

**LAPORAN PRAKTIKUM  
STRUKTUR DATA DAN ALGORITMA  
MODUL 4  
“LINKED LIST CIRCULAR DAN NON CIRCULAR”**



**DISUSUN OLEH:  
TSAQIF KANZ AHMAD  
2311102075  
IF-11-B**

**DOSEN:**

**WAHYU ANDI SAPUTRA S.Pd., M.Eng.**

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

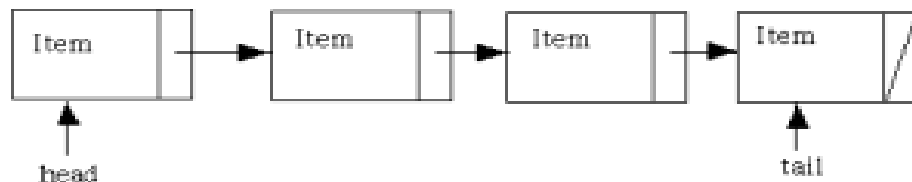
## **TUJUAN PRAKTIKUM**

1. Mengetahui dan memahami linked list circular dan non circular.
2. Membuat linked list circular dan non circular.
3. Mengaplikasikan linked list circular dan non circular pada program yang dibuat.

## 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 satu sama lain. Pointer terakhir (Tail) pada linked list selalu memiliki nilai NULL untuk menunjukan data terakhir dalam listnya. Kelebihan pada Linked List Non-Circular adalah implementasi lebih mudah karena tidak ada koneksi melingka dan membutuhkan sedikit memori karena pointer next pada node terakhir menunjuk ke NULL. Untuk kekurangannya yaitu pada Operasi akses acak (mencari node berdasarkan indeks) lebih lambat karena harus melalui traversal dari awal list dan tidak dapat digunakan untuk kasus dimana akses acak diperlukan secara efisien



Contoh Gambar Linked List Non-Circular

## 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;
};
```

### 3. 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)

```

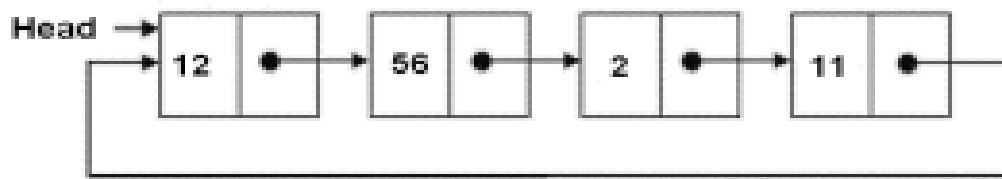
```

    {
        cout << helper->data << ends;
        helper = helper->next;
    }
}

```

## 2). LINKED LIST CIRCULAR

Linked list circular merupakan linked list yang tidak memiliki akhir karena node terakhir (tail) tidak mempunyai nilai 'NULL', namun terhubung dengan node pertama (head). Saat menggunakan linked list circular kita memerlukan dummy node atau node peniru yang biasa disebut dengan node current agar 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 kali, seperti playlist lagu, daftar antrian pesanan, atau penggunaan memori berulang dalam suatu aplikasi. Kekurangan pada Linked list circular yaitu perlu membutuhkan sedikit lebih banyak memori karena pointer next pada node terakhir menunjuk kembali ke node pertama. Linked list circular dapat digambarkan sebagai berikut.



Contoh Gambar Linked List Circular

## OPERASI 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
    buat Node(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;
    }
}
```

## PENJELASAN GUIDED

1). Program Linked List Non Circular.

### SOURCE CODE

```
#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() {
    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;
    }
}

// Tambah Depan
void insertBelakang(int nilai) {
    // Buat Node Baru
    Node *baru = new Node;
    baru->data = nilai;
    baru->next = NULL;
    if (isEmpty()) {
        head = tail = baru;
    } else {
        tail->next = baru;
        tail = baru;
    }
}
```



```

}

// Hitung jumlah List
int hitungList() {
    Node *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 = new Node();
        baru->data = data;

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

// Hapus depan
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;
    }
}

// Hapus belakang
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;
    }
}

// Hapus tengah
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;
    }
}

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

```

```

// Ubah tengah
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;
    }
}

```

```

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

```

```

// Hapus list
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;
}

```

```

// Tampilkan list
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;
}

```

## SCREENSHOT OUTPUT

The screenshot shows a terminal window with the following output:

```

PS C:\Users\ACER> cd "C:\Users\ACER\AppData\Local\Temp\" ; if ($?) { g++ tempCodeRunnerFile.cpp
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 C:\Users\ACER\AppData\Local\Temp>

```

Overlaid on the terminal is a code runner window titled "ppu" with a menu bar (File, Edit, View) and a settings icon. The window displays the following information:

```

Nama : TSAQIF KANZ AHMAD
Nim  : 2311102075
Kelas: IF-11-B

```

At the bottom of the code runner window, the status bar shows: "Ln 3, Col 15 | 57 characters | 100% | Window | UTF-8".

## **DESKRIPSI PROGRAM**

*Program diatas adalah program mengimplementasikan Linked List Non-Circular yang digunakan untuk menyimpan data integer. Program tersebut terdiri dari beberapa fungsi untuk mengelola list yaitu Fungsi utama(main) untuk melakukan beberapa operasi pada list untuk mendemostrasikan fungsionalitas program, Fungsi inisialisasi(int()) yang mengatur Head dan Tail menjadi NULL, Fungsi pemeriksaan(isEmpty()) untuk mengembalikan true jika list kosong, false sebaliknya, Fungsi Penyisipan(insert...(int nilai)) untuk menyisipkan elemen baru dengan nilai pada setiap posisi list, Fungsi Penghapusan untuk menghapus elemen pada setiap posisi list, Fungsi Pengubahan nilai untuk mengubah nilai elemen pada setiap posisi list, serta fungsi lainnya berupa 'hitungList()', 'clearList()' dan 'tampil()' untuk menghitung, menghapus serta menampilkan semua elemen pada list tersebut.*

## 2). Program Linked List Circular.

### 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() {
    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;
}

```

### SCREENSHOT OUTPUT :

The screenshot shows a terminal window with the following output:

```

PS C:\Users\ACER> cd "C:\Users\ACER\AppData\Local\Temp\" ; if ($?) { g++ tempCodeRunnerFile.cpp
Ayam
Bebek Ayam
Bebek Ayam Cicak
Bebek Ayam Cicak Domba
Bebek Ayam Cicak
Ayam Cicak
Ayam Sapi Cicak
PS C:\Users\ACER\AppData\Local\Temp>

```

Overlaid on the terminal is a Notepad++ window titled "Nama" with the following content:

```

Nama : TSAQIF KANZ AHMAD
Nim  : 2311102075
Kelas: IF-11-B

```

The Notepad++ status bar at the bottom indicates: Ln 3, Col 15 | 57 characters | 100% | Window | UTF-8.

## **DESKRIPSI PROGRAM**

*Program diatas adalah program implementasi Linked List Circular yang menyimpan data bertipe string. Terdapat Perubahan pada Struktur data dan perubahan Variabel. Pada perubahan struktur data terdapat Struct Node yang berubah tipe data menjadi string untuk menyimpan teks. Pada perubahan Variabel terdapat head,tail,baru,bantu,hapus yang digunakan untuk penunjuk node dalam list. Pada program tersebut terdapat berbagai fungsi untuk mengelola list seperti penyisipan, penghapusan, perubahan nilai dan pengecekan khususnya pada data string.*

## PENJELASAN UNGUIDED

Buatlah program menu Single Linked List Non-Circular untuk menyimpan Nama dan NIM mahasiswa dengan menggunakan inputan dari user. Lakukan operasi berikut.

1. Buatlah menu untuk menambahkan, mengubah, menghapus, dan melihat Nama dan NIM mahasiswa, berikut contoh tampilan output dari nomor 1:

\* 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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi :
```

- Tampilan Operasi Tambah :

```
-Tambah Depan

Masukkan Nama :
Masukkan NIM :

Data telah ditambahkan
```

```
-Tambah Tengah

Masukkan Nama :
Masukkan NIM :
Masukkan Posisi :

Data telah ditambahkan
```

```
-Hapus Belakang

Data (nama mahasiswa yang dihapus) berhasil dihapus
```

- Tampilan Operasi Ubah :

```
-Ubah Belakang

Masukkan nama :
Masukkan NIM :

Data (nama lama) telah diganti dengan data (nama baru)
```

```

-Ubah Belakang

Masukkan nama :
Masukkan NIM :
Masukkan posisi :

Data (nama lama) telah diganti dengan data (nama baru)

```

- Tampilan Operasi Data:

```

DATA MAHASISWA

NAMA NIM
Nama1 NIM1
Nama2 NIM2

```

**Buatlah tampilan output sebgus dan secantik mungkin sesuai kreatifitas anda masing-masing, jangan terpaku pada contoh output yang diberikan.**

2. Setelah membuat menu tersebut, masukan data sesuai urutan berikut, lalu tampilkan data yang telah dimasukan (Gunakan insert depan, belakang atau tengah).

Nama	NIM
Jawad	23300001
[Nama Anda]	[NIM Anda]
Farrel	23300003
Denis	23300005
Anis	23300008
Bowo	23300015
Gahar	23300040
Udin	23300048
Ucok	23300050
Budi	23300099

3. Lakukan perintah berikut:
  - a) Tambahkan data berikut diantara Farrel dan Denis:  
**Wati 23300004**
  - b) Hapus data Denis
  - c) Tambahkan data berikut di awal:  
**Owi 23300000**
  - d) Tambahkan data berikut di akhir:  
**David 23300100**
  - e) Ubah data Udin menjadi data berikut:  
**Idin 23300045**
  - f) Ubah data terkahir menjadi berikut:  
**Lucy 23300101**

- g) Hapus data awal
- h) Ubah data awal menjadi berikut:  
**Bagas 2330002**
- i) Hapus data akhir
- j) Tampilkan seluruh data

## SOURCE CODE

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

struct mahasiswa{
    string nama;
    string nim;
};

struct node{
    mahasiswa ITTP;
    node *next;
};

node *head, *tail, *bantu, *hapus, *before, *baru;

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

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;
}

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";
}

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;
    }
}

int hitungList(){
    int penghitung = 0;
    node *bantu;
    bantu = head;
    while (bantu != NULL){
        penghitung++;
        bantu = bantu->next;
    }
    return penghitung;
}

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;
    }
}

void ubahDepan(mahasiswa data){
    string namaBefore = head->ITTP.nama;
    head->ITTP.nama = data.nama;
    head->ITTP.nim = data.nim;
    cout << "data " << namaBefore << " telah diganθ dengan data " <<
data.nama << endl;
}

void ubahBelakang(mahasiswa data){
    string namaBefore = tail->ITTP.nama;
    tail->ITTP.nama = data.nama;
    tail->ITTP.nim = data.nim;
    cout << "data " << namaBefore << " telah diganθ dengan data " <<
data.nama << endl;
}

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;
    }
}

void tampil(){
    node *bantu = head;
    cout << "Nama " << " Nim\n";

```

```

while (bantu != NULL){
    cout << bantu->ITTP.nama << " " << bantu->ITTP.nim << endl;
    bantu = bantu->next;
}
}

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";
}

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";
}

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";
    }
}

void hapusList(){
    bantu = head;
    while (bantu != NULL){
        hapus = bantu;
        delete hapus;
        bantu = bantu->next;
    }
    init();
    cout << "\nsemua data berhasil dihapus\n";
}

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;
}

```

## SCREENSHOT OUTPUT

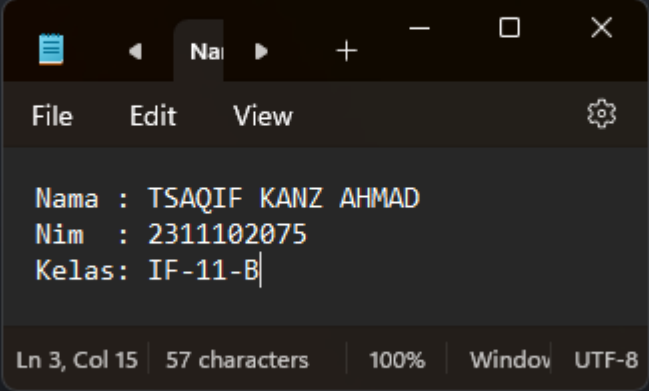
1. Buatlah menu untuk menambahkan, mengubah, menghapus, dan melihat Nama dan NIM mahasiswa.

