LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL IV
SINGLY LINKED LIST

Disusun Oleh :

NAMA : Damanik, Yohanes Geovan Ondova

NIM : 103112400022

Dosen

FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

A. Dasar Teori

Linked List merupakan struktur data dinamis yang terdiri dari serangkaian elemen (node) yang saling terhubung melalui pointer. Berbeda dengan array yang bersifat statis, linked list dapat tumbuh dan menyusut secara fleksibel sesuai kebutuhan. Setiap node dalam Singly Linked List terdiri dari dua bagian utama: data yang menyimpan informasi dan pointer next yang menunjuk ke node berikutnya. Struktur ini hanya memiliki satu arah penelusuran dari node pertama (head/first) hingga node terakhir yang menunjuk ke NULL.

Operasi dasar dalam Linked List meliputi pembuatan list (createList), alokasi memori (alokasi), penyisipan (insertFirst, insertLast, insertAfter), penghapusan (deleteFirst, deleteLast, deleteAfter), penelusuran (printInfo), dan pembebasan memori (dealokasi). Keunggulan utama Singly Linked List terletak pada kemudahan dalam melakukan penyisipan dan penghapusan node di tengah list tanpa perlu menggeser elemen lainnya, meskipun hanya mendukung penelusuran satu arah maju.

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

Guided 1

Singlylist.cpp

```
#include "Singlylist.h"

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

address alokasi(infotype x) {
    address P = new ElmtList;
    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;
    }
}
```

Singlylist.h

```
#ifndef SINGLYLIST_H_INCLUDED
#define SINGLYLIST_H_INCLUDED
#define Nil NULL

#include <iostream>


typedef int infotype;
typedef struct ElmtList *address;

struct ElmtList {
    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 // SINGLYLIST_H_INCLUDED
```

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

   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;
}
```

Output

```
PS C:\Users\Lenovo\Documents\STRUKKTUR DATA MODUL 3> cd "c:\Users\Lenovo
$?) { g++ main.cpp -o main } ; if ($?) { .\main }
Input nama = Geovan
Input nilai = 99
Input nilai = 98
rata - rata = 98.5
PS C:\Users\Lenovo\Documents\STRUKKTUR DATA MODUL 3>
```

Deskripsi:

Program ini merupakan implementasi Singly Linked List dalam C++ yang terdiri dari tiga file: header file (Singlylist.h) yang mendefinisikan struktur data list dan deklarasi fungsi, file implementasi (Singlylist.cpp) yang berisi definisi fungsi-fungsi operasi linked list seperti CreateList, alokasi, insertFirst, insertLast, dan printInfo, serta file main (main.cpp) yang mendemonstrasikan penggunaan linked list dengan membuat list kosong, mengalokasikan lima elemen berisi nilai integer (9, 12, 8, 0, 2) menggunakan insertLast, dan menampilkan seluruh isi list secara berurutan sehingga output yang dihasilkan adalah "9 12 8 0 2". Program ini menunjukkan konsep dasar linked list dimana setiap elemen dialokasikan secara dinamis dan dihubungkan melalui pointer, dengan operasi penambahan elemen di akhir list dan traversal untuk menampilkan data..

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

Unguided 1

Playlist.cpp

```
#include "Playlist.h"

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

address alokasi(Lagu laguBaru) {
    address P = new Node;
    P->info = laguBaru;
    P->next = NULL;
    return P;
}

void dealokasi(address P) {
    delete P;
}

void insertFirst(Playlist &L, address P) {
    P->next = L.first;
```

```
    L.first = P;
}

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

void insertAfter3(Playlist &L, address P) {
    if (L.first == NULL) {
        cout << "Playlist masih kosong!\n";
        return;
    }

    address Q = L.first;
    int count = 1;
    while (Q != NULL && count < 3) {
        Q = Q->next;
        count++;
    }

    if (Q == NULL) {
        cout << "Jumlah lagu kurang dari 3, ditambahkan di akhir playlist.\n";
        insertLast(L, P);
    } else {
        P->next = Q->next;
        Q->next = P;
    }
}

void deleteByJudul(Playlist &L, string judul) {
    if (L.first == NULL) {
        cout << "Playlist kosong!\n";
        return;
    }

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

    while (P != NULL && P->info.judul != judul) {
        prev = P;
        P = P->next;
    }
```

```
    if (P == NULL) {
        cout << "Lagu dengan judul '" << judul << "' tidak ditemukan.\n";
    } else {
        if (prev == NULL) {
            L.first = P->next;
        } else {
            prev->next = P->next;
        }
        cout << "Lagu '" << judul << "' berhasil dihapus.\n";
        dealokasi(P);
    }
}

void printPlaylist(Playlist L) {
    if (L.first == NULL) {
        cout << "Playlist kosong.\n";
        return;
    }

    address P = L.first;
    int i = 1;
    cout << "\n=== Daftar Lagu dalam Playlist ===\n";
    while (P != NULL) {
        cout << i << ". Judul: " << P->info.judul << endl;
        cout << "   Penyanyi: " << P->info.penyanyi << endl;
        cout << "   Durasi: " << P->info.durasi << " menit\n";
        cout << "----------------------------------\n";
        P = P->next;
        i++;
    }
}
```

Playlist.h

