

**LAPORAN PRAKTIKUM STRUKTUR  
DATA DAN ALGORITMA**

**MODUL VIII  
ALGORITMA SEARCHING**



**Disusun Oleh :**

NAMA : RIFKY DWI MAHARDIKA

NIM : 2311102043

**Dosen**

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

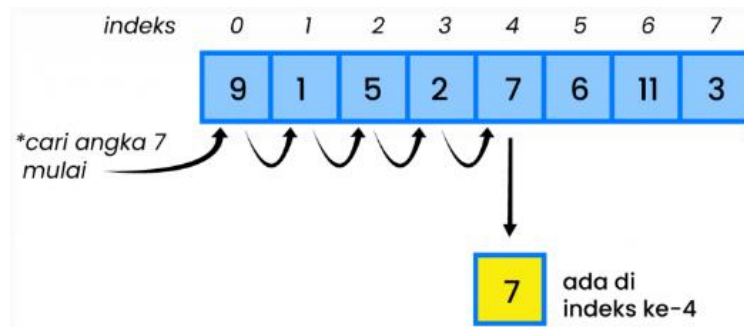
**PROGRAM STUDI S1 TEKNIK INFORMATIKA  
FAKULTAS INFORMATIKA  
INSTITUT TEKNOLOGI TELKOM PURWOKERTO  
2024**

## A. Dasar Teori

Algoritma pencarian adalah formula unik yang digunakan mesin pencari untuk mengambil informasi spesifik yang disimpan dalam struktur data. Algoritma pencarian inilah yang menentukan penting tidaknya suatu halaman web dan kontennya. Dalam ilmu komputer, tingkat kecepatan sebuah aplikasi terletak pada ketepatan penggunaan algoritma pencarian. Algoritma pencarian yang terdapat dalam mesin pencari menggunakan kata kunci untuk menangkap semua data yang ada. Efisiensi algoritma pencarian diukur dengan sebanyak apa perbandingan kunci pencarian dilakukan. Notasi yang digunakan dalam algoritma pencarian adalah  $O(n)$ , dimana  $n$  adalah jumlah perbandingan yang dilakukan.

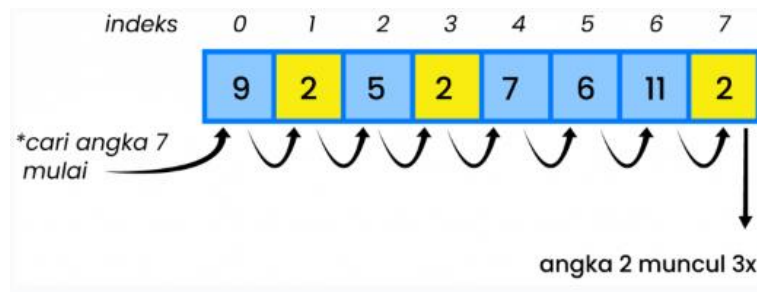
### a. Sequential Search

Algoritma pencarian Sekuensial adalah salah satu algoritma pencarian data yang biasa digunakan untuk data yang berpola acak atau belum terurut. Algoritma ini akan mencari data sesuai kata kunci yang diberikan mulai dari elemen awal pada array hingga elemen akhir array. Kemungkinan terbaik (best case) ketika menggunakan algoritma ini adalah jika data yang dicari terletak di indeks awal array sehingga hanya membutuhkan sedikit waktu pencarian. Sedangkan kemungkinan terburuknya (worst case) adalah jika data yang dicari ternyata terletak dibagian akhir dari array sehingga pencarian data akan memakan waktu yang lama. Berikut adalah ilustrasi dari pencarian sekuensial:



**Gambar Ilustrasi Sequential Search**

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.

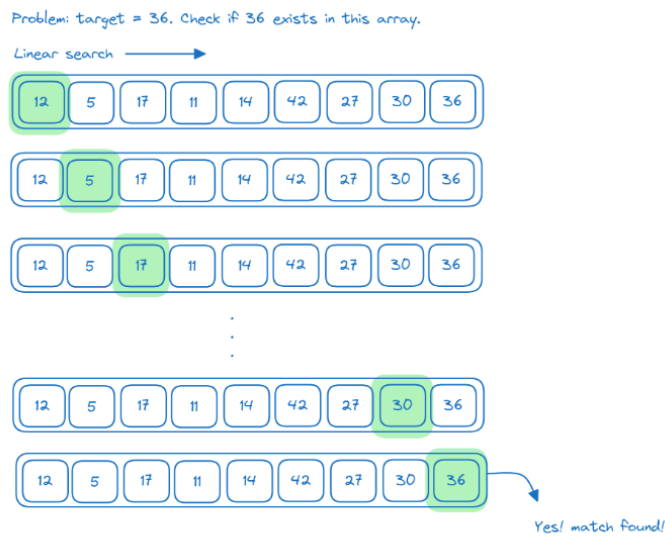


**Gambar Ilustrasi Sequential Search dengan Counter**

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.