- Tampilan Menu

```
PS C:\Users\ACER> cd "C:\Users\ACER\AppData\Local\Temp\" ; if ($?) { g++ tempCodeRunnerFile.cpp
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	File Edit View
Masukkan Nama : Tsaqif Masukkan NIM : 2311102075 Data Tsaqif berhasil diinput!	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B

Pilih Operasi :> 2 tambah belakang	File Edit View
Masukkan Nama : Rizky Masukkan NIM : 2311102061	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B

Pilih Operasi :> 3 tambah tengah nama : Naufal NIM : 2311102078 Posisi: 2	File Edit View
	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B

- Tampilan operasi mengubah

Pilih Operasi :> 6 ubah tengah	File Edit View
Masukkan Nama : Bintang Masukkan NIM : 2311102052 Masukkan posisi data yang akan diubah : 2	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B

- Tampilan Operasi Menghapus

Pilih Operasi :> 8 hapus belakang Data Tsaqif berhasil dihapus	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
----------------------------------------------------------------------	----------------------------------------------------------------

- Tampilan Operasi Data

Pilih Operasi :> 11 Nama Nim Tsaqif 2311102075 Bintang 2311102052	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
----------------------------------------------------------------------------	----------------------------------------------------------------

2) Masukan data pada menu baru sesuai urutan lalu tampilkan data tersebut menggunakan insert.

- Tampilan Depan

Pilih Operasi :> 1 tambah depan  Masukkan Nama : Jawad Masukkan NIM : 23300001 Data Jawad berhasil diinput!	File Edit View  Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
----------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------

- Tampilan Tengah

Pilih Operasi :> 3 tambah tengah nama : Tsaqif NIM : 2311102075 Posisi: 2	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
---------------------------------------------------------------------------------------	----------------------------------------------------------------

Pilih Operasi :> 3 tambah tengah nama : Farrel NIM : 23300003 Posisi: 3	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
-------------------------------------------------------------------------------------	----------------------------------------------------------------

Pilih Operasi :> 3 tambah tengah nama : Denis NIM : 23300005 Posisi: 4	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
------------------------------------------------------------------------------------	----------------------------------------------------------------

Pilih Operasi :> 3  
tambah tengah  
nama : Anis  
NIM : 23300008  
Posisi: 5

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

Pilih Operasi :> 3  
tambah tengah  
nama : Bowo  
NIM : 23300015  
Posisi: 6

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

Pilih Operasi :> 3  
tambah tengah  
nama : Gahar  
NIM : 23300040  
Posisi: 7

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

Pilih Operasi :> 3  
tambah tengah  
nama : Udin  
NIM : 23300048  
Posisi: 8

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

Pilih Operasi :> 3  
tambah tengah  
nama : Ukok  
NIM : 23300050  
Posisi: 9

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

- Tampilan Belakang

Pilih Operasi :> 2  
tambah belakang

Masukkan Nama : Budi  
Masukkan NIM : 23300099

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

- Tampilan Semua Data

```
Pilih Operasi :> 11
Nama Nim
Jawad 23300001
Tsaqif 2311102075
Farrel 23300003
Denis 23300005
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099
```

File Edit View

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

Ln 3, Col 15 | 57 characters | 100% | Window UTF-8

3. Lakukan perintah berikut :

- a). Tambahkan data berikut diantara Farrel dan Denis : **Wati 23300004**

```
Pilih Operasi :> 3
tambah tengah
nama : Wati
NIM : 2330004
Posisi: 4
```

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

- b). Hapus data Denis

```
Pilih Operasi :> 9
hapus tengah
Nama Nim
Jawad 23300001
Tsaqif 2311102075
Farrel 23300003
Wati 23300004
Denis 23300005
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099
```

File Edit View

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

Ln 3, Col 15 | 57 characters | 100% | Window UTF-8

```
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 : 2330000
Data Owi berhasil diinput!
```

Nama : TSAQIF KANZ AHMAD  
Nim : 2311102075  
Kelas: IF-11-B|

d). Tambahkan data berikut diakhir : **David 23300100**

Pilih Operasi :> 2 tambah belakang	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
Masukkan Nama : David Masukkan NIM : 23300100	

e). Ubah data data Udin menjadi : **Idin 23300045**

Pilih Operasi :> 6 ubah tengah	File Edit View Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
Masukkan Nama : Idin Masukkan NIM : 23300045 Masukkan posisi data yang akan diubah : 9	

f). Ubah data terakhir menjadi berikut : **Lucy 23300101**

Pilih Operasi :> 5 ubah belakang Masukkan Nama : Lucy Masukkan NIM : 23300101 data David telah digan f dengan data Lucy	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
-------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------

g). Hapus data Awal

Pilih Operasi :> 7 hapus depan Data Owi berhasil dihapus	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
----------------------------------------------------------------	----------------------------------------------------------------

h). Ubah data awal menjadi : **Bagas 23300002**

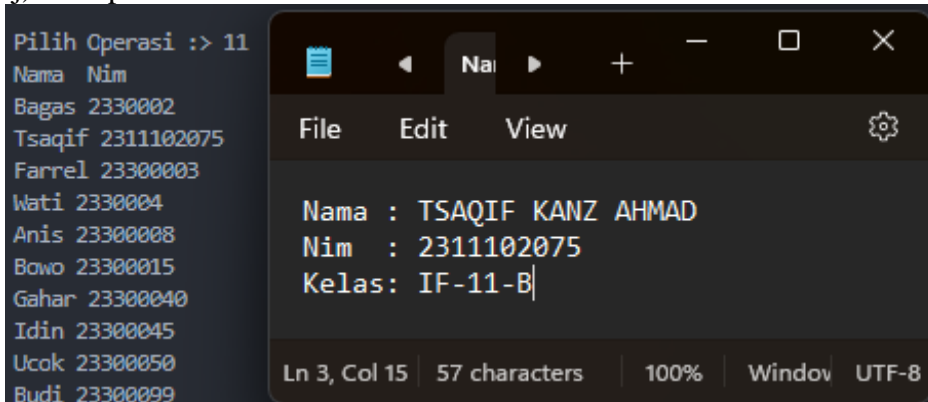
Pilih Operasi :> 4 ubah depan Masukkan Nama : Bagas Masukkan NIM : 23300002 data Jawad telah digan f dengan data Bagas	File Edit View Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------

i). Hapus data akhir

Pilih Operasi :> 8 hapus belakang Data Bagas berhasil dihapus	Nama : TSAQIF KANZ AHMAD Nim : 2311102075 Kelas: IF-11-B
---------------------------------------------------------------------	----------------------------------------------------------------



j). Tampilkan seluruh data



