

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

MODUL 4

“LINKED LIST CIRCULAR DAN NON CIRCULAR”



Disusun Oleh :
NAMA : Ibnu Rizal Mutaqim
NIM : 2311102067

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;  
  
};
```

3) Pengecekan Kondisi Linked List

```
bool isEmpty()

{

if (head == NULL && tail ==

NULL) {

return true;

}

else

{

return false;

}

}
```

4) Penambahan Simpul

```
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) Menampilkan 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;

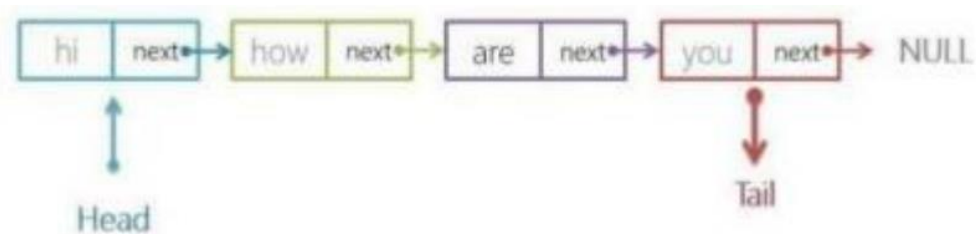
}

}

}

```

digambarkan sebagai berikut:



Gambar 1 *Single Linked List Non Circular*

2. 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: Linked List Non Circular

Source code:

```
#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;
}

```

Screenshots Output

```

PS D:\KULIAH\KULIAH SEMESTER 2\PRAKTIKUM STRUKTUR DATA DAN ALGORITMA\PRAKTIKUM WEEK 4> cd "
Guided1 } ; if ($?) { .\Guided1 }
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:\KULIAH\KULIAH SEMESTER 2\PRAKTIKUM STRUKTUR DATA DAN ALGORITMA\PRAKTIKUM WEEK 4>

```

Nama : Ibnu Rizal Mutaqimppp
 NIM : 2311102067
 Kelas : IF-11-B

Deskripsi:

Kode tersebut merupakan Linked List non-Circular. Struktur data ini terdiri dari simpul-simpul berurutan dengan setiap simpul(node) menyimpan nilai integer dan pointer ke simpul berikutnya. Fungsi-fungsi yang disediakan meliputi inisialisasi, pengecekan kekosongan, penambahan simpul di depan atau di belakang, penambahan simpul di tengah berdasarkan posisi, penghapusan simpul di depan, di belakang, atau di tengah, pengubahan nilai data simpul di depan, di belakang, atau di tengah, penghapusan seluruh isi Linked List, dan tampilan isi Linked List. Ini memungkinkan penyimpanan dan manipulasi data dengan fleksibilitas dalam program.

Guided 2: 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;
    }
}

```

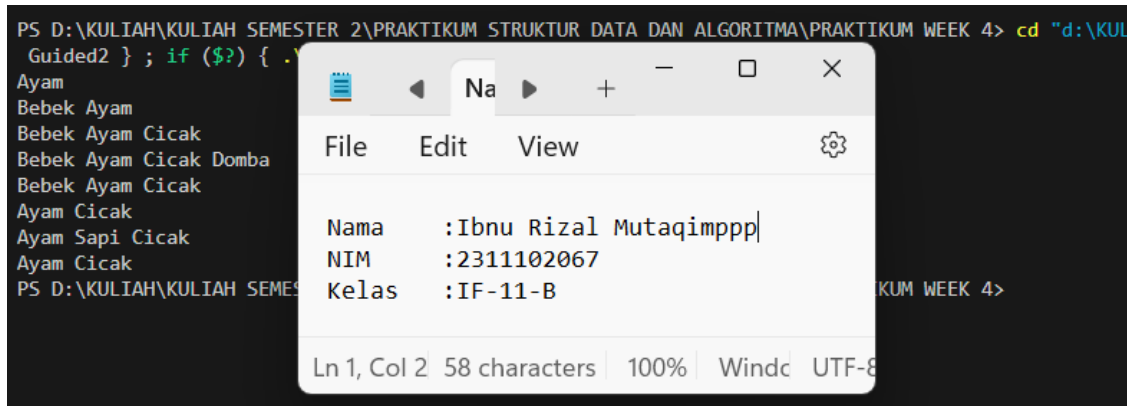
```

        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;
}

```


Screenshots Output



Deskripsi:

Kode tersebut merupakan Linked List Circular ,di mana setiap simpul(node) menyimpan data string dan memiliki pointer yang menunjuk ke simpul berikutnya dalam lingkaran. Fungsi-fungsi yang disediakan mencakup inisialisasi, pengecekan kekosongan, pembuatan simpul baru, penambahan di depan, di belakang, atau di tengah, penghapusan di depan, di belakang, atau di tengah, penghapusan seluruh isi, dan tampilan isi Linked List. Ini memungkinkan penggunaan struktur data fleksibel untuk penyimpanan dan manipulasi data dalam program.

