

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VIII
ALGORITMA SEARCHING**



Disusun Oleh :

Muhammad Arsyad Zaidan (2311102058)

Dosen

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

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

B. 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:

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

Algoritma pencarian berurutan dapat dituliskan sebagai berikut :

1) $i \leftarrow 0$

- 2) ketemu \leftarrow false
- 3) Selama (tidak ketemu) dan $(i \leq N)$ kerjakan baris 4
- 4) Jika $(Data[i] = x)$ maka ketemu \leftarrow true, jika tidak $i \leftarrow i + 1$
- 5) Jika (ketemu) maka i adalah indeks dari data yang dicari, jika tidak data tidak ditemukan.

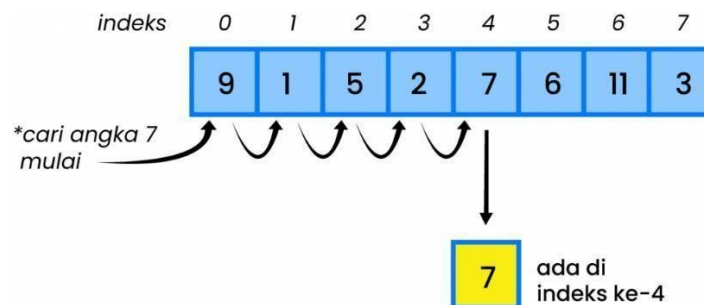
Di bawah ini merupakan fungsi untuk mencari data menggunakan pencarian sekuensial.

```
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}



Gambar 1. Ilustrasi Sequential Search

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 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:

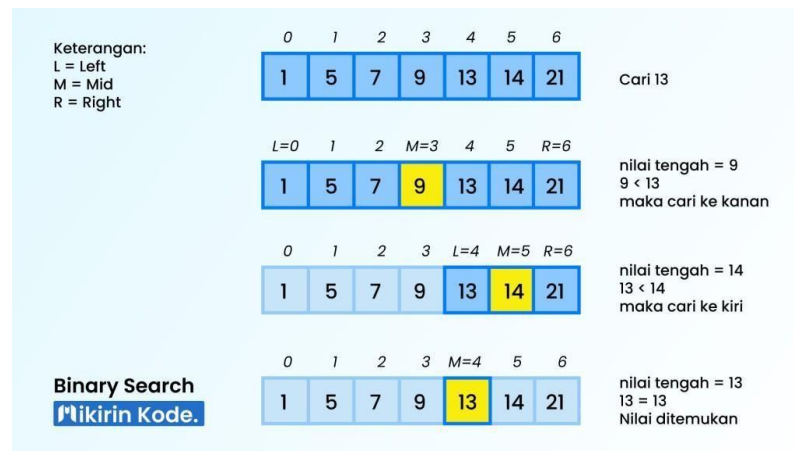
- Data diambil dari posisi 1 sampai posisi akhir N.
- Kemudian data akan dibagi menjadi dua untuk mendapatkan posisi data tengah.
- Selanjutnya data yang dicari akan dibandingkan dengan data yang berada di posisi tengah, apakah lebih besar atau lebih kecil.
- Apabila data yang dicari lebih besar dari 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) $L \leq 0$
- 2) $R \leq N - 1$
- 3) ketemu \leq false
- 4) Selama $(L \leq R)$ dan (tidak ketemu) kerjakan baris 5 sampai dengan 8 5) $m \leq (L + R) / 2$
- 6) Jika $(Data[m] = x)$ maka ketemu \leq true
- 7) Jika $(x < Data[m])$ maka $R \leq m - 1$

- 8) Jika ($x > \text{Data}[m]$) maka $L \leftarrow m + 1$
- 9) Jika (ketemu) maka m adalah indeks dari data yang dicari, jika tidak data tidak ditemukan

Contoh dari Binary Search, yaitu:



Gambar 2. Ilustrasi Binary Search

- 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.

1. Latihan Guided dan Unguided

1) 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;
    //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<<"\nAngka "<<cari<<" ditemukan pada indeks ke- "<<i<<endl;
    }else
    {
        cout<<"\nAngka "<<cari<<" Tidak ditemukan pada data"<<endl;
    }

    return 0;
}
```

