LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL IV & V
PENGENALAN BAHASA C++ (BAGIAN KEDUA)ABSTRACT DATA TYPE (ADT)

Disusun Oleh :
NAMA : ABYAN RAHMAN AL FARIZ
NIM : 103112430021

Dosen
FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO

2025

A. Dasar Teori

Single Linked List adalah sebuah field pointer-nya hanya satu buah saja dan satu arah serta pada akhir node yang nodenya saling terhubung satu sama lain. Jadi Setiap node pada linked list mempunyai field yang berisi pointer ke node berikutnya, dan juga memiliki field yang berisi data. Node terakhir akan menunjuk ke NULL yang akan digunakan sebagai kondisi berhenti pada saat pembacaan isi linked list.

B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1 (Singlylist.h)

```
#ifndef SINGLYLIST_H_INCLUDED
#define SINGLYLIST_H_INCLUDED

#include <iostream>

#define Nil NULL

typedef int infotype;
typedef struct ElmList *address;

struct ElmList{
    infotype info;
    address next;
};

struct List{
    address First;
};

void CreateList(List &L);
address alokasi(infotype x);
void dealokasi(address &P);
void insertFirst(List &L, address P);
void insertLast(List &L, address P);
void printInfo(List L);

#endif
```

Guided 1 (Singlylist.cpp)

```cpp
#include "Singlylist.h"

void CreateList(List &L){
    L.First = Nil;
}

address alokasi(infotype x){
    address P = new ElmList;
    P->info = x;
    P->next = Nil;
    return P;
}

void dealokasi(address &P){
    delete P;
}

void insertFirst(List &L, address P){
    P->next = L.First;
    L.First = P;
}

void insertLast(List &L, address P){
    if(L.First == Nil){
        //Jika list kosong, insertLast sama dengan insertFirst
        insertFirst(L, P);
    }else{
        //Jika list tidak kosong, cari elemen terakhir
        address Last = L.First;
        while(Last->next != Nil){
            Last = Last->next;
        }
        //Sambungkan elemen terakhir ke elemen baru(P)
        Last->next = P;
    }
}

void printInfo(List L) {
    address P = L.First;
```

```
    if (P == Nil) {
        std::cout << "List Kosong!" << std::endl;
    } else {
        while (P != Nil) {
            std::cout << P->info << " ";
            P = P->next;
        }
        std::cout << std::endl;
    }
}
```

Guided 1 (main.cpp)

```
 #include <iostream>
#include <cstdlib>
#include "Singlylist.h"
#include "Singlylist.cpp"


using namespace std;

int main() {
    List L;
    address P; // Cukup satu pointer untuk digunakan berulang kali

    CreateList(L);

    cout << "Mengisi list menggunakan insertLast..." <<endl;

    //Mengisi list sesuai urutan
    P = alokasi(9);
    insertLast(L, P);

    P = alokasi(12);
    insertLast(L, P);

    P = alokasi(8);
    insertLast(L, P);
```

```
    P = alokasi(0);
    insertLast(L, P);

    P = alokasi(2);
    insertLast(L, P);

    cout << "Isi list sekarang adalah: ";
    printInfo(L);

    system("pause");
    return 0;
}
```

Screenshots Output:

```
PS D:\TelkomUniversity\Mata Kuliah\Semester 3\Struktur Data\Praktikum>  & 'c:\Users\ACER\.vscode\extensions\ms-vscode.cpptools-1.28.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-voemkjfk.tkm' '--stdout=Microsoft-MIEngine-Out-hwpl3jhk.rfk' '--stderr=Microsoft-MIEngine-Error-4ln5chxo.fik' '--pid=Microsoft-MIEngine-Pid-yw01kl5i.j3p' '--dbgExe=C:\Users\ACER\mingw32\bin\gdb.exe' '--interpreter=mi'
Mengisi list menggunakan insertLast...
Isi list sekarang adalah: 9 12 8 0 2
Press any key to continue . . .
```

Deskripsi:

Secara keseluruhan, ketiga file ini berisi tentang implementasi Abstract Data Type (ADT) Singly Linked List dalam bahasa C++. Program ini menunjukkan bagaimana sebuah list dapat dibangun menggunakan struktur data dinamis yang terdiri dari node-node saling terhubung melalui pointer. File Singlylist.h mendefinisikan struktur data dan deklarasi fungsi dasar, Singlylist.cpp berisi implementasi dari fungsi-fungsi yang mengatur pembuatan, penambahan, dan penampilan elemen dalam list, sedangkan main.cpp berfungsi sebagai driver yang menguji penggunaan ADT tersebut dengan menambahkan beberapa data ke dalam list dan menampilkannya. Dengan menerapkan prinsip abstraksi dan modularitas, program ini memperlihatkan cara pengelolaan data yang efisien, terstruktur, dan mudah dikembangkan tanpa perlu mengubah keseluruhan sistem.

