

LAPORAN PRAKTIKUM

MODUL IV LINKED LIST CIRCULAR DAN NON CIRCULAR



Disusun oleh:
Muhammad Rifki Fadhilah
NIM: 2311102032

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

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
PURWOKERTO
2024**

BAB I

TUJUAN PRAKTIKUM

1. Mahasiswa mengetahui dan memahami linked list circular dan non circular
2. Mahasiswa membuat linked list circular dan non circular.
3. Mahasiswa dapat mengaplikasikan atau menerapkan linked list circular dan non circular pada program yang dibuat

BAB II

DASAR TEORI

1. Single Linked List Circular

Pengertian:

- a. Single: artinya field pointer-nya hanya satu buah saja dan satu arah.
- b. Linked List: artinya node-node tersebut saling terhubung satu sama lain.
- c. Circular: artinya pointer next-nya akan menunjuk pada dirinya sendiri sehingga berputar
- d. Setiap node pada linked list mempunyai field yang berisi pointer ke node berikutnya, dan juga memiliki field yang berisi data.
- e. Pada akhir linked list, node terakhir akan menunjuk ke node terdepan sehingga linked list tersebut berputar.

Deklarasi Node:

```
typedef struct TNode{  
    int data;  
    TNode *next;  
};
```

2. Single Linked List Non Circular

Pengertian:

- a. Single: Field pointer-nya hanya satu buah saja dan satu arah serta pada akhir node, pointer-nya menunjuk NULL
- b. Linked List: artinya node-node tersebut saling satu sama lain
- c. Setiap node pada linked list mempunyai field yang berisi pointer ke node berikutnya, dan juga memiliki field yang berisi data.
- d. Node terakhir akan menunjuk ke NULL yang akan digunakan sebagai kondisi berhenti pada saat pembacaan isi linked list.

Deklarasi Node:

```
typedef struct TNode{  
    int data;  
    TNode *next;  
};
```

Penjelasan:

- a. Pembuatan struct bernama TNode yang berisi 2 field, yaitu field data bertipe integer dan field **next** yang bertipe pointer dari TNode
- b. Setelah pembuatan struct, buat variabel head yang bertipe pointer dari TNode yang berguna sebagai kepala linked list.

BAB IV

GUIDED

1. Guided 1

Source code

```
#include <iostream>

using namespace std;

// PROGRAM SINGLE LINKED LIST NON-CIRCULAR

// Deklarasi struct node
struct Node
{
    int data;
    Node *next;
};

Node *head; // Deklarasi head
Node *tail; // Deklarasi tail

// Inisialisasi Node
void init()
{
    head = NULL;
    tail = NULL;
}

// Pengecekan apakah linked list kosong
bool isEmpty()
{
    if (head == NULL)
    {
        return true;
    }
}
```

```

    }
    else
    {
        return false;
    }
}

// Tambah depan
void insertDepan(int nilai)
{
    // buat node baru
    Node *baru = new Node();
    baru->data = nilai;
    baru->next = NULL;
    if (isEmpty() == true)
    {
        head = tail = baru;
        head->next = NULL;
    }
    else
    {
        baru->next = head;
        head = baru;
    }
}

// Tambah belakang
void insertBelakang(int nilai)
{
    // buat node baru
    Node *baru = new Node();
    baru->data = nilai;
    baru->next = NULL;
    if (isEmpty() == true)

```

```

        {
            head = tail = baru;
            head->next = NULL;
        }
    else
    {
        tail->next = baru;
        tail = baru;
    }
}

// Hitung jumlah list
int hitungList()
{
    Node *hitung;
    hitung = head;
    int jumlah = 0;
    while (hitung != NULL)
    {
        jumlah++;
        hitung = hitung->next;
    }
    return jumlah;
}

// Tambah tengah
void insertTengah(int data, int posisi)
{
    if (posisi < 1 || posisi > hitungList())
    {
        cout << "Posisi di luar jangkauan" << endl;
    }
    else if (posisi == 1)
    {

```

```

        cout << "Posisi bukan posisi tengah" << endl;
    }
    else
    {
        Node *baru, *bantu;
        baru = new Node();
        baru->data = data;

        // tranversing
        bantu = head;
        int nomor = 1;
        while (nomor < posisi - 1)
        {
            bantu = bantu->next;
            nomor++;
        }

        baru->next = bantu->next;
        bantu->next = baru;
    }
}

// Hapus depan
void hapusDepan()
{
    Node *hapus;
    if (isEmpty() == false)
    {
        if (head->next != NULL)
        {
            hapus = head;
            head = head->next;
            delete hapus;
        }
    }
}

