

**LAPORAN PRAKTIKUM  
STRUKTUR DATA DAN ALGORITMA**

**MODUL VIII**



**DISUSUN OLEH :**

**Nama : Fahrur Rizqi**

**NIM : 2311102059**

**Dosen**

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

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

## A. Dasar Teori

### 1. Pencarain sekuensial (sequential search)

Pencarian sekuensial (sequensial search) atau sering disebut pencarian linier menggunakan prinsip sebagai berikut : data yang ada dibandingkan satu persatu secara berurutan dengan yang dicari.

Pada dasarnya, pencarian ini hanya melakukan pegulangan dari 1 sampai dengan jumlah data. Pada setiap perulangan, dibandingkan data ke-i dengan yang dicari. Apabila sama, berarti data telah ditemukan. Sebaliknya apabila sampai akhir pengulangan, tidak ada yang sama berarti data tidak ada.

#### Algoritma Linear Searching:

1. Input x (data yang dicari)
2. Bandingkan x dengan data ke-i sampai n
3. Jika ada data yang sama dengan x maka cetak pesan "ada"
4. Jika tidak ada data yang sama dengan x cetak pesan "tidak ada"

#### Contoh Program pencarian sekuensial menggunakan algoritma linier searching

##### Source code :

```
#include <iostream>

using namespace std;

main() {
    int i;
    int cari, ketemu;
    int A[100];

    cout<<"PROGRAM SEARCHING Liniear\n";
    cout<<"masukan 7 buah data : \n\n";
    for(i=1; i<=7; i++)
    {
        cout<<"masukan data ke-"<<i<<" = ";
        cin>>A[i];
    }
    cout<<endl;
    cout<<"Input bilangan yang cari : ";cin>>cari;
    cout<<endl;

    ketemu=0;
    for(i=0; i<=7; i++)
    {
        if (A[i]==cari)
        {
            ketemu=1;
            cout<<"Data ditemukan pada indeks ke-"<<i<<endl;
        }
    }
}
```

```

    }
    if (ketemu==0)
    {
        cout<<"Data tidak ditemukan"<<endl;
    }
}

```

## 2. Pencarian Bagi Dua (Binary Search)

Salah satu keuntungan data yang terurut adalah memudah pencarian, yang dalam hal ini adalah pencarian bagi dua. Sebenarnya dalam kehidupan sehari-hari kita sering menerapkan algoritma ini. Untuk mencari kata tertentu dalam kamus (misalnya kamus bahasa Inggris), kita tidak membuka kamus tersebut dari halaman awal sampai halaman akhir satu persatu, namun kita mencari dengan cara membelah atau membagi halaman-halaman buku tersebut. Begitu seterusnya sampai kita menemukan kata yang dicari

### 1. Langkah Pertama:

Bagi 2 elemen larik pada elemen tengah. Elemen tengah adalah elemen dengan indeks  $k = (Ia + Ib) \div 2$ . (Elemen tengah,  $L[k]$ , membagi larik menjadi 2 bagian  $L [Ia...k-1]$  dan bagian kanan  $L[k+1...Ib]$ )

### 2. Langkah Kedua:

Periksa apakah  $L[k]=X$ . Jika  $L[k]=X$ , pencarian dihentikan sebab  $X$  sudah ditemukan, tetapi jika tidak, harus ditentukan apakah pencarian pada larik bagian kiri atau larik bagian kanan. Jika  $L[k] < X$  maka pencarian dilakukan pada larik kiri. Sebaliknya jika  $L[k] > X$  maka pencarian dilakukan pada larik bagian kanan.

### 3. Langkah Ketiga:

Ulangi langkah 1 sampai  $X$  atau  $Ia > Ib$ .

## Contoh Program pencarian biner (binary search)

### Source code :

```

#include <iostream>

using namespace std;

int main() {
    const int arraySize = 5;
    int target;
    int array[arraySize] = {1, 2, 3, 4, 5};
    int first, mid, last;

    cout << "Masukan angka yang dicari: ";
    cin >> target;

    first = 0;
    last = arraySize - 1; // Mengubah inisialisasi last ke
    arraySize - 1

```

```

while (first <= last) {
    mid = (first + last) / 2;
    if (target > array[mid]) {
        first = mid + 1;
    } else if (target < array[mid]) {
        last = mid - 1; // Mengubah last ke mid - 1
    } else {
        break; // Menggunakan break untuk keluar dari loop
ketika target ditemukan
    }
}

if (first <= last) { // Memeriksa apakah target ditemukan
    cout << "Angka ditemukan." << endl;
} else {
    cout << "Angka tidak ditemukan." << endl;
}

return 0; // Mengembalikan 0 untuk menandakan program selesai
dengan sukses
}

```

## B. Guided

### Guided 1

Sourcode :

```

#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;
}

```

Output:

```

PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8> cd "c:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8\" ; if ($?) { g++ guided1.cpp -o guided1 } ; if ($?) { .\guided1 }
Program Sequential Search Sederhana

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

angka 10 ditemukan pada indeks ke -9
PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Lap