## b. Binary Search

Binary Search adalah algoritma pencarian untuk menemukan posisi elemen dalam array yang diurutkan. Ada dua metode yang dapat digunakan dalam algoritma pencarian biner, yakni metode iterasi dan rekursif. Metode iterasi adalah metode perulangan, sedangkan metode rekursif adalah metode yang mengikuti pendekatan bagi dan taklukkan.



**Gambar contoh Binary Search**

Jadi pencarian linier atau berurutan adalah yang terbaik yang dapat Anda lakukan ketika mencari melalui urutan yang tidak diurutkan.

## 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 << "\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-" << i
    << endl;
    }
    else
    {
        cout << cari << " tidak dapat ditemukan pada data." << endl;
    }
    return 0;
}
```

### Screenshots Output

```
PS C:\Users\ASUS\Documents\semester 2\struktur data algoritma\laprak 8> cd "C:\Users\ASUS\Documents\semester 2\struktur data algoritma\laprak 8\" ; if ($?) { g++ guided1.cpp -o guided1 }
Program Sequential Search Sederhana

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

angka 10 ditemukan pada indeks ke-9
```

Nama : Rifky Dwi Mahardika  
NIM : 2311102043

## **Deskripsi**

Program ini mengimplementasikan algoritma pencarian berurutan (Sequential Search) untuk menemukan elemen tertentu dalam sebuah array. Program dimulai dengan mengimpor pustaka input/output standar dan mendefinisikan fungsi utama “main”. Di dalam fungsi “main”, sebuah array “data” berukuran 10 diinisialisasi dengan nilai-nilai tertentu, dan variabel “cari” ditetapkan dengan nilai elemen yang ingin dicari (dalam hal ini 10). Variabel boolean “ketemu” digunakan untuk menandai apakah elemen yang dicari ditemukan atau tidak. Algoritma pencarian berurutan dilakukan dengan iterasi melalui setiap elemen array menggunakan loop “for”, dan jika elemen yang dicari ditemukan, variabel “ketemu” diatur menjadi “true” dan loop berhenti. Setelah pencarian selesai, program menampilkan hasil pencarian: apakah elemen tersebut ditemukan beserta indeksnya atau tidak ditemukan sama sekali. Output program menampilkan bahwa elemen 10 ditemukan pada indeks ke-9 dalam array.

## Guided 2

```
#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;
}

```

## Screenshots Output



```

PS C:\Users\ASUS\Documents\semester 2\struktur data algoritma\laprak 8>
ak 8\" ; if ($?) { g++ guided2.cpp -o guided2 } ; if ($?) { .\guided2 }
        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

Program ini menggabungkan algoritma pengurutan selection sort dengan pencarian biner (binary search) untuk menemukan elemen tertentu dalam array. Setelah mengimpor

pustaka yang diperlukan dan mendefinisikan array “data” berukuran 7 dengan nilai awal tertentu, program mendeklarasikan variabel global “cari” yang akan menyimpan elemen yang dicari. Fungsi “selection\_sort” mengurutkan array “data” dengan algoritma selection sort, yang melibatkan iterasi melalui elemen-elemen array untuk menemukan nilai minimum dan menukarnya dengan nilai di posisi saat ini. Fungsi “binarysearch” kemudian mencari elemen “cari” dalam array yang sudah diurutkan menggunakan algoritma pencarian biner, yang membagi array menjadi dua bagian pada setiap langkah untuk mempersempit pencarian hingga elemen ditemukan atau seluruh array telah diperiksa. Fungsi utama “main” menampilkan data awal, meminta pengguna memasukkan nilai yang akan dicari, mengurutkan data menggunakan “selection\_sort”, menampilkan data yang sudah diurutkan, dan memanggil “binarysearch” untuk mencari elemen tersebut. Jika elemen ditemukan, program menampilkan indeksinya; jika tidak, program menyatakan bahwa elemen tidak ditemukan. Program diakhiri dengan menunggu input dari pengguna sebelum keluar.



