LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL 13
MULTI LINKED LIST

NAMA : Najwa Areefa Ghaisani

NIM : 103122400028

Dosen

Diah Septiani S.Kom M.Cs

PROGRAM STUDI REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO

2025

A. Dasar Teori

Multi List merupakan sekumpulan list yang berbeda yang memiliki suatu keterhubungan satu sama lain. Tiap elemen dalam multi link list dapat membentuk list sendiri.Biasanya ada yang bersifat sebagai list induk dan list anak[1] .

Guided

Screenshots Output:

Deskripsi:

B. Unguided

Unguided

circularlist.h

```
#ifndef CIRCULARLIST_H
#define CIRCULARLIST_H

// Najwa Areefa Ghaisani_103122400028

#include <iostream>
using namespace std;

typedef struct mahasiswa{
   string nama;
   string nim;
   char jenis_kelamin;
   float ipk;
} infotype;

typedef struct elmlist *address;

typedef struct elmlist {
   infotype info;
   address next;
} elmlist;

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 insertAfter(List &L, address Prec, address P);
void deleteFirst(List &L, address &P);
void deleteLast(List &L, address &P);
void deleteAfter(List &L, address Prec, address &P);
address findElm(List L, string nim);
```

```
void printInfo(List L);

#endif
```

circularlist.cpp

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

// Najwa Areefa Ghaisani_103122400028

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

   }
```

```
}

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

void insertAfter(List &L, address Prec, address P) {
   if (Prec != NULL) {
      P -> next = Prec -> next;
      Prec -> next = P;
   }
}

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

void deleteLast(List &L, address &P) {
   if (L.first != NULL) {
      address Q = L.first;
      address Prev = NULL;
```

```
        while (Q -> next != L.first) {
            Prev = Q;
            Q = Q -> next;
        }
        P = Q;
        if (Prev == NULL) {
            L.first = NULL;
        } else {
            Prev -> next = L.first;
        }
        P -> next = NULL;
    }
}

void deleteAfter(List &L, address Prec, address &P) {
    if (Prec != NULL && Prec -> next != NULL) {
        P = Prec -> next;
        Prec -> next = P -> next;
        P -> next = NULL;
    }
}

address findElm(List L, string nim) {
    if (L.first == NULL) return NULL;

    address P = L.first;
    do {
        if (P -> info.nim == nim) {
            return P;
        }
        P = P -> next;
    } while (P != L.first);
    return NULL;
}

void printInfo(List L) {
    if (L.first == NULL) {
        cout << "List Kosongg!" << endl;
        return;
    }
    address P = L.first;
    do {
```

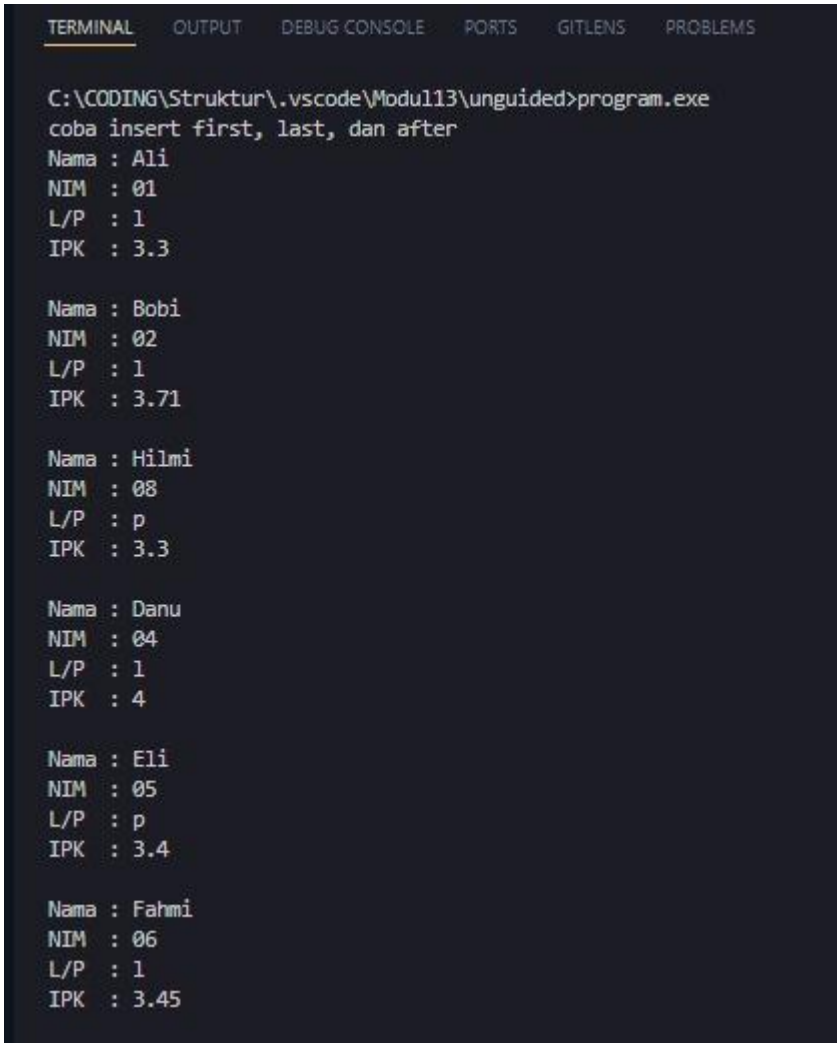
```
        cout << "Nama : " << P -> info.nama << endl;
        cout << "NIM  : " << P -> info.nim << endl;
        cout << "L/P  : " << P -> info.jenis_kelamin << endl;
        cout << "IPK  : " << P -> info.ipk << endl;
        cout << endl;
        P = P -> next;
    } while (P != L.first);
}
```

main.cpp

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

// Najwa Areefa Ghaisani_103122400028

address createData (string nama, string nim, char jenis_kelamin, float ipk)
{
    infotype x;
    x.nama = nama;
    x.nim = nim;
    x.jenis_kelamin = jenis_kelamin;
    x.ipk = ipk;
    return alokasi(x);
}

int main() {
    List L, A, B, L2;
    address P1 = NULL;
    address P2 = NULL;
    infotype x;
    createList(L);

    cout << "coba insert first, last, dan after" << endl;
    P1 = createData("Danu", "04", 'l', 4.0);
    insertFirst(L, P1);

    P1 = createData("Fahmi", "06", 'l', 3.45);
    insertLast(L, P1);

    P1 = createData("Bobi", "02", 'l', 3.71);
```

```
    insertFirst(L, P1);

    P1 = createData("Ali", "01", 'l', 3.3);
    insertFirst(L, P1);

    P1 = createData("Gita", "07", 'p', 3.75);
    insertLast(L, P1);

    x.nim = "07";
    P1 = findElm(L, x.nim);
    P2 = createData("Cindi", "03", 'p', 3.5);
    insertAfter(L, P1, P2);

    x.nim = "02";
    P1 = findElm(L, x.nim);
    P2 = createData("Hilmi", "08", 'p', 3.3);
    insertAfter(L, P1, P2);

    x.nim = "04";
    P1 = findElm(L, x.nim);
    P2 = createData("Eli", "05", 'p', 3.4);
    insertAfter(L, P1, P2);

    printInfo(L);
    return 0;
}
```

