

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL 8
ALGORITMA SEARCHING**



Disusun Oleh :
Muhammad Hamzah Haifan Ma'ruf
231102091

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

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

A. Dasar Teori

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.

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.

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.

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 teratur 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 tengah = $(high + low)/2$
- 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 perbandingan. Dengan kondisi posisi high berubah yaitu $(mid - 1)$ 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 (mid + 1) dan posisi high tetap.

Kelebihan dan Kekurangan Binary Search:

- 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.
- 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.

B. Guided

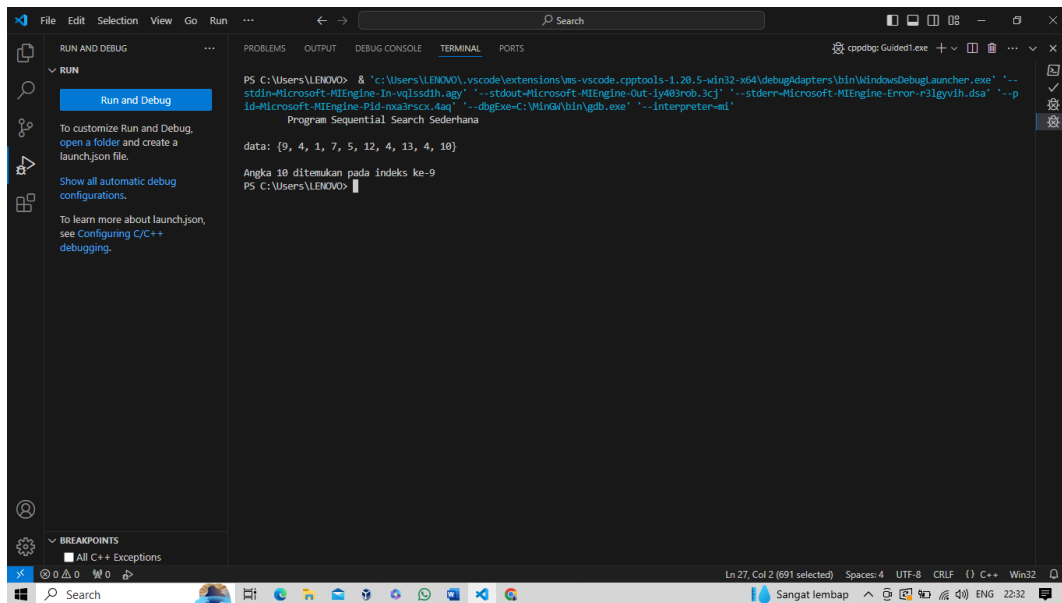
Guided 1

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 << "\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 :



Deskripsi :

Program di atas adalah mencari suatu nilai tertentu dalam sebuah array. Dimulai dengan mendeklarasikan sebuah array data yang berisi 10 elemen dan inisialisasi nilai yang akan dicari (cari) yaitu 10. Kemudian, variabel ketemu digunakan sebagai indikator apakah nilai yang dicari ditemukan atau tidak, dan i sebagai indeks. Dalam loop for, program memeriksa setiap elemen dalam array data untuk melihat apakah elemen tersebut sama dengan nilai cari. Jika ditemukan, variabel ketemu diatur ke true dan loop dihentikan menggunakan break. Setelah pencarian selesai, program mencetak hasil pencarian ke layar. Jika nilai ditemukan, program menampilkan indeks di mana nilai tersebut berada; jika tidak, program memberi tahu bahwa nilai tersebut tidak ditemukan dalam array.

Guided 2

Source Code

```
#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;
    }
}
```

```

    }
    }
    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;
}

