

**LAPORAN  
STRUKTUR DATA DAN ALGORITMA**

**MODUL IV**



**DISUSUN OLEH :**

Nama: Ahmad Titana Nanda Pramudya  
Nim : (2311102042)

**Dosen**

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

**PROGRAM STUDI S1 TEKNIK INFORMATIKA  
FAKULTAS INFORMATIKA  
INSTITUT TEKNOLOGI TELKOM PURWOKERTO  
2024**

## A. Dasar Teori

### 1. Linked List Non Circular

Linked list non circular merupakan linked list dengan node pertama (head) dan node terakhir (tail) yang tidak saling terhubung. Pointer terakhir (tail) pada Linked List ini selalu bernilai 'NULL' sebagai pertanda data terakhir dalam list-nya. Linked list non circular dapat digambarkan sebagai berikut.

### OPERASI PADA LINKED LIST NON CIRCULAR

#### 1. Deklarasi Simpul (Node)

```
struct node
{
    int data;
    node *next;
};
```

#### 2. Membuat dan Menginisialisasi Pointer Head dan Tail

```
node *head, *tail;

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

## 2. Pengecekan Kondisi Linked List

```
bool isEmpty()
{
    if (head == NULL && tail ==
        NULL) {
        return true;
    }
    else
    {
        return false;
    }
}
```

## 4. Penambahan Simpul (Node)

```
void insertBelakang(string
dataUser) {
    if (isEmpty() == true)
    {
        node *baru = new node;
        baru->data = dataUser;
        head = baru;
        tail = baru;
        baru->next = NULL;
    }
    else
    {
        node *baru = new node;
        baru->data = dataUser;
        baru->next = NULL;
        tail->next = baru;
    }
}
```

```
    tail = baru;
}
};
```

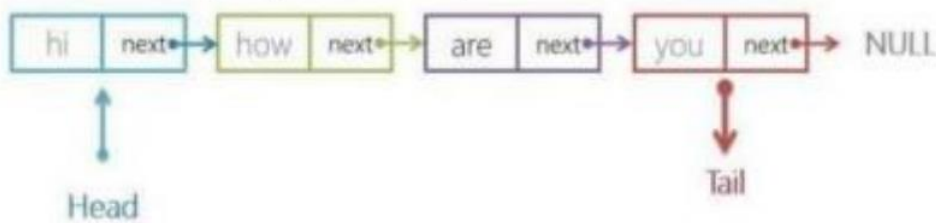
## 5. Penghapusan Simpul (Node)

```
void hapusDepan()
{
    if (isEmpty() == true)
    {
        cout << "List kosong!" <<
endl; }
    else
    {
        node *helper;
        helper = head;
        if (head == tail)
        {
            head = NULL;
            tail = NULL;
            delete helper;
        }
        else
        {
            head = head->next;
            helper->next = NULL;
            delete helper;
        }
    }
}
```

## 6. Tampil Data Linked List

```
void tampil()
{
    if (isEmpty() == true)
    {
        cout << "List kosong!" << endl;
    }
    else
    {
        node *helper;
        helper = head;
        while (helper != NULL)
        {
            cout << helper->data << ends;
            helper = helper->next;
        }
    }
}
```

digambarkan sebagai berikut.



## 2. 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. Linked list circular dapat.

## OPERASI PADA LINKED LIST CIRCULAR

### 1. Deklarasi Simpul (Node)

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

### 2. Membuat dan Menginisialisasi Pointer Head dan Tail

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

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

### 3. Pengecekan Kondisi Linked List

```
int isEmpty()
{
    if (head == NULL)
        return 1; // true
    else
        return 0; // false
}
```

### 4. Pembuatan Simpul (Node)

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

## 5. Penambahan Simpul (Node)

```
// Tambah Depan
void insertDepan(string
data) {
// Buat Node baru
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;
}
}
```

## 6. Penghapusan Simpul (Node)

```
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;
hapus->next = NULL;
delete hapus;
}
}
```

## 7. Menampilkan Data Linked List

