

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VIII
ALGORITMA SEARCHING**



**Disusun Oleh :
WIDARI DWI HAYATI
2311102060**

**Dosen :
WAHYU ANDI SAPUTRA, S.Pd., M.Eng.**

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

A. Dasar Teori

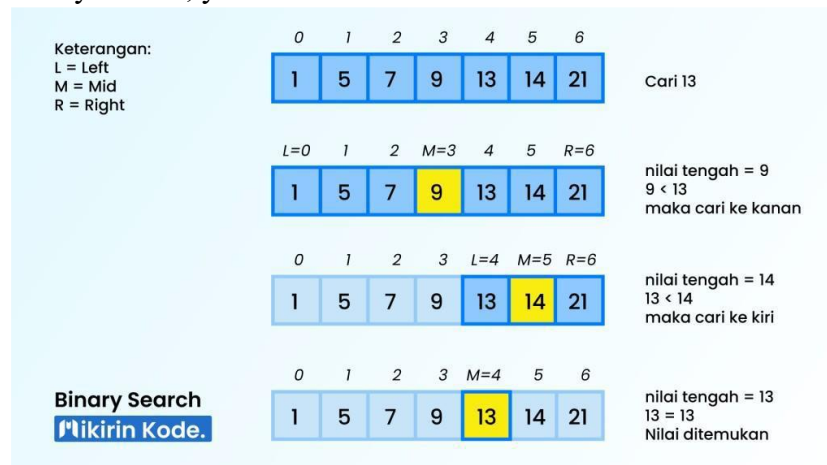
Searching adalah proses mencari suatu nilai tertentu pada kumpulan data. Hasil pencarian dapat menghasilkan tiga kondisi: data ditemukan, data ditemukan lebih dari satu, atau data tidak ditemukan. Pencarian dapat juga diartikan sebagai proses mencari data pada array dengan mengulangi setiap indeks baris atau kolomnya. Ada dua jenis algoritma pencarian, yaitu Sequential Search dan Binary Search.

Sequential Search adalah algoritma pencarian yang umum digunakan untuk data yang berpola acak atau belum terurut. Cara kerjanya adalah dengan membandingkan setiap elemen pada array satu per satu hingga data ditemukan atau hingga akhir array. Proses perulangan pada pencarian akan terjadi sebanyak jumlah N elemen pada array.

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.

Binary Search adalah algoritma pencarian yang termasuk dalam interval search, yang digunakan pada array/list dengan elemen terurut. Cara kerjanya adalah dengan membagi data menjadi dua untuk setiap tahap pencarian. Data harus diurutkan terlebih dahulu sebelum menggunakan algoritma ini. Data yang dicari akan dibandingkan dengan data tengah, apabila data yang dicari lebih besar atau lebih kecil, maka pencarian akan dilakukan pada bagian yang sesuai.

Contoh dari Binary Search, yaitu:



Terdapat sebuah array yang menampung 7 elemen seperti ilustrasi di atas. Nilai yang akan dicari pada array tersebut adalah 13. Jadi karena konsep dari binary search ini adalah membagi array menjadi dua bagian, maka pertama tama kita cari nilai tengahnya dulu, total elemen dibagi 2 yaitu $7/2 = 4.5$ dan kita bulatkan jadi 4. Maka elemen ke empat pada array adalah nilai tengahnya, yaitu angka 9 pada indeks ke 3. Kemudian kita cek apakah $13 > 9$ atau $13 < 9$? 13 lebih besar dari 9, maka kemungkinan besar angka 13 berada setelah 9 atau di sebelah kanan. Selanjutnya kita cari ke kanan dan kita dapat mengabaikan elemen yang ada di kiri. Setelah ituujungnya kita cari lagi nilai tengahnya, didapatlah angka 14 sebagai nilai tengah. Lalu, kita bandingkan apakah $13 > 14$ atau $13 < 14$? Ternyata 13 lebih kecil dari 14, maka selanjutnya kita cari ke kiri. Karna tersisa 1 elemen saja, maka elemen tersebut adalah nilai tengahnya. Setelah dicek ternyata elemen pada indeks ke-4 adalah elemen yang dicari, maka telah selesai proses pencariannya.