C. Unguided/Tugas (berisi screenshot source code & output program disertai

penjelasannya)

Unguided 1 (Playlist.cpp)

```cpp
#include "Playlist.h"
#include <iostream>
#include <iomanip>
using namespace std;

void createPlaylist(Playlist &L) {
    L.first = NULL;
}

address alokasi(string judul, string penyanyi, float durasi) {
    address P = new Node;
    P->data.judul = judul;
    P->data.penyanyi = penyanyi;
    P->data.durasi = durasi;
    P->next = NULL;
    return P;
}

void dealokasi(address P) {
    delete P;
}

void insertFirst(Playlist &L, address P) {
    if (isEmpty(L)) {
        L.first = P;
    } else {
        P->next = L.first;
        L.first = P;
    }
}

void insertLast(Playlist &L, address P) {
    if (isEmpty(L)) {
        L.first = P;
    } else {
        address last = L.first;
        while (last->next != NULL) {
            last = last->next;
        }
        last->next = P;
```

```
    }
}

void insertAfter(Playlist &L, address P, int posisi) {
    if (isEmpty(L)) {
        L.first = P;
        return;
    }

    address current = L.first;
    int count = 1;

    while (current != NULL && count < posisi) {
        current = current->next;
        count++;
    }

    if (current != NULL) {
        P->next = current->next;
        current->next = P;
    } else {
        insertLast(L, P);
    }
}

void deleteLagu(Playlist &L, string judul) {
    if (isEmpty(L)) {
        cout << "Playlist kosong!" << endl;
        return;
    }

    address current = L.first;
    address prev = NULL;

    while (current != NULL && current->data.judul != judul) {
        prev = current;
        current = current->next;
    }

    if (current == NULL) {
        cout << "Lagu dengan judul '" << judul << "' tidak
ditemukan!" << endl;
        return;
```

```
    }

    if (prev == NULL) {
        L.first = current->next;
    } else {
        prev->next = current->next;
    }

    dealokasi(current);
    cout << "Lagu '" << judul << "' berhasil dihapus dari
playlist!" << endl;
}

void displayPlaylist(Playlist L) {
    if (isEmpty(L)) {
        cout << "Playlist kosong!" << endl;
        return;
    }

    cout << "\n=== PLAYLIST LAGU ===" << endl;
    cout <<
"-----------------------------------------------------" <<
endl;
    cout << left << setw(20) << "Judul"
         << setw(20) << "Penyanyi"
         << setw(10) << "Durasi" << endl;
    cout <<
"-----------------------------------------------------" <<
endl;

    address current = L.first;
    int nomor = 1;
    float totalDurasi = 0;

    while (current != NULL) {
        cout << nomor << ". "
             << left << setw(17) << current->data.judul
             << setw(20) << current->data.penyanyi
             << setw(10) << current->data.durasi << " menit" <<
endl;
        totalDurasi += current->data.durasi;
        current = current->next;
        nomor++;
```

```
    }

    cout <<
"-------------------------------------------------------" <<
endl;
    cout << "Total durasi playlist: " << totalDurasi << " menit"
<< endl;
    cout << "Jumlah lagu: " << (nomor - 1) << " lagu" << endl;
}

bool isEmpty(Playlist L) {
    return L.first == NULL;
}
```

Unguided 1 (Playlist.h)

```
#ifndef PLAYLIST_H
#define PLAYLIST_H

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

struct Lagu {
    string judul;
    string penyanyi;
    float durasi;
};

struct Node {
    Lagu data;
    Node* next;
};

typedef Node* address;
```

```
struct Playlist {
    address first;
};

void createPlaylist(Playlist &L);
address alokasi(string judul, string penyanyi, float durasi);
void dealokasi(address P);
void insertFirst(Playlist &L, address P);
void insertLast(Playlist &L, address P);
void insertAfter(Playlist &L, address P, int posisi);
void deleteLagu(Playlist &L, string judul);
void displayPlaylist(Playlist L);
bool isEmpty(Playlist L);

#endif
```