```
void tampil()
{
if (isEmpty() == 0)
{
tail = head;
do
{
```



```
cout << tail->data << ends;
tail = tail->next;
} while (tail != head);
cout << endl;
}
}
```

## B. Guided

### Guided 1

Sourcode :

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

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

Node *head;
Node *tail;

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

bool isEmpty() {
    return head == NULL;
}

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

void insertBelakang(int nilai) {
```

```

    Node *baru = new Node;
    baru->data = nilai;
    baru->next = NULL;
    if (isEmpty()) {
        head = tail = baru;
    } else {
        tail->next = baru;
        tail = baru;
    }
}

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

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

void hapusDepan() {
    if (!isEmpty()) {
        Node *hapus = head;
        if (head->next != NULL) {
            head = head->next;
        } else {
            head = tail = NULL;
        }
        delete hapus;
    }
}

```

```

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

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

void hapusTengah(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 *bantu = head;
        Node *hapus;
        Node *sebelum = NULL;
        int nomor = 1;
        while (nomor < posisi) {
            sebelum = bantu;
            bantu = bantu->next;
            nomor++;
        }
        hapus = bantu;
        if (sebelum != NULL) {
            sebelum->next = bantu->next;
        } else {
            head = bantu->next;
        }
        delete hapus;
    }
}

```

```

void ubahDepan(int data) {
    if (!isEmpty()) {
        head->data = data;
    } else {
        cout << "List masih kosong!" << endl;
    }
}

void ubahTengah(int data, int posisi) {
    if (!isEmpty()) {
        if (posisi < 1 || posisi > hitungList()) {
            cout << "Posisi di luar jangkauan" << endl;
        } else if (posisi == 1) {
            cout << "Posisi bukan posisi tengah" << endl;
        } else {
            Node *bantu = head;
            int nomor = 1;
            while (nomor < posisi) {
                bantu = bantu->next;
                nomor++;
            }
            bantu->data = data;
        }
    } else {
        cout << "List masih kosong!" << endl;
    }
}

void ubahBelakang(int data) {
    if (!isEmpty()) {
        tail->data = data;
    } else {
        cout << "List masih kosong!" << endl;
    }
}

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

void tampil() {

```

```

Node *bantu = head;
if (!isEmpty()) {
    while (bantu != NULL) {
        cout << bantu->data << " ";
        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 :

```
g.exe=C:\msys64\usr\bin\gdb.exe --interpreter=mi
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:\contoh> & 'c:\Users\Eko Puji Susanto\.vscode\extensions
```

Deskripsi :

Program diatas merupakan program Double Linked List Non-Circular (DLLNC). Pada program awal Mendeklarasikan Struct dinamain Node berisi 3 field yaitu field data bertipe data integer, field next bertipe pointer dan field prev bertipe pointer. Mendeklarasikan pointer head, tail baru, bantu, bantu2, dan hapus ke struct node. Pada program ini memiliki 15 fungsi, untuk fungsi pertama yaitu fungsi init yaitu untuk memberikan nilai awal node pada list kosong dengan memberikan nilai NULL pada node head dan tail., selanjutnya isEmpty Operasi untuk memeriksa apakah suatu linked list masih kosong , selanjutnya fungsi create untuk membuat node baru, Fungsi selanjutnya yaitu insertdepan, insertbelakang dan insert tengah Operasi untuk menambahkan satu node ke dalam linked list didepan, dibelakang dan ditengah. Fungsi delete Operasi untuk menghapus node pada linked list. Yang berada di depan, belakang ataupun tengah, selanjutnya fungsi countlist untuk menghitung jumlah list, fungsi selanjutnya merubah list yang berada didepan, dibelakang dan ditengah. Fungsi selanjutnya yaitu clearlist untuk menghapus semua data list yang sudah diinputkan oleh user. Dan fungsi terakhir yaitu display untk menampilkan list yang sudah diinputkan. Selanjutnya program utama dan pengaplikasian program tersebut, berawal memanggil fungsi init dan seterusnya seperti di dalam program yang di atas.

## Guided 2

Sourcode :

```
#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;
    }
    cout << "List berhasil terhapus!" << endl;
}