```

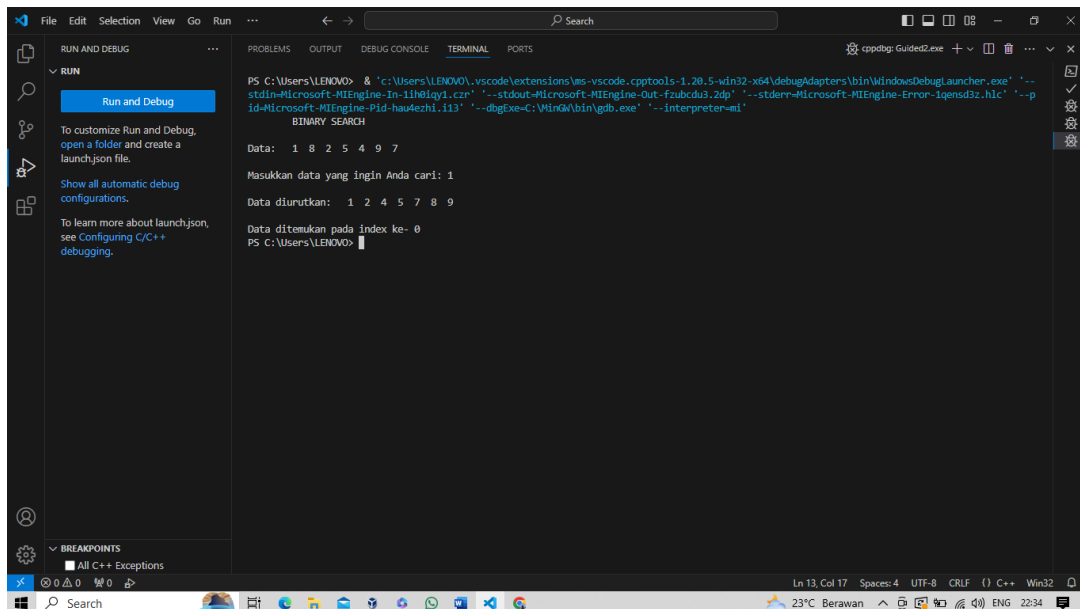
```

    binarysearch();

    _getche();
    return 0;
}

```

Screenshots Output :



Deskripsi :

Program tersebut terdiri dari dua fungsi utama, yaitu `selection_sort` untuk mengurutkan array menggunakan algoritma selection sort dan `binarysearch` untuk melakukan pencarian binary search pada array yang sudah diurutkan. Setelah pengurutan, program meminta pengguna untuk memasukkan nilai yang ingin dicari, kemudian mencari nilai tersebut menggunakan binary search. Hasil pencarian, baik jika nilai ditemukan atau tidak, ditampilkan kepada pengguna. Program juga menyertakan tampilan data awal sebelum pengurutan dan data setelah diurutkan menggunakan selection sort.

C. Unguided

Unguided 1

Source Code

```

#include <iostream>
using namespace std;

void selectionSort(string &huruf, int n)
{

```

```

int i, j, min_091;
for (i = 0; i < n - 1; i++)
{
    min_091 = i;
    for (j = i + 1; j < n; j++)
        if (huruf[j] < huruf[min_091])
            min_091 = j;
    char temp = huruf[i];
    huruf[i] = huruf[min_091];
    huruf[min_091] = temp;
}
}

int binarySearch(string Huruf_091, int kiri, int kanan, char target)
{
    while (kiri <= kanan)
    {
        int mid = kiri + (kanan - kiri) / 2;
        if (Huruf_091[mid] == target)
            return mid;
        if (Huruf_091[mid] < target)
            kiri = mid + 1;
        else
            kanan = mid - 1;
    }
    return -1;
}

int main()
{
    string kalimat_091;
    char input;
    cout << "=====" << endl;
    cout << "Program Menentukan Indeks Pada Huruf" << endl;
    cout << "=====" << endl;
    cout << "Masukkan kalimat yang anda inginkan: ";
    getline(cin, kalimat_091);
    cout << "Masukkan huruf yang anda ingin cari: ";
    cin >> input;
    cout << endl;
    selectionSort(kalimat_091, kalimat_091.size());
    int result = binarySearch(kalimat_091, 0, kalimat_091.size() - 1,
input);
    if (result == -1)
    {
        cout << "Huruf yang anda cari tidak ditemukan!" << endl;
    }
    else
    {

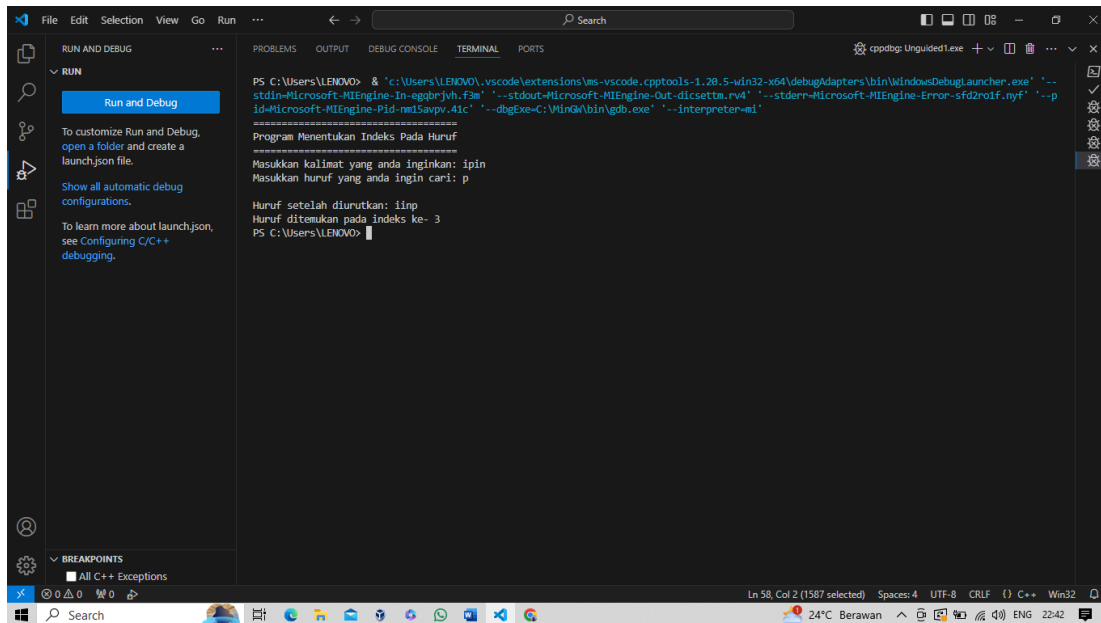
```

```

        cout << "Huruf setelah diurutkan: " << kalimat_091 << endl;
        cout << "Huruf ditemukan pada indeks ke- " << result << endl;
    }
    return 0;
}