## C. UNGUIDED

### Unguided 1

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

void selection_sort(char arr[], int n) {
    int i, j, min_idx;
    char temp;
    for (i = 0; i < n-1; i++) {
        min_idx = i;
        for (j = i+1; j < n; j++)
            if (arr[j] < arr[min_idx])
                min_idx = j;
        temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}

int binary_search(char arr[], int n, char x) {
    int l = 0, r = n - 1;
    while (l <= r) {
        int m = l + (r - l) / 2;
        if (arr[m] == x) return m;
        if (arr[m] < x) l = m + 1;
        else r = m - 1;
    }
    return -1;
}

int main() {
    string kalimat;
    char cari;

    cout << "===== " << endl;
    cout << "    Program Binary Search    " << endl;
    cout << "===== " << endl;

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

    cout << "Masukkan huruf yang ingin dicari: ";
    cin >> cari;
```

```

string temp;
for (char c : kalimat) {
    if (!isspace(c)) {
        temp += c;
    }
}

int n = temp.size();
char data[n + 1];
strcpy(data, temp.c_str());

selection_sort(data, n);

cout << "\nHuruf-huruf yang diurutkan: ";
for (int i = 0; i < n; i++) {
    cout << data[i] << " ";
}
cout << endl;

int result = binary_search(data, n, cari);

cout << "\nHasil Pencarian:" << endl;
cout << "-----"
-----" << endl;
if (result != -1) {
    cout << "Huruf '" << cari << "' ditemukan pada indeks ke-" <<
result << " dalam array yang diurutkan." << endl;
} else {
    cout << "Huruf '" << cari << "' tidak ditemukan dalam kalimat."
<< endl;
}
cout << "-----"
-----" << endl;

return 0;
}

```

## Screenshots Output

```
PS C:\Users\ASUS\Documents\semester 2\struktur data algoritma\laprak 8> cd "C:\Use
ak 8\" ; if ($?) { g++ unguided1.cpp -o unguided1 } ;
=====
Program Binary Search
=====
Masukkan kalimat: Saya suka makan nasi goreng
Masukkan huruf yang ingin dicari: u

Huruf-huruf yang diurutkan: S a a a a a e g g i k k m n n n o r s s u y

Hasil Pencarian:
-----
Huruf 'u' ditemukan pada indeks ke-21 dalam array yang diurutkan.
-----
```

Nama : Rifky Dwi Mahardika  
NIM : 2311102043

## Deskripsi

Program ini menggunakan algoritma selection sort dan binary search untuk mencari huruf dalam sebuah kalimat yang diberikan pengguna. Program mengimpor “iostream” dan “cstring”, menggunakan “selection\_sort” untuk mengurutkan huruf-huruf dalam array karakter dengan mencari elemen terkecil dan menukarnya dengan elemen di posisi saat ini, dan “binary\_search” untuk mencari huruf yang diminta pengguna dalam array yang sudah diurutkan. Dalam fungsi “main”, program meminta pengguna memasukkan kalimat dan huruf yang ingin dicari, menghapus spasi dari kalimat, membentuk array karakter, mengurutkan array menggunakan “selection\_sort”, dan menampilkan huruf-huruf yang sudah diurutkan. Kemudian, program menggunakan “binary\_search” untuk mencari huruf yang diminta dan menampilkan hasil pencariannya dalam format yang rapi dan mudah dipahami, termasuk antarmuka yang menampilkan header program, huruf-huruf yang diurutkan, dan hasil pencarian.

## Unguided 2

```
#include <iostream>
#include <string>
#include <algorithm>

using namespace std;

int countVowels(string sentence) {
    int count = 0;
    string vowels = "aeiou";
    // Mengubah kalimat menjadi lowercase agar pencarian case
    insensitive
    transform(sentence.begin(), sentence.end(), sentence.begin(),
    ::tolower);
    for (char c : sentence) {
        if (vowels.find(c) != string::npos) {
            count++;
        }
    }
    return count;
}

int main() {
    string sentence;

    cout << "=====" << endl;
    cout << "          Program Penghitung Vokal          " << endl;
    cout << "=====" << endl;

    cout << "\nMasukkan kalimat: ";
    getline(cin, sentence);

    int vowelCount = countVowels(sentence);

    cout << "\n" << endl;
    cout << "Jumlah Vokal: " << endl;
    cout << "-----" << endl;
    cout << "kalimat yang dimasukkan memiliki " << vowelCount << " huruf
vokal." << endl;
    cout << "-----" << endl;
    return 0;
}
```

## Screenshots Output

```
PS C:\Users\ASUS\Documents\semester 2\struktur data algoritma\laprak 8> cd "c:\Users\AS
ak 8\" ; if ($?) { g++ unguided2.cpp -o unguided2 } ; if ($?) { .\unguided2 }
=====
Program Penghitung Vokal
=====