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
```

```
File Edit View
Nama : Ahmad Titana Nanda | Pramudya
Nim : 2311102042
Kelas: IF-11-B
```

Deskripsi :

Program diatas sama halnya dengan linked list sebelumnya hanya saja program ini single linked list circular yang mana pointer tail nya ke head lagi tidak menuju ke NULL. Untuk program pertamanya struct node memiliki 2 field, field data bertipe string dan field next yang bertipe pointer dari node. Pada program ini memiliki 12 fungsi yaitu, yang pertama fungsi penginisialisasi untuk menyiapkan list ditetapkan bahwa head = NULL dan tail = head . Pengecekan nilai node menggunakan percabangan bertipe data integer jika nilai head sama dengan NULL maka true sebaliknya jika tidak maka false. Fungsi insert depan digunakan untuk menambahkan data baru ke dalam linked list dengan posisi node di depan sekaligus menjadi Node head. Fungsi insert belakang digunakan untuk menambahkan data baru ke dalam linked list dengan posisi di belakang yang otomatis akan menjadi Node Tail. Selanjutnya fungsi insert tengah sama halnya kaya fungsi sebelumnya yaitu menambahkan hanya beda dalam posisinya yaitu di tengah untuk proses nya harus menggunakan program hitunglist dan tranversing untuk mengetahui posisi node tersebut. Fungsi Hapus depan yaitu menghapus node yang berada di depan dengan posisi node menjadi head. Fungsi Hapus Belakang yaitu menghapus node tail dengan posisi di belakang. Fungsi menghapus tengah sama halnya dengan fungsi hapus depan dan belakang hanya saja ini diposisi di tengah . Fungsi ubah depan yaitu mengubah nilai node depan yang posisinya menjadi head. Fungsi ubah tengah yaitu mengubah nilai node tengah. Fungsi ubah belakang yaitu mengubah nilai node yang berada dibelakang yang posisinya menjadi tail. Fungsi hapus list yaitu menghapus semua linked list yang sudah diinputkan oleh user. Fungsi yang terakhir yaitu fungsi menampilkan list yaitu menampilkan seluruh data list yang sudah diinputkan oleh user. Program utama yaitu memanggil fungsi init mengenisialisasikan untuk menyiapkan list dengan head dan tail untuk pertama kali, pada tahap ini ditetapkan bahwa head = NULL dan tail = head. Memanggilkan fungsi insertdepan dengan data Ayam, fungsi tampil. Memanggil fungsi Insert Depan dengan data Bebek , fungsi tampil. Memanggil dengan data Cicak, fungsi tampil. Memanggil insert Belakang dengan data Domba. Memanggil fungsi tampil, penjelasan sementara dalam tampilannya yaitu menjadi Bebek Ayam Cicak Domba. Memanggil fungsi hapus belakang, fungsi tampil. Memanggil fungsi hapusdepan, fungsi tampil, dengan tampilannya yaitu menjadi Ayam Cicak. Memanggil fungsi inserttengah dengan data node barunya yaitu sapi dengan posisi di angka 2, fungsi tampil dengan tampilan Ayam Sapi Cicak. Memanggil fungsi hapustengah dengan posisi yang berada di angka 2, fungsi tampil dengan tampilan menjadi Ayam Cicak.

## C. Unguided

### Unguided 1

#### Sourcode

```
#include <iostream>
#include <iomanip>
using namespace std;
//Membuat struct dengan variabel mahasiswa terdapat 2 field
struct mahasiswa
{
    string nama;//Field nama berØpe data string
    string nim;//Field nim berØpe data integer
};
//Struct node terdapat 2 field
struct node
{
    mahasiswa ITTP;// field ITTP berØpe inisialisasi struct mahasiswa
    node *next;// Field next berØpe pointer node
};
//inisialisasi
node *head, *tail, *bantu, *hapus, *before, *baru;
//Menginisialisasikan nilai head & tail dengan nilai NULL
void init()
{
    head = NULL;
    tail = NULL;
}
//Pengecekan Nilai
bool isEmpty()
{
    if (head == NULL)
    {
        return true;
    }
    else
    {
        return false;
    }
}
mahasiswa Pendataan()
{
    mahasiswa ITTP;
    cout << "\nMasukkan Nama\t: ";
    cin.ignore();
    getline(cin, ITTP.nama);
    cout << "Masukkan NIM\t: ";
```

```

cin >> ITTP.nim;
return ITTP;
}
// Fungsi Tambah depan
void insertDepan(mahasiswa ITTP)
{
    node *baru = new node;
    baru->ITTP.nama = ITTP.nama;
    baru->ITTP.nim = ITTP.nim;
    baru->next = NULL;
    if (isEmpty() == true)
    {
        head = tail = baru;
        tail->next = NULL;
    }

    else
    {
        baru->next = head;
        head = baru;
    }
    cout << "Data " << ITTP.nama << " berhasil diinput!\n";