B. 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;

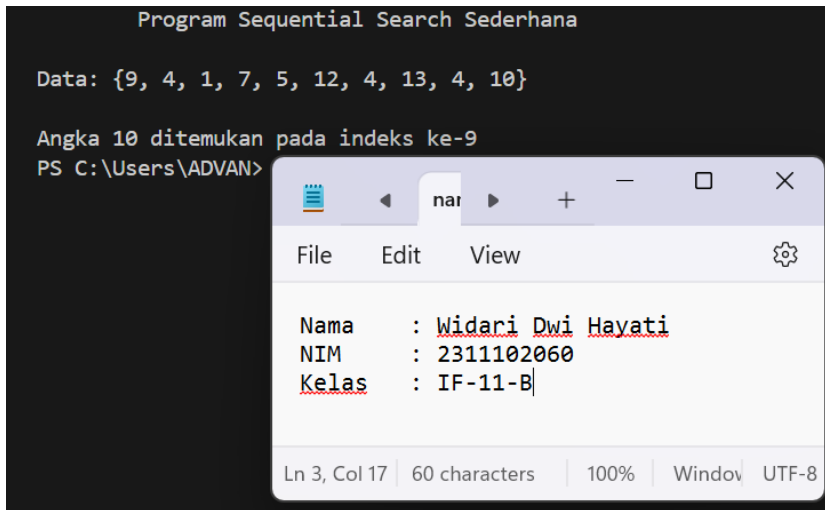
    // algoritma Sequential Search
    for (i = 0; i < n; i++){
        if(data[i] == cari){
            ketemu = true;
            break;
        }
    }

    cout << "\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 << "\nAngka " << cari << " tidak dapat ditemukan pada data." << endl;
    }

    return 0;
}
```

Screenshots Output



The screenshot shows a terminal window titled "Program Sequential Search Sederhana" with the following output:

```
Data: {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}
Angka 10 ditemukan pada indeks ke-9
PS C:\Users\ADVAN>
```

Overlaid on the terminal is a text editor window titled "nar" with a menu bar (File, Edit, View) and a settings icon. The editor contains the following text:

```
Nama      : Widari Dwi Hayati
NIM       : 2311102060
Kelas  : IF-11-B
```

The status bar at the bottom of the editor shows "Ln 3, Col 17 | 60 characters | 100% | Window | UTF-8".

Deskripsi:

Program di atas adalah sebuah implementasi algoritma Sequential Search yang akan mencari nilai yang dicari dalam array dan menampilkan indeks ke-berapa nilai tersebut ditemukan. Jika nilai tidak ditemukan, program akan menampilkan pesan bahwa nilai tidak dapat ditemukan.

Fungsi program terdiri dari mencari nilai yang dicari dalam array menggunakan algoritma Sequential Search, menampilkan indeks ke-berapa nilai tersebut ditemukan jika nilai ditemukan, dan menampilkan pesan bahwa nilai tidak dapat ditemukan jika nilai tidak ditemukan. Dengan input berupa nilai yang dicari (cari) = 10 dan array data = {9, 4, 1, 7, 5, 12, 4, 13, 4, 10}. Serta output jika nilai ditemukan: "Angka <nilai> ditemukan pada indeks ke-<indeks>" dan jika nilai tidak ditemukan: "Angka <nilai> tidak dapat ditemukan pada data."

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 datas[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(datas[j] < datas[min])
            {
                min=j;
            }
        }
        temp = datas[i];
        datas[i] = datas[min];
        datas[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(datas[tengah] == cari)
        {
            b_flag = 1;
            break;
        }
        else if(datas[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) << datas[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) << datas[x];
    cout << endl;
    binarysearch();
    _getche();
    return EXIT_SUCCESS;
}

```

Screenshots Output

```

BINARY SEARCH

Data :   1   8   2   5   4   9   7

Masukkan data yang ingin Anda cari :9

Data diurutkan :   1   2   4   5   7   8   9

Data ditemukan pada index ke- 6
PS C:\Users\ADVAN>

```

nar

File Edit View

Nama : Widari Dwi Hayati
NIM : 2311102060
Kelas : IF-11-B

Ln 3, Col 17 | 60 characters | 100% | Window UTF-8

Deskripsi:

Program di atas adalah sebuah implementasi algoritma Binary Search yang digunakan untuk mencari sebuah nilai dalam sebuah array yang sudah diurutkan. Program ini akan mencari nilai yang dicari dalam array dan menampilkan indeks ke-berapa nilai tersebut ditemukan. Jika nilai tidak ditemukan, program akan menampilkan pesan bahwa nilai tidak dapat ditemukan.

Fungsi program terdiri dari mencari nilai yang dicari dalam array menggunakan algoritma Binary Search, menampilkan indeks ke-berapa nilai tersebut ditemukan jika nilai ditemukan, menampilkan pesan bahwa nilai tidak dapat ditemukan jika nilai tidak ditemukan, dan mengurutkan array menggunakan algoritma Selection Sort sebelum melakukan pencarian. Dengan input berupa nilai yang dicari (cari) = input dari user dan array data = {1, 8, 2, 5, 4, 9, 7}. Serta output berupa jika nilai ditemukan: "Data ditemukan pada index ke- <indeks>", jika nilai tidak ditemukan: "Data tidak ditemukan", data awal, dan data setelah diurutkan.

