

**LAPORAN PRAKTIKUM
STRUKTUR DATA DAN ALGORITMA
MODUL 8
“SEARCHING”**



**DISUSUN OLEH:
TSAQIF KANZ AHMAD
2311102075
IF-11-B**

DOSEN:

WAHYU ANDI SAPUTRA S.Pd., M.Eng.

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

TUJUAN PRAKTIKUM

1. Menunjukkan beberapa algoritma dalam pencarian (Searching).
2. Menunjukkan bahwa pencarian merupakan suatu persoalan yang bisa diselesaikan dengan beberapa algoritma yang berbeda.
3. dapat memilih algoritma yang paling sesuai untuk menyelesaikan suatu permasalahan pemrograman

DASAR TEORI

1). PENGERTIAN SEARCHING

Searching(Pencarian) adalah proses mencari nilai tertentu dalam sekumpulan data. Hasil pencarian dapat dibagi dalam salah satu dari tiga kondisi yaitu data ditemukan, data ditemukan lebih dari satu kali atau data tidak ditemukan. Searching juga dapat dilihat sebagai proses mencari data dalam suatu array dengan cara memeriksa setiap indeks baris atau setiap indeks kolom satu per satu dengan menggunakan teknik pengulangan untuk mencari datanya. Ada 2 metode dalam algoritma Searching :

a. SEQUENTIAL SEARCH(Pencarian Berurutan)

Sequential Search adalah metode searching yang paling sederhana. Pada metode ini, data diurutkan satu per satu dari awal hingga akhir untuk mencari data yang diinginkan. Jika data yang dicari tidak ditemukan, maka algoritma akan mengembalikan nilai false. Pencarian ini hanya melakukan pengulangan dari 1 sampai dengan jumlah data. Pada setiap pengulangan, dibandingkan data ke-i dengan yang dicari. Apabila sama, berarti data telah ditemukan. Sebaliknya apabila sampai akhir pengulangan tidak ada data yang sama, berarti data tidak ada. Pada kasus yang paling buruk, untuk N elemen data harus dilakukan pencarian sebanyak N kali pula.

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 ($\text{Data}[i] = x$) 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.