}
//Fungsi Tambah Belakang
void insertBelakang(mahasiswa ITTP)
{
    node *baru = new node;
    baru->ITTP.nama = ITTP.nama;
    baru->ITTP.nim = ITTP.nim;
    baru->next = NULL;
    if (isEmpty() == true)
    {
        head = tail = baru;
        tail->next = NULL;
    }
    else
    {
        tail->next = baru;
        tail = baru;
    }
}
//Hitung List
int hitungList()
{
    int penghitung = 0;

    node *bantu;
    bantu = head;

```

```

while (bantu != NULL)
{
    penghitung++;
    bantu = bantu->next;
}
return penghitung;
}
//Fungsi Tambah tengah
void insertTengah(mahasiswa iden0tas, int posisi)
{
    node *baru = new node;
    baru->ITTP.nama = iden0tas.nama;
    baru->ITTP.nim = iden0tas.nim;
    node *bantu;
    if (posisi < 1 || posisi > hitungList())
    {
        cout << "posisi diluar jangkauan";
    }
    else if (posisi == 1)
    {
        cout << "INi bukan posisi tengah\n";
    }
    else
    {
        bantu = head;
        int penghitung = 1;
        while (penghitung != posisi - 1)
        {
            penghitung++;
            bantu = bantu->next;
        }
        baru->next = bantu->next;
        bantu->next = baru;
    }
}
//Fungsi Ubah depan
void ubahDepan(mahasiswa data)
{
    string namaBefore = head->ITTP.nama;
    head->ITTP.nama = data.nama;
    head->ITTP.nim = data.nim;
    cout << "data " << namaBefore << " telah digan0 dengan data " << data.nama
    << endl;
}
//Fungsi Ubah Belakang
void ubahBelakang(mahasiswa data)
{
    string namaBefore = tail->ITTP.nama;

```

```

tail->ITTP.nama = data.nama;
tail->ITTP.nim = data.nim;
cout << "data " << namaBefore << " telah digan0 dengan data " << data.nama
<< endl;
}
//Fungsi Ubah Tengah
void ubahTengah(mahasiswa data)
{
int posisi;
cout << "\nMasukkan posisi data yang akan diubah : ";
cin >> posisi;

if (posisi < 1 || posisi > hitungList())
{
cout << "\nPosisi diluar jangkauan\n";
}
else if (posisi == 1)
{
cout << "\nBukan posisi tengah\n";
}
else
{
bantu = head;
int penghitung = 1;
while (penghitung != posisi)
{
penghitung++;
bantu = bantu->next;
}
bantu->ITTP.nama = data.nama;
bantu->ITTP.nim = data.nim;
}
}
//Fungsi Menampilkan List
void tampil()
{
node *bantu = head;
cout << "Nama "
<< " Nim\n";
while (bantu != NULL)
{
cout << bantu->ITTP.nama << " " << bantu->ITTP.nim << endl;
bantu = bantu->next;
}
}
//Fungsi Hapus Depan
void hapusDepan()

```

```

{
string dataBefore = head->ITTP.nama;
hapus = head;
if (head != tail)
{
head = head->next;
delete hapus;
}
else
{
head = tail = NULL;
}
cout << "Data " << dataBefore << " berhasil dihapus\n";
}
//Fungsi Hapus Belakang
void hapusBelakang()
{
string dataBefore = head->ITTP.nama;
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;
}
cout << "Data " << dataBefore << " berhasil dihapus\n";
}
//Fungsi Hapus Tengah
void hapusTengah()
{
tampil();
cout << endl;
if (isEmpty() == false)
{
back:
int posisi;
cout << "Masukkan Posisi yang dihapus : ";
cin >> posisi;
if (posisi < 1 || posisi > hitungList())

```



```

{
cout << "\nPosisi di luar jangkauan!\n";

cout << "Masukkan posisi baru\n";
goto back;

}
else if (posisi == 1 || posisi == hitungList())
{
cout << "\nBukan Posisi tengah\n";

cout << "Masukkan posisi baru\n";
goto back;
}
else
{
bantu = head;
int penghitung = 1;
while (penghitung <= posisi)
{
if (penghitung == posisi - 1)
{
before = bantu;
}
if (penghitung == posisi)
{
hapus = bantu;
}
bantu = bantu->next;
penghitung++;
}
string dataBefore = hapus->ITTP.nama;
before->next = bantu;
delete hapus;
cout << "\nData " << dataBefore << " berhasil dihapus!\n";