C. Unguided/Tugas

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;
#include <conio.h>
#include <iomanip>
#include <string>

string kalimat;
char cari;

void binarysearch()
{
    //searching
    int awal, akhir, tengah, b_flag = 0;
    awal = 0;
    akhir = kalimat.length() - 1;
    while (b_flag == 0 && awal <= akhir)
    {
        tengah = (awal + akhir) / 2;
        if(kalimat[tengah] == cari)
        {
            b_flag = 1;
            break;
        }
        else if(kalimat[tengah] < cari)
            awal = tengah + 1;
        else
            akhir = tengah - 1;
    }

    if(b_flag == 1)
        cout<<"\n Huruf ditemukan pada index ke- " << tengah << endl;
    else
        cout<<"\n Huruf tidak ditemukan\n";
}

int main()
{
    cout<<"\t BINARY SEARCH " << endl;
    cout<<"\n Masukkan kalimat : ";
    getline(cin, kalimat);
    cout << "\n Masukkan huruf yang ingin Anda cari :";
    cin >> cari;
```

```

cout << "\n Kalimat diurutkan : ";
//urutkan kalimat dengan selection sort
for(int i = 0; i < kalimat.length(); i++)
{
    for(int j = i + 1; j < kalimat.length(); j++)
    {
        if(kalimat[j] < kalimat[i])
        {
            char temp = kalimat[i];
            kalimat[i] = kalimat[j];
            kalimat[j] = temp;
        }
    }
}
//tampilkan kalimat setelah diurutkan
for(int x = 0; x < kalimat.length(); x++)
    cout << kalimat[x];
cout << endl;
binarysearch();
_getche();
return EXIT_SUCCESS;
}

```

Screenshots Output

The screenshot shows a terminal window with the following output:

```

BINARY SEARCH

Masukkan kalimat : Widari Dwi Hayati

Masukkan huruf yang ingin Anda cari :r

Kalimat diurutkan :  DHWaaadiiirtwy

Huruf ditemukan pada index ke- 13
PS C:\Users\ADVAN>

```

Overlaid on the terminal is a Notepad window titled 'nar'. It contains the following text:

```

Nama      : Widari Dwi Hayati
NIM       : 2311102060
Kelas  : IF-11-B|

```

The Notepad window's status bar at the bottom indicates: Ln 3, Col 17 | 60 characters | 100% | Window | UTF-8.

Deskripsi:

Program di atas adalah sebuah implementasi algoritma Binary Search yang akan mencari huruf yang dicari dalam kalimat dan menampilkan indeks ke-berapa huruf tersebut ditemukan. Jika huruf tidak ditemukan, program akan menampilkan pesan bahwa huruf tidak dapat ditemukan.

Fungsi program terdiri dari mencari huruf yang dicari dalam kalimat menggunakan algoritma Binary Search, menampilkan indeks ke-berapa huruf tersebut ditemukan jika huruf ditemukan. ~~Menampilkan, menampilkan~~ pesan bahwa huruf tidak dapat ditemukan jika huruf tidak ditemukan. ~~Mengurutkan, dan mengurutkan~~ kalimat menggunakan algoritma Selection Sort sebelum melakukan pencarian. Dengan input berupa kalimat (kalimat) = input dari user dan huruf yang dicari (cari) = input dari user. Serta output berupa jika huruf ditemukan: "Huruf ditemukan pada index ke- <indeks>", jika huruf tidak ditemukan: "Huruf tidak ditemukan", dan kalimat setelah diurutkan.

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

Source Code

```
#include <iostream>
using namespace std;
#include <conio.h>
#include <iomanip>
#include <string>

char datas[26] = {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};
int cari;

void selection_sort()
{
    char temp;
    int min, i, j;
    for(i = 0; i < 26; i++)
    {
        min = i;
        for(j = i + 1; j < 26; j++)
        {
            if(datas[j] < datas[min])
            {
                min=j;
            }
        }
        temp = datas[i];
        datas[i] = datas[min];
        datas[min] = temp;
    }
}

void binarysearch(char x)
{
    //searching
    int awal, akhir, tengah, b_flag = 0;
    awal = 0;
    akhir = 26;
    while (b_flag == 0 && awal <= akhir)
    {
        tengah = (awal + akhir) / 2;
        if(datas[tengah] == x)
        {
            b_flag = 1;
            break;
        }
    }
}
```

