

**LAPORAN PRAKTIKUM STRUKTUR
LINKED LIST CIRCULAR DAN NON CIRCULAR**

**MODUL 4
LINKED LIST CIRCULAR DAN NON
CIRCULAR**



Disusun Oleh:

Muhamad ihsan

2311102077

Dosen

Wahyu Andi Saputra, S.pd., M,Eng

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMARIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

2024

Modul 4

A. Tujuan

1. Mahasiswa dapat memahami konsep Circular Linked List dan Non Circular.
2. Mahasiswa mampu mengimplementasikan Circular Linked List dan Non Circular dalam pemrograman.

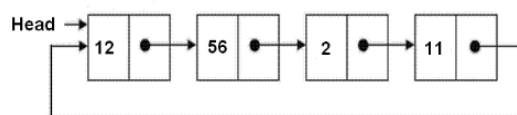
B. Dasar Teori

1. Linked List Circular

Linked list circular merupakan linked list yang tidak memiliki akhir karena node terakhir (tail) tidak bernilai 'NULL', tetapi terhubung dengan node pertama (head). Saat menggunakan linked list circular kita membutuhkan dummy node atau node pengecoh yang biasanya dinamakan dengan node current supaya program dapat berhenti menghitung data ketika node current mencapai node pertama (head). Linked list circular dapat digunakan untuk menyimpan data yang perlu diakses secara berulang, seperti daftar putar lagu, daftar pesan dalam antrian, atau penggunaan memori berulang dalam suatu aplikasi.

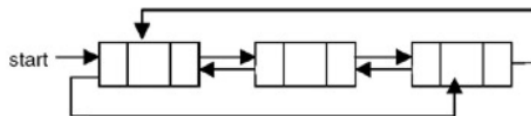
Ada 2 jenis Circular Linked List, yaitu :

1. Circular Single Linked List



Gambar . Ilustrasi Circular Single Linked List

2. Circular Double Linked List



Gambar . Ilustrasi Circular Double Linked List

2. Non Circular Linked List

Linked List merupakan suatu bentuk struktur data yang berisi Kumpulan data yang disebut sebagai node yang tersusun secara sekuensial, saling sambung menyambung, dan dinamis. Linked List sering juga disebut sebagai sebutan data berantai. Cara untuk menghubungkan satu node dengan node lainnya maka Linked List menggunakan pointer sebagai petunjuk node selanjutnya atau bisa disebut next.

Secara teoritis data pointer bersifat dinamis, sehingga suatu pointer dapat dijadikan sebuah list berkait atau Linked List

C. GUIDE Guided 1 : Linked List Non Circular

```
#include <iostream>

using namespace std;

/// PROGRAM SINGLE LINKED LIST NON-CIRCULAR

// Deklarasi Struct Node

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

Node *head;
Node *tail;

// Inisialisasi Node

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

// Pengecekan

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;
    tail->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;
        tail->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 diluar 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;
        }
        else
        {
            }
        }
    }

```

```

    }
    else
    {
        delete hapus;
        head = tail =
            NULL;
        cout << "List 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;
        }
        else
        {
        }
    }
    else

```

```

    {
        delete hapus;
        head = tail = NULL;
        cout << "List kosong!" << endl;
    }
}

// Hapus Tengah
void hapusTengah(int posisi)
{
    Node *bantu, *hapus, *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 << "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
        {
            bantu = head;
            int nomor = 1;
            while (nomor < posisi)
            {
                bantu = bantu->next;
                nomor++;
            }
            bantu->data = data;
        }
    }
    else
    {
        cout << "List masih kosong!" << endl;
    }
}

// Ubah Belakang
void ubahBelakang(int data)
{
    if (isEmpty() == 0)
    {
        tail->data = data;
    }

    else
    {
        cout << "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;
}

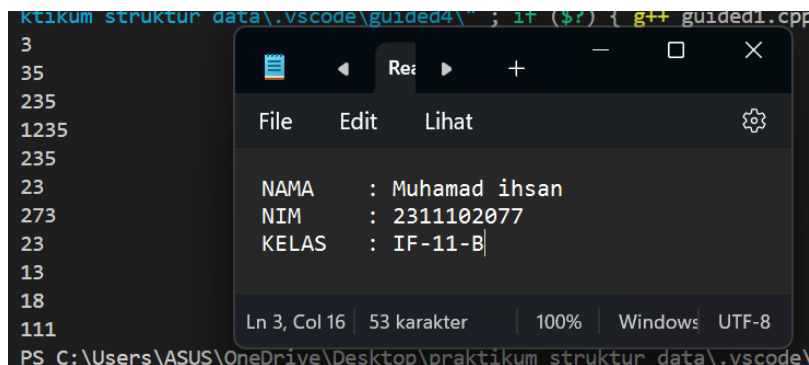
// Tampilkan List
void tampil()
{
    Node *bantu;
    bantu = head;
    if (isEmpty() == false)
    {
        while (bantu != NULL)
        {
            cout << bantu->data << ends;
            bantu = bantu->next;
        }

        cout << endl;
    }
    else