}
}
else
{
cout << "\n!!! List Data Kosong !!!\n";

}
}
//Fungsi Hapus List
void hapusList()
{

```

```

bantu = head;
while (bantu != NULL)
{
hapus = bantu;
delete hapus;
bantu = bantu->next;
}
init();
cout << "\nsemua data berhasil dihapus\n";
}
//Program Utama
int main()
{
init();
mahasiswa ITTP;
back:
int operasi, posisi;
cout << " PROGRAM SINGLE LINKED LIST NON-CIRCULAR" << endl;
cout << " =====\n\n" << 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 Teangah" << endl;
cout << "10.Hapus list" << endl;
cout << "11.Tampilkan" << endl;
cout << "0. Exit" << endl;

cout << "\nPilih Operasi :> ";
cin >> operasi;

switch (operasi)
{
case 1:
cout << "tambah depan\n";
insertDepan(Pendataan());
cout << endl;
goto back;
break;

case 2:
cout << "tambah belakang\n";
insertBelakang(Pendataan());
cout << endl;

```

```
goto back;

break;
case 3:
cout << "tambah tengah\n";
cout << "nama : ";
cin >> ITTP.nama;
cout << "NIM : ";
cin >> ITTP.nim;
cout << "Posisi: ";
cin >> posisi;
insertTengah(ITTP, posisi);
cout << endl;
goto back;
break;
case 4:
cout << "ubah depan\n";
ubahDepan(Pendataan());
cout << endl;
goto back;
break;
case 5:
cout << "ubah belakang\n";
ubahBelakang(Pendataan());
cout << endl;
goto back;
break;
case 6:
cout << "ubah tengah\n";
ubahTengah(Pendataan());
cout << endl;
goto back;

break;
case 7:
cout << "hapus depan\n";
hapusDepan();
cout << endl;
goto back;
break;
case 8:
cout << "hapus belakang\n";
hapusBelakang();
cout << endl;
goto back;
break;
case 9:
cout << "hapus tengah\n";
hapusTengah();
```

```
cout << endl;
goto back;
break;
case 10:
cout << "hapus list\n";
hapusList();
cout << endl;
goto back;
break;
case 11:
tampil();
cout << endl;
goto back;
break;

case 0:
cout << "\nEXIT PROGRAM\n";
break;

default:
cout << "\nSalah input operasi\n";
cout << endl;
goto back;
break;
}

return 0;
}
```

Output :

1. Buat menu menambahkan, mengubah, menghapus dan melihat nama dan nim

- Tampilan menu

```
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 Teengah
10. Hapus list
11. Tampilkan
0. Exit

Pilih Operasi :> 
```

- Tampilan operasi tambah

```
Pilih Operasi :> 1
tambah depan

Masukkan Nama : Ahmad
Masukkan NIM : 2311102042
Data Ahmad berhasil diinput!
```

```
Pilih Operasi :> 2
tambah belakang

Masukkan Nama : Arvan
Masukkan NIM : 2311102074
```

```
Pilih Operasi :> 3
tambah tengah
nama : Aji
NIM : 2311102064
Posisi: 2
```

```
Pilih Operasi :> 3
tambah tengah
nama : Ihsan
NIM : 2311102064
Posisi: 3

Nama : Ahmad Titana Nanda Pramudya
Nim : 2311102042
Kelas: IF-11-B
```

- Tampilkan hapus

```
Pilih Operasi :> 8
hapus belakang
Data Ahmad berhasil dihapus

Nama : Ahmad Titana Nanda Pramudya
Nim : 2311102042
Kelas: IF-11-B
```

```
Pilih Operasi :> 9
hapus tengah
Nama Nim
Ahmad 2311102042
Aji 2311102064
Ihsan 2311102064

Masukkan Posisi yang dihapus : 2
Data Aji berhasil dihapus!
```

File Edit View

Nama : Ahmad Titana Nanda Pramudya  
Nim : 2311102042  
Kelas: IF-11-B

Ln 6, Col 1 | 1 of 71 characters | 100% | Window UTF-8

- Ubah data belakang

```
Pilih Operasi :> 5
ubah belakang

Masukkan Nama : Haikal
Masukkan NIM : 2311102066
data Ihsan telah diganti dengan data Haikal

