

**LAPORAN PRAKTIKUM  
STRUKTUR DATA DAN ALGORITMA**

**MODUL 8  
ALGORITMA SEARCHING**



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

**Disusun oleh:**

**HAIKAL SATRIATAMA**

**2311102066**

**IF-11-B**

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

**2024**

# **BAB I**

## **DASAR TEORI**

### **1. 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 :

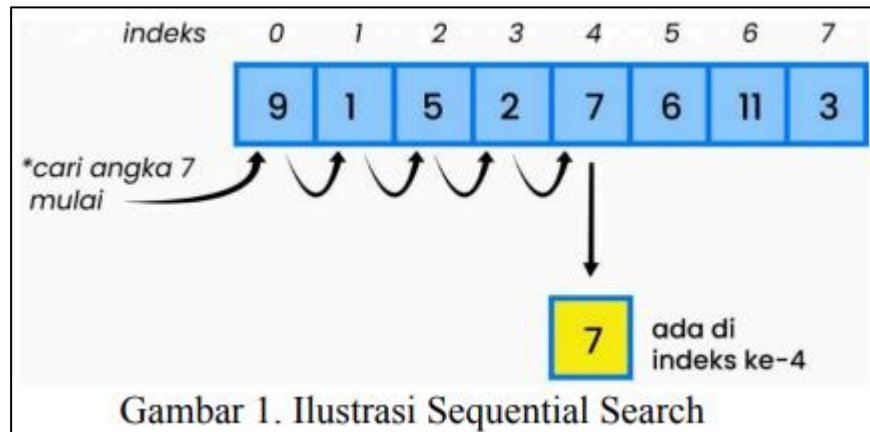
- 1)  $i \leftarrow 0$
- 2)  $ketemu \leftarrow false$
- 3) Selama (tidak  $ketemu$ ) dan  $(i \leq N)$  kerjakan baris 4
- 4) Jika  $(Data[i] = x)$  maka  $ketemu \leftarrow true$ , jika tidak  $i \leftarrow i + 1$

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

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, makapencarian 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.

## BAB II

### GUIDED

#### LATIHAN – GUIDED

##### 1. Guided 1

Buatlah sebuah project dengan menggunakan sequential search sederhana untuk melakukan pencarian data.

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

##### Screenshoot program

Program Sequential Search Sederhana	Haikal Satriatama
data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}	2311102066
angka 10 ditemukan pada indeks ke - 9	

### Deskripsi program

Dalam setiap iterasi, program membandingkan nilai elemen yang sedang diperiksa dengan nilai yang dicari. Jika nilainya cocok, variabel ketemu diubah menjadi true, dan perulangan dihentikan menggunakan perintah break. Jika perulangan selesai tanpa menemukan nilai yang dicari, variabel ketemu tetap bernilai false.

### 2. Guided 2

Buatlah sebuah project untuk melakukan pencarian data dengan menggunakan Binary Search.

### 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;
}

```

**Screenshoot program**

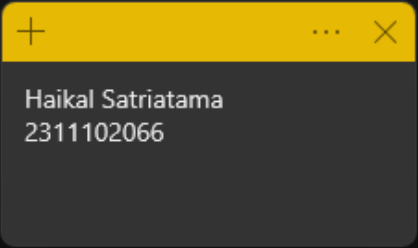
```
BINARY SEARCH

Data : 1 8 2 5 4 9 7

Masukkan data yang ingin Anda cari :1

Data diurutkan : 1 2 4 5 7 8 9

Data ditemukan pada index ke- 0
```



### Deskripsi program

Algoritma utama yang diterapkan dalam program ini adalah Selection Sort untuk mengurutkan array dan Binary Search untuk melakukan pencarian pada array yang sudah terurut.



## BAB III

### UNGUIDED

#### TUGAS – UNGUIDED

##### 1. Unguided 1

Buatlah sebuah program untuk mencari sebuah huruf pada sebuah kalimat yang sudah di input dengan menggunakan Binary Search!

##### Source Code

```
#include <iostream>

using namespace std;
void selectionSort(string &huruf, int a_63)
{
    int x, y, min;
    for (x = 0; x < a_63 - 1; x++)
    {
        min = x;
        for (y = x + 1; y < a_63; y++)
            if (huruf[y] < huruf[min])
                min = y;
        char temp = huruf[x];
        huruf[x] = huruf[min];
        huruf[min] = temp;
    }
}