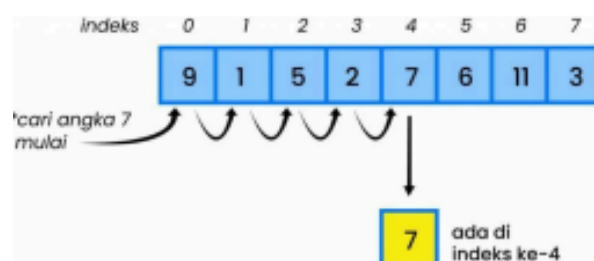
Di bawah ini merupakan fungsi untuk mencari data menggunakan pencarian sekuensial.

```
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 akan dicari. Apabila data tidak ditemukan maka fungsi diatas akan mengembalikan nilai-1.

Contoh dari Sequential Search :

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 dari indeks 0 yaitu angka 9, kemudian dibandingkan dengan angka yang akan dicari. Jika tidak sama maka pencarian dilanjutkan ke indeks berikutnya.
- Pada index pertama, yaitu angka 1, juga bukan merupakan angka yang diinginkan, Oleh karena itu pencarian Praktikum Struktur Data dan Algoritma 2 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 adalah algoritma pencarian yang digunakan untuk menemukan posisi nilai target dalam array yang diurutkan. Ia bekerja dengan membagi interval pencarian menjadi dua berulang kali hingga nilai target ditemukan atau interval tersebut kosong. Interval pencarian dikurangi setengahnya dengan membandingkan elemen target dengan nilai tengah ruang pencarian. Prinsip pencarian biner yaitu pertama diambil posisi awal 0 dan posisi akhir = $N - 1$, kemudian dicari posisi data tengah dengan rumus $(\text{posisi awal} + \text{posisi akhir}) / 2$. Kemudian data yang dicari dibandingkan dengan data tengah. Jika lebih kecil, proses dilakukan kembali tetapi posisi akhir dianggap sama dengan posisi tengah - 1. Jika lebih besar, proses dilakukan kembali tetapi posisi awal dianggap sama dengan posisi tengah + 1. Demikian seterusnya sampai data tengah sama dengan yang dicari. Untuk lebih jelasnya perhatikan contoh berikut : Misalnya ingin mencari data 17 pada sekumpulan data berikut :

3	9	11	12	15	17	20	23	31	35
awal				tengah					akhir

Pertama a dicari data tengah, dengan rumus $(0 + 9) / 2 = 4$. Berarti data tengah adalah data ke-4, yaitu 15. Data yang dicari, yaitu 17, dibandingkan dengan data tengah karena $17 < 15$, berarti proses dilanjutkan tetapi kali ini posisi awal dianggap sama dengan posisi tengah +1 atau 5.

3	9	11	12	15	17	20	23	31	35
				awal		tengah			akhir

Data tengah yang baru didapat dengan rumus $(5 + 9) / 2 = 7$. Berarti data tengah yang baru adalah data ke-7, yaitu 23. Data yang dicari yaitu 17 dibandingkan dengan data tengah ini. Karena $17 < 23$, berarti proses dilanjutkan tetapi kali ini posisi akhir dianggap sama dengan posisi tengah -1 atau 6.

3	9	11	12	15	17	20	23	31	35
				awal=tengah		akhir			

Data tengah yang baru didapat dengan rumus $(5 + 6) / 2 = 5$. Berarti data tengah yang baru adalah data ke-5, yaitu 17. data yang dicari dibandingkan dengan data tengah ini dan ternyata sama. Jadi data ditemukan pada indeks ke-5. Pencarian biner ini akan berakhir jika data ditemukan atau posisi awal lebih besar daripada posisi akhir. Jika posisi sudah lebih besar daripada posisi akhir berarti data tidak ditemukan. Untuk lebih

jelasanya perhatikan contoh pencarian data 16 pada data diatas. Prosesnya hampir sama dengan pencarian data 17. Tetapi setelah posisi awal 5 dan posisi akhir 6, data tidak ditemukan dan $16 < 17$, maka posisi akhir menjadi posisi tengah - 1 atau = 4 sedangkan posisi awal = 5.

3	9	11	12	15	17	20	23	31	35
				akhir	awal				

Disini dapat dilihat bahwa posisi awal lebih besar daripada posisi akhir, yang artinya data tidak ditemukan.

Algoritma pencarian biner dapat dituliskan sebagai berikut :

- 1 $L \leftarrow 0$
- 2 $R \leftarrow N - 1$
- 3 ketemu \leftarrow false
- 4 Selama ($L \leq R$) dan (tidak ketemu) kerjakan baris 5 sampai dengan 8
- 5 $m \leftarrow (L + R) / 2$
- 6 Jika ($Data[m] = x$) maka ketemu \leftarrow true
- 7 Jika ($x < Data[m]$) maka $R \leftarrow m - 1$
- 8 Jika ($x > Data[m]$) maka $L \leftarrow m + 1$
- 9 Jika (ketemu) maka m adalah indeks dari data yang dicari, jika tidak data tidak ditemukan

Di bawah ini merupakan fungsi untuk mencari data menggunakan pencarian biner.