```

```
{  
    cout << "List masih kosong!" << endl;  
}  
}  
int main()  
{  
    init();  
    insertDepan(3);  
    tampil();  
    insertBelakang(5);  
    tampil();  
    insertDepan(2);  
    tampil();  
    insertDepan(1);  
    tampil();  
    hapusDepan();  
    tampil();  
    hapusBelakang();  
    tampil();  
    insertTengah(7, 2);  
    tampil();  
    hapusTengah(2);  
    tampil();  
    ubahDepan(1);  
    tampil();  
    ubahBelakang(8);  
    tampil();  
    ubahTengah(11, 2);  
    tampil();  
  
    return 0;  
}
```

Output:



```
3
35
235
1235
235
23
273
23
13
18
111
PS C:\Users\ASUS\OneDrive\Desktop\praktikum struktur data\.vscode\
```

File Edit Lihat

NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B

Ln 3, Col 16 | 53 karakter | 100% | Windows | UTF-8

Deskripsi:

program ini membuat data menggunakan Non Circular Linked List, program ini memiliki 14 Fungsi atau prosedur yang di setiap fungsi atau prosedurnya memiliki kegunaan masing-masing, Seperti yang digunakan di dalam main program ada init , void ini adalah untuk menginisialisasikan Node awal yang head dan tail nya masih bernilai NULL,Lalu ada insertDepan , fungsi ini kegunaanya untuk menambahkan node di awal data, insertBelakang kegunaannya untuk menambahkan data setelah node awal dibuat,hapusBelakang untuk menghapus node yang berada di belakang data. insertTengah menambahkan node di Tengah-tengah data yang sudah dibuat, hapusTengah untuk menghapus node yang berada di urutan Tengah atau setelah awal dan sebelum akhir,ubahDepan berguna untuk mengedit node yang sudah dibuat diawal data, ubahTengah untuk mengedit node yang telah dibuat di Tengah-tengah data atau setelah awal node sampai sebelum akhir node.ubahBelakang berfungsi untuk mengubah data atau node yang berada di urutan belakang.

Guided 2 :

```
#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() {
    return head == NULL;
}

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()) {
        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()) {
        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()) {
        head = baru;
        tail = head;
        baru->next = head;
    } else {
        baru->data = data;
        int nomor = 1;
        bantu = head;
        while (nomor < posisi - 1) {
            bantu = bantu->next;
            nomor++;
        }
        baru->next = bantu->next;
        bantu->next = baru;
    }
}

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

void hapusBelakang() {
    if (!isEmpty()) {
        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;
            hapus->next = NULL;
            delete hapus;
        }
    }
}

```

```

    } else {
        cout << "List masih kosong!" << endl;
    }
}

```

```

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

```

```

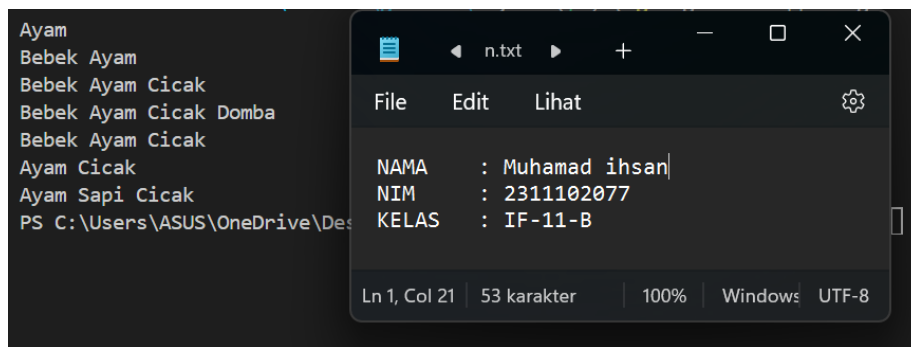
void clearList() {
    if (head != NULL) {
        hapus = head->next;
        while (hapus != head) {
            bantu = hapus->next;
            delete hapus;
            hapus = bantu;
        }
        delete head;
        head = NULL;
    }
}