```

    }
    else if(datas[tengah] < x)
        awal = tengah + 1;
    else
        akhir = tengah - 1;
    }

    if(b_flag != 1)
        cout<<"\n Huruf tidak ditemukan\n";
}

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

    cout << "\t PROGRAM MENGHITUNG JUMLAH HURUF VOKAL" << endl;
    cout << "\n Masukkan kalimat : ";
    getline(cin, kalimat);

    selection_sort();

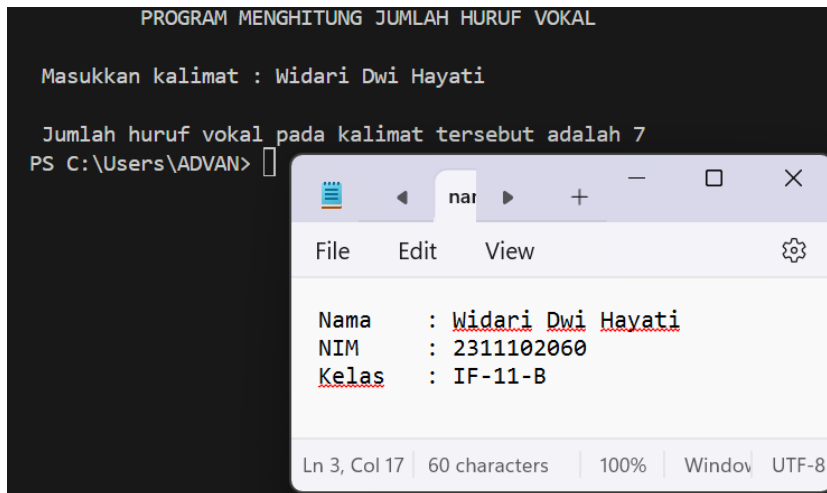
    for(int i = 0; i < kalimat.length(); i++)
    {
        if(kalimat[i] >= 'a' && kalimat[i] <= 'z')
        {
            binarysearch(kalimat[i]);
            if(kalimat[i] == 'a' || kalimat[i] == 'i' || kalimat[i] == 'u' || kalimat[i] == 'e' ||
kalimat[i] == 'o')
            {
                jumlah_vokal++;
            }
        }
        else if(kalimat[i] >= 'A' && kalimat[i] <= 'Z')
        {
            binarysearch(kalimat[i] + 32);
            if(kalimat[i] == 'A' || kalimat[i] == 'I' || kalimat[i] == 'U' || kalimat[i] == 'E' ||
kalimat[i] == 'O')
            {
                jumlah_vokal++;
            }
        }
    }

    cout << "\n Jumlah huruf vokal pada kalimat tersebut adalah " << jumlah_vokal <<
endl;
    _getche();
}

```

```
    return EXIT_SUCCESS;
}
```

Screenshots Output



Deskripsi:

Program di atas adalah sebuah implementasi algoritma Binary Search yang akan mencari setiap huruf dalam kalimat dan mengecek apakah huruf tersebut adalah huruf vokal atau tidak. Jika huruf tersebut adalah huruf vokal, program akan menambah jumlah huruf vokal. Setelah melakukan pencarian pada setiap huruf dalam kalimat, program akan menampilkan jumlah huruf vokal yang ditemukan.

Fungsi program terdiri dari mencari setiap huruf dalam kalimat menggunakan algoritma Binary Search, mengecek apakah huruf tersebut adalah huruf vokal atau tidak, menambah jumlah huruf vokal jika huruf tersebut adalah huruf vokal, dan menampilkan jumlah huruf vokal yang ditemukan. Dengan input berupa kalimat (kalimat) = input dari user. Serta output berupa Jumlah huruf vokal yang ditemukan.

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 main(){
    int n = 10;
    int data[n] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int cari = 4;
    int jumlah_kemunculan = 0;

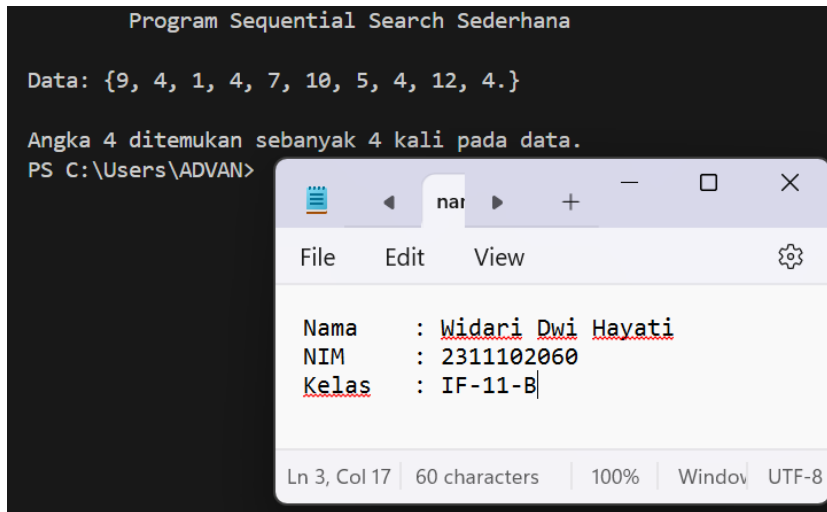
    // algoritma Sequential Search
    for (int i = 0; i < n; i++){
        if(data[i] == cari){
            jumlah_kemunculan++;
        }
    }

    cout << "\tProgram Sequential Search Sederhana\n" << endl;
    cout << "Data: {9, 4, 1, 4, 7, 10, 5, 4, 12, 4.}" << endl;

    if (jumlah_kemunculan > 0){
        cout << "\nAngka " << cari << " ditemukan sebanyak " << jumlah_kemunculan <<
        " kali pada data." << endl;
    } else {
        cout << "\nAngka " << cari << " tidak dapat ditemukan pada data." << endl;
    }

    return 0;
}
```