```



```

        else
        {
            head = tail = NULL;
        }
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// Hapus belakang
void hapusBelakang()
{
    Node *hapus;
    Node *bantu;
    if (isEmpty() == false)
    {
        if (head != tail)
        {
            hapus = tail;
            bantu = head;
            while (bantu->next != tail)
            {
                bantu = bantu->next;
            }
            tail = bantu;
            tail->next = NULL;
            delete hapus;
        }
        else
        {
            head = tail = NULL;
        }
    }
}

```

```

    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}
// Hapus tengah
void hapusTengah(int posisi)
{
    Node *hapus, *bantu, *sebelum;
    if (posisi < 1 || posisi > hitungList())
    {
        cout << "Posisi di luar jangkauan" << endl;
    }
    else if (posisi == 1)
    {
        cout << "Posisi bukan posisi tengah" << endl;
    }
    else
    {
        int nomor = 1;
        bantu = head;
        while (nomor <= posisi)
        {
            if (nomor == posisi - 1)
            {
                sebelum = bantu;
            }
            if (nomor == posisi)
            {
                hapus = bantu;
            }
            bantu = bantu->next;
            nomor++;
        }
    }
}

```

```

        }
        sebelum->next = bantu;
        delete hapus;
    }
}

// ubah depan
void ubahDepan(int data)
{
    if (isEmpty() == 0)
    {
        head->data = data;
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// ubah tengah
void ubahTengah(int data, int posisi)
{
    Node *bantu;
    if (isEmpty() == 0)
    {
        if (posisi < 1 || posisi > hitungList())
        {
            cout << "Posisi di luar jangkauan" << endl;
        }
        else if (posisi == 1)
        {
            cout << "Posisi bukan posisi tengah" << endl;
        }
        else

```

```

        {
            int nomor = 1;
            bantu = head;
            while (nomor < posisi)
            {
                bantu = bantu->next;
                nomor++;
            }
            bantu->data = data;
        }
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// ubah belakang
void ubahBelakang(int data)
{
    if (isEmpty() == 0)
    {
        tail->data = data;
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// Hapus list
void clearList()
{
    Node *bantu, *hapus;

```

```

        bantu = head;
        while (bantu != NULL)
        {
            hapus = bantu;
            bantu = bantu->next;
            delete hapus;
        }
        head = tail = NULL;
        cout << "List berhasil terhapus!" << endl;
    }

    // Tampilkan list
    void tampilList()
    {
        Node *bantu;
        bantu = head;
        if (isEmpty() == false)
        {
            while (bantu != NULL)
            {
                cout << bantu->data << " ";
                bantu = bantu->next;
            }
            cout << endl;
        }
        else
        {
            cout << "Linked list masih kosong" << endl;
        }
    }

    int main()
    {
        init();
    }

```

```
        insertDepan(3);  
        tampilList();  
        insertBelakang(5);  
        tampilList();  
        insertDepan(2);  
        tampilList();  
        insertDepan(1);  
        tampilList();  
        hapusDepan();  
        tampilList();  
        hapusBelakang();  
        tampilList();  
        insertTengah(7, 2);  
        tampilList();  
        hapusTengah(2);  
        tampilList();  
        ubahDepan(1);  
        tampilList();  
        ubahBelakang(8);  
        tampilList();  
        ubahTengah(11, 2);  
        tampilList();  
  
        return 0;  
    }
```

Screenshoot program

```

PS D:\Project VS Code\C++\Semester 2\Praktikum Struktur Data\modul4\output> & .\guide1.exe'
3
3 5
2 3 5
1 2 3 5
2 3 5
2 3
2 7 3
2 3
1 3
1 8
1 11
PS D:\Project VS Code\C++\Semester 2\Praktikum Struktur Data\modul4\output>

```

Deskripsi program

Program diatas adalah implementasi dari single linked list non-circular. Program ini memberikan fungsi-fungsi penting untuk manipulasi linked list, seperti penambahan, penghapusan, dan pengubahan nilai node. Berikut penjelasan setiap fungsi dalam program tersebut:

- Node struct:** Struktur data untuk merepresentasikan sebuah node dalam linked list, terdiri dari dua bagian, yaitu data (int) dan pointer next (Node*) yang menunjuk ke node berikutnya.
- init():** Fungsi untuk menginisialisasi linked list dengan mengatur head dan tail menjadi NULL, menandakan linked list kosong.
- isEmpty():** Fungsi untuk memeriksa apakah linked list kosong. Mengembalikan nilai true jika head NULL dan false jika tidak.
- insertDepan(int nilai):** Fungsi untuk menambahkan node baru di depan linked list dengan nilai tertentu. Jika linked list kosong, node baru akan menjadi head dan tail. Jika tidak, node baru akan menjadi head dan next-nya akan menunjuk ke node sebelumnya head.
- insertBelakang(int nilai):** Fungsi untuk menambahkan node baru di belakang linked list dengan nilai tertentu. Jika linked list kosong, node baru akan menjadi head dan tail. Jika tidak, node baru akan

menjadi tail dan next dari node sebelumnya tail akan menunjuk ke node baru.

- f. `hitungList()`: Fungsi untuk menghitung jumlah node dalam linked list. Fungsi ini mengembalikan jumlah node dalam linked list.
- g. `insertTengah(int data, int posisi)`: Fungsi untuk menambahkan node baru di posisi tengah linked list dengan nilai tertentu. Fungsi ini memerlukan parameter posisi di mana node baru akan dimasukkan. Jika posisi di luar jangkauan, akan menampilkan pesan kesalahan. Jika posisi adalah 1, akan menampilkan pesan bahwa posisi bukan posisi tengah. Jika posisi valid, fungsi akan mencari node pada posisi sebelumnya dan menyisipkan node baru di antara node tersebut dengan menggunakan pointer next.
- h. `hapusDepan()`: Fungsi untuk menghapus node pertama (depan) dari linked list. Jika linked list tidak kosong, head akan diubah menjadi node kedua, jika ada, atau NULL jika tidak ada node lain.
- i. `hapusBelakang()`: Fungsi untuk menghapus node terakhir (belakang) dari linked list. Jika linked list tidak kosong, tail akan diubah menjadi node sebelumnya, jika ada, atau NULL jika tidak ada node lain.
- j. `hapusTengah(int posisi)`: Fungsi untuk menghapus node pada posisi tengah linked list. Fungsi ini bekerja mirip dengan `insertTengah()`, namun kali ini node yang dituju akan dihapus dari linked list.
- k. `ubahDepan(int data)`: Fungsi untuk mengubah nilai data node pertama (depan) linked list.
- l. `ubahTengah(int data, int posisi)`: Fungsi untuk mengubah nilai data node pada posisi tengah linked list. Fungsi ini bekerja mirip dengan `insertTengah()`, namun kali ini nilai data node yang dituju akan diubah.

- m. `ubahBelakang(int data)`: Fungsi untuk mengubah nilai data node terakhir (belakang) linked list.
- n. `clearList()`: Fungsi untuk menghapus semua node dari linked list, sehingga linked list menjadi kosong.
- o. `tampilList()`: Fungsi untuk menampilkan semua nilai data dari setiap node dalam linked list.
- p. `main()`: Fungsi utama program yang melakukan pengujian fungsi-fungsi linked list yang telah didefinisikan di atas. Program ini membuat linked list baru, menambahkan, menghapus, dan mengubah node, serta menampilkan isi linked list setiap kali ada perubahan.

2. Guided 2

Source code

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

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

Node *head, *tail, *baru, *bantu, *hapus;

void init() {
    head = NULL;
    tail = head;
}

int isEmpty() {
    if (head == NULL)
        return 1;
    else
        return 0;
}

void buatNode(string data) {
    baru = new Node;
    baru->data = data;
    baru->next = NULL;
}

int hitungList() {
    bantu = head;
    int jumlah = 0;
    while (bantu != NULL) {
```

```

        jumlah++;
        bantu = bantu->next;
    }
    return jumlah;
}

void insertDepan(string data) {
    buatNode(data);
    if (isEmpty() == 1) {
        head = baru;
        tail = head;
        baru->next = head;
    } else {
        while (tail->next != head) {
            tail = tail->next;
        }
        baru->next = head;
        head = baru;
        tail->next = head;
    }
}

void insertBelakang(string data) {
    buatNode(data);
    if (isEmpty() == 1) {
        head = baru;
        tail = head;
        baru->next = head;
    } else {
        while (tail->next != head) {
            tail = tail->next;
        }
        tail->next = baru;
        baru->next = head;
    }
}

```