```

```
void tampil() {
    if (!isEmpty()) {
        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;
}
```

Output:



```
Ayam
Bebek Ayam
Bebek Ayam Cicak
Bebek Ayam Cicak Domba
Bebek Ayam Cicak
Ayam Cicak
Ayam Sapi Cicak
PS C:\Users\ASUS\OneDrive\Desktop>
```

File Edit Lihat

NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B

Ln 1, Col 21 53 karakter 100% Windows UTF-8

Deskripsi:

program ini membuat menggunakan Circular Linked List, pada main program diawali dengan init, yang mana head nya masih NULL dan tailnya diisi variable head, lalu insertDepan dengan data yang diisi ayam menggunakan parameter, insertDepan lagi dengan parameter bebek, insertBelakang dengan nilai parameter cicak, insertBelakang lagi dengan parameter yang berisi nilai Domba, lalu menghapus data belakang dengan hapusBelakang yaitu data domba, kemudian hapus depan yaitu data ayam, menambah data di Tengah dengan value sapi dan di posisikan ke 2, menghapus data Tengah di posisi ke 2, dan yang terakhir menampilkan semua data yang telah di inialisasi sebelumnya menggunakan fungsi tampil.

D. UNGUIDED

Buatlah program menu Linked List Non Circular untuk menyimpan Nama dan NIM mahasiswa, dengan menggunakan input dari user.

SOURCE CODE

```
#include <iostream>

using namespace std;

/// PROGRAM SINGLE LINKED LIST NON-CIRCULAR

// Deklarasi Struct Node

struct Node {

    string nama;

    string nim;

    Node *next;

};

Node *head;

Node *tail;

// Inisialisasi Node

void init() {

    head = NULL;

    tail = NULL;
```

```

}

// Pengecekan
bool isEmpty()
{
    if (head == NULL)
        return true;
    else
        return false;
}

// Tambah Depan
void insertDepan(string nama, string nim)
{
    // Buat Node baru
    Node *baru = new Node;
    baru->nama = nama;
    baru->nim = nim;
    baru->next = NULL;
    if (isEmpty() == true)
    {
        head = tail = baru;
        tail->next = NULL;
    }
    else
    {
        baru->next = head;
        head = baru;
    }
}

// Tambah Belakang
void insertBelakang(string nama, string nim)
{
    // Buat Node baru

```

```
Node *baru = new Node;
baru->nama = nama;
baru->nim = nim;
baru->next = NULL;
if (isEmpty() == true)
{
    head = tail = baru;
    tail->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(string nama, string nim, int posisi)
{
    if (posisi < 1 || posisi > hitungList())
```

```

{
    cout << "Posisi diluar jangkauan" << endl;
}
else if (posisi == 1)
{
    cout << "Posisi bukan posisi tengah" << endl;
}
else
{
    Node *baru, *bantu;
    baru = new Node();
    baru->nama = nama;
    baru->nim = nim;
    // 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 << "List 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 << "List kosong!" << endl;
}
}

// Hapus Tengah
void hapusTengah(int posisi)
{
    Node *bantu, *hapus, *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(string nama, string nim)
{
    if (isEmpty() == 0)
    {
        head->nama = nama;
        head->nim = nim;
    }
    else
    {
        cout << "List masih kosong!" << endl;
    }
}

// Ubah Tengah
void ubahTengah(string nama, string nim, 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
    {
        bantu = head;
        int nomor = 1;
        while (nomor < posisi)
        {
            bantu = bantu->next;
            nomor++;
        }
        bantu->nama = nama;
        bantu->nim = nim;
    }
}

// Ubah Belakang
void ubahBelakang(string nama, string nim)
{

```

```
        if (isEmpty() == 0)
        {
            tail->nama = nama;
            tail->nim = nim;
        }
        else
        {
            cout << "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 tampil()
{
    Node *bantu;
    bantu = head;
    if (!isEmpty())
    {
        cout << endl << endl;
    }
}
```

```

        cout << "\t\t      Output Data" << endl;
        cout << "\t\t_____ " << endl << endl;
        cout << " Nama\t\t|   Nim\t\t|" << endl;
        while (bantu != NULL)
        {
            cout << bantu->nama << "\t\t|" << bantu->nim
<< "\t" << endl;
            bantu = bantu->next;
        }
    }
else
{
    cout << "List masih kosong!" << endl;
    return;
}
}

int main()
{

    string nama;
    string nim;
    int pilihan;
    int posisi;

    do
    {
        cout << endl;
        cout << "\t\tData Mahasiswa" << endl;
        cout << "1. Tambah Depan" << endl;
        cout << "2. Tambah Belakang" << endl;
        cout << "3. Tambah Tengah" << endl;
        cout << "4. Ubah Depan" << endl;
        cout << "5. Ubah Belakang" << endl;