```

Screenshots Output :



Deskripsi :

Program tersebut terdiri dari dua fungsi utama, yaitu selectionSort untuk mengurutkan huruf-huruf dalam kalimat secara alfabetis menggunakan algoritma selection sort, dan binarySearch untuk melakukan pencarian biner terhadap huruf yang diurutkan. Setelah pengurutan, pengguna diminta untuk memasukkan kalimat dan huruf yang ingin dicari. Program kemudian mengurutkan huruf dalam kalimat dan mencari huruf yang diinginkan menggunakan binary search. Hasil pencarian, baik jika huruf ditemukan atau tidak, ditampilkan kepada pengguna bersama dengan indeksnya dalam kalimat yang sudah diurutkan.

Unguided 2

Source Code

```

#include <iostream>
using namespace std;

int main()
{
    string kalimat_091;
    int count=0;

```



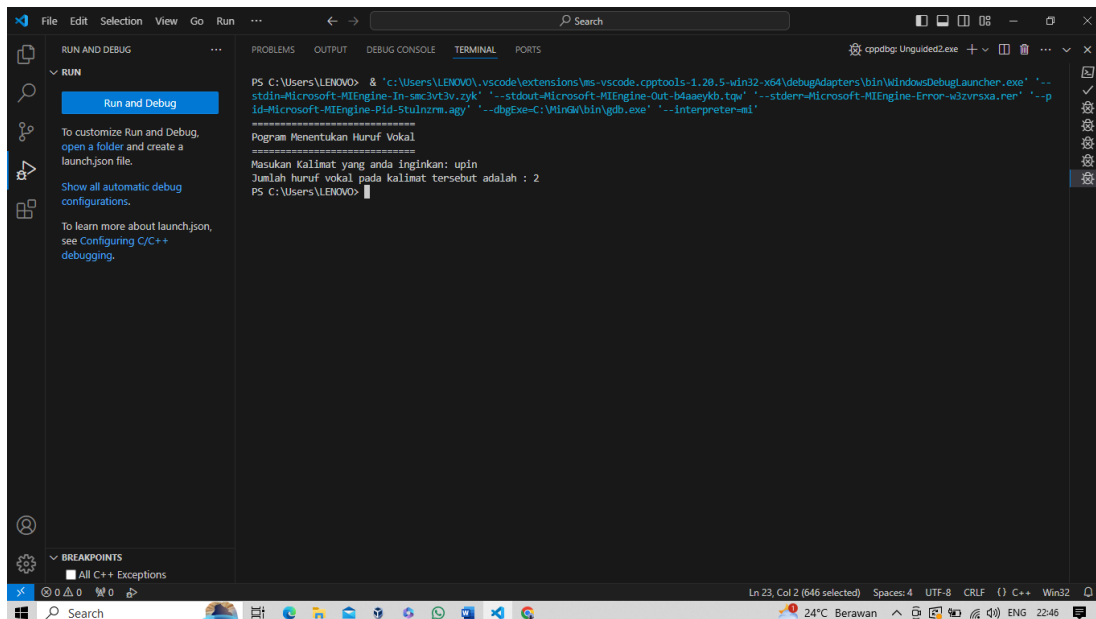
```

cout << "======" << endl;
cout << "Pogram Menentukan Huruf Vokal" << endl;
cout << "======" << endl;
cout << "Masukan Kalimat yang anda inginkan: " ;
cin >> kalimat_091;

for (int i=0; i<kalimat_091.length(); i++)
{
    if(kalimat_091[i]=='a' || kalimat_091[i]=='i' ||
kalimat_091[i]=='u' || kalimat_091[i]=='e' || kalimat_091[i]=='o')
    {
        count++;
    }
}
cout << "Jumlah huruf vokal pada kalimat tersebut adalah : " << count;
}