Screenshots Output:



Deskripsi:

Program ini mengimplementasikan struktur data untuk mengelola data mahasiswa yang terdiri dari nama, NIM, jenis kelamin, dan IPK. Circular Linked List adalah struktur data linear di mana setiap elemen (node) terhubung dengan elemen lainnya secara berantai, dan elemen terakhir terhubung kembali ke elemen pertama sehingga membentuk struktur melingkar tanpa ada node yang menunjuk ke NULL.

Program ini menyediakan berbagai operasi dasar linked list seperti createList untuk inisialisasi list kosong, alokasi untuk membuat node baru, insertFirst untuk menyisipkan node di awal list, insertLast untuk menyisipkan

di akhir, dan insertAfter untuk menyisipkan setelah node tertentu. Selain itu, tersedia juga operasi penghapusan seperti deleteFirst, deleteLast, dan deleteAfter, serta fungsi findElm untuk mencari mahasiswa berdasarkan NIM dan printInfo untuk menampilkan seluruh data mahasiswa.

Dalam program utama, dilakukan pengisian data 8 mahasiswa menggunakan kombinasi operasi insert, di mana beberapa mahasiswa dimasukkan di awal (Ali, Bobi, Danu), beberapa di akhir (Fahmi, Gita), dan sisanya disisipkan setelah mahasiswa tertentu menggunakan fungsi findElm untuk mencari posisi (Cindi setelah Gita, Hilmi setelah Bobi, Eli setelah Danu). Hasil akhir program menampilkan daftar lengkap mahasiswa dengan format yang terstruktur, menampilkan nama, NIM, jenis kelamin, dan IPK setiap mahasiswa dalam baris terpisah, yang membuktikan bahwa operasi circular linked list berjalan dengan benar dan data tersimpan secara melingkar di mana node terakhir menunjuk kembali ke node pertama.

C. Kesimpulan

Dari praktikum Modul 13 tentang Multi Linked List dapat disimpulkan bahwa Circular Linked List adalah struktur data dinamis di mana node terakhir terhubung kembali ke node pertama membentuk lingkaran. Implementasi circular linked list memerlukan penanganan khusus pada operasi insert dan delete untuk menjaga sifat circular-nya, terutama dalam memastikan node terakhir selalu menunjuk ke first. Struktur ini memiliki keunggulan dalam traversal yang dapat dimulai dari node mana saja dan cocok untuk aplikasi yang membutuhkan akses data secara berulang seperti playlist atau scheduling.

Operasi-operasi dasar seperti insertFirst, insertLast, insertAfter, deleteFirst, deleteLast, dan findElm telah berhasil diimplementasikan dengan kompleksitas waktu umumnya O(n) karena perlu traversal list. Melalui praktikum ini, mahasiswa memahami konsep pointer, dynamic memory allocation, dan bagaimana memanipulasi struktur data yang lebih kompleks dibanding single atau double linked list biasa.

D. Referensi

[1] A. U. Q., *Draft Modul Struktur Data SE Ganjil-2526*. Universitas, 2025.