```

```

        cout << "6. Ubah Tengah" << endl;
        cout << "7. Hapus Depan" << endl;
        cout << "8. Hapus Belakang" << endl;
        cout << "9. Hapus Tengah" << endl;
        cout << "10. Hapus List" << endl;
        cout << "11. Tampilkan" << endl;
        cout << "12. Exit" << endl;
        cout << "input (1-12): ";
        cin >> pilihan;
        switch (pilihan)
        {
        case 1:
        {
                cout << "manu  depan\n\n";
                cout << "Masukkan Nama :";
                cin.ignore();
                getline(cin, nama);
                cout << "Masukkan Nim: ";
                cin.ignore();
                cin >> nim;
                insertDepan(nama, nim);
                cout << "Data " << nama << " berhasil
diinput!";

                break;
        }
        case 2:
        {
                cout << "manu  belakang\n\n";
                cout << "Masukkan Nama :";
                cin.ignore();
                getline(cin, nama);
                cout << "Masukkan Nim: ";

```

```

        cin >> nim;
        insertBelakang(nama, nim);
        cout << "Data " << nama << " berhasil
diinput!";
        break;
    }
    case 3:
    {

        cout << "manu  tengah\n\n";
        cout << "Masukkan Nama :";
        cin.ignore();
        getline(cin, nama);
        cout << "Masukkan Nim: ";
        cin >> nim;
        cout << "Masukan Posisi: ";
        cin >> posisi;
        insertTengah(nama, nim, posisi);
        cout << "Data " << nama << " Berhasil
diinput!" << endl;
        break;
    }
    case 4:
    {

        cout << "manu ubah depan\n\n";
        cout << "Masukkan Nama :";
        cin.ignore();
        getline(cin, nama);
        cout << "Masukkan Nim: ";
        cin >> nim;
        ubahDepan(nama, nim);
        cout << "data depan berhasil diubah";
        break;
    }

```

```

    }

    case 5:
    {
        cout << "manu ubah belakang\n\n";
        cout << "Masukkan Nama :";
        cin.ignore();
        getline(cin, nama);
        cout << "Masukkan Nim: ";
        cin >> nim;
        ubahBelakang(nama, nim);
        cout << "data belakang berhasil diubah";
        break;
    }

    case 6:
    {
        cout << "manu ubah tengah\n\n";
        cout << "Masukkan Nama :";
        cin.ignore();
        getline(cin, nama);
        cout << "Masukkan Nim: ";
        cin >> nim;
        cout << "Masukkan Posisi: ";
        cin >> posisi;
        ubahTengah(nama, nim, posisi);
        cout << "data tengah berhasil diubah";
        break;
    }

    case 7:
    {
        cout << "manu hapus depan\n\n";
        hapusDepan();
        cout << "Data Depan berhasil terhapus!";
    }