```
int BinarySearch(int x)
{
    int L = 0, R = Max-1, m;
    bool ketemu = false;
    while((L <= R) && (!ketemu))
    {
        m = (L + R) / 2;
        if(Data[m] == x)
            ketemu = true;
        else if (x < data[m])
            R = m - 1;
        else
            L = m + 1;
    }
    if(ketemu)
        return m;
    else
        return -1;
}
```

Fungsi diatas akan mengembalikan indeks dari data yang dicari. Apabila data tidak ditemukan maka fungsi diatas akan mengembalikan nilai -1. Jumlah perbandingan minimum pada pencarian biner adalah 1 kali, yaitu apabila data yang dicari tepat berada di tengah-tengah. Jumlah perbandingan maksimum yang dilakukan dengan pencarian biner dapat dicari menggunakan rumus logaritma, yaitu : $C = 2 \log(N)$.

PENJELASAN GUIDED

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

SOURCE CODE

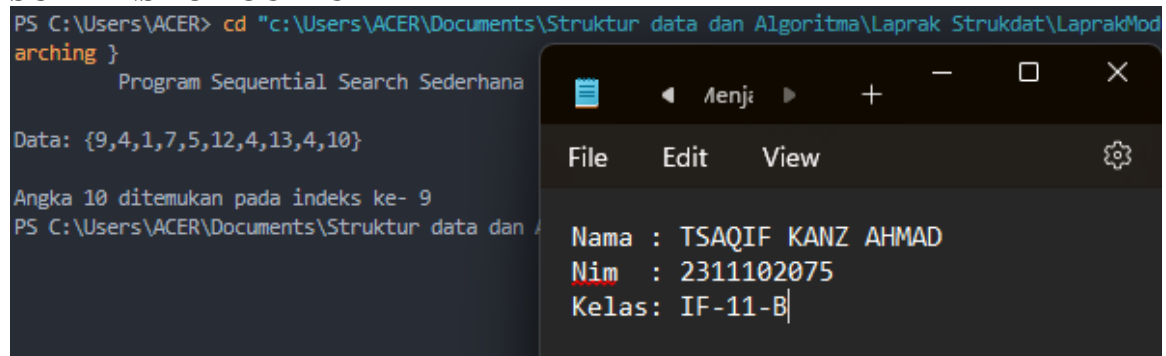
```
#include <iostream>
using namespace std;

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;
    //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<<"\nAngka "<<cari<<" Tidak ditemukan pada data"<<endl;
    }

    return 0;
}
```

SCREENSHOT OUTPUT



```
PS C:\Users\ACER> cd "c:\Users\ACER\Documents\Struktur data dan Algoritma\Laprak Strukdat\LaprakMod
arching"
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\ACER\Documents\Struktur data dan
```

Nama : TSAQIF KANZ AHMAD
Nim : 2311102075
Kelas: IF-11-B

DESKRIPSI PROGRAM

Program di atas adalah program pencarian data menggunakan sequential search untuk mencari nilai tertentu dalam array. Algoritma ini bekerja dengan memeriksa setiap elemen dalam array satu persatu hingga menemukan nilai yang dicari atau hingga semua elemen telah diperiksa. Jika nilai ditemukan, program akan mengatur 'ketemu' menjadi true dan menghentikan loop.

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;
}

```

SCREENSHOT OUTPUT

```

PS C:\Users\ACER> cd "c:\Users\ACER\Documents\Struktur data dan Algoritma\Laprak Strukdat\Lapr
arching }
    BINARY SEARCH

Data :  1  8  2  5  4  9  7

Masukkan DATA yang ingin Anda cari :8

Data diurutkan :  1  2  4  5  7  8  9

Data ditemukan pada index ke-5

```

Nama : TSAQIF KANZ AHMAD
 Nim : 2311102075
 Kelas: IF-11-B

DESKRIPSI PROGRAM

Program di atas adalah program melakukan pencarian data menggunakan binary search pada sebuah array yang diurutkan menggunakan algoritma selection sort. Program menampilkan data awal dari array, kemudian meminta pengguna memasukan nilai yang ingin dicari. Pada array 'DATA' diurutkan menggunakan selection sort dan program menampilkan array yang sudah diurutkan. Program akan melakukan pencarian biner untuk menemukan nilai yang dimasukan oleh pengguna dalam array yang sudah diurutkan. Kemudian program akan menampilkan hasil pencarian.