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

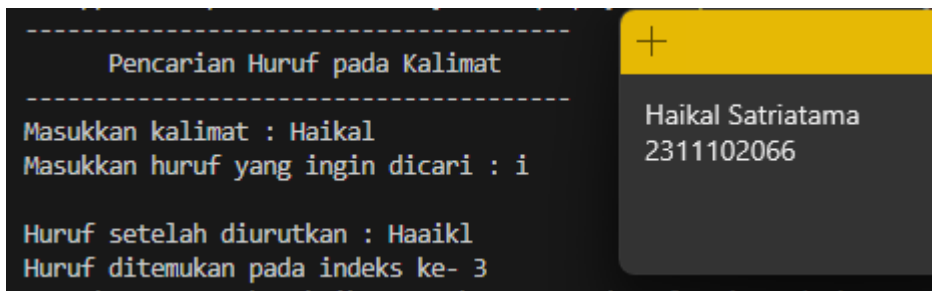
int main()
{
    string kalimat;
    char input;
    cout << "-----" << endl;
    cout << "        Pencarian Huruf pada Kalimat        " << endl;
    cout << "-----" << endl;
```

```

    cout << "Masukkan kalimat : ";
    getline(cin, kalimat);
    cout << "Masukkan huruf yang ingin dicari : ";
    cin >> input;
    cout << endl;
    selectionSort(kalimat, kalimat.size());
    int result = binarySearch(kalimat, 0, kalimat.size() - 1, input);
    if (result == -1)
    {
        cout << "Huruf yang Anda cari tidak ditemukan!" << endl;
    }
    else
    {
        cout << "Huruf setelah diurutkan : " << kalimat << endl;
        cout << "Huruf ditemukan pada indeks ke- " << result << endl;
    }
    return 0;
}

```

### Screenshot Program



### Deskripsi program

Program akan memanggil fungsi `binarySearch` untuk mencari posisi atau indeks dari huruf yang dicari dalam kalimat yang telah diurutkan. Fungsi ini menggunakan algoritma pencarian biner yang sangat efisien untuk mencari elemen dalam kumpulan data yang telah diurutkan.

Jika huruf yang dicari ditemukan dalam kalimat, program akan menampilkan kalimat yang telah diurutkan dan posisi atau indeks dari huruf yang dicari. Namun, jika huruf yang dicari tidak ditemukan dalam kalimat, program akan menampilkan pesan bahwa huruf tidak ditemukan.

## 2. Guided 2

Buatlah sebuah program yang dapat menghitung banyaknya huruf vocal dalam sebuah kalimat!

### Source code

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

using namespace std;

int main() {
    string kalimat;
    int jumlah = 0;

    cout << "-----" << endl;
    cout << "    Program Menghitung Huruf Vokal    " << endl;
    cout << "-----" << endl;

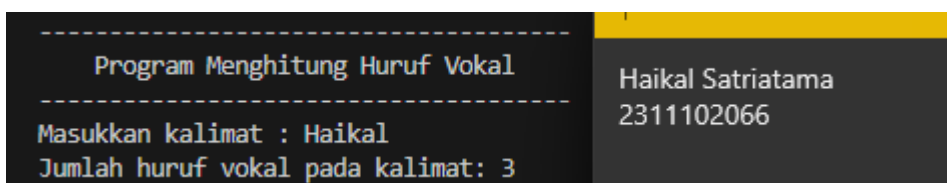
    cout << "Masukkan kalimat : ";
    getline(cin, kalimat);

    for (int n_63 = 0; n_63 < kalimat.length(); n_63++) {
        char a = kalimat[n_63];
        if (a == 'a' || a == 'i' || a == 'u' || a == 'e' || a == 'o'
||
            a == 'A' || a == 'I' || a == 'U' || a == 'E' || a == 'O')
        {
            jumlah++;
        }
    }

    cout << "Jumlah huruf vokal pada kalimat: " << jumlah << endl;

    return 0;
}
```

### Screenshoot program



### Deskripsi program

Program ini bertujuan untuk menghitung jumlah huruf vokal yang terdapat dalam sebuah kalimat yang dimasukkan oleh pengguna. Pada awalnya, program akan menampilkan judul "Program Menghitung Huruf Vokal" dan meminta pengguna untuk

memasukkan sebuah kalimat. Setelah pengguna memasukkan kalimat tersebut, program akan melakukan perulangan untuk memeriksa setiap karakter dalam kalimat tersebut. Jika karakter tersebut merupakan huruf vokal (a, i, u, e, o) baik huruf besar maupun huruf kecil, maka program akan menambahkan jumlah vokal yang ditemukan. Setelah selesai memeriksa seluruh karakter dalam kalimat, program akan menampilkan jumlah huruf vokal yang ditemukan pada kalimat yang dimasukkan oleh pengguna.