Unguided

Unguided 1:

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

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 identitas, int posisi)
{
    node *baru = new node;
    baru->ITTP.nama = identitas.nama;
    baru->ITTP.nim = identitas.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 diganti 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 diganti 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;
}
```

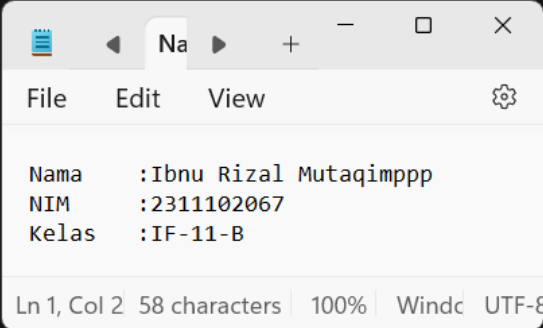
Screenshots Output

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

Tampilan Menu:

```
PS D:\KULIAH\KULIAH SEMESTER 2\PRAKTIKUM STRUKTUR DATA DAN ALGORITMA\PRAKTIKUM WEEK 4> cd
; if ($?) { g++ unguided1.cpp -o unguided1 } ; if ($?) { .\unguided1 }
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 :>
```



The screenshot shows a terminal window with a menu for a 'PROGRAM SINGLE LINKED LIST NON-CIRCULAR'. The menu options are: 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, and 0. Exit. A pop-up window titled 'Na' is overlaid on the terminal, displaying the following data: Nama : Ibnu Rizal Mutaqimppp, NIM : 2311102067, and Kelas : IF-11-B. The status bar of the pop-up window shows 'Ln 1, Col 2 58 characters', '100%', 'Windc', and 'UTF-8'.

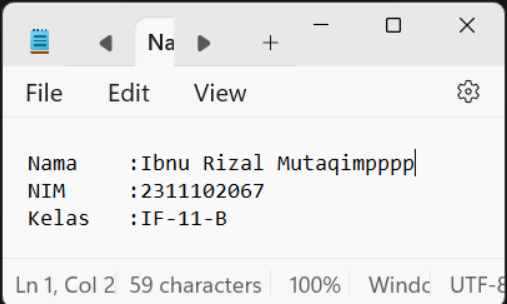
Tampilan Operasi Tambah:

- Depan

```
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 :> 1
tambah depan

Masukkan Nama : Jawad
Masukkan NIM : 23300001
Data Jawad berhasil diinput!
```



The screenshot shows the terminal window after selecting option 1. The prompt 'Pilih Operasi :>' is followed by '1' and the text 'tambah depan'. Below this, the prompts 'Masukkan Nama :', 'Masukkan NIM :', and 'Data Jawad berhasil diinput!' are shown. A pop-up window titled 'Na' is overlaid on the terminal, displaying the following data: Nama : Ibnu Rizal Mutaqimppp, NIM : 2311102067, and Kelas : IF-11-B. The status bar of the pop-up window shows 'Ln 1, Col 2 59 characters', '100%', 'Windc', and 'UTF-8'.

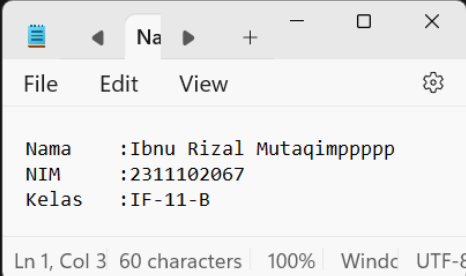
- Tengah

```

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 Teangah
10.Hapus list
11.Tampilkan
0. Exit

