LAPORAN PRAKTIKUM
STRUKTUR DATA DAN ALGORITMA

MODUL 8

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

Disusun oleh:

ABDA FIRAS RAHMAN

2311102049

IF-11-B

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

BAB I
DASAR TEORI
ALGORITMA SEARCHING

1. Pengertian

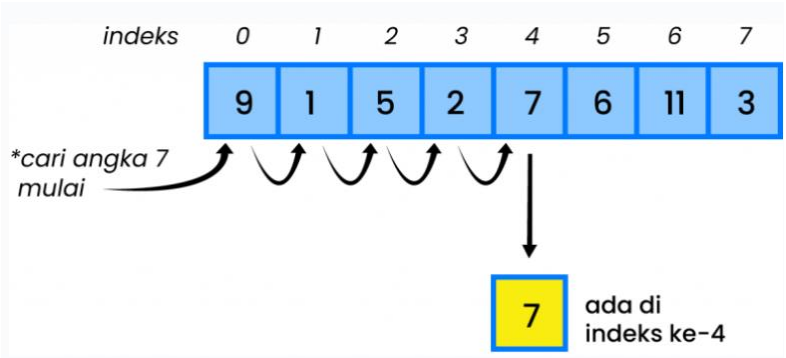
Pencarian (Searching) yaitu proses menemukan suatu nilai tertentu pada kumpulan data. Hasil pencarian adalah salah satu dari tiga keadaan ini: data ditemukan, data ditemukan lebih dari satu, atau data tidak ditemukan. Searching juga dapat dianggap sebagai proses pencarian suatu data di dalam sebuah array 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

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.

Konsep Pencarian Sekuensial:

- 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



Ilustrasi Sequential Search

b. Binary Search

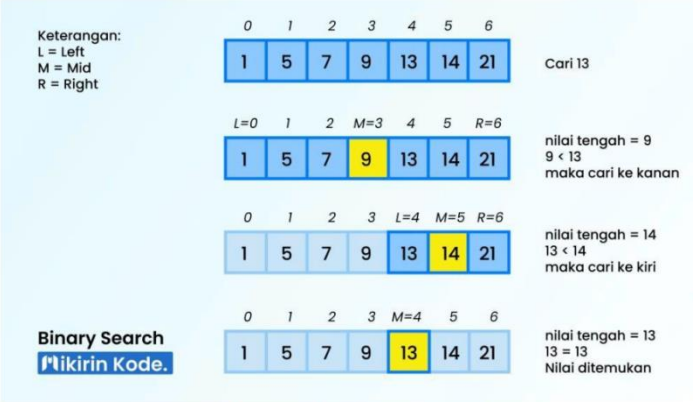
Binary Search merupakan sebuah teknik pencarian data dengancara berulang kali membagi separuh dari jumlah data yang dicari sampai sehingga memperkecil lokasi pencarian menjadi satu data. Dengan teknik ini kita akanmembuang setengah dari jumlah data. Apabila ditemukan kecocokan data maka program akan mengembalikan output, jika tidak pencarian akan terus berlanjut hingga akhir dari pembagian jumlah data tersebut. Algotihma ini biasanya banyak digunakan untuk mencari di program dengan jumlah data yang banyak, dimana kompleksitas dari algorithma ini adalah O(log n) di mana n adalah jumlah item. Pada saat menggunakan binary search, data yang berada di dalam array harus diurutkan terlebih dahulu.

Konsep Binary Search:

- Data diambil dari posisi 1 sampai posisi akhir N.
- Kemudian data akan dibagi menjadi dua untuk mendapatkan posisi data tengah.
- Selanjutnya data yang dicari akan dibandingkan dengan data yang berada di posisi tengah, apakah lebih besar atau lebih kecil.
- Apabila data yang dicari lebih besar dari data tengah, maka dapat dipastikan bahwa data yang dicari

kemungkinan berada di sebelah kanan dari datatengah. Proses pencarian selanjutnya akan dilakukan pembagian data menjadi dua bagian pada bagian kanan dengan acuan posisi data tengah akan menjadi posisi awal untuk pembagian tersebut.

- Apabila data yang dicari lebih kecil dari data tengah, maka dapat dipastikan bahwa data yang dicari kemungkinan berada di sebelah kiri dari data Tengah Proses pencarian selanjutnya akan dilakukan pembagian data menjadi dua bagian pada bagian kiri. Dengan acuan posisi data tengah akan menjadi posisi akhir untuk pembagian selanjutnya.