PENJELASAN UNGUIDED

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

SOURCE CODE

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

void selection_sort(vector<char>& data) {
    int n = data.size();
    for (int i = 0; i < n - 1; ++i) {
        int min_idx = i;
        for (int j = i + 1; j < n; ++j) {
            if (data[j] < data[min_idx]) {
                min_idx = j;
            }
        }
        swap(data[i], data[min_idx]);
    }
}

bool binary_search(const vector<char>& data, char target) {
    int left = 0, right = data.size() - 1;
    while (left <= right) {
        int middle = left + (right - left) / 2;
        if (data[middle] == target) {
            return true;
        } else if (data[middle] < target) {
            left = middle + 1;
        } else {
            right = middle - 1;
        }
    }
    return false;
}

int main() {
    string input;
    char target;

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

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

    // Mengubah kalimat menjadi array karakter dan mengurutkannya
    vector<char> data(input.begin(), input.end());
```

```

// Mengurutkan array karakter
selection_sort(data);

// Menampilkan array karakter setelah diurutkan
cout << "Kalimat setelah diurutkan: ";
for (char c : data) {
    cout << c << ' ';
}
cout << endl;

// Melakukan pencarian biner
if (binary_search(data, target)) {
    cout << "Huruf '" << target << "' ditemukan dalam kalimat." <<
endl;
} else {
    cout << "Huruf '" << target << "' tidak ditemukan dalam kalimat."
<< endl;
}

return 0;
}

```

SCREENSHOT OUTPUT

<pre> PS C:\Users\ACER> cd "C:\Users\ACER\AppData\Local\Temp\" ; if (\$?) { g++ t Masukkan sebuah kalimat: Telkom University Pwt Masukkan huruf yang ingin dicari: t Kalimat setelah diurutkan: P T U e e i i k l m n o r s t t v w y Huruf 't' ditemukan dalam kalimat. PS C:\Users\ACER\AppData\Local\Temp> </pre>	<pre> Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B </pre>
--	---

DESKRIPSI PROGRAM

Program diatas adalah program mencari sebuah huruf pada sebuah kalimat yang sudah di input dengan menggunakan Binary Search. Program meminta pengguna untuk memasukan sebuah kalimat dan pengguna juga diminta untuk memasukan huruf yang ingin dicari. Kemudian mengkonversikan kalimat yang dimasukan menjadi array karakter menggunakan 'vector<char>', kemudian menggunakan fungsi 'selection_sort' untuk mengurutkan array karakter. Pada fungsi 'binary_search' untuk mencari huruf yang dimasukan dalam array yang sudah diurutkan. Fungsi ini akan mengembalikan 'true' jika huruf ditemukan maupun sebaliknya dan program akan menampilkan hasil apakah huruf yang dicari ditemukan dalam kalimat atau tidak

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

SOURCE CODE

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

// Fungsi untuk mengecek apakah sebuah karakter adalah huruf vokal
bool isVowel(char c) {
    c = tolower(c); // Mengubah karakter menjadi huruf kecil
    return (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');
}

// Fungsi untuk menghitung jumlah huruf vokal dalam kalimat
int countVowels(const string& sentence) {
    int count = 0;
    for (char c : sentence) {
        if (isVowel(c)) {
            count++;
        }
    }
    return count;
}

int main() {
    string sentence;

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

    int vowelCount = countVowels(sentence);

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

    return 0;
}
```

SCREENSHOOT OUTPUT

PS C:\Users\ACER> cd "C:\Users\ACER\AppData\Local\Temp"	Nama : TSAQIF KANZ AHMAD
Masukkan sebuah kalimat: Tsaqif Kanz Ahamd	Nim : 2311102075
Jumlah huruf vokal dalam kalimat: 5	Kelas: IF-11-B
PS C:\Users\ACER\AppData\Local\Temp> █	