Nama : Ahmad Titana Nanda Pramudya
Nim : 2311102042
Kelas: IF-11-B
```

- Tampilkan operasi

```
Pilih Operasi :> 11
Nama Nim
Ahmad 2311102042
Haikal 2311102066

File Edit View

Nama : Ahmad Titana Nanda Pramudya
Nim : 2311102042
Kelas: IF-11-B
```

2. Masukkan Data pada menu baru sesuai urutan, lalu tampilkan data yang telah di masukan menggunakan insert.

- Tampilan tambah depan

Pilih Operasi :> 1 tambah depan	File Edit View
Masukkan Nama : Jawad	Nama : Ahmad Titana Nanda Pramudya
Masukkan NIM : 23300001	Nim : 2311102042
Data Jawad berhasil diinput!	Kelas: IF-11-B

- Tampilan tambah belakang

Pilih Operasi :> 2 tambah belakang	Nama : Ahmad Titana Nanda Pramudya
Masukkan Nama : Budi	Nim : 2311102042
Masukkan NIM : 23300099	Kelas: IF-11-B

- Tampilan tambah tengah

Pilih Operasi :> 3 tambah tengah nama : Ahmad NIM : 2311102042 Posisi: 2	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
--	--

Pilih Operasi :> 3 tambah tengah nama : Farrel NIM : 23300003 Posisi: 3	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
---	--

Pilih Operasi :> 3 tambah tengah nama : Denis NIM : 23300005 Posisi: 4	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
--	--

Pilih Operasi :> 3 tambah tengah nama : Anis NIM : 23300008 Posisi: 5	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
---	--

Pilih Operasi :> 3 tambah tengah nama : Bowo NIM : 233300015 Posisi: 6	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
--	--

Pilih Operasi :> 3 tambah tengah nama : Gahar NIM : 23300040 Posisi: 7	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
--	--

Pilih Operasi :> 3 tambah tengah nama : Udin NIM : 23300048 Posisi: 8	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
---	--

Pilih Operasi :> 3 tambah tengah nama : Ucok NIM : 23300050 Posisi: 9	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
---	--

- Tampilan operasi

Pilih Operasi :> 11 Nama Nim Jawad 23300001 Ahmad 2311102042 Farrel 23300003 Denis 23300005 Anis 23300008 Bowo 23300015 Gahar 23300040 Udin 23300048 Ucok 23300050 Budi 23300099	<div> <div>Nama Ah</div> <div>File Edit View</div> <div> Nama : Ahmad Titana Nanda Pramudya  Nim : 2311102042  Kelas: IF-11-B </div> <div>Ln 3, Col 15   67 characters   100%   Window UTF-8</div> </div>
---	---

### 3. Lakukan Perintah berikut :

- Tambahkan data berikut diantara farel dan denis :

Wati 23300004

Pilih Operasi :> 3 tambah tengah nama : Wati NIM : 23300004 Posisi: 4	<div> <div></div> <div>File Edit View</div> <div> Nama : Ahmad Titana Nanda Pramudya  Nim : 2311102042  Kelas: IF-11-B </div> </div>
---	--

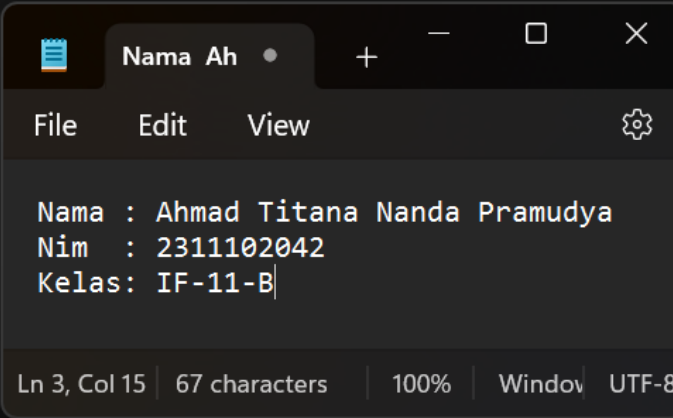


b. Hapus data denis

```
Pilih Operasi :> 9
hapus tengah
Nama Nim
Jawad 23300001
Ahmad 2311102042
Farrel 23300003
Wati 23300004
Denis 23300005
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099