Masukkan kalimat: cuaca hari ini cukup cerah dan berawan

Jumlah Vokal:
-----
kalimat yang dimasukkan memiliki 15 huruf vokal.
-----
```

+

Nama : Rifky Dwi Mahardika  
NIM : 2311102043

## Deskripsi

Program tersebut adalah untuk menghitung jumlah huruf vokal dalam sebuah kalimat yang dimasukkan oleh pengguna. Dengan menggunakan “iostream”, “string”, dan “algorithm”, program melakukan transformasi pada kalimat agar pencarian huruf vokal tidak sensitif terhadap huruf besar/kecil. Fungsi “countVowels” digunakan untuk menghitung jumlah huruf vokal dalam kalimat, di mana setiap karakter dari kalimat diperiksa apakah terdapat dalam string yang berisi huruf vokal (a, e, i, o, u). Hasil perhitungan tersebut kemudian ditampilkan kepada pengguna melalui tampilan yang informatif dan menarik, meliputi header program, permintaan untuk memasukkan kalimat, serta hasil perhitungan jumlah huruf vokal dalam kalimat tersebut.

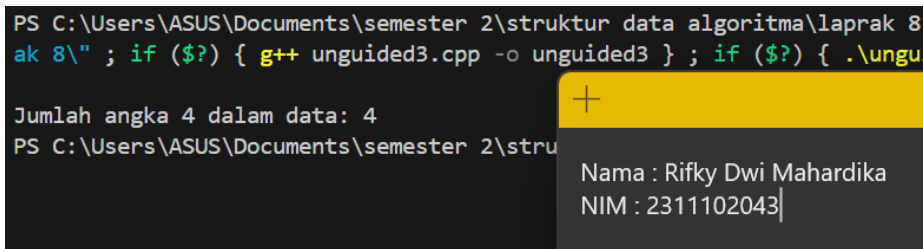
### Unguided 3

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

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

    //array dan menghitung jumlah kemunculan angka 4
    int size = sizeof(data) / sizeof(data[0]);
    int target = 4;
    int count = 0;
    for (int i = 0; i < size; i++)
    {
        if (data[i] == target)
        {
            count++;
        }
    }
    cout << "\n";
    cout << "Jumlah angka 4 dalam data: " << count << endl;
    return 0;
}
```

### Screenshots Output



```
PS C:\Users\ASUS\Documents\semester 2\struktur data algoritma\laprak 8
ak 8\" ; if ($?) { g++ unguided3.cpp -o unguided3 } ; if ($?) { .\ungu
Jumlah angka 4 dalam data: 4
PS C:\Users\ASUS\Documents\semester 2\stru
Nama : Rifky Dwi Mahardika
NIM : 2311102043
```

### Deskripsi

Program tersebut adalah untuk menghitung berapa kali angka 4 muncul dalam array yang telah didefinisikan. Pertama-tama, program mendefinisikan array “data” yang berisi serangkaian bilangan bulat. Variabel target diinisialisasi dengan nilai 4, yang merupakan angka yang ingin dihitung kemunculannya dalam array. Lalu, program menggunakan loop for untuk mengiterasi melalui setiap elemen array. Setiap kali program menemukan elemen yang sama dengan target (yaitu, angka 4), variabel count akan bertambah satu.

#### **D. Kesimpulan**

Algoritma searching adalah alat penting dalam ilmu komputer yang digunakan untuk menemukan item tertentu dalam kumpulan data. Algoritme ini dirancang untuk menavigasi struktur data secara efisien untuk menemukan informasi yang diinginkan, menjadikannya dasar dalam berbagai aplikasi seperti database, mesin pencari web, dan banyak lagi. Searching adalah proses mendasar untuk menemukan elemen atau item tertentu dalam kumpulan data. Kumpulan data ini dapat mengambil berbagai bentuk, seperti array, atau representasi terstruktur lainnya. Tujuan utama searching adalah untuk menentukan apakah elemen yang diinginkan ada dalam data, dan jika ada, untuk mengidentifikasi lokasi tepatnya atau mengambilnya kembali.

#### **E. Referensi**

[1] Asisten Praktikum. 2024. Modul VIII : ALGORITMA SEARCHING

[2]Bala Priya C. 12 Juli 2023. Binary Search – Algorithm and Time Complexity Explained. Diakses pada 2 Juni 2024, dari <https://www.freecodecamp.org/news/binary-search-algorithm-and-time-complexity-explained/>

[3] Muhammad Wafa. 06 Desember 2021. Sequential Search – Algoritma Pencarian. Diakses pada 2 Juni 2024, dari <https://mikirinkode.com/sequential-search/>