Gambar 2. Ilustrasi Binary Search

UNGUIDED

```
#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 << " diytemukan pada indeks ke-" <<
i << endl;
    } else {
        cout << cari << " tidak dapat ditemukan pada data." << endl;
    }

    return 0;
}
```

OUTPUT

```
        program sequential search sederhana

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

 Angka 10 diytemukan pada indeks ke-9
```

```
NAMA    :       ABDA FIRAS RAHMAN
KELAS   :       IF-11-B
NIM     :       2311102049
```

DESKRIPSI PROGRAM

Program dimulai dengan mendefinisikan array bernama 'data' dengan 10 elemen berisi nilai bilangan bulat. Kemudian, program menentukan nilai yang akan dicari dalam array tersebut, yaitu 10, yang disimpan dalam variabel 'cari'. Sebuah variabel boolean 'ketemu' diinisialisasi dengan nilai false untuk melacak apakah nilai yang dicari ditemukan dalam array atau tidak. Selanjutnya, program menggunakan perulangan for untuk memeriksa setiap elemen dalam array secara berurutan. Jika elemen yang sedang diperiksa sama dengan nilai 'cari', maka variabel 'ketemu' diubah menjadi true dan perulangan dihentikan dengan perintah 'break'. Setelah perulangan selesai, program menampilkan isi array dan memberikan output apakah nilai yang dicari ditemukan dalam array atau tidak. Jika ditemukan, program akan menampilkan indeks dari elemen tersebut. Jika tidak ditemukan, program akan menampilkan pesan bahwa nilai tidak dapat ditemukan dalam array.

GUIDED 2

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

#include <conio.h>
#include <iomanip>
int nomer[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 (nomer[j] < nomer[min])
            {
                min = j;
            }
        }
        temp = nomer[i];
        nomer[i] = nomer[min];
        nomer[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 (nomer[tengah] == cari)
        {
            b_flag = 1;
            break;
        }
        else if (nomer[tengah] < cari)
            awal = tengah + 1;
        else
            akhir = tengah - 1;
    }
    if (b_flag == 1)
        cout << "\n nomer ditemukan pada index ke- "<<tengah<<endl;
```

```
            else cout
             << "\n nomer tidak ditemukan\n";
}
int main()
{
    cout << "\t BINARY SEARCH " << endl;
    cout << "\n nomer : ";
    // tampilkan nomer awal
    for (int x = 0; x < 7; x++)
        cout << setw(3) << nomer[x];
    cout << endl;
    cout << "\n Masukkan nomer yang ingin Anda cari :";
    cin >> cari;
    cout << "\n nomer diurutkan : "; // urutkan nomer dengan selection
sort
    selection_sort(); // tampilkan nomer setelah diurutkan
    for (int x = 0; x < 7; x++)
        cout << setw(3) << nomer[x];
    cout << endl;
    binarysearch();
    _getche();
    return EXIT_SUCCESS;
}
```

OUTPUT

```
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
```

```
NAMA    :       ABDA FIRAS RAHMAN
KELAS   :       IF-11-B
NIM     :       2311102049

Ln 3, Col 18 | 58 characters | 100% | Win
```

DESKRIPSI PROGRAM

Program ini mendemonstrasikan penggunaan algoritma Selection Sort dan Binary Search pada array integer. Pertama, program mendeklarasikan array bernama "nomer" dengan 7 elemen yang berisi angka acak. Kemudian, fungsi "selection_sort()" digunakan untuk mengurutkan array tersebut secara ascending menggunakan algoritma Selection Sort. Setelah array terurut, program meminta pengguna untuk memasukkan angka yang ingin dicari dalam array. Fungsi "binarysearch()" kemudian melakukan pencarian angka yang diinputkan pengguna dengan menggunakan algoritma Binary Search pada array yang sudah terurut. Jika

angka ditemukan, program akan menampilkan indeks di mana angka tersebut berada dalam array. Jika tidak ditemukan, program akan memberikan pemberitahuan bahwa angka tidak ada dalam array.

UNGUIDED 1

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

bool binarySearch(string sentence, char target)
{
    int kiri = 0;
    int kanan = sentence.length() - 1;
    while (kiri <= kanan)
    {
        int mid = kiri + (kanan - kiri) / 2;
        if (sentence[mid] == target)
        {
            return true;
        }
        if (sentence[mid] < target)
        {
            kiri = mid + 1;
        }
        else
        {
            kanan = mid - 1;
        }
    }
    return false;
}
int main()
{
    string sentence;
    char tuju;
    cout << "Masukkan Kalimat atau nama lengkap: ";
    getline(cin, sentence);
    cout << "Masukkan huruf yang ingin dicari: ";
    cin >> tuju;
    transform(sentence.begin(), sentence.end(), sentence.begin(),
::tolower); // Mengubah kalimat menjadi lowercase agar pencariancase
intensitive
    bool found = binarySearch(sentence, tolower(tuju)); // Mencari huruf
dalam kalimat menggunakan Binary
    if (found)
```

```
    {
        cout << "Huruf " << tuju << " ditemukan dalam kalimat." << endl;
    }
    else
    {
        cout << "Huruf " << tuju << " tidak ditemukan dalam kalimat." <<
endl;
    }
    return 0;
}
```

OUTPUT

```
 unguided1.cpp -o unguided1 } ; if ($?) { .\ung
Masukkan Kalimat atau nama lengkap: abda firas
Masukkan huruf yang ingin dicari: i
Huruf i ditemukan dalam kalimat.
```

```
NAMA    :       ABDA FIRAS RAHMAN
KELAS   :       IF-11-B
NIM     :       2311102049
```