Masukkan Posisi yang dihapus : 5

Data Denis berhasil dihapus!
```

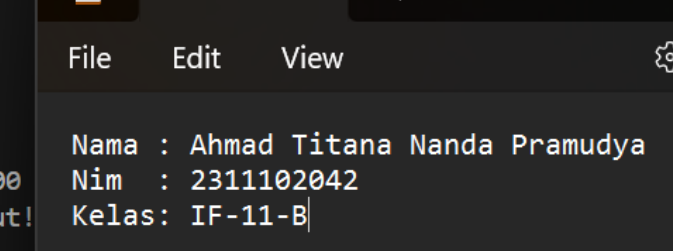


c. Tambahkan data awal :

Owi 23300000

```
Pilih Operasi :> 1
tambah depan

Masukkan Nama : Owi
Masukkan NIM : 23300000
Data Owi berhasil diinput!
```

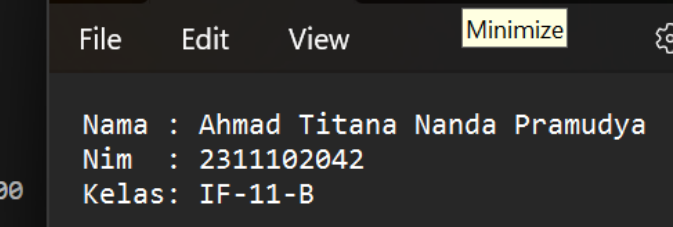


d. Tambahkan data akhir :

David 23300100

```
Pilih Operasi :> 2
tambah belakang

Masukkan Nama : David
Masukkan NIM : 23300100
```



- e. Ubah data udin menjadi idin 23300045 :

Pilih Operasi :> 6 ubah tengah  Masukkan Nama : Idin Masukkan NIM : 23300045  Masukkan posisi data yang akan diubah : 9	File Edit View Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B Ln 3, Col 15 67 characters 100% Window UTF-8
---	--

- f. Ubah data terakhir menjadi Lucy 23300002:

Pilih Operasi :> 5 ubah belakang  Masukkan Nama : Lucy Masukkan NIM : 23300101 data David telah diganθ dengan data Lucy	File Edit View Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B Ln 3, Col 15 67 characters 100% Window UTF-8
--	--

- g. Hapus data awal :

Pilih Operasi :> 7 hapus depan Data Owi berhasil dihapus	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
--	--

- h. Ubah data awal menjadi :

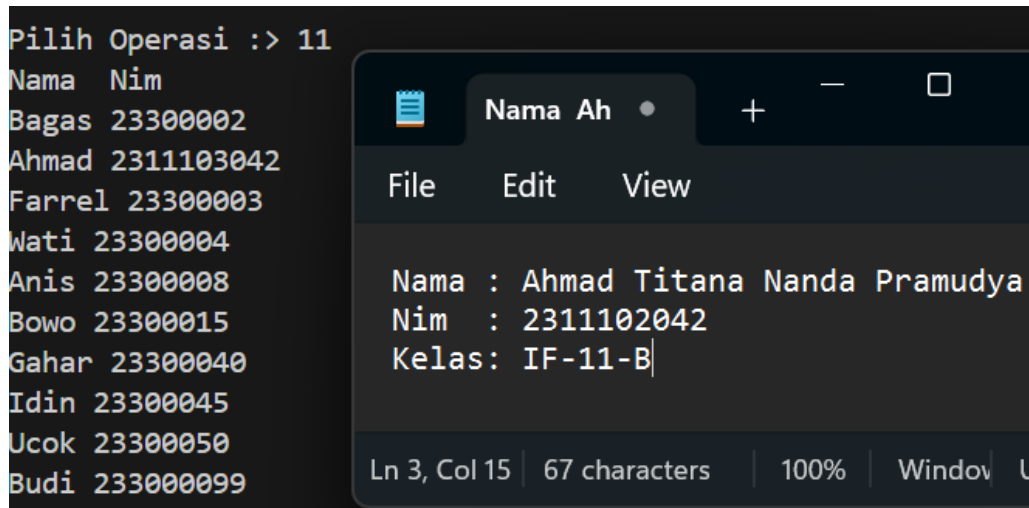
Bagas 23300002

Pilih Operasi :> 4 ubah depan  Masukkan Nama : Bagas Masukkan NIM : 23300002 data Jawad telah diganθ dengan data Bagas	File Edit View Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
---	--

- i. Hapus data akhir :

Pilih Operasi :> 8 hapus belakang Data Bagas berhasil dihapus	Nama : Ahmad Titana Nanda Pramudya Nim : 2311102042 Kelas: IF-11-B
---	--

j. Tampilkan seluruh data :