Unguided 1 (main.cpp)

```
#include <iostream>
#include "Playlist.h"
using namespace std;

void menu() {
    cout << "\n=== PROGRAM PLAYLIST LAGU ===" << endl;
    cout << "1. Tambah lagu di awal playlist" << endl;
    cout << "2. Tambah lagu di akhir playlist" << endl;
    cout << "3. Tambah lagu setelah lagu ke-3" << endl;
    cout << "4. Hapus lagu berdasarkan judul" << endl;
    cout << "5. Tampilkan seluruh playlist" << endl;
    cout << "6. Keluar" << endl;
    cout << "Pilih menu: ";
}

int main() {
    Playlist myPlaylist;
    createPlaylist(myPlaylist);

    int pilihan;
    string judul, penyanyi;
    float durasi;
```

```
    insertFirst(myPlaylist, alokasi("Bohemian Rhapsody", "Queen",
5.55));
    insertLast(myPlaylist, alokasi("Yesterday", "The Beatles",
2.05));
    insertLast(myPlaylist, alokasi("Hotel California", "Eagles",
6.30));

    do {
        menu();
        cin >> pilihan;
        cin.ignore();

        switch (pilihan) {
            case 1:
                cout << "\n--- Tambah Lagu di Awal ---" << endl;
                cout << "Judul: "; getline(cin, judul);
                cout << "Penyanyi: "; getline(cin, penyanyi);
                cout << "Durasi (menit): "; cin >> durasi;
                insertFirst(myPlaylist, alokasi(judul, penyanyi,
durasi));
                cout << "Lagu berhasil ditambahkan di awal
playlist!" << endl;
                break;

            case 2:
                cout << "\n--- Tambah Lagu di Akhir ---" << endl;
                cout << "Judul: "; getline(cin, judul);
                cout << "Penyanyi: "; getline(cin, penyanyi);
                cout << "Durasi (menit): "; cin >> durasi;
                insertLast(myPlaylist, alokasi(judul, penyanyi,
durasi));
                cout << "Lagu berhasil ditambahkan di akhir
playlist!" << endl;
                break;

            case 3:
                cout << "\n--- Tambah Lagu Setelah Lagu ke-3 ---"
<< endl;
                cout << "Judul: "; getline(cin, judul);
                cout << "Penyanyi: "; getline(cin, penyanyi);
                cout << "Durasi (menit): "; cin >> durasi;
                insertAfter(myPlaylist, alokasi(judul, penyanyi,
```

```
durasi), 3);
                cout << "Lagu berhasil ditambahkan setelah lagu
ke-3!" << endl;
                break;

            case 4:
                cout << "\n--- Hapus Lagu ---" << endl;
                cout << "Judul lagu yang akan dihapus: ";
                getline(cin, judul);
                deleteLagu(myPlaylist, judul);
                break;

            case 5:
                displayPlaylist(myPlaylist);
                break;

            case 6:
                cout << "Terima kasih telah menggunakan program!"
<< endl;
                break;

            default:
                cout << "Pilihan tidak valid!" << endl;
        }

    } while (pilihan != 6);

    return 0;
}
```