Pilih Operasi :> 3
tambah tengah
nama : Farrel
NIM : 23300003
Posisi: 3

```



Tampilan Opersai Hapus

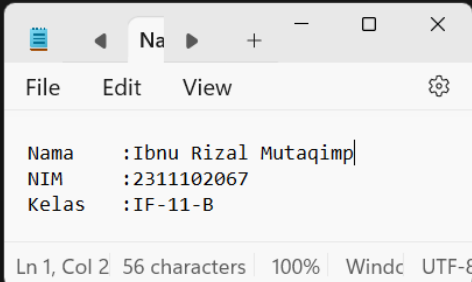
- Belakang

```

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 Teangah
10.Hapus list
11.Tampilkan
0. Exit

Pilih Operasi :> 8
hapus belakang
Data Jawad berhasil dihapus

```



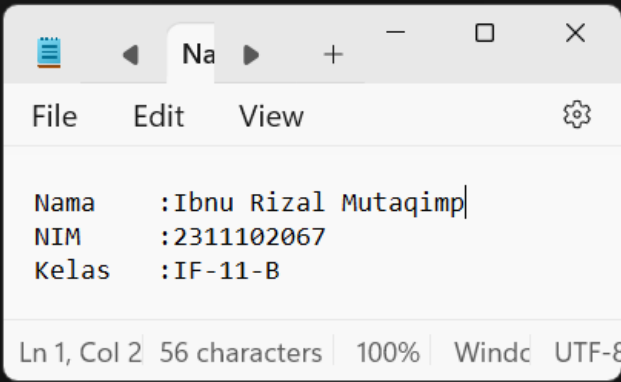
- Tengah

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

Pilih Operasi :> 9
hapus tengah
Nama Nim
Jawad 23300001
Ibnu 2311102067
Farrel 23300003

Masukkan Posisi yang dihapus : 2

Data Ibnu berhasil dihapus!
```



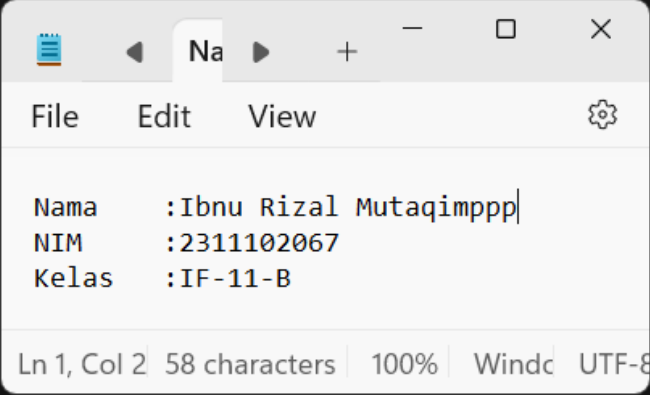
Tampilan Operasi Ubah

- Ubah Belakang


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

Pilih Operasi :> 5
ubah belakang

Masukkan Nama : Denis
Masukkan NIM : 23300005
data Farrel telah digan|f dengan data Denis
```



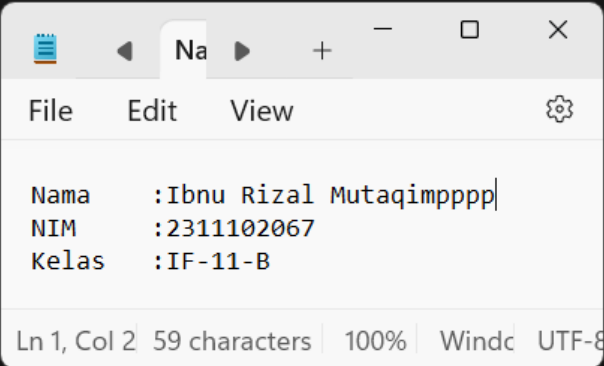
- Ubah Tengah

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

Pilih Operasi :> 6
ubah tengah

Masukkan Nama : Anis
Masukkan NIM : 23300008

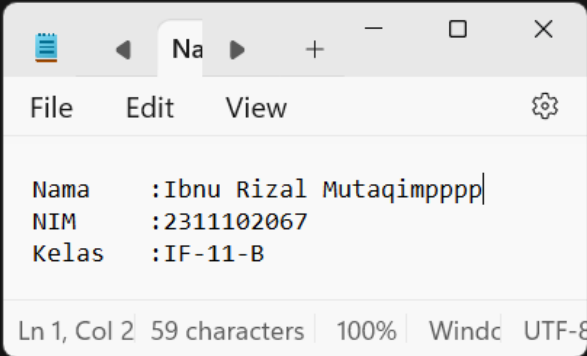
Masukkan posisi data yang akan diubah : 5
```