Output Code

The screenshot shows a VS Code interface. The terminal window at the bottom displays the following output:

```
PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cmake-tools\bin\windowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-Input.txt' '--stdout=Microsoft-MIEngine-Output.txt' '--stderr=Microsoft-MIEngine-Error-rxdrdtim.jpa' '--pid=Microsoft-MIEngine-Pid-yhhzanld'
Program Sequential Search Sederhana

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

Angka 10 ditemukan pada indeks ke- 9
PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan>
```

The editor window above the terminal shows a C++ file named 'as'. The code in the editor is:

```
#include <iostream>
using namespace std;
#include <conio.h>
#include <iomanip>
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];
```

Deskripsi Code

Pemrograman yang menerapkan algoritma searching yang menginialisasi sequential search, berfungsi mencari data secara berurut. Data didefinisikan sebagai array yang terdiri dari sepuluh elemen sedangkan pencarian mewakili nilai yang dicari (10). Di sisi lain, ditemukan menunjukkan sebuah bendera yang menunjukkan apakah nilainya telah ditemukan; itu diinisialisasi ke false. Proses Pencarian: Perulangan for memindai setiap elemen yang ada dalam larik data. Ketika elemen yang dipertimbangkan (data[i]) cocok dengan pencarian, find disetel ke true dan ini mengakibatkan pemutusan perulangan. Keluaran: Jika find benar, umumkan bahwa nilai telah ditemukan dan berikan detail mengenai indeksnya. Cetak nilai yang tidak ditemukan jika salah.

Guided2

Source Code

```
#include <iostream>
using namespace std;
#include <conio.h>
#include <iomanip>
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;
    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";
}

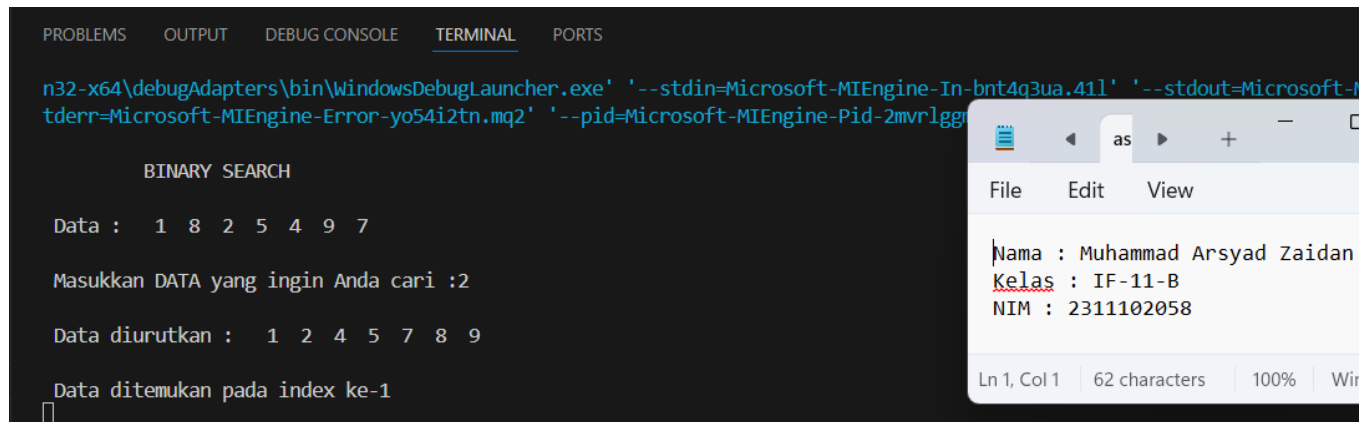
int main()
{
    cout << "\t BINARY SEARCH " << endl;
    cout << "\n Data : ";
    // tampilkan DATA awal
    for (int x = 0; x < 7; x++)
        cout << setw(3) << DATA[x];
    cout << endl;
    cout << "\n Masukkan DATA yang ingin Anda cari :";
    cin >> cari;
    cout << "\n Data diurutkan : ";
    // 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;
}