```
Pilih Operasi :> 11
Nama Nim
Bagas 23300002
Ahmad 2311103042
Farrel 23300003
Wati 23300004
Anis 23300008
Bowo 23300015
Gahar 23300040
Idin 23300045
Ucok 23300050
Budi 233000099
```

Nama Ah

File Edit View

Nama : Ahmad Titana Nanda Pramudya  
Nim : 2311102042  
Kelas: IF-11-B

Ln 3, Col 15 | 67 characters | 100% | Window

Deskripsi :

Program diatas merupakan program linked list yang digunakan untuk menampilkan program dengan linked list bertipe single linked list non – circular. Pada awal program membuat struct dinamain Mahasiswa berisi dua field, yaitu field nama bertipe data string dan field nim yang bertipe data integer. Membuat struct dengan variabel node memiliki 2 field yaitu field ITTP bertipe inisialisasi struct mahasiswa dan field next bertipe pointer node. Pada program ini memiliki 12 fungsi yaitu, yang pertama fungsi penginisialisasi untuk menyiapkan list ditetapkan bahwa head = NULL dan tail = NULL. Pengecekan nilai node menggunakan percabangan bertipe data boolean jika nilai head sama dengan NULL maka true sebaliknya jika tidak maka false. Inisialisasi struct mahasiswa dengan variabel pendataan yang isinya program penginputan nama dan nim. Fungsi insert depan digunakan untuk menambahkan data baru ke dalam linked list dengan posisi node di depan sekaligus menjadi Node head. Fungsi insert belakang digunakan untuk menambahkan data baru ke dalam linked list dengan posisi di belakang yang otomatis akan menjadi Node Tail. Selanjutnya fungsi insert tengah sama halnya kaya fungsi sebelumnya yaitu menambahkan hanya beda dalam posisinya yaitu di tengah untuk proses nya harus menggunakan program hitunglist dan tranversing untuk mengetahui posisi node tersebut. Fungsi Hapus depan yaitu menghapus node yang berada di depan dengan posisi node menjadi head. Fungsi Hapus Belakang yaitu menghapus node tail dengan posisi di belakang. Fungsi menghapus tengah sama halnya dengan fungsi hapus depan dan belakang hanya saja ini diposisi di tengah . Fungsi ubah depan yaitu mengubah nilai node depan yang posisinya menjadi head. Fungsi ubah tengah yaitu mengubah nilai node tengah. Fungsi ubah belakang yaitu mengubah nilai node yang berada dibelakang yang posisinya menjadi tail. Fungsi hapus list yaitu menghapus semua linked list yang sudah diinputkan oleh user. Fungsi yang terakhir yaitu fungsi menampilkan list yaitu menampilkan seluruh data list yang sudah diinputkan oleh user. Pada fungsi utama terdapat pemanggilan fungsi inisialisasi yaitu untuk menyiapkan linked list. Selanjutnya terdapat menu yang terdiri dari 11 case menggunakan perulangan switch. Untuk pilihan :

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. Hapus list
- 11. Tampilkan
- 0. Exit

#### **D. Kesimpulan**

Linked list adalah suatu simpul (node) yang dikaitkan dengan simpul yang lain dalam suatu urutan tertentu. Suatu linked list dikatakan single linked list apabila hanya ada satu pointer yang menghubungkan setiap node (satu arah “next”). Perancangan Linked list mempunyai 2 yaitu single linked list dan double linked list. Beberapa operasi pada linked list bisa diterapkan, pada praktikum ini terdapat 12 pengoperasian dengan menggunakan fungsi yaitu penginisialisasian, menambahkan, mengubah, mengurangi dan menampilkan.

#### **E. Referensi**

Referensi Sekolah Ilmu Komputer (SICS) Binus University. (2017, 15 Maret). Doubly Linked List. Diakses pada 29 Maret 2024, dari

<https://socs.binus.ac.id/2017/03/15/doublylinked-list/>

Mikirin Kode. (n.d.). Single Linked List. Diakses pada 29 Maret 2024, dari <https://mikirinkode.com/single-linked-list/>