### 3. Guided 3

Diketahui data = 9, 4, 1, 4, 7, 10, 5, 4, 12, 4. Hitunglah berapa banyak angka 4 dengan menggunakan algoritma Sequential Search!

#### Source code

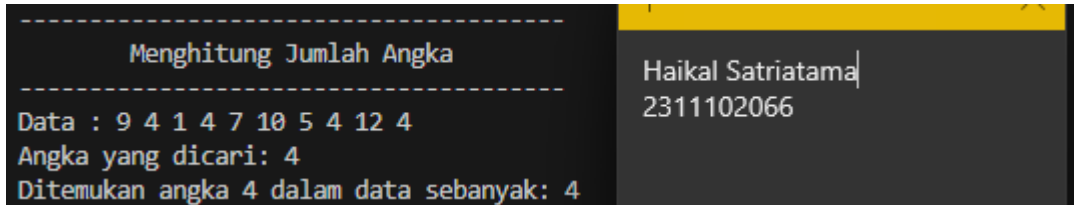
```
#include <iostream>

using namespace std;
int hitungAngka(const int array[], int size, int target)
{
    int jumlah = 0;
    for (int a_63 = 0; a_63 < size; a_63++)
    {
        if (array[a_63] == target)
        {
            jumlah++;
        }
    }
    return jumlah;
}

int main()
{
    const int size = 10;
    int array[size] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int target = 4;
    int jumlah = hitungAngka(array, size, target);
    cout << "-----" << endl;
    cout << "          Menghitung Jumlah Angka          " << endl;
    cout << "-----" << endl;
    cout << "Data : ";
    for (int element : array)
    {
        cout << element << " ";
    }
    cout << "\nAngka yang dicari: " << target << endl;
    cout << "Ditemukan angka " << target << " dalam data sebanyak: "
    << jumlah << endl;
```

```
    return 0;  
}
```

### Screenshoot program



The screenshot shows a terminal window with the title "Menghitung Jumlah Angka". The output displays the data array "Data : 9 4 1 4 7 10 5 4 12 4", the target number "Angka yang dicari: 4", and the result "Ditemukan angka 4 dalam data sebanyak: 4". On the right side of the terminal, the user's name "Haikal Satriatama" and ID "2311102066" are visible.

### Deskripsi program

Program ini mencari dan menghitung jumlah kemunculan angka tertentu dalam sebuah array (larik) yang telah didefinisikan. Program mendefinisikan sebuah fungsi bernama "hitungAngka" yang memeriksa setiap elemen dalam array tersebut. Jika elemen sama dengan angka yang dicari, sebuah penghitung akan bertambah. Fungsi kemudian mengembalikan nilai penghitung sebagai hasil. Di dalam fungsi utama, program membuat array dengan angka-angka tertentu, menentukan angka yang ingin dicari, memanggil fungsi "hitungAngka", dan menyimpan hasilnya. Terakhir, program menampilkan data array, angka yang dicari, serta jumlah kemunculan angka tersebut dalam array ke layar.

## **BAB IV**

### **KESIMPULAN**

Praktikum yang mempelajari Algoritma Search membahas dua algoritma utama, yaitu Algoritma Sequential dan Binary Search. Algoritma Sequential Search mencari sebuah elemen dalam sebuah array secara berurutan dari awal hingga akhir array. Meskipun sederhana, algoritma ini memiliki kompleksitas waktu rata-rata untuk kasus terburuk sebesar  $O(n)$ , di mana  $n$  adalah jumlah elemen dalam array. Algoritma ini cocok digunakan untuk array yang tidak terurut atau jumlah elemennya relatif kecil.

Di sisi lain, Algoritma Binary Search hanya dapat digunakan pada array yang telah terurut secara ascending atau descending. Algoritma ini membagi array menjadi dua bagian dan mencari elemen target di salah satu bagian tersebut secara rekursif. Dengan kompleksitas waktu rata-rata untuk kasus terburuk sebesar  $O(\log n)$ , Algoritma Binary Search terbukti jauh lebih efisien daripada Algoritma Sequential Search, terutama untuk array yang besar dan telah terurut.

Dalam praktikum ini, peserta mempelajari implementasi kedua algoritma tersebut serta menganalisis kompleksitas waktu masing-masing algoritma. Pemahaman tentang kedua algoritma ini penting dalam bidang algoritma dan struktur data, karena algoritma pencarian merupakan operasi fundamental dalam banyak aplikasi dan sistem. Algoritma yang tepat dapat meningkatkan efisiensi dan kinerja program secara signifikan.

## DAFTAR PUSTAKA

[1] Asisten Praktikum, "*Modul 8 Searching Seaching*", Learning Management System, 2024.