```



```
}
```

Output Code



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

n32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-bnt4q3ua.411' '--stdout=Microsoft-MIEngine-Error-yo54i2tn.mq2' '--pid=Microsoft-MIEngine-Pid-2mvr1gg

BINARY SEARCH

Data :  1  8  2  5  4  9  7

Masukkan DATA yang ingin Anda cari :2

Data diurutkan :  1  2  4  5  7  8  9

Data ditemukan pada index ke-1
```

as

File Edit View

Nama : Muhammad Arsyad Zaidan
Kelas : IF-11-B
NIM : 2311102058

Ln 1, Col 1 | 62 characters | 100% | Win

Deskripsi Code

Pada pemrograman guided2 memberikan selection sorting sebagai sarana yang melakukan pencarian data secara berurut. Teknik pengurutan yang digunakan adalah seleksi sort dimana dalam suatu array DATA terdapat 7 elemen yang disusun dari yang terkecil hingga yang terbesar. Array baru setelah diurutkan menjadi [1, 2, 4, 5, 7, 8, 9]. Pencarian biner adalah ketika mencari nilai dalam array DATA yang diurutkan dimana kita menetapkan titik awal pada 0 dan titik akhir pada 6. Indeks tengah dihitung dan kemudian dibandingkan dengan pencarian: jika sama maka nilainya ditemukan; jika pembaruan lebih kecil mulai ke tengah +1; jika pembaruan lebih besar berakhir di tengah - 1. Ulangi hingga nilai ditemukan atau awal melebihi akhir — jika suatu nilai ditemukan, keluarkan indeksnya; jika tidak, umumkan bahwa nilainya tidak ditemukan. Algoritma ini dikenal sebagai pencarian biner dan efisien untuk mencari dalam array yang diurutkan.

2) Unguided

Unguided1

Source Code

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

```

using namespace std;

string kalimat;
char c;

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

void selection_sort()
{
    int min, i, j;
    char temp;
    toLower();
    for (i = 0; i < kalimat.length(); i++)
    {
        min = i;
        for (j = i + 1; j < kalimat.length(); j++)
        {
            if (kalimat[j] < kalimat[min])
            {
                min = j;
            }
        }
        temp = kalimat[i];
        kalimat[i] = kalimat[min];
        kalimat[min] = temp;
    }
}

void binarysearch()
{
    int awal, akhir, tengah, b_flag = 0;
    awal = 0;
    akhir = kalimat.length();
    while (b_flag == 0 && awal <= akhir)
    {
        tengah = (awal + akhir) / 2;
        if (kalimat[tengah] == c)
        {
            b_flag = 1;
            break;
        }
        else if (kalimat[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, kalimat);
    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 < kalimat.length(); x++)
        cout << kalimat[x];
    cout << endl;
    binarysearch();
    return EXIT_SUCCESS;
}

```

Output Code

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan> & 'c:\Users\ASUS\.vscode\extension
n32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-
tderr=Microsoft-MIEngine-Error-aa3s5qmu.nm0' '--pid=Microsoft-MIEngine-Pid-h23h5u0

        BINARY SEARCH
Masukkan kalimat : momo suka menari

Masukkan karakter yang ingin Anda cari : m

Kalimat yang diurutkan berdasarkan karakter :  aaeiimmnoorsu

Karakter 'm' ditemukan pada index ke - 8
PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan>

```

File Edit View
 Nama : Muhammad Arsyad Zaidan
 Kelas : IF-11-B
 NIM : 2311102058
 Ln 1, Col 1 62 characters 100% Win

Deskripsi Code

Pemrograman ini menggunakan pencarian dengan algoritma pencarian binary search. Selection sort yang berperan dalam mengurutkan data pada program tersebut. Fungsi mengurutkan karakter dalam kalimat berdasarkan abjad, yang mana algoritma menemukan karakter terkecil di kalimat sebelumnya dengan kalimat terkini. Terdapat binary search pada pemrograman, yang mana dapat

mencari karakter c dalam kalimat yang telah diurutkan. Jika c pada program ditemukan di program tersebut, maka indeksnya akan ditampilkan. Dan jika tidak, output tidak ditampilkan dan memunculkan pesan bahwa data tidak ditemukan.