Screenshots Output



```
Program Sequential Search Sederhana

Data: {9, 4, 1, 4, 7, 10, 5, 4, 12, 4}

Angka 4 ditemukan sebanyak 4 kali pada data.
PS C:\Users\ADVAN>
```

Nama : Widari Dwi Hayati
NIM : 2311102060
Kelas : IF-11-B

Ln 3, Col 17 | 60 characters | 100% | Window UTF-8

Deskripsi:

Program di atas adalah sebuah implementasi algoritma Sequential Search yang akan mencari nilai yang dicari dalam array dan menghitung jumlah kemunculan dari nilai tersebut dalam array. Jika nilai ditemukan, program akan menampilkan jumlah kemunculan dari nilai tersebut dalam array. Jika nilai tidak ditemukan, program akan menampilkan pesan bahwa nilai tidak dapat ditemukan.

Fungsi program terdiri dari mencari nilai yang dicari dalam array menggunakan algoritma Sequential Search, menghitung jumlah kemunculan dari nilai tersebut dalam array, menampilkan jumlah kemunculan dari nilai tersebut dalam array jika nilai ditemukan, dan menampilkan pesan bahwa nilai tidak dapat ditemukan jika nilai tidak ditemukan. Dengan input berupa nilai yang dicari (cari) = 4 dan array data = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4}. Serta output berupa jika nilai ditemukan: "Angka <nilai> ditemukan sebanyak <jumlah_kemunculan> kali pada data." dan jika nilai tidak ditemukan: "Angka <nilai> tidak dapat ditemukan pada data.".

D. Kesimpulan

Searching adalah proses mencari suatu nilai tertentu pada kumpulan data. Hasil pencarian dapat menghasilkan tiga kondisi: data ditemukan, data ditemukan lebih dari satu, atau data tidak ditemukan. Pencarian dapat juga diartikan sebagai proses mencari data pada array dengan mengulangi setiap indeks baris atau kolomnya. Ada dua jenis algoritma pencarian, yaitu Sequential Search dan Binary Search.

Sequential Search adalah algoritma pencarian yang umum digunakan untuk data yang berpola acak atau belum terurut. Cara kerjanya adalah dengan membandingkan setiap elemen pada array satu per satu hingga data ditemukan atau hingga akhir array. Proses perulangan pada pencarian akan terjadi sebanyak jumlah N elemen pada array.

Binary Search adalah algoritma pencarian yang termasuk dalam interval search, yang digunakan pada array/list dengan elemen terurut. Cara kerjanya adalah dengan membagi data menjadi dua untuk setiap tahap pencarian. Data harus diurutkan terlebih dahulu sebelum menggunakan algoritma ini. Data yang dicari akan dibandingkan dengan data tengah, apabila data yang dicari lebih besar atau lebih kecil, maka pencarian akan dilakukan pada bagian yang sesuai

E. Referensi (APA)

Stroustrup, B. (2021). A Tour of C++ (2nd ed.). Addison-Wesley Professional.

Lippman, S. L., Lajoie, J., & Moo, B. (2020). C++ Primer (6th ed.). Addison-Wesley Professional.

Malik, D. S. (2014). C++ Programming: From Problem Analysis to Program Design (7th ed.). Cengage Learning.