```
#ifndef PLAYLIST_H_INCLUDED
#define PLAYLIST_H_INCLUDED

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

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

typedef struct Node *address;
```

```
struct Node {
   Lagu info;
   address next;
};

struct Playlist {
   address first;
};

// Prototipe fungsi
void createList(Playlist &L);
address alokasi(Lagu laguBaru);
void dealokasi(address P);

void insertFirst(Playlist &L, address P);
void insertLast(Playlist &L, address P);
void insertAfter3(Playlist &L, address P);
void deleteByJudul(Playlist &L, string judul);
void printPlaylist(Playlist L);

#endif
#endif
```

Main.cpp

```
#include "Playlist.h"
#include <iostream>
#include "Playlist.cpp"

using namespace std;

int main() {
   Playlist L;
   createList(L);

   Lagu lagu1 = {"Hati-Hati di Jalan", "Tulus", 4.2};
   Lagu lagu2 = {"Cinta Luar Biasa", "Andmesh", 3.8};
   Lagu lagu3 = {"Celengan Rindu", "Fiersa Besari", 4.0};
   Lagu lagu4 = {"Laskar Pelangi", "Nidji", 5.0};
   Lagu lagu5 = {"Akad", "Payung Teduh", 4.3};

   insertFirst(L, alokasi(lagu1));
   insertLast(L, alokasi(lagu2));
   insertLast(L, alokasi(lagu3));
   insertAfter3(L, alokasi(lagu4)); // setelah lagu ke-3
   insertLast(L, alokasi(lagu5));

   printPlaylist(L);
```

```
    cout << "\nMenghapus lagu dengan judul 'Cinta Luar Biasa'...\n";
    deleteByJudul(L, "Cinta Luar Biasa");

    printPlaylist(L);

    return 0;
}
```

Output

```
PS C:\Users\Lenovo\Documents\STRUKTUR DATA MODUL 4> cd "c:\Users\Lenovo\Documer
) { g++ mainn.cpp -o mainn } ; if ($?) { .\mainn }

=== Daftar Lagu dalam Playlist ===
1. Judul: Hati-Hati di Jalan
   Penyanyi: Tulus
   Durasi: 4.2 menit
---------------------------------
2. Judul: Cinta Luar Biasa
   Penyanyi: Andmesh
   Durasi: 3.8 menit
---------------------------------
3. Judul: Celengan Rindu
   Penyanyi: Fiersa Besari
   Durasi: 4 menit
---------------------------------
4. Judul: Laskar Pelangi
   Penyanyi: Nidji
   Durasi: 5 menit
---------------------------------
5. Judul: Akad
   Penyanyi: Payung Teduh
   Durasi: 4.3 menit
---------------------------------
```

```
Menghapus lagu dengan judul 'Cinta Luar Biasa'...
Lagu 'Cinta Luar Biasa' berhasil dihapus.

=== Daftar Lagu dalam Playlist ===
1. Judul: Hati-Hati di Jalan
1. Judul: Hati-Hati di Jalan
   Penyanyi: Tulus
   Durasi: 4.2 menit
---------------------------------
2. Judul: Celengan Rindu
   Penyanyi: Fiersa Besari
   Durasi: 4 menit
---------------------------------
3. Judul: Laskar Pelangi
   Penyanyi: Nidji
   Durasi: 5 menit
---------------------------------
4. Judul: Akad
   Penyanyi: Payung Teduh
   Durasi: 4.3 menit
---------------------------------
PS C:\Users\Lenovo\Documents\STRUKTUR DATA MODUL 4>
```

Deskripsi:

Program ini merupakan implementasi manajemen playlist lagu menggunakan struktur data Singly Linked List dalam bahasa C++, yang terdiri dari tiga file utama: header file (Playlist.h) untuk deklarasi struktur data dan fungsi, file implementasi (playlist.cpp) yang berisi definisi fungsi-fungsi operasi linked list, dan file main (mainn.cpp) sebagai driver program. Program ini memungkinkan pembuatan playlist dengan operasi tambah lagu di awal (insertFirst), di akhir (insertLast), setelah elemen ketiga (insertAfter3), serta penghapusan lagu berdasarkan judul (deleteByJudul). Dalam contoh eksekusi, program membuat playlist berisi lima lagu dengan informasi judul, penyanyi, dan durasi, kemudian mendemonstrasikan penghapusan satu lagu sebelum menampilkan seluruh daftar playlist yang tersisa, menunjukkan kemampuan manipulasi data lagu secara dinamis menggunakan linked list.

D. Kesimpulan

Praktikum Singly Linked List ini telah berhasil mengimplementasikan struktur data dinamis baik untuk data numerik sederhana maupun data kompleks berupa playlist lagu, yang menunjukkan keunggulan linked list dalam fleksibilitas penambahan dan penghapusan elemen tanpa perlu menggeser data lainnya. Melalui program guided dan unguided, praktikum ini membuktikan efektivitas operasi dasar linked list seperti insertFirst, insertLast, insertAfter, deleteByJudul, dan traversal, sekaligus mendemonstrasikan kemampuan linked list dalam mengelola data yang terstruktur dengan alokasi memori dinamis serta kemudahan modifikasi data dibandingkan array statis.

E. Referensi

https://www.programiz.com/dsa/linked-list

https://www.programiz.com/cpp-programming/pointers

https://www.geeksforgeeks.org/cpp/vector-insert-function-in-cpp-stl/