Unguided2

Source Code

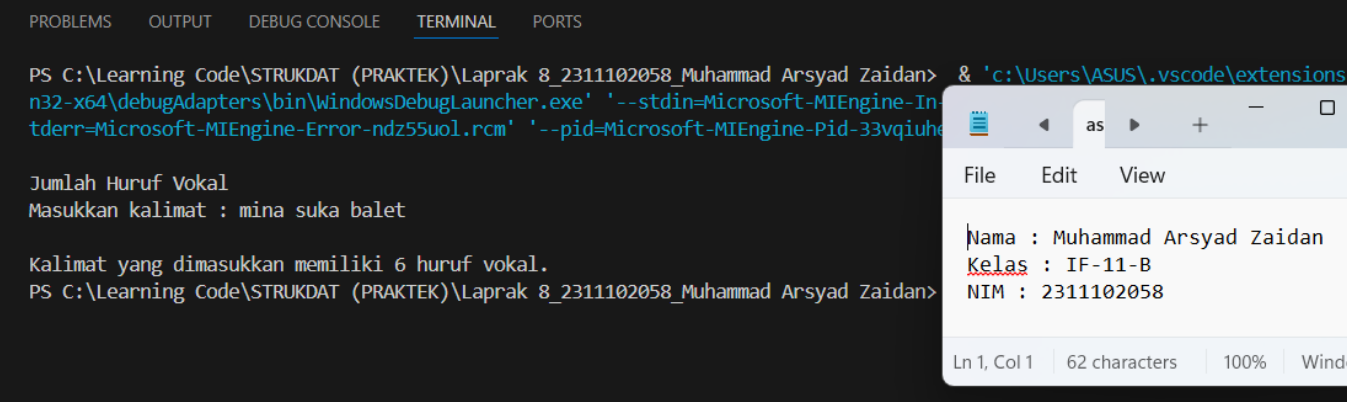
```
#include <iostream>
using namespace std;
int main()
{
    string kalimat;
    int jumlah = 0;

    cout << "Jumlah Huruf Vokal" << endl;
    cout << "Masukkan kalimat : ";
    getline(cin, kalimat);

    for (int i = 0; i < kalimat.length(); i++) {
        char c = tolower(kalimat[i]);
        if (c == 'a' || c == 'i' || c == 'u' || c == 'e' || c == 'o') jumlah++;
    }

    cout << "\nKalimat yang dimasukkan memiliki " << jumlah << " huruf vokal.";
    return 0;
}
```

Output Code



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-Input-tderr=Microsoft-MIEngine-Error-ndz55uol.rcm' '--pid=Microsoft-MIEngine-Pid-33vqiuh

Jumlah Huruf Vokal
Masukkan kalimat : mina suka balet

Kalimat yang dimasukkan memiliki 6 huruf vokal.
PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan>
```

Deskripsi Code

Pemrograman singkat pada code tersebut memakai sequential search untuk mengecek atau memeriksa huruf vokal dari kalimat yang diinputkan oleh user.

Program menginisialisasi variabel angka ke 0 untuk menghitung jumlah vokal. Menggunakan looping agar dapat memeriksa karakter secara berulang. Jika karakternya adalah salah satu huruf vokal ('a', 'i', 'u', 'e', 'o'), variabel bilangan bertambah 1.

Unguided3

Source Code

```
#include <iostream>
#include <iomanip>
#include <conio.h>
using namespace std;