```



```
        break;
    }
    case 8:
    {
        cout << "manu hapus belakang\n\n";
        hapusBelakang();
        cout << "Data Belakang berhasil terhapus!";
        break;
    }
    case 9:
    {
        cout << "manu hapus tengah\n\n";
        cout << "Masukkan Posisi: ";
        cin >> posisi;
        hapusTengah(posisi);
        cout << "Data Tengah berhasil terhapus!";
        break;
    }
    case 10:
    {
        clearList();
        break;
    }
    case 11:
    {
        tampil();
        break;
    }
    case 12:
    {
        cout << "Terima Kasih!" << endl;
    }
}
```

```

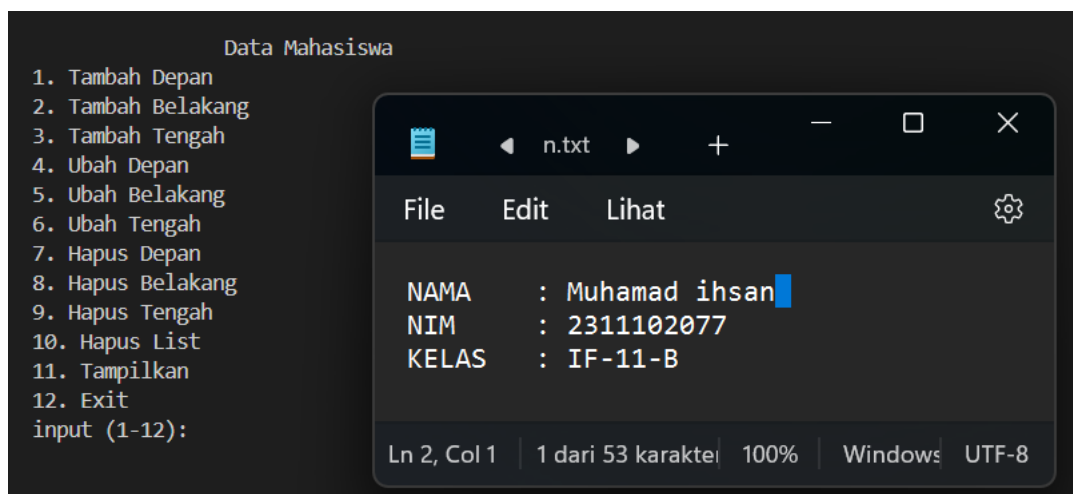
        default:
        {
            cout << "Pilihan tidak Valid!" << endl;
            break;
        }
    }
} while (pilihan != 12);

return 0;
}

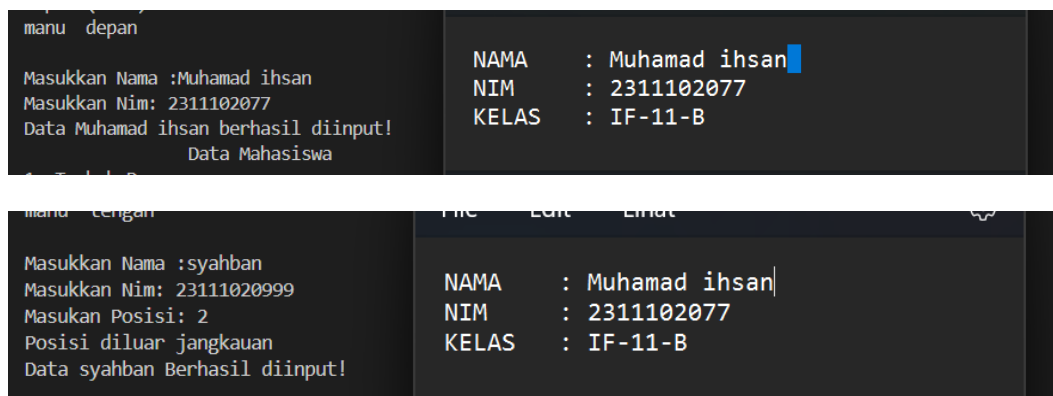
```

SCREENSHOT OUTPUT

- Tampilan menu



- Tampilan operasi tambahan



- Operasi hapus

```

hapus belakang
NAMA      : Muhamad ihsan
NIM       : 2311102077
KELAS    : IF-11-B
Belakang berhasil terhapus!
Data Mahasiswa

```

```

hapus tengah
Masukkan Posisi: 2
Posisi di luar jangkauan
Data Tengah berhasil terhapus!
Data Mahasiswa

```

- Tampilan operasi ubah

```

ubah tengah
Masukkan Nama :youlost
Masukkan Nim: 9999999
Masukkan Posisi: 2
List masih kosong!
Data tengah berhasil diubah

```

```

manu ubah belakang
Masukkan Nama :mentri
Masukkan Nim: 333333
List masih kosong!
data belakang berhasil diubah

```

- Operasi tampilan data

```

Output Data
-----
Nama      | Nim      |
jawa      | 23300001 |
ihsan     | 2311102077 |
farrel    | 23300003 |
denis     | 23300005 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
udin      | 23300048 |
ucok      | 23300050 |
budi      | 23300099 |

```

-Tambahkan data berikut diantara Farrel dan Denis:

```

Output Data
-----
Nama      | Nim      |
jawa      | 23300001 |
ihsan     | 2311102077 |
farrel    | 23300003 |
wati      | 23300004 |
denis     | 23300005 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
udin      | 23300048 |
ucok      | 23300050 |
budi      | 23300099 |

