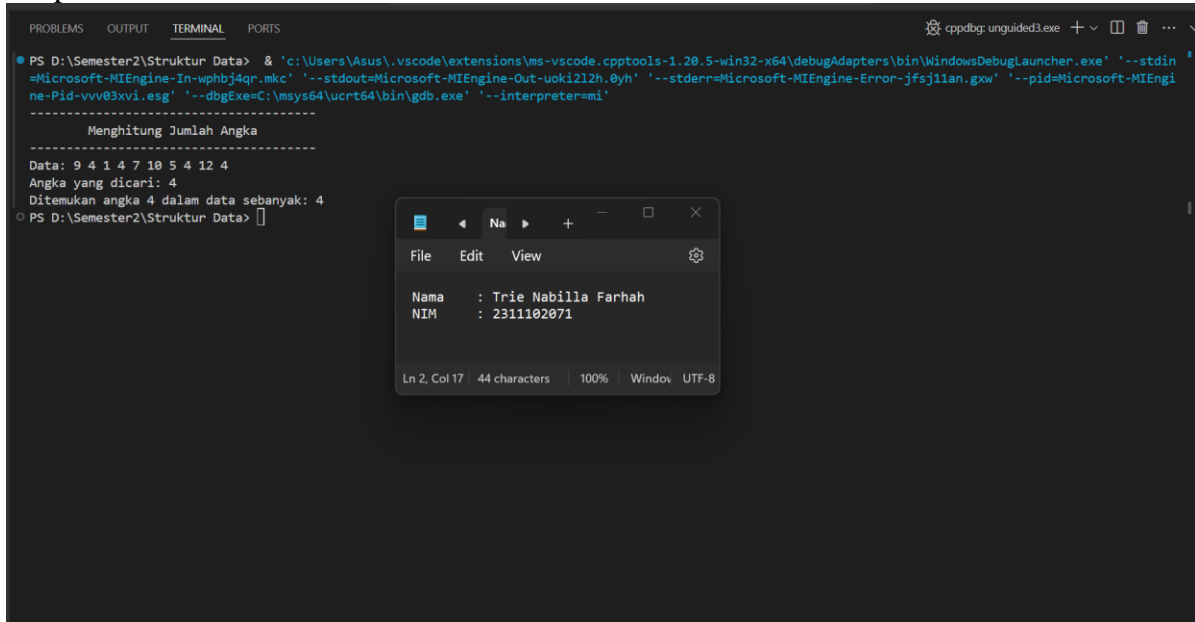
Source Code

```
#include <iostream>

using namespace std;
int hitungAngka(const int array[], int size, int target)
{
    int jumlah = 0;
    for (int i = 0; i < size; i++)
    {
        if (array[i] == target)
        {
            jumlah++;
        }
    }
    return jumlah;
}

int main()
{
    const int size = 10;
    int array[size] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int target = 4;
    int jumlah = hitungAngka(array, size, target);
    cout << "-----" <<
endl;
    cout << "          Menghitung Jumlah Angka          " <<
endl;
    cout << "-----" <<
endl;
    cout << "Data: ";
    for (int element : array)
    {
        cout << element << " ";
    }
    cout << "\nAngka yang dicari: " << target << endl;
    cout << "Ditemukan angka " << target << " dalam data
sebanyak: " << jumlah << endl;
    return 0;
}
```


Output



The screenshot shows a VS Code terminal window with the following output:

```
PS D:\Semester2\Struktur Data> & 'c:\Users\Asus\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin  
=Microsoft-MIEngine-In-wphbj4qr.mkc' '--stdout=Microsoft-MIEngine-Out-uoki2l2h.0yh' '--stderr=Microsoft-MIEngine-Error-jfsj1lan.gxw' '--pid=Microsoft-MIEngi  
ne-Pid-vvv03xvi.esg' '--dbgExe=c:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
```

Menghitung Jumlah Angka

Data: 9 4 1 4 7 10 5 4 12 4
Angka yang dicari: 4
Ditemukan angka 4 dalam data sebanyak: 4
PS D:\Semester2\Struktur Data>

Overlaid on the terminal is a small window titled 'Na' with the following content:

```
File Edit View  
  
Nama : Trie Nabilla Farhah  
NIM : 2311102071  
  
Ln 2, Col 17 44 characters 100% Window UTF-8
```

Deskripsi Program

Program ini dapat digunakan untuk menghitung jumlah kemunculan sebuah angka tertentu (angka target) dalam sebuah array.

- Variabel-variabel yang diperlukan dideklarasikan dan diinisialisasi untuk memulai program. Variabel array adalah array integer yang menyimpan data angka; variabel size adalah variabel integer yang menyimpan ukuran array; variabel target adalah variabel integer yang menyimpan angka yang dicari; dan variabel jumlah adalah variabel integer yang menyimpan jumlah kali angka target muncul dalam array.
- Dengan parameter array, ukuran array, dan angka target, program memanggil fungsi `hitungAngka()` untuk mengembalikan jumlah angka target yang muncul dalam array.
- Dengan menggunakan sintaks `cout`, program akan menampilkan hasil fungsi `hitungAngka()` setelah selesai. Program akan menampilkan data array, angka yang dicari, dan jumlah kemunculan angka dalam array.

D. KESIMPULAN

Pencarian sequential, juga dikenal sebagai pencarian linier, adalah teknik pencarian yang meninjau setiap komponen satu per satu dari awal hingga akhir. Metode ini mudah digunakan, sederhana, dan tidak memerlukan data terurut. Namun, karena perlu memeriksa setiap elemen (waktu pencarian $O(n)$), metode ini tidak efisien untuk jumlah data besar.

Pencarian biner, juga dikenal sebagai pencarian biner, adalah teknik pencarian yang lebih efisien untuk data besar dengan syarat data sudah terurut. Membedakan data menjadi dua dan membandingkan elemen tengah adalah proses pencarian. Dengan waktu pencarian $O(\log n)$, metode ini jauh lebih cepat daripada pencarian urut, tetapi memerlukan data terurut dan implementasinya lebih kompleks.

E. REFERENSI

Asisten praktikum, “*Modul 8 Algoritma Searching*”, learning Management System, 2024

Mochamad, A. Radja, S.P, Dhamar, G.A, Diski, A.(2023).” *IMPLEMENTASI METODE SEQUENTIAL SEARCHUNTUK PENGELOLAAN DATA BARANG PADA SISTEM APLIKASI SIKILAT CARGO*”. Jurnal Ilmu Komputer Dan Pendidikan. 1(2), 283-287.

Tirago4.(2016, 11 Oktober). *Pencarian (searching) Didalam Algoritma*. Diakses pada 1 Juni 2024, dari <https://tirago4.wordpress.com/2016/10/11/pencarian-searching-didalam-algoritma/>

Muhammad Wafa. (2021, 6 Desember). *Sequential Search – Algoritma Pencarian* . Diakses pada 1 Juni 2024, dari <https://mikirinkode.com/sequential-search/>

Muhammad Wafa. (2021, 7 Desember). *Binary Search* . Diakses pada 1 Juni 2024, dari <https://mikirinkode.com/binary-search/>