```

    }
}

void insertTengah(string data, int posisi) {
    if (isEmpty() == 1) {
        head = baru;
        tail = head;
        baru->next = head;
    } else {
        baru = new Node;
        baru->data = data;
        int nomor = 1;
        bantu = head;
        while (nomor < posisi - 1 && bantu->next != head) {
            bantu = bantu->next;
            nomor++;
        }
        if (posisi == 1) {
            insertDepan(data);
        } else if (posisi <= hitungList()) {
            baru->next = bantu->next;
            bantu->next = baru;
        } else {
            cout << "Posisi diluar jangkauan" << endl;
        }
    }
}

void hapusDepan() {
    if (isEmpty() == 0) {
        hapus = head;
        tail = head;
        if (hapus->next == head) {
            head = NULL;

```

```

        tail = NULL;
        delete hapus;
    } else {
        while (tail->next != hapus) {
            tail = tail->next;
        }
        head = head->next;
        tail->next = head;
    }
} else {
    cout << "List masih kosong!" << endl;
}
}

void hapusBelakang() {
    if (isEmpty() == 0) {
        hapus = head;
        tail = head;
        if (hapus->next == head) {
            head = NULL;
            tail = NULL;
            delete hapus;
        } else {
            while (hapus->next != head) {
                hapus = hapus->next;
            }
            while (tail->next != hapus) {
                tail = tail->next;
            }
            tail->next = head;
        }
    } else {
        cout << "List masih kosong!" << endl;
    }
}

```

```

    }
}

void hapusTengah(int posisi) {
    if (isEmpty() == 0) {
        int nomor = 1;
        bantu = head;
        while (nomor < posisi - 1 && bantu->next != head) {
            bantu = bantu->next;
            nomor++;
        }
        if (posisi == 1) {
            hapusDepan();
        } else if (posisi <= hitungList()) {
            hapus = bantu->next;
            bantu->next = hapus->next;
            delete hapus;
        } else {
            cout << "Posisi diluar jangkauan" << endl;
        }
    } else {
        cout << "List masih kosong!" << endl;
    }
}

void clearList() {
    if (head != NULL) {
        hapus = head->next;
        while (hapus != head) {
            bantu = hapus->next;
            delete hapus;
            hapus = bantu;
        }
        delete head;
    }
}

```

```

        head = NULL;
    }
    cout << "List berhasil terhapus!" << endl;
}

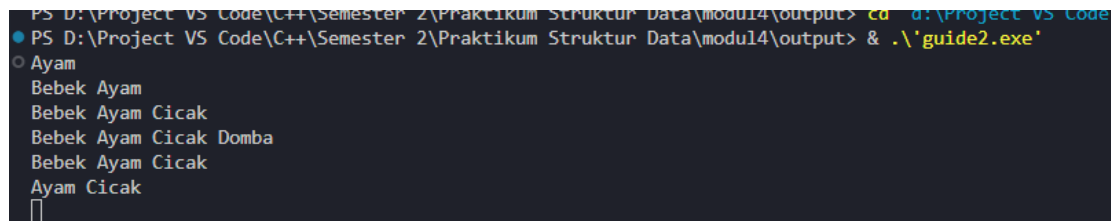
void tampil() {
    if (isEmpty() == 0) {
        tail = head;
        do {
            cout << tail->data << " ";
            tail = tail->next;
        } while (tail != head);
        cout << endl;
    } else {
        cout << "List masih kosong!" << endl;
    }
}

int main() {
    init();
    insertDepan("Ayam");
    tampil();
    insertDepan("Bebek");
    tampil();
    insertBelakang("Cicak");
    tampil();
    insertBelakang("Domba");
    tampil();
    hapusBelakang();
    tampil();
    hapusDepan();
    tampil();
    insertTengah("Sapi", 2);
    tampil();
}

```

