LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL 5
PENGENALAN CODE BLOCKS

Disusun Oleh :
NAMA :Muhamad Ilham Syahid
NIM :103112400155

Dosen
WAHYU ANDI SAPUTRA

PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

A. Dasar Teori

Singly Linked List adalah struktur data linier yang terdiri dari node-node yang saling terhubung satu arah. Setiap node memiliki **data (info)** dan **pointer (next)** yang menunjuk ke node berikutnya. List kosong jika `first = NULL`.

Modul 5 membahas **operasi searching**, yaitu proses pencarian data dengan menelusuri node dari awal hingga data ditemukan atau list berakhir. Searching penting untuk mendukung operasi lain seperti **insert, delete, dan update**.

Selain searching, modul ini juga mencakup operasi dasar Singly Linked List seperti **pembuatan list, penambahan, penghapusan, pencetakan data, serta manajemen memori (alokasi dan dealokasi)** menggunakan pointer dalam bahasa C/C++.

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

Guided 1

Singlylist.h

```
#ifndef SINGLYLIST_H
#define SINGLYLIST_H


#include <iostream>
using namespace std;


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 printInfo(List L);
address findElm(List L, infotype x);
int totalInfo(List L);


#endif
```

Guided 2

Singlylist.cpp

```
#include "Singlylist.h"


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


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


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


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


void printInfo(List L) {
address P = L.First;
while (P != NULL) {
cout << P->info << " ";
P = P->next;
}
cout << endl;
}
```

```
address findElm(List L, infotype x) {
address P = L.First;
while (P != NULL) {
if (P->info == x)
return P;
P = P->next;
}
return NULL;
}

int totalInfo(List L) {
address P = L.First;
int total = 0;
while (P != NULL) {
total += P->info;
P = P->next;
}
return total;
}
```

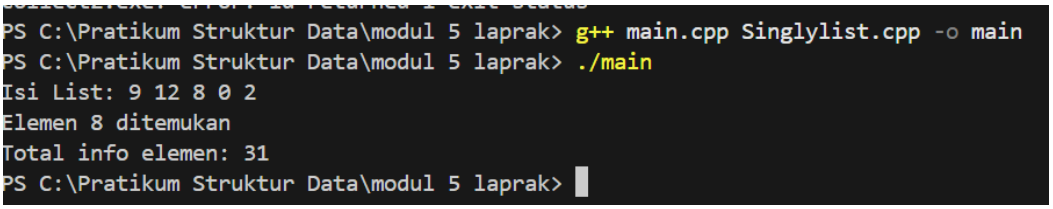
Guided 3

Main.cpp

```
#include "Singlylist.h"

int main() {
List L;
address P1, P2, P3, P4, P5;

CreateList(L);

P1 = alokasi(2);
insertFirst(L, P1);

P2 = alokasi(0);
insertFirst(L, P2);

P3 = alokasi(8);
insertFirst(L, P3);
```

```
P4 = alokasi(12);
insertFirst(L, P4);


P5 = alokasi(9);
insertFirst(L, P5);


cout << "Isi List: ";
printInfo(L);


if (findElm(L, 8) != NULL)
cout << "Elemen 8 ditemukan" << endl;
else
cout << "Elemen 8 tidak ditemukan" << endl;


cout << "Total info elemen: " << totalInfo(L) << endl;


return 0;
}
```

Screenshots Output



Deskripsi:

Program ini mengelola playlist lagu menggunakan struktur data **Single Linked List** dalam bahasa C++. Setiap lagu disimpan dalam sebuah node yang berisi judul, penyanyi, dan durasi lagu serta pointer ke node berikutnya. Program menyediakan operasi untuk menambah lagu di awal, di akhir, dan setelah lagu ke-3, menghapus lagu berdasarkan judul, serta menampilkan seluruh isi playlist. Struktur ini bersifat dinamis sehingga memudahkan penambahan dan penghapusan data.

C. Kesimpulan

Program ini menunjukkan cara penggunaan pointer dan linked list untuk menyimpan dan mengelola data secara dinamis, serta memperlihatkan operasi dasar seperti insert, searching, dan traversal pada Singly Linked List.

Referensi

Modul Praktikum Struktur Data – Modul 5: Singly Linked List**,**