DESKRIPSI PROGRAM

Program ini mengimplementasikan algoritma Binary Search untuk mencari keberadaan sebuah karakter dalam sebuah string (kalimat atau nama lengkap). Pertama, program meminta pengguna untuk memasukkan sebuah kalimat atau nama lengkap. Kemudian, program meminta pengguna untuk memasukkan sebuah karakter yang ingin dicari dalam kalimat tersebut. Selanjutnya, program mengonversi kalimat menjadi huruf kecil (lowercase) menggunakan fungsi "transform" agar pencarian tidak case-sensitive. Fungsi "binarySearch" yang mengimplementasikan algoritma Binary Search kemudian dipanggil untuk mencari keberadaan karakter dalam kalimat. Jika karakter ditemukan, program akan menampilkan pesan bahwa karakter tersebut ditemukan dalam kalimat. Jika tidak ditemukan, program akan menampilkan pesan bahwa karakter tidak ditemukan dalam kalimat.

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 insensitve
```

```
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 << "Masukkan kalimat : ";
    getline(cin, sentence);
    int vowelCount = countVowels(sentence);
    cout << " Jumlah huruf vokal dalam kalimat: " << vowelCount
         << endl;
    return 0;
}
```

OUTPUT

```
 unguided2.cpp -o unguided2 } ; if ($?) { .\ung
Masukkan kalimat : pirass123
 Jumlah huruf vokal dalam kalimat: 2
NAMA    :       ABDA FIRAS RAHMAN
KELAS   :       IF-11-B
NIM     :       2311102049
```

DESKRIPSI PROGRAM

Program ini berfungsi untuk menghitung jumlah huruf vokal (a, e, i, o, u) yang terdapat dalam sebuah kalimat yang diinputkan oleh pengguna. Pertama, program meminta pengguna untuk memasukkan sebuah kalimat. Kemudian, fungsi "countVowels" dipanggil dengan melewatkan kalimat yang diinputkan sebagai parameter. Di dalam fungsi ini, kalimat diubah menjadi huruf kecil (lowercase) menggunakan fungsi "transform" agar pencarian huruf vokal tidak case-sensitive. Selanjutnya, sebuah loop dijalankan untuk memeriksa setiap karakter dalam kalimat. Jika karakter tersebut merupakan huruf vokal, maka penghitung (count) akan ditingkatkan. Setelah loop selesai, fungsi akan mengembalikan jumlah total huruf vokal yang ditemukan. Di akhir program, jumlah huruf vokal yang dihitung akan ditampilkan kepada pengguna.

UNGUIDED 3

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

int main()
{
    int data[] = {9, 4, 1, 8, 8, 10, 5, 4, 12, 4, 7, 4, 8, 2, 3, 6, 8, 7, 6,
4, 3, 2, 4, 2, 8};
    int size = sizeof(data) / sizeof(data[0]); //Mendapatkan ukuran array
    int target = 8;
    int count = 0; // Variabel untuk menghitung jumlahkemunculan angka 4
    for (int i = 0; i < size; i++)
    {
        if (data[i] == target)
        {
            count++;
        }
    }
    cout << "Jumlah angka 8 dalam data ini: " << count << endl;
    return 0;
}
```

OUTPUT

```
S\MATA KULIAH SEMESTER 2\STRUKTUR DATA DAN ALG
 unguided3.cpp -o unguided3 } ; if ($?) { .\ung
Jumlah angka 8 dalam data ini: 5
PS C:\Users\LENOVO\OneDrive\Documents\DOKUMEN
```

```
NAMA    :       ABDA FIRAS RAHMAN
KELAS   :       IF-11-B
NIM     :       2311102049
```

DESKRIPSI PROGRAM

Program ini berfungsi untuk menghitung jumlah kemunculan angka tertentu (dalam kasus ini angka 8) dalam sebuah array of integers. Pertama, program mendeklarasikan sebuah array bernama "data" yang berisi 25 elemen integer. Kemudian, ukuran array dihitung menggunakan ekspresi "sizeof(data) / sizeof(data[0])". Angka yang ingin dicari kemunculannya dalam array dideklarasikan sebagai variabel "target" dengan nilai 8. Sebuah variabel "count" diinisialisasi dengan nilai 0 untuk menghitung jumlah kemunculan angka target. Selanjutnya, sebuah loop "for" dijalankan untuk memeriksa setiap elemen dalam array. Jika elemen yang diperiksa sama dengan nilai target, maka variabel "count" akan ditingkatkan. Setelah loop selesai, program akan menampilkan jumlah kemunculan angka target dalam array tersebut.

DAFTAR PUSTAKA

1) Sequential Search – Algoritma Pencarian
https://mikirinkode.com/sequential-search/

2) BINARY SEARCH
https://socs.binus.ac.id/2019/12/26/binary-search/