```

Deskripsi :

Program ini merupakan implementasi dari algoritma pencarian berurutan untuk mencari angka dalam sebuah array. Program dimulai dengan menginisialisasi array data yang berisi 10 elemen dan angka cari yang akan dicari. Kemudian, program menggunakan loop for untuk memeriksa setiap elemen dalam array. Jika elemen yang dicari ditemukan, variabel ketemu diubah menjadi true dan loop dihentikan. Setelah proses pencarian selesai, program menampilkan isi array dan hasil pencarian di konsol. Jika angka yang dicari ditemukan, program mencetak indeks tempat angka tersebut berada; jika tidak ditemukan, program mencetak pesan bahwa angka tidak ditemukan. Program ditutup dengan mengembalikan nilai 0 sebagai tanda bahwa program berakhir dengan sukses.

## Guided 2

Sourcode :

```

#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;
}

```

Output :

```

PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8> cd "c:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8\" ; if ($?) { g++ guided2.cpp -o guided2 } ; if ($?) { .\guided2 }
    BINARY SEARCH

Data awal:   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
PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8>

```

Deskripsi:

Program ini merupakan implementasi dari algoritma Selection Sort dan pencarian biner (Binary Search). Array `DATA` berisi 7 elemen diurutkan menggunakan fungsi `selection\_sort`, yang menukar elemen terkecil dengan elemen saat ini. Fungsi `binarysearch` mencari angka dalam array yang telah diurutkan dengan membagi array menjadi dua bagian hingga angka ditemukan atau semua elemen diperiksa. Di dalam fungsi `main`, program menampilkan elemen awal array, meminta pengguna memasukkan angka yang akan dicari, mengurutkan array, menampilkan array yang sudah diurutkan, lalu mencari angka tersebut menggunakan pencarian biner. Hasil pencarian, apakah angka ditemukan atau tidak, ditampilkan kepada pengguna. Program diakhiri dengan mengembalikan `EXIT\_SUCCESS` sebagai tanda berakhirnya program dengan sukses.

## C. Unguided

### Unguided 1

Sourcode:

```
#include <iostream>
```

```

#include <string>
#include <algorithm>
using namespace std;

int binarySearch(string kalimat, char huruf)
{
    sort(kalimat.begin(), kalimat.end());

    int left = 0;
    int right = kalimat.length() - 1;

    while (left <= right)
    {
        int mid = left + (right - left) / 2;

        if (kalimat[mid] == huruf)
        {
            return mid;
        }
        else if (kalimat[mid] < huruf)
        {
            left = mid + 1;
        }
        else
        {
            right = mid - 1;
        }
    }

    return -1;
}

int main()
{
    string kalimat;
    char huruf;

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

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

    int indeks = binarySearch(kalimat, huruf);

    if (indeks == -1)
    {
        cout << "Huruf " << huruf << " tidak ditemukan dalam kalimat." << endl;
    }
}

```



```

    else
    {
        cout << "Huruf " << huruf << " ditemukan pada indeks " << indeks << " dalam
kalimat." << endl;
    }

    return 0;
}

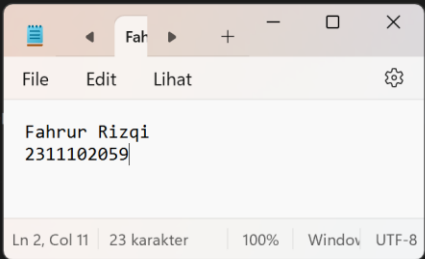
```

Output:

```

PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8> cd "c:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8\" ; if ($?) { g++ unguided1.cpp -o unguided1 } ; if ($?) { .\unguided1 }
Masukkan kalimat: sayang
Masukkan huruf yang ingin dicari: y
Huruf y ditemukan pada indeks 5 dalam kalimat.
PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Lap

```



Deskripsi :

Program ini menerima sebuah kalimat dan karakter dari user, mengubah semua karakter dalam kalimat menjadi huruf kecil, mengurutkan kalimat tersebut dengan algoritma selection sort, dan melakukan pencarian biner untuk menemukan karakter yang dimaksud. Setelah user memasukkan kalimat dan karakter, program mengubah karakter menjadi huruf kecil, mengurutkan kalimat, dan menampilkannya. Kemudian, pencarian biner dilakukan pada kalimat yang sudah diurutkan untuk menemukan karakter, menampilkan indeksinya jika ditemukan atau pesan jika tidak ditemukan.