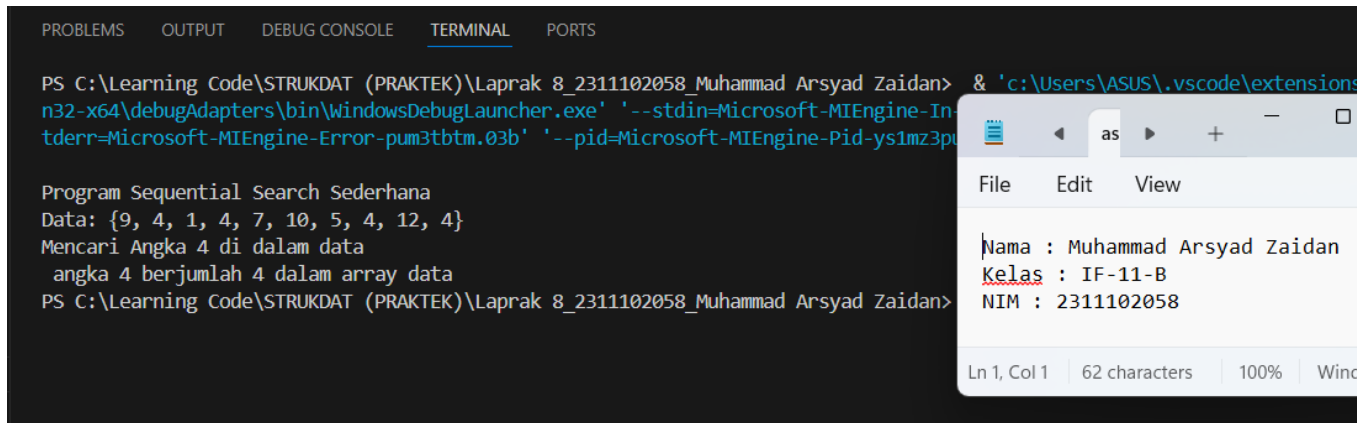
int DATA[10]={ 9, 4, 1, 4, 7, 10, 5, 4, 12, 4};

void sequentialsearch(int angka){
    int temp=0;
    for (int i = 0; i < 10; i++)
    {
        if (DATA[i] == angka)
        {
            temp=temp+1;
        }
    }
}

cout<<" angka "<<angka<<" berjumlah "<< temp<<" dalam array data";
}

int main(){
    int angka=4;
    cout<<"Program Sequential Search Sederhana"<<endl;
    cout<<"Data: {9, 4, 1, 4, 7, 10, 5, 4, 12, 4} "<<endl;
    cout<<"Mencari Angka 4 di dalam data"<<endl;
    sequentialsearch(angka);
}
```

Output Code



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools\bin\windowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-IntelliSense-Error-pum3tbtm.03b' '--pid=Microsoft-MIEngine-Pid-ys1mz3pu

Program Sequential Search Sederhana
Data: {9, 4, 1, 4, 7, 10, 5, 4, 12, 4}
Mencari Angka 4 di dalam data
    angka 4 berjumlah 4 dalam array data
PS C:\Learning Code\STRUKDAT (PRAKTEK)\Laprak 8_2311102058_Muhammad Arsyad Zaidan>
```

File Edit View

Nama : Muhammad Arsyad Zaidan
Kelas : IF-11-B
NIM : 2311102058

Ln 1, Col 1 | 62 characters | 100% | Win

Deskripsi Code

Program yang menghitung jumlah kemunculan angka tertentu pada array, diurutkan dengan sequential search. dalam array DATA. Variabel temp diinisialisasi ke 0 dan digunakan untuk menghitung jumlah kemunculan suatu angka. Algoritma ini menggunakan loop for untuk memeriksa setiap elemen dalam array DATA. Jika elemen yang diperiksa (DATA[i]) sama dengan angka, variabel temp bertambah 1. Keluaran: Setelah selesai, program mencetak jumlah kemunculan angka dalam larik DATA.

2. Kesimpulan

Dari pratikum ini, mahasiswa dapat mempelajari mengimplementasikan algoritma searching, 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, sequential searching dan binary searching. Dalam pengembangan grafis, C++ digunakan untuk memastikan performa yang tinggi dan akses langsung ke sumber daya sistem seperti memori dan perangkat keras. C++ juga digunakan dalam pengembangan engine game, editor grafis, dan library atau framework grafis lainnya.

3. Referensi

- [1] Tim Asisten Asprak, "Algoritma Searching", Learning Management System, 2024.
- [2] Kinasih, T. (2023, April 12). "Mengenal Apa Itu Bahasa Pemrograman C++ Dan Penggunaannya. Kuncie". <https://www.kuncie.com/posts/bahasa-pemrograman-cpp/#:~:text=Dalam%20pengembangan%20grafis%2C%20C%2B%2B%20digunakan,library%20atau%20framework%20grafis%20>. Terakhir kali diakses pada tanggal 3 Juni 2024