```
hapusTengah(2);  
tampil();  
return 0;  
}
```

Screenshoot program



```
PS D:\Project VS Code\C++\Semester 2\Praktikum Struktur Data\modul4\output> cd ..\Project VS Code  
PS D:\Project VS Code\C++\Semester 2\Praktikum Struktur Data\modul4\output> & .\'guide2.exe'  
Ayam  
Bebek Ayam  
Bebek Ayam Cicak  
Bebek Ayam Cicak Domba  
Bebek Ayam Cicak  
Ayam Cicak  
█
```

Deskripsi program

Program di atas adalah contoh implementasi circular linked list dalam bahasa pemrograman C++. Circular linked list adalah struktur data yang terdiri dari sejumlah node, di mana setiap node memiliki dua bagian utama: data dan pointer yang menunjuk ke node berikutnya dalam list. Pada circular linked list, pointer dari node terakhir akan menunjuk kembali ke node pertama, membentuk lingkaran atau struktur "circular". Program ini menggunakan beberapa fungsi untuk mengelola circular linked list, seperti menyisipkan node di depan, di belakang, di tengah, menghapus node, dan menampilkan isi dari list tersebut.

LATIHAN KELAS - UNGUIDED

1. Unguided 1

Source code

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

struct Mahasiswa {
    string nama;
    string nim;
    Mahasiswa* next;
};

Mahasiswa* head = nullptr;

void tambahDepan() {
    Mahasiswa* baru = new Mahasiswa();
    cout << "Masukkan Nama : ";
    cin >> baru->nama;
    cout << "Masukkan NIM : ";
    cin >> baru->nim;
    baru->next = head;
    head = baru;
    cout << "\nData telah ditambahkan\n\n";
}

void tambahBelakang() {
    Mahasiswa* baru = new Mahasiswa();
    cout << "Masukkan Nama : ";
    cin >> baru->nama;
    cout << "Masukkan NIM : ";
    cin >> baru->nim;
```

```

baru->next = nullptr;
if (head == nullptr) {
    head = baru;
} else {
    Mahasiswa* temp = head;
    while (temp->next != nullptr) {
        temp = temp->next;
    }
    temp->next = baru;
}
cout << "\nData telah ditambahkan\n\n";
}

void tambahTengah() {
    Mahasiswa* baru = new Mahasiswa();
    cout << "Masukkan Nama : ";
    cin >> baru->nama;
    cout << "Masukkan NIM : ";
    cin >> baru->nim;
    cout << "Masukkan Posisi : ";
    int pos;
    cin >> pos;
    if (pos == 1) {
        baru->next = head;
        head = baru;
    } else {
        Mahasiswa* temp = head;
        for (int i = 1; i < pos - 1; i++) {
            if (temp->next != nullptr) {
                temp = temp->next;
            } else {
                cout << "Posisi tidak valid\n";
                return;
            }
        }
    }
}

```

```

    }
    baru->next = temp->next;
    temp->next = baru;
}
cout << "\nData telah ditambahkan\n\n";
}

void ubahDepan() {
    if (head == nullptr) {
        cout << "Linked List kosong\n";
    } else {
        cout << "Masukkan nama baru : ";
        string namaBaru;
        cin >> namaBaru;
        cout << "Masukkan NIM baru : ";
        string nimBaru;
        cin >> nimBaru;
        head->nama = namaBaru;
        head->nim = nimBaru;
        cout << "Data berhasil diubah\n";
    }
}

void ubahBelakang() {
    if (head == nullptr) {
        cout << "Linked List kosong\n";
    } else if (head->next == nullptr) {
        cout << "Masukkan nama baru : ";
        string namaBaru;
        cin >> namaBaru;
        cout << "Masukkan NIM baru : ";
        string nimBaru;
        cin >> nimBaru;
        head->nama = namaBaru;
    }
}

```

```

        head->nim = nimBaru;
        cout << "Data berhasil diubah\n";
    } else {
        Mahasiswa* temp = head;
        while (temp->next != nullptr) {
            temp = temp->next;
        }
        cout << "Masukkan nama baru : ";
        string namaBaru;
        cin >> namaBaru;
        cout << "Masukkan NIM baru : ";
        string nimBaru;
        cin >> nimBaru;
        temp->nama = namaBaru;
        temp->nim = nimBaru;
        cout << "Data berhasil diubah\n";
    }
}

void ubahTengah() {
    if (head == nullptr) {
        cout << "Linked List kosong\n";
    } else {
        cout << "Masukkan nama mahasiswa yang ingin diubah : ";
        string namaCari;
        cin >> namaCari;
        Mahasiswa* temp = head;
        bool ditemukan = false;
        while (temp != nullptr) {
            if (temp->nama == namaCari) {
                ditemukan = true;
                break;
            }
            temp = temp->next;
        }
    }
}

