# LAPORAN PRAKTIKUM

# MODUL 8
# ALGORITMA SEARCHING

**Disusun oleh:**
**Tegar Bangkit Wijaya**
**NIM: 2311102027**

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

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

# BAB I
# TUJUAN PRAKTIKUM

## A. TUJUAN PRAKTIKUM

1. Menyajikan beberapa algoritma pencarian.
2. Menunjukkan bahwa pencarian dapat diselesaikan dengan berbagai algoritma.
3. Memilih algoritma yang paling cocok untuk menyelesaikan suatu permasalahan pemrograman.

# BAB II

# DASAR TEORI

## B. DASAR TEORI

Pencarian (Searching) adalah 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 per satu pada setiap indeks baris atau setiap indeks kolomnya dengan menggunakan teknik perulangan untuk melakukan pencarian data.

Terdapat dua 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, di mana data dalam array dibaca satu demi satu dan diurutkan dari indeks terkecil ke indeks terbesar, maupun sebaliknya.

Konsep Sequential Search:

- 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 ← 0`

2. `ketemu ← false`

3. Selama (tidak ketemu) dan (i <= N) kerjakan baris 4:

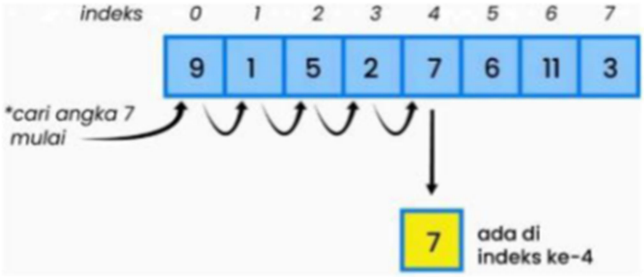
   - Jika (Data[i] = x) maka ketemu ← true, jika tidak i ← i + 1

4. Jika (ketemu) maka i adalah indeks dari data yang dicari, jika tidak data tidak ditemukan.

Dibawah 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}




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 merupakan salah satu metode pencarian yang termasuk dalam interval search, di mana algoritma ini digunakan untuk mencari data dalam array/list yang sudah terurut. Dalam metode ini, data harus diurutkan terlebih dahulu, dan pencarian dilakukan dengan membagi data menjadi dua bagian secara logis pada setiap tahap pencarian.

Pada penerapannya, algoritma Binary Search sering digabungkan dengan algoritma pengurutan (sorting) karena data yang akan dicari harus sudah terurut terlebih dahulu.

Konsep Binary Search:

- Data diambil dari posisi awal (1) hingga posisi akhir (N).

- Data kemudian dibagi menjadi dua bagian untuk menentukan posisi tengah.

- Data yang dicari kemudian dibandingkan dengan data di posisi tengah; apakah lebih besar atau lebih kecil.

- Jika data yang dicari lebih besar dari data di posisi tengah, maka pencarian dilanjutkan pada bagian kanan dari data tengah. Proses pencarian ini dilakukan dengan membagi data pada bagian kanan, menggunakan posisi tengah sebagai posisi awal yang baru.

Dengan demikian, Binary Search adalah metode efisien untuk mencari data dalam kumpulan data yang sudah terurut, menggunakan prinsip pembagian data menjadi dua secara berulang hingga data yang dicari ditemukan atau dipastikan tidak ada.

# BAB III

## GUIDED

### 1. 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 << "\nAngka " << cari << " ditemukan pada indeks ke-"
<< i << endl;
    }
    else
    {
        cout << cari << " tidak dapat ditemukan pada data." <<
endl;
    }
```

```
    return 0;
}
```

**Screenshoot program**


**Deskripsi program**

Program ini bertujuan untuk mencari keberadaan suatu angka dalam sebuah array dengan menelusuri setiap elemen satu per satu dari awal hingga akhir array. Pertama, program mendefinisikan sebuah array yang berisi 10 elemen dan beberapa angka sebagai data yang akan dicari. Kemudian, program menggunakan algoritma Sequential Search untuk mencari angka tertentu dalam array tersebut. Jika angka tersebut ditemukan, program akan menampilkan pesan bahwa angka tersebut ditemukan beserta indeksnya dalam array. Jika tidak ditemukan, program akan menampilkan pesan bahwa angka tersebut tidak ada dalam data.

2. **Guided 2**

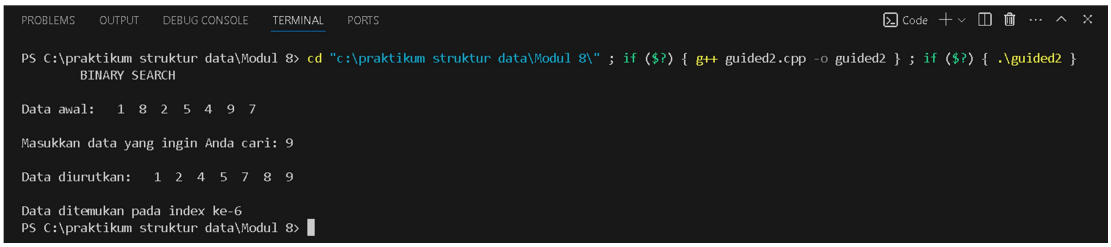
**Source code**