Screenshots Output:

```
PS D:\TelkomUniversity\Mata Kuliah\Semester 3\Struktur Data\Praktikum\Modul 4 -\Unguided> ./program
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 1

--- Tambah Lagu di Awal ---
Judul: Terbuang Dalam Waktu
Penyanyi: Barasuara
Durasi (menit): 4.41
Lagu berhasil ditambahkan di awal playlist!

=== PROGRAM PLAYLIST LAGU ===
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 5

=== PLAYLIST LAGU ===
-----------------------------------------------------
Judul              Penyanyi           Durasi
-----------------------------------------------------
1. Terbuang Dalam WaktuBarasuara          4.41       menit
2. Bohemian RhapsodyQueen              5.55       menit
3. Yesterday        The Beatles        2.05       menit
4. Hotel California Eagles             6.3        menit
-----------------------------------------------------
Total durasi playlist: 18.31 menit
Jumlah lagu: 4 lagu
```

```
PS D:\TelkomUniversity\Mata Kuliah\Semester 3\Struktur Data\Praktikum\Modul 4 -\Unguided> ./program
=== PROGRAM PLAYLIST LAGU ===
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 2

--- Tambah Lagu di Akhir ---
Judul: Everyday
Penyanyi: Ariana Grande
Durasi (menit): 3.14
Lagu berhasil ditambahkan di akhir playlist!

=== PROGRAM PLAYLIST LAGU ===
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 5

=== PLAYLIST LAGU ===
-----------------------------------------------------
Judul               Penyanyi            Durasi
-----------------------------------------------------
1. Terbuang Dalam WaktuBarasuara           4.41       menit
2. Bohemian RhapsodyQueen               5.55       menit
3. Yesterday        The Beatles         2.05       menit
4. Hotel California Eagles              6.3        menit
5. Everyday         Ariana Grande       3.14       menit
-----------------------------------------------------
Total durasi playlist: 21.45 menit
```

```
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 3

--- Tambah Lagu Setelah Lagu ke-3 ---
Judul: Tampar
Penyanyi: Juicy Luicy
Durasi (menit): 3.22
Lagu berhasil ditambahkan setelah lagu ke-3!

=== PROGRAM PLAYLIST LAGU ===
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 5

=== PLAYLIST LAGU ===
-----------------------------------------------------
Judul               Penyanyi            Durasi
-----------------------------------------------------
1. Terbuang Dalam WaktuBarasuara           4.41       menit
2. Bohemian RhapsodyQueen                  5.55       menit
3. Yesterday        The Beatles            2.05       menit
4. Tampar           Juicy Luicy            3.22       menit
5. Hotel California Eagles                 6.3        menit
6. Everyday         Ariana Grande          3.14       menit
-----------------------------------------------------
Total durasi playlist: 24.67 menit
```

```
PS D:\TelkomUniversity\Mata Kuliah\Semester 3\Struktur Data\Praktikum\Modul 4 -\Unguided> ./program
=== PROGRAM PLAYLIST LAGU ===
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 4

--- Hapus Lagu ---
Judul lagu yang akan dihapus: Bohemian Rhapsody
Lagu 'Bohemian Rhapsody' berhasil dihapus dari playlist!

=== PROGRAM PLAYLIST LAGU ===
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 5

=== PLAYLIST LAGU ===
------------------------------------------------------
Judul               Penyanyi            Durasi
------------------------------------------------------
1. Terbuang Dalam WaktuBarasuara           4.41       menit
2. Yesterday        The Beatles         2.05       menit
3. Tampar           Juicy Luicy         3.22       menit
4. Hotel California Eagles              6.3        menit
5. Everyday         Ariana Grande       3.14       menit
------------------------------------------------------
Total durasi playlist: 19.12 menit
Jumlah lagu: 5 lagu
```

```
=== PROGRAM PLAYLIST LAGU ===
1. Tambah lagu di awal playlist
2. Tambah lagu di akhir playlist
3. Tambah lagu setelah lagu ke-3
4. Hapus lagu berdasarkan judul
5. Tampilkan seluruh playlist
6. Keluar
Pilih menu: 6
Terima kasih telah menggunakan program!
```

Deskripsi:

Secara keseluruhan, ketiga file ini berisi tentang implementasi Abstract Data Type (ADT) Playlist Lagu menggunakan struktur data Singly Linked List dalam bahasa C++. Program ini dibuat untuk mengelola daftar lagu dengan fitur menambah, menampilkan, dan menghapus data lagu secara dinamis. File Playlist.h berfungsi sebagai tempat pendefinisian struktur data dan deklarasi fungsi-fungsi yang digunakan, Playlist.cpp berisi implementasi dari fungsi-fungsi tersebut seperti pembuatan playlist, penambahan lagu di berbagai posisi, penghapusan lagu, serta penampilan daftar lagu,

sedangkan main.cpp digunakan sebagai driver program untuk menjalankan seluruh proses pengelolaan playlist melalui menu interaktif.

D. Kesimpulan

Kesimpulannya, materi Singly Linked List mengajarkan cara mengelola data secara dinamis menggunakan konsep node yang saling terhubung melalui pointer. Struktur ini memungkinkan penambahan dan penghapusan data tanpa perlu menggeser elemen lain seperti pada array, sehingga lebih efisien dalam penggunaan memori. Melalui implementasi program yang telah dibuat, dapat dipahami bahwa setiap node dalam singly linked list terdiri dari dua bagian utama, yaitu data (info) dan pointer next yang menunjuk ke elemen berikutnya.

E. Referensi

https://daismabali.com/artikel_detail/54/1/Mengenal-Single-Linked-List-dalam-Struktur-Data.html

https://www.trivusi.web.id/2022/07/struktur-data-linked-list.html

https://terapan-ti.vokasi.unesa.ac.id/post/memahami-konsep-dan-jenis-jenis-linked-list-dalam-struktur-data