```

```

    }
    if (ditemukan) {
        cout << "Masukkan nama baru : ";
        string namaBaru;
        cin >> namaBaru;
        cout << "Masukkan NIM baru : ";
        string nimBaru;
        cin >> nimBaru;
        temp->nama = namaBaru;
        temp->nim = nimBaru;
        cout << "Data berhasil diubah\n";
    } else {
        cout << "Nama mahasiswa tidak ditemukan\n";
    }
}

void hapusDepan() {
    if (head == nullptr) {
        cout << "Linked List kosong\n";
    } else {
        Mahasiswa* temp = head;
        head = head->next;
        delete temp;
        cout << "Data berhasil dihapus\n";
    }
}

void hapusBelakang() {
    if (head == nullptr) {
        cout << "Linked List kosong\n";
    } else if (head->next == nullptr) {
        delete head;
        head = nullptr;
    }
}

```

```

    } else {
        Mahasiswa* temp = head;
        while (temp->next->next != nullptr) {
            temp = temp->next;
        }
        delete temp->next;
        temp->next = nullptr;
    }
    cout << "Data berhasil dihapus\n";
}

void hapusTengah() {
    if (head == nullptr) {
        cout << "Linked List kosong\n";
    } else {
        cout << "Masukkan nama mahasiswa yang ingin dihapus : ";
        string namaCari;
        cin >> namaCari;
        Mahasiswa* temp = head;
        Mahasiswa* prev = nullptr;
        bool ditemukan = false;
        while (temp != nullptr) {
            if (temp->nama == namaCari) {
                ditemukan = true;
                break;
            }
            prev = temp;
            temp = temp->next;
        }
        if (ditemukan) {
            if (prev == nullptr) {
                // Hapus data di depan
                head = temp->next;
                delete temp;
            }
        }
    }
}

```

```

        } else {
            prev->next = temp->next;
            delete temp;
        }
        cout << "Data berhasil dihapus\n";
    } else {
        cout << "Nama mahasiswa tidak ditemukan\n";
    }
}

}

void hapusList() {
    Mahasiswa* temp = head;
    while (temp != nullptr) {
        Mahasiswa* hapus = temp;
        temp = temp->next;
        delete hapus;
    }
    head = nullptr;
    cout << "Linked List berhasil dihapus\n";
}

void tampilkan() {
    if (head == nullptr) {
        cout << "Linked List kosong\n";
    } else {
        cout << "DATA MAHASISWA\n";
        cout << "NAMA\t\tNIM\n";
        Mahasiswa* temp = head;
        while (temp != nullptr) {
            cout << temp->nama << "\t\t" << temp->nim << endl;
            temp = temp->next;
            cout << endl;
        }
    }
}

```

```

    }
}

int main() {
    int choice;
    do {
        cout << "PROGRAM SINGLE LINKED LIST NON-CIRCULAR\n\n";
        cout << "1. Tambah Depan\n";
        cout << "2. Tambah Belakang\n";
        cout << "3. Tambah Tengah\n";
        cout << "4. Ubah Depan\n";
        cout << "5. Ubah Belakang\n";
        cout << "6. Ubah Tengah\n";
        cout << "7. Hapus Depan\n";
        cout << "8. Hapus Belakang\n";
        cout << "9. Hapus Tengah\n";
        cout << "10. Tampilkan\n";
        cout << "11. Keluar\n";
        cout << "Pilih Operasi : ";
        cin >> choice;
        switch (choice) {
            case 1:
                tambahDepan();
                break;
            case 2:
                tambahBelakang();
                break;
            case 3:
                tambahTengah();
                break;
            case 4:
                ubahDepan();
                break;
            case 5:

```



```
        ubahBelakang();
        break;
    case 6:
        ubahTengah();
        break;
    case 7:
        hapusDepan();
        break;
    case 8:
        hapusBelakang();
        break;
    case 9:
        hapusTengah();
        break;
    case 10:
        tampilkan();
        break;
    case 0:
        cout << "Keluar dari program\n";
        break;
    default:
        cout << "Pilihan tidak valid\n";
        break;
    }
} while (choice != 0);

return 0;
}
```

Screenshoot program

PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar

Pilih Operasi : 10

DATA MAHASISWA

NAMA	NIM
Jawad	23300001
Fadhil	2311102032
Farrel	23300003
Denis	23300005
Anis	23300008
Bowo	23300015
Gahar	23300040
Udin	23300048
Ucok	23300050
Budi	23300099

- a. Tambah data Wati(2330004) diantara Farrel dan Denis:

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

```
1. Tambah Depan  
2. Tambah Belakang  
3. Tambah Tengah  
4. Ubah Depan  
5. Ubah Belakang  
6. Ubah Tengah  
7. Hapus Depan  
8. Hapus Belakang  
9. Hapus Tengah  
10. Tampilkan  
11. Keluar  
Pilih Operasi : 3  
Masukkan Nama : Wati  
Masukkan NIM : 2330004  
Masukkan Posisi : 4
```

```
Data telah ditambahkan
```

PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar

Pilih Operasi : 10

DATA MAHASISWA

NAMA	NIM
Jawad	23300001
Fadhil	2311102032
Farrel	23300003
Wati	23300004
Denis	23300005
Anis	23300008
Bowo	23300015
Gahar	23300040
Udin	23300048
Ucok	23300050
Budi	23300099

b. Hapus Data Denis

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 9
Masukkan nama mahasiswa yang ingin dihapus : Denis
Data berhasil dihapus
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 10
DATA MAHASISWA
NAMA          NIM
Jawad          23300001

Fadhil         2311102032

Farrel         23300003

Wati           23300004

Anis           23300008

Bowo           23300015

Gahar          23300040

Udin           23300048

Ucok           23300050

Budi           23300099
```

c. Tambahkan data Owi(2330000) di awal

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar

Pilih Operasi : 1

Masukkan Nama : Owi

Masukkan NIM : 2330000

Data telah ditambahkan

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar

Pilih Operasi : 10

DATA MAHASISWA

NAMA	NIM
Owi	2330000
Jawad	23300001
Fadhil	2311102032
Farrel	23300003
Wati	23300004
Anis	23300008
Bowo	23300015
Gahar	23300040
Udin	23300048
Ucok	23300050
Budi	23300099

d. Tambahkan data David(23300100) di akhir

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

```
1. Tambah Depan  
2. Tambah Belakang  
3. Tambah Tengah  
4. Ubah Depan  
5. Ubah Belakang  
6. Ubah Tengah  
7. Hapus Depan  
8. Hapus Belakang  
9. Hapus Tengah  
10. Tampilkan  
11. Keluar  
Pilih Operasi : 2  
Masukkan Nama : David  
Masukkan NIM : 23300100
```

Data telah ditambahkan

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

```
1. Tambah Depan  
2. Tambah Belakang  
3. Tambah Tengah  
4. Ubah Depan  
5. Ubah Belakang  
6. Ubah Tengah  
7. Hapus Depan  
8. Hapus Belakang  
9. Hapus Tengah  
10. Tampilkan  
11. Keluar  
Pilih Operasi : 10  
DATA MAHASISWA  
NAMA      NIM  
Owi       23300000  
  
Jawad     23300001  
  
Fadhil    2311102032  
  
Farrel    23300003  
  
Wati      23300004  
  
Anis      23300008  
  
Bowo      23300015  
  
Gahar     23300040  
  
Udin      23300048  
  
Ucok      23300050  
  
Budi      23300099  
  
David     23300100
```

e. Ubah data Udin menjadi Idin(23300045)

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 6
Masukkan nama mahasiswa yang ingin diubah : Udin
Masukkan nama baru : Idin
Masukkan NIM baru : 23300045
Data berhasil diubah
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 10
DATA MAHASISWA
NAMA      NIM
Owi        23300000

Jawad      23300001

Fadhil     2311102032

Farrel     23300003

Wati       23300004

Anis       23300008

Bowo       23300015

Gahar      23300040

Idin       23300045

Ucok       23300050

Budi       23300099

David      23300100
```


- f. Ubah data terakhir menjadi Lucy(23300101)

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar

Pilih Operasi : 5

Masukkan nama baru : Lucy

Masukkan NIM baru : 23300101

Data berhasil diubah

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar

Pilih Operasi : 10

DATA MAHASISWA

NAMA	NIM
Owi	2330000
Jawad	23300001
Fadhil	2311102032
Farrel	23300003
Wati	2330004
Anis	23300008
Bowo	23300015
Gahar	23300040
Idin	23300045
Ucok	23300050
Budi	23300099
Lucy	23300101

g. Hapus data awal

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 7
Data berhasil dihapus
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 10
DATA MAHASISWA
NAMA      NIM
Jawad      23300001

Fadhil      2311102032

Farrel      23300003

Wati        23300004

Anis        23300008

Bowo        23300015

Gahar       23300040

Idin        23300045

Ucok        23300050

Budi        23300099

Lucy        23300101
```

h. Ubah data awal menjadi Bagus(23300002)

PROGRAM SINGLE LINKED LIST NON-CIRCULAR