```
#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;
}
```

**Screenshoot program**




**Deskripsi program**

Program ini bertujuan untuk mencari keberadaan suatu angka dalam array yang telah diurutkan secara menaik. Pertama, program mendefinisikan sebuah array dengan panjang 7 dan beberapa angka sebagai data yang akan dicari. Kemudian, program

mengimplementasikan fungsi `selection_sort()` untuk mengurutkan data dalam array menggunakan algoritma Selection Sort. Setelah itu, program meminta pengguna untuk memasukkan angka yang ingin dicari. Setelah angka dimasukkan, program menampilkan data yang telah diurutkan. Selanjutnya, program mencari angka yang dimasukkan oleh pengguna dalam array yang telah diurutkan menggunakan algoritma Binary Search. Jika angka tersebut ditemukan, program akan menampilkan indeks tempat angka tersebut ditemukan. Jika tidak ditemukan, program akan memberikan pesan bahwa data tidak ditemukan dalam array.

# LATIHAN KELAS - UNGUIDED

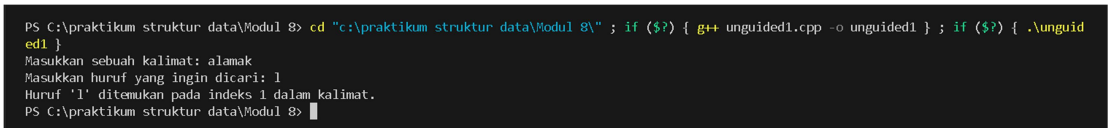
## 1. Unguided 1

### Source code

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

// Fungsi untuk melakukan pencarian linear pada string
int linearSearch(const string &kalimat, char hurufDicari)
{
    for (int i = 0; i < kalimat.length(); ++i)
    {
        if (kalimat[i] == hurufDicari)
        {
            return i; // Mengembalikan indeks jika huruf ditemukan
        }
    }
    return -1; // Mengembalikan -1 jika huruf tidak ditemukan
}

int main()
{
    string kalimat;
    char hurufDicari;
    cout << "Masukkan sebuah kalimat: ";
    getline(cin, kalimat);
    cout << "Masukkan huruf yang ingin dicari: ";
    cin >> hurufDicari;

    // Melakukan pencarian linear pada kalimat
    int hasilPencarian = linearSearch(kalimat, hurufDicari);
    if (hasilPencarian != -1)
    {
        cout << "Huruf '" << hurufDicari << "' ditemukan pada
indeks " << hasilPencarian << " dalam kalimat." << endl;
    }
    else
```

```
    {
        cout << "Huruf '" << hurufDicari << "' tidak ditemukan
dalam kalimat." << endl;
    }

    return 0;
}
```

**Screenshoot program**




**Deskripsi program**

Program ini bertujuan untuk mencari indeks pertama kemunculan suatu huruf dalam sebuah kalimat yang dimasukkan oleh pengguna. Pada awalnya, program menggunakan `#include` untuk menyertakan pustaka yang diperlukan seperti `iostream` dan `string`. Kemudian, program mendefinisikan sebuah fungsi bernama `linearSearch()` yang menerima dua parameter: sebuah string (kalimat) dan sebuah karakter (hurufDicari). Fungsi ini melakukan pencarian linear pada string untuk mencari huruf yang diinginkan dan mengembalikan indeks pertama kemunculan huruf tersebut dalam kalimat. Dalam fungsi `main()`, program meminta pengguna untuk memasukkan sebuah kalimat dan sebuah huruf yang ingin dicari. Setelah itu, program memanggil fungsi `linearSearch()` untuk mencari huruf yang dimasukkan dalam kalimat, kemudian menampilkan hasilnya ke layar. Jika huruf tersebut ditemukan, program akan menampilkan indeks pertama kemunculannya dalam kalimat. Jika tidak ditemukan, program akan menampilkan pesan bahwa huruf tersebut tidak ditemukan dalam kalimat.