## Unguided 2

Sourcode

```

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

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

    int jumlahVokal = 0;
    char vokal[] = {'a', 'i', 'u', 'e', 'o', 'A', 'I', 'U', 'E', 'O'};
    int ukuranVokal = sizeof(vokal) / sizeof(vokal[0]);

    for (char huruf : kalimat)

```

```

{
    for (int i = 0; i < ukuranVokal; i++)
    {
        if (huruf == vokal[i])
        {
            jumlahVokal++;
            break;
        }
    }
}

cout << "Jumlah huruf vokal dalam kalimat: " << jumlahVokal << endl;

return 0;
}

```

Output :

```

PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8> cd "c:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8\" ; if ($?) { g++ unguided2.cpp -o unguided2 } ; if ($?) { .\unguided2 }
Masukkan kalimat: sayang
Jumlah huruf vokal dalam kalimat: 2
PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8>

```

Deskripsi :

Program ini menghitung jumlah huruf vokal dalam sebuah kalimat yang dimasukkan oleh user. Setelah memasukkan kalimat, program mengonversi setiap karakter menjadi huruf kecil dan memeriksa apakah karakter tersebut adalah vokal (a, i, u, e, o). Jika iya, program menambah jumlah huruf vokal. Setelah itu, program menampilkan total huruf vokal yang ditemukan dalam kalimat.

### Unguided 3

Sourcode

```

#include <iostream>
using namespace std;
int countOccurrences(int arr[], int size, int target)
{
    int count = 0;
    for (int i = 0; i < size; i++)
    {
        if (arr[i] == target)
        {
            count++;
        }
    }
}

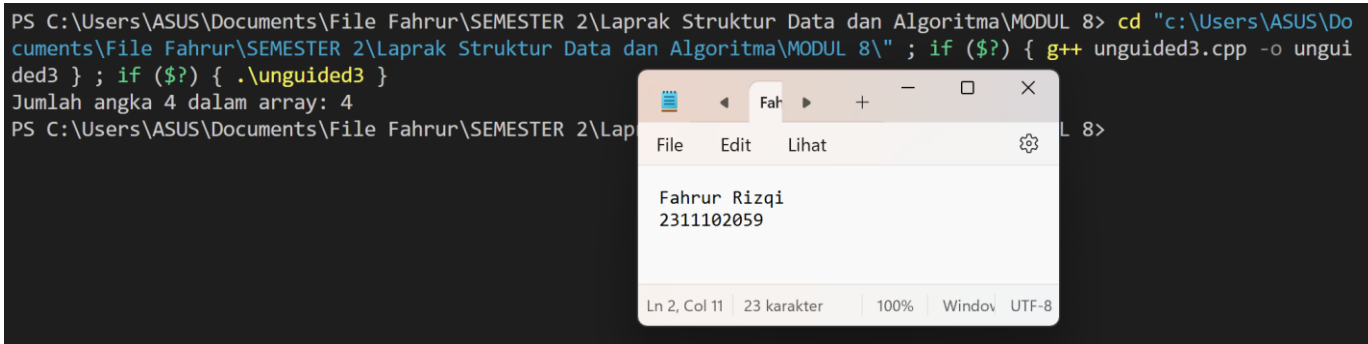
```

```

    }
    return count;
}
int main()
{
    int data[] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int target = 4;
    int occurrences = countOccurrences(data, sizeof(data) / sizeof(data[0]),
target);
    cout << "Jumlah angka " << target << " dalam array: " << occurrences << endl;
    return 0;
}

```

Output:



The screenshot shows a terminal window with the following commands and output:

```

PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8> cd "c:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Laprak Struktur Data dan Algoritma\MODUL 8\" ; if ($?) { g++ unguided3.cpp -o unguided3 } ; if ($?) { .\unguided3 }
Jumlah angka 4 dalam array: 4
PS C:\Users\ASUS\Documents\File Fahrur\SEMESTER 2\Lap

```

Overlaid on the terminal is a Notepad++ window titled 'Fahrur Rizqi' with the following text:

```

Fahrur Rizqi
2311102059

```

The Notepad++ status bar at the bottom indicates 'Ln 2, Col 11', '23 karakter', '100%', 'Window', and 'UTF-8'.

Deskripsi :

Kode ini merupakan implementasi sederhana dari algoritma pencarian sekuensial. Kode ini mendefinisikan sebuah array global bernama `DATA` yang berisi 10 elemen. Fungsi `sequentialsearch` bertugas menghitung berapa kali angka tertentu muncul dalam array tersebut dengan menggunakan loop untuk memeriksa setiap elemen. Fungsi ini kemudian menampilkan jumlah kemunculan angka tersebut di konsol. Pada fungsi `main`, angka yang akan dicari (angka 4) dideklarasikan, informasi awal ditampilkan, dan `sequentialsearch` dipanggil untuk mencari angka 4 dalam array serta menampilkan hasilnya.

## Kesimpulan

Pencarian sequential mengecek setiap elemen secara berurutan hingga menemukan yang dicari, cocok untuk data tidak terurut namun cenderung lambat. Pencarian binary membagi data menjadi dua bagian dan memeriksa bagian tengah, memberikan kinerja lebih cepat untuk data terurut dengan kompleksitas waktu logaritmik. Pemilihan metode tergantung pada sifat data dan kebutuhan kinerja.

## **Referensi**

Asisten Pratikum “Modul 8 ALGORITMA SEARCHING”, Learning Management System, 2024.

Badri,”ALGORITMA DAN PEMROGRAMAN PENCARIAN (SEARCHING)”,2023.

<https://www.teachmesoft.com/2019/03/pencarian-searching-c-disertai-contoh.html>