Tampilan Operasi Tampilkan Data

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teengah
10. Hapus list
11. Tampilkan
0. Exit

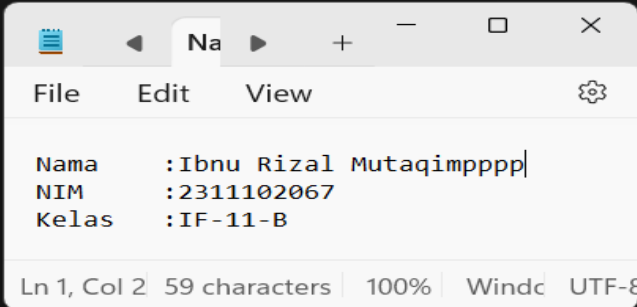
Pilih Operasi :> 11
Nama Nim
Jawad 23300001
Ibnu 2311102067
Farrel 23300003
Denis 23300005
Anis 23300008
Bowo 23300015
```



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

```
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 :> 11
Nama Nim
Jawad 23300001
Ibnu 2311102067
Farrel 23300003
Denis 23300005
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099
```



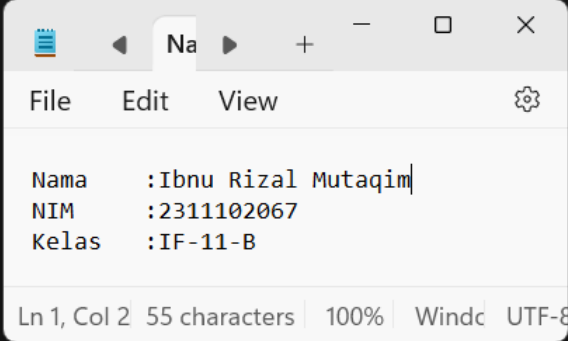
3. Lakukan perintah berikut:

a) Tambahkan data berikut diantara Farrel dan Denis:

Wati 2330004

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

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

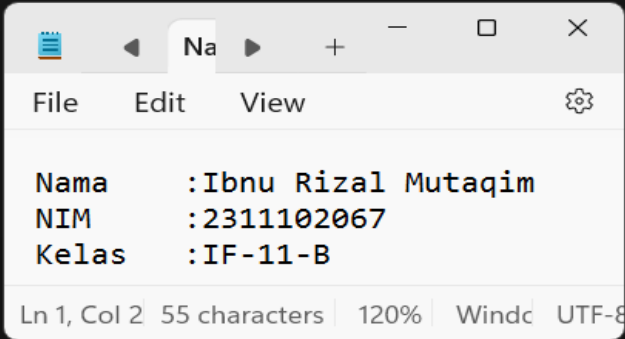


b) Hapus data Denis

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teangah
10.Hapus list
11.Tampilkan
0. Exit

Pilih Operasi :> 9
hapus tengah
Nama Nim
Jawad 23300001
Ibnu 2311102067
Farrel 23300003
Wati 2330004
Denis 23300005
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099

Masukkan Posisi yang dihapus : 5
Data Denis berhasil dihapus!
```



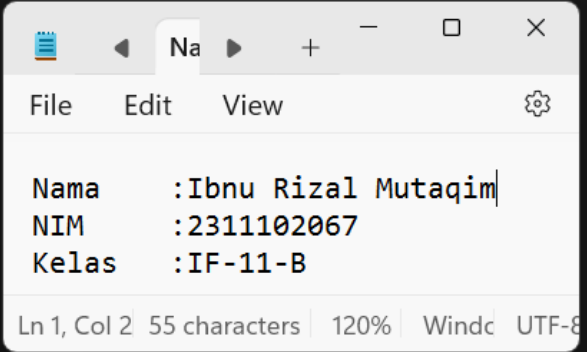
c) Tambahkan data berikut di awal:

Owi 2330000

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

Pilih Operasi :> 1
tambah depan

Masukkan Nama : Owi
Masukkan NIM : 2330000
Data Owi berhasil diinput!
```



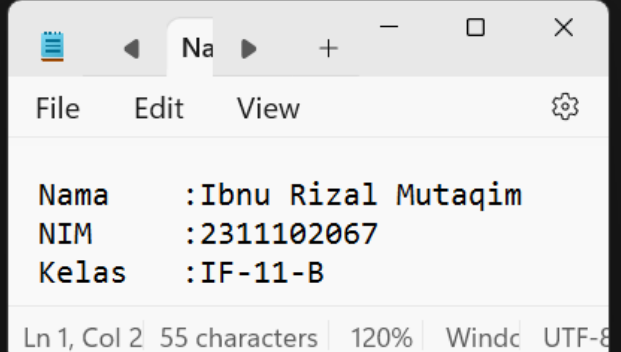
d) Tambahkan data berikut di akhir:

David 23300100

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

Pilih Operasi :> 2
tambah belakang

Masukkan Nama : David
Masukkan NIM : 23300100
```



e) e) Ubah data Udin menjadi data berikut:

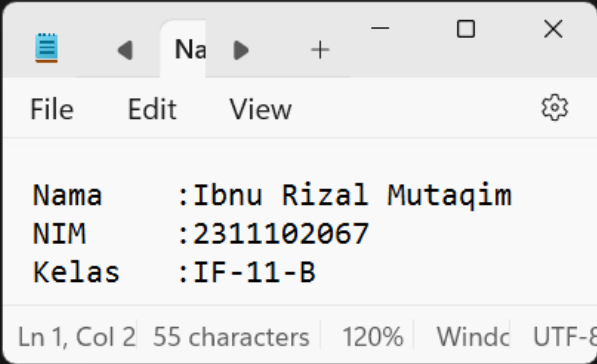
Idin 23300045

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teangah
10.Hapus list
11.Tampilkan
0. Exit

Pilih Operasi :> 6
ubah tengah

Masukkan Nama : Idin
Masukkan NIM : 23300045

Masukkan posisi data yang akan diubah : 9
```



The screenshot shows a Notepad window titled 'Na' with a menu bar (File, Edit, View) and a status bar (Ln 1, Col 2, 55 characters, 120%, Windc, UTF-8). The text content is as follows:

Nama	: Ibnu Rizal Mutaqim
NIM	: 2311102067
Kelas	: IF-11-B

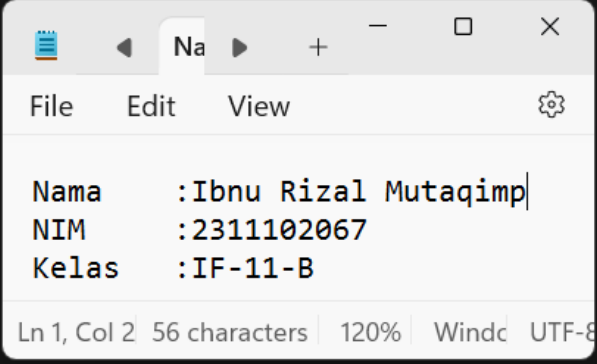
f) f) Ubah data terkahir menjadi berikut:

Lucy 23300101

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teangah
10.Hapus list
11.Tampilkan
0. Exit

Pilih Operasi :> 5
ubah belakang

Masukkan Nama : Lucy
Masukkan NIM : 23300101
data David telah digan|f dengan data Lucy
```



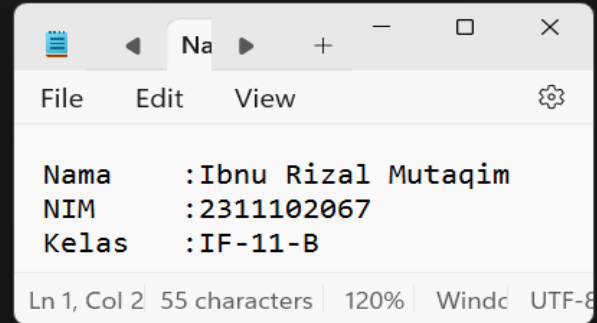
The screenshot shows a Notepad window titled 'Na' with a menu bar (File, Edit, View) and a status bar (Ln 1, Col 2, 56 characters, 120%, Windc, UTF-8). The text content is as follows:

Nama	: Ibnu Rizal Mutaqimp
NIM	: 2311102067
Kelas	: IF-11-B

g) g) Hapus data awal

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teangah
10.Hapus list
11.Tampilkan
0. Exit