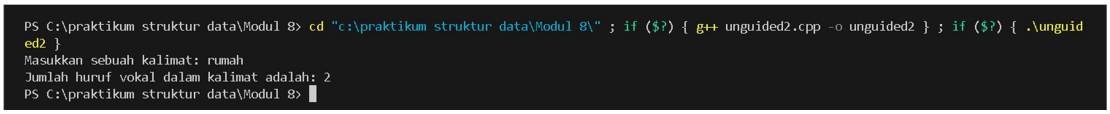
## 2. Unguided 2

### Source code

```
#include <iostream>
#include <string>
#include <cctype>

using namespace std;

// Fungsi untuk menghitung jumlah huruf vokal dalam sebuah kalimat
int hitungVokal(const string &teks)
{
    int jumlahVokal = 0;
    for (char karakter : teks)
    {
        // Mengubah huruf menjadi huruf kecil untuk memudahkan
perbandingan
        char huruf = tolower(karakter);
        // Memeriksa apakah karakter tersebut adalah huruf vokal
        if (huruf == 'a' || huruf == 'e' || huruf == 'i' || huruf
== 'o' || huruf == 'u')
        {
            jumlahVokal++;
        }
    }
    return jumlahVokal;
}

int main()
{
    string teks;
    cout << "Masukkan sebuah kalimat: ";
    getline(cin, teks);

    // Menghitung jumlah huruf vokal dalam teks
    int totalVokal = hitungVokal(teks);
    cout << "Jumlah huruf vokal dalam kalimat adalah: " <<
```

```
totalVokal << endl;

    return 0;
}
```

**Screenshoot program**




**Deskripsi program**

Program tersebut adalah contoh implementasi yang bertujuan untuk menghitung jumlah huruf vokal dalam sebuah kalimat yang diberikan oleh pengguna. Pertama-tama, program menggunakan `#include` untuk memasukkan pustaka yang diperlukan seperti iostream, string, dan cctype. Selanjutnya, program mendefinisikan sebuah fungsi bernama `hitungVokal()` yang menerima string sebagai parameter dan mengembalikan jumlah huruf vokal dalam string tersebut. Fungsi ini menggunakan perulangan `for` untuk menelusuri setiap karakter dalam string dan menggunakan fungsi `tolower()` untuk mengubah karakter menjadi huruf kecil agar memudahkan perbandingan.

Di dalam fungsi `main()`, program meminta pengguna untuk memasukkan sebuah kalimat menggunakan fungsi `getline()` dan menyimpannya dalam variabel `teks`. Setelah itu, program memanggil fungsi `hitungVokal()` untuk menghitung jumlah huruf vokal dalam kalimat yang dimasukkan, kemudian menampilkan hasilnya ke layar.

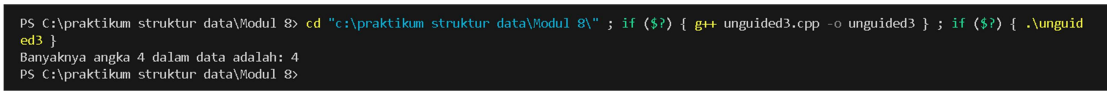
3. **Unguided 3**

**Source code**

```
#include <iostream>
using namespace std;
// Fungsi untuk melakukan pencarian Sequential Search
```

```cpp
int cariSequensial(int data[], int ukuran, int target)
{
    int hitung = 0;
    for (int i = 0; i < ukuran; ++i)
    {
        if (data[i] == target)
        {
            hitung++;
        }
    }
    return hitung;
}
int main()
{
    int data[] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int ukuran = sizeof(data) / sizeof(data[0]);
    int target = 4;
    // Melakukan pencarian Sequential
    int jumlah = cariSequensial(data, ukuran, target);
    cout << "Banyaknya angka 4 dalam data adalah: " << jumlah <<
endl;
    return 0;
}
```

**Screenshoot program**




**Deskripsi program**

Program tersebut bertujuan untuk mencari berapa kali suatu angka tertentu muncul dalam sebuah array. Awalnya, program mendefinisikan sebuah array dengan sejumlah angka yang akan dicari. Kemudian, program menghitung ukuran array tersebut. Selanjutnya, program memanggil fungsi `cariSequensial()` untuk melakukan pencarian berulang kali terhadap angka target dalam array menggunakan algoritma Sequential Search. Fungsi ini akan mengembalikan jumlah kemunculan

angka target dalam array. Setelah pencarian selesai, program akan menampilkan jumlah kemunculan angka target tersebut dalam array.

# BAB IV

# KESIMPULAN

Kesimpulan dari program ini dapat dirangkum sebagai berikut:

1.Penggunaan struktur dasar bahasa C++ terlihat dalam program ini, termasuk tipe data string dan fungsi bawaan seperti getline() untuk mengambil input dari pengguna.

2.Program menerapkan algoritma pencarian linear untuk mencari karakter dalam sebuah string. Algoritma ini sederhana dan cocok untuk data kecil atau tidak terurut.

3.Implementasi fungsi dasar bahasa C++ seperti perulangan for dan if digunakan untuk iterasi dan pengecekan kondisi.

4.Program memberikan pengalaman interaktif kepada pengguna dengan memperbolehkan mereka melihat hasil pencarian huruf dalam kalimat yang dimasukkan.