```
Pilih Operasi :> 11
Nama Nim
Bagas 2330002
Tsaqif 2311102075
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
Idin 23300045
Ucok 23300050
Budi 23300099

Nama : TSAQIF KANZ AHMAD
Nim : 2311102075
Kelas: IF-11-B

Ln 3, Col 15 | 57 characters | 100% | Window | UTF-8
```

## DESKRIPSI PROGRAM

Program diatas adalah program menu Single Linked List Non-Circular untuk menyimpan data mahasiswa yang terdiri dari nama dan NIM. Program ini digunakan dengan menu operasi yang jelas dengan menggunakan perulangan switch dan menyediakan berbagai fungsi untuk menambah, mengubah dan menghapus data mahasiswa dalam list tersebut. Fungsi tersebut terdapat 12 diantaranya yaitu Fungsi utama yaitu pemanggilan inisialisasi untuk menyiapkan linked list, Fungsi pengecekan, Fungsi tambah depan, belakang dan tengah, fungsi ubah depan, belakang dan tengah, Fungsi menampilkan list, Fungsi hapus depan, belakang dan tengah. Pada struktur data tersebut terdapat dua 'struct untuk menyimpan data nama dan nim yang bertipe string serta 'struct node untuk menyimpan data mahasiswa dan pointer ke node berikutnya. Pada variabel program tersebut terdapat berbagai tipe data yaitu head dan tail untuk menunjuk ke node pertama ke node terakhir dalam list, 'bantu digunakan untuk iterasi dalam list, 'hapus untuk menyimpan node yang akan dihapus, 'before untuk menyimpan node sebelum node akan dihapus dan 'baru untuk membuat node baru.

## KESIMPULAN

**Linked list** adalah cara yang efisien untuk menyimpan data yang saling terkait, dimana setiap elemen (Node) memiliki data dan pointer pada elemen berikutnya. Linked list circular memiliki pointer terakhir yang menunjuk kembali ke node pertama yang membentuk lingkaran. Kelebihannya adalah akses lebih cepat, namun implementasi lebih kompleks dan penggunaan memori sedikit lebih banyak. Pada Linked list Non-circular memiliki pointer terakhir yang menunjuk ke NULL, sehingga tidak ada koneksi melingkar. Kelebihannya adalah implementasi yang lebih mudah dan penggunaan memori yang lebih hemat, namun aksesnya acak untuk mencari node yang sesuai indeks sehingga aksesnya lebih lambat.

## **DAFTAR PUSTAKA**

[1] Asisten Praktikum, "Modul 4 Linked List Circular dan Non-Circular", Learning ManagementSystem, 2024.

[2] GeeksforGeeks – Circular Linked list : <https://www.geeksforgeeks.org/circular-linked-list/>

[3] Modul Kuliah Struktur data – Linked List – Universitas Esa Unggul :  
[https://lms-paralel.esaunggul.ac.id/pluginfile.php?file=%2F86227%2Fmod\\_resource%2Fcontent%2F1%2FModul%20Struktur%20Data-Linked%20List.pdf](https://lms-paralel.esaunggul.ac.id/pluginfile.php?file=%2F86227%2Fmod_resource%2Fcontent%2F1%2FModul%20Struktur%20Data-Linked%20List.pdf)