```

Screenshots Output :



Deskripsi :

Program di atas adalah untuk menghitung jumlah huruf vokal dalam sebuah kalimat yang dimasukkan oleh pengguna. Setelah pengguna memasukkan kalimat, program akan melakukan iterasi melalui setiap karakter dalam kalimat tersebut. Setiap karakter akan diperiksa apakah merupakan huruf vokal ('a', 'i', 'u', 'e', atau 'o'). Jika karakter tersebut adalah huruf vokal, maka variabel hitung akan bertambah satu. Setelah iterasi selesai, program akan mencetak jumlah huruf vokal yang ditemukan dalam kalimat tersebut.

Unguided 3

Source Code

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

int HitungAngka( const int array[], int size, int target_091) {
    int count = 0;

    for (int i = 0; i < size; i++) {
        if (array[i] == target_091) {
            count++;
        }
    }

    return count;
}

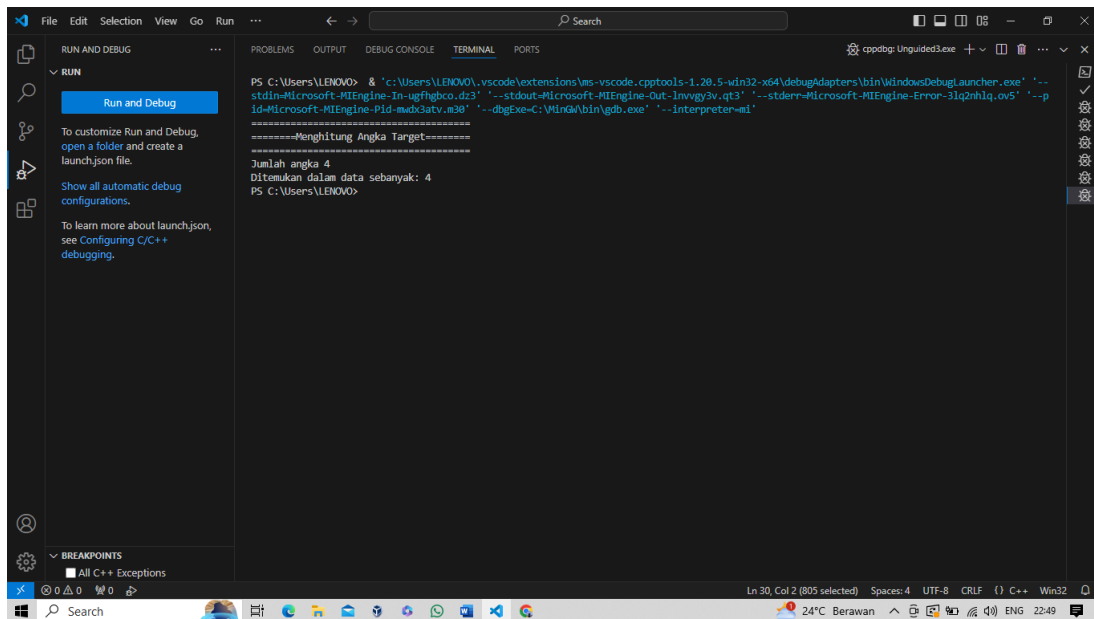
int main() {
    const int size = 10;
    int array[size] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int target_091 = 4;

    int count = HitungAngka(array, size, target_091);

    cout << "=====" << endl;
    cout << "=====Menghitung Angka Target=====" << endl;
    cout << "=====" << endl;
    cout << "Jumlah angka " << target_091 << endl;
    cout << "Ditemukan dalam data sebanyak: " << count << endl;

    return 0;
}
```

Screenshots Output :