DESKRIPSI PORGRAM

Program diatas adalah program menghitung banyaknya huruf vokal dalam sebuah kalimat. Program pertama menggunakan fungsi 'isVowel(char c)' untuk mengecek apakah karakter 'c' adalah huruf vokal atau bukan. Karakter tersebut diubah menjadi huruf kecil menggunakan 'tolower' untuk memastikan pengecekan tidak peka terhadap huruf besar/kecil. Program kedua menggunakan fungsi 'countVowel(const string& sentence)' untuk menghitung jumlah huruf vokal dalam sebuah kalimat. Kemudian menggunakan loop 'for' untuk iterasi melalui setiap karakter dalam kalimat menggunakan 'isVowel' untuk mengecek apakah karakter tersebut vokal. Jika karakter

adalah vokal, variabel 'count' ditambah 1. Program ketiga menggunakan fungsi 'main' untuk mengambil input kalimat dari pengguna menggunakan 'getline' untuk memungkinkan input mengandung spasi. Pada tipe data 'countVowel' untuk menghitung jumlah huruf vokal dalam kalimat dan menampilkan hasil perhitungan.

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;

// Fungsi untuk menghitung berapa banyak angka 4 dalam array menggunakan
sequential search
int countNumber(const int data[], int size, int target) {
    int count = 0;
    for (int i = 0; i < size; i++) {
        if (data[i] == target) {
            count++;
        }
    }
    return count;
}

int main() {
    int data[] = {9, 4, 1, 4, 7, 10, 5, 4, 12, 4};
    int size = sizeof(data) / sizeof(data[0]);
    int target = 4;

    int count = countNumber(data, size, target);

    cout << "Jumlah angka " << target << " dalam array adalah: " << count
    << endl;

    return 0;
}
```

SCREENSHOOT OUTPUT

PS C:\Users\ACER> cd "C:\Users\ACER\AppData\Local\Temp"	Nama : TSAQIF KANZ AHMAD
Jumlah angka 4 dalam array adalah: 4	Nim : 2311102075
PS C:\Users\ACER\AppData\Local\Temp>	Kelas: IF-11-B

DESKRIPSI PROGRAM

Program diatas program untuk menghitung berapa banyak angka 4 dalam sebuah array menggunakan algoritma pencarian berurutan. Pada program tersebut terdapat 2 fungsi yaitu fungsi 'countNumber' dan fungsi 'main'. Pada fungsi 'countNumber' terdapat array yang menerima 'data', ukuran array 'size', dan angka target 'target' yang ingin dihitung kemunculannya dengan menggunakan loop 'for' untuk iterasi melalui setiap elemen array. Jika elemen array sama dengan angka target, maka variabel 'count' ditambah 1 dan mengembalikan nilai 'count' setelah iterasi selesai. Kemudian ada fungsi 'main' yang mendeklarasikan dan menginisialisasi array 'data' dengan nilai yang diberikan. Kemudian menghitung array menggunakan 'sizeof', serta mendeklarasikan variabel 'target' yang berisi angka 4. Memanggil fungsi 'countNumber' untuk menghitung jumlah kemuculan angka 4 dalam array dan menampilkan hasil perhitungannya.

KESIMPULAN

Searching atau pencarian adalah proses menemukan informasi atau elemen tertentu dalam kumpulan data atau ruang pencarian. Dalam ilmu komputer, pencarian adalah salah satu tugas fundamental yang berhubungan dengan algoritma dan struktur data. pencarian mencakup pemahaman tentang berbagai algoritma pencarian, struktur data yang mendukung pencarian efisien, serta aplikasi dan optimasi yang relevan. Pemilihan algoritma pencarian yang tepat sangat tergantung pada sifat data dan konteks penggunaannya.

DAFTAR PUSTAKA

[1] Asisten Praktikum, “Modul 8 SEARCHING” : [Modul 8 - Algoritma Searching \(1\).pdf](#)

[2] Politeknik Elektronika Negeri Surabaya – Pencarian (Searching) :
<https://yuliana.lecturer.pens.ac.id/Struktur%20Data%20C/Teori%20SD%20-%20pdf/Data%20Structure%20-%20Bab%208.pdf>