```

- Hapus data denis

```

Output Data
-----
Nama      | Nim      |
jawa      | 23300001 |
ihsan     | 2311102077 |
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
udin      | 23300048 |
ucok      | 23300050 |
budi      | 23300099 |

File Edit Lihat
NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B
Ln 1, Col 21 | 53 karakter | 100%

```

- Tambah data awal

```

Output Data
-----
Nama      | Nim      |
owi       | 23300000 |
jawa      | 23300001 |
ihsan     | 2311102077 |
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
udin      | 23300048 |
ucok      | 23300050 |
budi      | 23300099 |

File Edit Lihat
NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B
Ln 1, Col 21 | 53 karakter | 100%

```

- Tambah data akhir

```

Output Data
-----
Nama      | Nim      |
owi       | 23300000 |
jawa      | 23300001 |
ihsan     | 2311102077 |
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
udin      | 23300048 |
ucok      | 23300050 |
budi      | 23300099 |
david     | 23300100 |

File Edit Lihat
NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B
Ln 1, Col 21 | 53 karakter | 100%

```

- Ubah data udin

```

Output Data
-----
Nama      | Nim      |
owi       | 23300000 |
jawa      | 23300001 |
ihsan     | 2311102077 |
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
idin      | 23300045 |
ucok      | 23300050 |
budi      | 23300099 |
david     | 23300100 |

File Edit Lihat
NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B
Ln 1, Col 21 | 53 karakter | 100%

```

- Ubah data terakhir menjadi lucky

```

Output Data
-----
Nama      | Nim      |
owi       | 2330000  |
jawad     | 23300001 |
ihсан     | 2311102077
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
idin      | 23300045 |
ucok      | 23300050 |
budi      | 23300099 |
lucy      | 23300101 |

```

n.txt
File Edit Lihat
NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B
Ln 1, Col 21 | 53 karakter | 100%

- Hapus data awal

```

Output Data
-----
Nama      | Nim      |
jawad     | 23300001 |
ihسان     | 2311102077
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
idin      | 23300045 |
ucok      | 23300050 |
budi      | 23300099 |
lucy      | 23300101 |

```

n.txt
File Edit Lihat
NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B
Ln 1, Col 21 | 53 karakter | 100%

Data Mahasiswa

- Ubah data awal menjadi bagas

```

Output Data
-----
Nama      | Nim      |
bagas     | 23300002 |
ihسان     | 2311102077
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
idin      | 23300045 |
ucok      | 23300050 |
budi      | 23300099 |
lucy      | 23300101 |

```

n.txt
File Edit Lihat
NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B
Ln 1, Col 21 | 53 karakter | 100%

- Hapus data akhir

```
Output Data
-----
Nama      | Nim      |
bagas     | 2330002  |
ihсан     | 2311102077 |
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
idin      | 23300045 |
ucok      | 23300050 |
budi      | 23300099 |

Data Mahasiswa
```

n.txt

File Edit Lihat

NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B

Ln 1, Col 21 | 53 karakter | 100%

- Tampilkan seluruh data

```
Output Data
-----
Nama      | Nim      |
bagas     | 2330002  |
ihسان     | 2311102077 |
farrel    | 23300003 |
wati      | 23300004 |
anis      | 23300008 |
bowo      | 23300015 |
gahar     | 23300040 |
idin      | 23300045 |
ucok      | 23300050 |
budi      | 23300099 |

Data Mahasiswa
```

n.txt

File Edit Lihat

NAMA : Muhamad ihsan
NIM : 2311102077
KELAS : IF-11-B

Ln 1, Col 21 | 53 karakter | 100%

DESKRIPSI:

Program ini membuat program menu Linked List Non Circular untuk menyimpan Nama dan NIM mahasiswa, dengan menggunakan input dari user.

E. KESIMPULAN

Kesimpulan nya linked list circular tidak memiliki akhir karena node terakhir (tail) tidak bernilai 'NULL' dan untuk non circular selalu bernilai 'NULL'.

F. REFERENSI

<https://repository.unikom.ac.id/32762/1/Bab%20VII%20-%20Circular%20Linked%20List.pdf>
AKSU, Mustafa, and Ali KARCI. "Skip ring/circular skip list: circular linked list based new data structure." *structure* 6.5 (2015).
REPOSITORY UNIVERSITAS DIAN NUSWANTORO (2014. 29 Juli) SINGLE LINKED LIST NON CIRCULAR. Diakses pada 24 September 2018, dari https://repository.dinus.ac.id/docs/ajar/Pertemuan_11_Single_Linked_List_Circular.pdf