Pilih Operasi :> 7
hapus depan
Data Owi berhasil dihapus
```



The screenshot shows a Notepad window titled 'Na' with a menu bar (File, Edit, View) and a status bar (Ln 1, Col 2, 55 characters, 120%, Windc, UTF-8). The text content is as follows:

Nama	: Ibnu Rizal Mutaqim
NIM	: 2311102067
Kelas	: IF-11-B

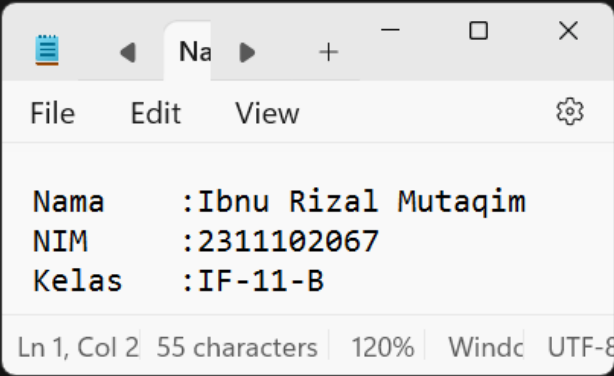
h) h) Ubah data awal menjadi berikut:

Bagas 2330002

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

Pilih Operasi :> 4
ubah depan

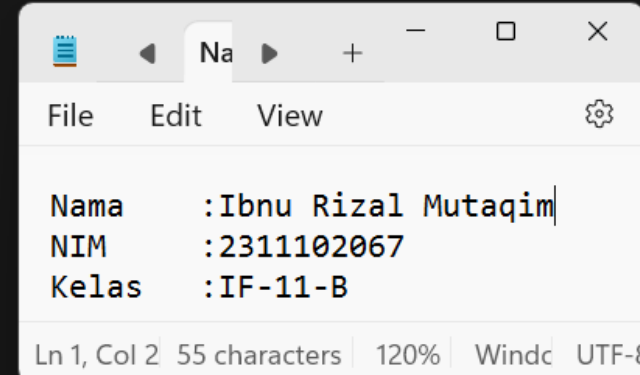
Masukkan Nama : Bagas
Masukkan NIM : 2330002
data Jawad telah diganti dengan data Bagas
```



i) Hapus data akhir

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teangah
10.Hapus list
11.Tampilkan
0. Exit




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



j) Tampilkan seluruh data

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teengah
10.Hapus list
11.Tampilkan
0. Exit

Pilih Operasi :> 11
Nama  Nim
Bagas 2330002
Ibnu 2311102067
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
Idin 23300045
Ucok 23300050
Budi 23300099
```

 Na  + - □ ×
File Edit View 
Nama : Ibnu Rizal Mutaqim
NIM : 2311102067
Kelas : IF-11-B
Ln 1, Col 2 | 55 characters | 120% | Windc UTF-8

C. Kesimpulan

Kesimpulan singkatnya adalah:

Linked list merupakan salah satu struktur data yang terdiri dari simpul-simpul yang terhubung melalui pointer. Ada dua jenis utama linked list: Non Circular dan circular. Linked list non circular adalah struktur data di mana setiap simpul memiliki pointer yang menunjuk ke simpul berikutnya dalam urutan linear. Tidak ada simpul yang menunjuk kembali ke simpul sebelumnya, sehingga linked list non circular memiliki awal dan akhir yang jelas. Di sisi lain, linked list circular adalah jenis linked list di mana simpul terakhir memiliki pointer yang menunjuk kembali ke simpul pertama, membentuk lingkaran. Dalam linked list circular, traversal dapat dimulai dari simpul manapun dan berakhir ketika kembali ke simpul awal. Meskipun operasi-operasi pada kedua jenis linked list mirip, perlu diperhatikan bahwa manipulasi linked list circular harus dilakukan dengan hati-hati untuk menghindari kebingungan atau looping tak terbatas.

2. Referensi (APA)

[1] Asisten Pratikum “Modul 4 Linked List Circular Dan Non Circular”, Learning Management System, 2024.

[2] Imam Ibnu Badri. (2019, maret). Single Linked List. Diakses pada 9 april 2024.

<https://www.teachmesoft.com/2019/03/single-linked-list-c-disertai-contoh.html>

[3] Imam Ibnu Badri. (2019, maret). Double Linked List. Diakses pada 9 april 2024.

<https://www.teachmesoft.com/2019/03/double-link-list-c-disertai-contoh.html>