```
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 4
Masukkan nama baru : Bagas
Masukkan NIM baru : 2330002
Data berhasil diubah
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
```

```
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 10
DATA MAHASISWA
NAMA      NIM
Bagas      2330002

Fadhil     2311102032

Farrel      23300003

Wati        23300004

Anis        23300008

Bowo        23300015

Gahar       23300040

Idin        23300045

Ucok        23300050

Budi        23300099

Lucy        23300101
```

i. Hapus data akhir

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 8
Data berhasil dihapus
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 10
DATA MAHASISWA
NAMA          NIM
Bagas          2330002

Fadhil         2311102032

Farrel         23300003

Wati           2330004

Anis           23300008

Bowo           23300015

Gahar          23300040

Idin           23300045

Ucok           23300050

Budi           23300099
```

j. Tampilkan seluruh data

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Tampilkan
11. Keluar
Pilih Operasi : 10
DATA MAHASISWA
NAMA          NIM
Bagas          2330002

Fadhil         2311102032

Farrel         23300003

Wati           2330004

Anis           23300008

Bowo           23300015

Gahar          23300040

Idin           23300045

Ucok           23300050

Budi           23300099
```

Deskripsi program

Program C++ di atas merupakan implementasi dari linked list non-circular untuk mengelola data mahasiswa berupa nama dan nim. Program ini memiliki fungsi untuk menambah, mengubah, dan menghapus data mahasiswa dalam linked list, serta menampilkan seluruh data mahasiswa.

Struktur data Mahasiswa digunakan untuk menyimpan informasi nama, NIM, dan pointer next yang menunjukkan ke node berikutnya dalam linked list. Variabel head menyimpan alamat node pertama dalam linked list.

Fungsi utama program ini antara lain:

- a. tambahDepan(): Menambahkan data mahasiswa baru ke awal linked list.
- b. tambahBelakang(): Menambahkan data mahasiswa baru ke akhir linked list.
- c. tambahTengah(): Menambahkan data mahasiswa baru di posisi tertentu dalam linked list.
- d. ubahDepan(): Mengubah data mahasiswa pertama dalam linked list.
- e. ubahBelakang(): Mengubah data mahasiswa terakhir dalam linked list.
- f. ubahTengah(): Mengubah data mahasiswa pada posisi tertentu dalam linked list.
- g. hapusDepan(): Menghapus data mahasiswa pertama dalam linked list.
- h. hapusBelakang(): Menghapus data mahasiswa terakhir dalam linked list.
- i. hapusTengah(): Menghapus data mahasiswa pada posisi tertentu dalam linked list.
- j. tampilkan(): Menampilkan seluruh data mahasiswa dalam linked list

Program ini menggunakan loop **do-while** untuk menampilkan menu operasi kepada pengguna dan terus berjalan hingga pengguna memilih untuk keluar dari program.

BAB IV

KESIMPULAN

Single Linked List Circular adalah struktur data yang terdiri dari sejumlah node yang saling terhubung. Setiap node memiliki dua bagian utama: data dan pointer yang menunjuk ke node berikutnya dalam linked list. Yang membedakan single linked list circular dengan yang non circular adalah pada single linked list circular, pointer next pada node terakhir menunjuk kembali ke node pertama, membentuk suatu lingkaran atau loop. Hal ini memungkinkan untuk melakukan traversal dari node pertama ke node terakhir dengan melintasi setiap node tepat satu kali.

Sementara itu, Single Linked List Non Circular juga memiliki struktur yang serupa dengan single linked list circular, namun pada node terakhir, pointer next-nya menunjuk ke NULL. Hal ini berarti tidak ada loop atau lingkaran dalam linked list, dan node terakhir berfungsi sebagai penanda akhir dari linked list tersebut. Saat melakukan traversal pada single linked list non circular, traversal akan berhenti saat mencapai node terakhir yang menunjuk ke NULL.

BAB V

DAFTAR PUSTAKA

1. Asisten Praktikum. (2024). Modul IV : Linked List Circular dan Non Circular
2. SlideServe. Rachmat, A. C., S.Kom. Single Linked List Circular. . Diakses pada 13 April 2024, dari <https://www.slideserve.com/nam/struktur-data-7-single-linked-list-circular>
3. SlideServe. Rachmat, A. C., S.Kom. Single Linked List Non Circular. . Diakses pada 13 April 2024, dari <https://www.slideserve.com/shear/struktur-data-6-single-linked-list-non-circular>