```
PS C:\Users\LENOVO> & 'c:\Users\LENOVO\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--
stdin=Microsoft-MIEngine-In-ugthgco.dz3' '--stdout=Microsoft-MIEngine-Out-Invgy3v.q3' '--stderr=Microsoft-MIEngine-Error-3lq2nhlq.ov5' '--p
id=Microsoft-MIEngine-Pid-mad3atv.x38' '--dbgExe=C:\Windows\bin\gdb.exe' '--Interpreter=ml'

=====Menghitung Angka Target=====
Jumlah angka 4
Ditemukan dalam data sebanyak: 4
PS C:\Users\LENOVO>
```

Deskripsi :

Program di atas adalah sebuah program dalam bahasa C++ yang memiliki tujuan untuk menghitung jumlah kemunculan suatu angka target dalam sebuah array. Program tersebut terdiri dari dua fungsi utama, yaitu `HitungAngka` yang menerima sebuah array, ukuran array, dan angka target sebagai parameter dan mengembalikan jumlah kemunculan angka target dalam array, dan fungsi `main` yang merupakan fungsi utama program.

Dalam fungsi `HitungAngka`, program melakukan iterasi melalui setiap elemen dalam array dan memeriksa apakah elemen tersebut sama dengan angka target. Jika iya, variabel `count` akan bertambah satu. Setelah iterasi selesai, nilai `count` akan dikembalikan. Di dalam fungsi `main`, program mendefinisikan sebuah array dengan ukuran 10 dan memasukkan angka-angka ke dalamnya. Selanjutnya, program memanggil fungsi `HitungAngka` dengan array tersebut, ukuran array, dan angka target sebagai argumen. Hasil perhitungan akan dicetak ke layar bersamaan dengan pesan yang menjelaskan bahwa program menghitung jumlah kemunculan angka target dalam data.

D. Kesimpulan

Pada pembelajarannya, kita bisa melihat beberapa konsep dasar pemrograman C++ algoritma searching yang diterapkan:

1. Penggunaan Fungsi

Program menggunakan dua fungsi utama, yaitu `'HitungAngka'` dan `'main'`. Fungsi `'HitungAngka'` digunakan untuk menghitung jumlah kemunculan angka

target dalam array, sementara fungsi ``main`` adalah fungsi utama yang menjalankan program.

2. Parameter dan Pengembalian Fungsi

Fungsi ``HitungAngka`` menerima tiga parameter, yaitu array, ukuran array, dan angka target. Fungsi tersebut mengembalikan jumlah kemunculan angka target dalam array.

3. Iterasi melalui Array

Dengan menggunakan loop ``for``, program melakukan iterasi melalui setiap elemen dalam array untuk memeriksa apakah elemen tersebut sama dengan angka target.

4. Pemanggilan Fungsi

Fungsi ``HitungAngka`` dipanggil di dalam fungsi ``main`` dengan argumen yang sesuai, yaitu array, ukuran array, dan angka target.

Pembelajaran dari codingan tersebut adalah pemahaman tentang penggunaan fungsi, penggunaan parameter dan pengembalian fungsi, serta cara mengakses dan memanipulasi elemen-elemen dalam sebuah array menggunakan loop.

E. Referensi

Alpro, B. (no date) *Searching dalam Bahasa Pemrograman C++*.

<http://sinaualpro.blogspot.com/2018/12/searching-dalam-bahasa-pemrograman-c.html>.

Sarjana, A. (2019) 'Belajar C++ Part 6 (Searching),' *Sarjana Pedia*, 16 February.

<http://www.sarjanapedia.com/2019/02/metode-searching-dalam-cpp.html>.

Wafa, M. (2022) *Program penerapan algoritma searching C++*. <https://mikirinkode.com/program-penerapan-algoritma-searching-c/>.