

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL IV
LINKED LIST CIRCULAR DAN NON CIRCULAR**



Disusun Oleh :

NAMA : D'sharlendita Febianda Aurelia
NIM : 2311102069

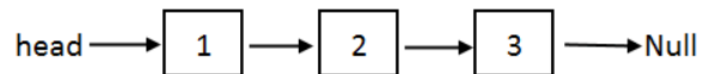
Dosen :

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

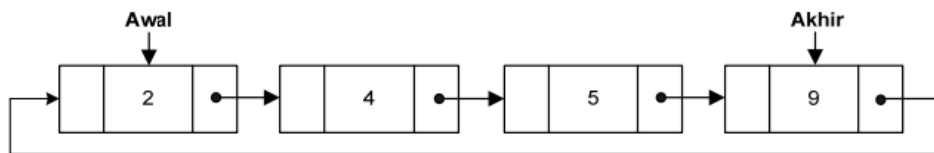
**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

A. Dasar Teori

Linked List Non Circular: Linked List Non Circular adalah suatu data struktur yang hanya memiliki satu variabel pointer, dimana pointer tersebut menunjuk ke node selanjutnya. Biasanya field pada tail menunjuk ke NULL.



Linked List Circular: Linked List Circular merupakan suatu struktur data yang dimana node terakhirnya akan menunjuk ke node terdepan, dan node terdepan akan menunjuk ke node terakhir. Hal ini membuat linked list berputar, sehingga tidak ada pointer yang menunjuk ke NULL.



B. Guided

Guided 1

Program 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 :

```

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:\parktikum struktur data>

```

Deskripsi:

Program ini adalah implementasi dari Linked List Non Circular. Program ini menyediakan beberapa fungsi seperti menambahkan data, mengubah data, menghapus data, menghapus list, menghitung jumlah list, dan menampilkan data.

Guided 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;
}

```

Screenshots Output :

```

Ayam
Bebek Ayam
Bebek Ayam Cicak
Bebek Ayam Cicak Domba
Bebek Ayam Cicak
Ayam Cicak
Ayam Sapi Cicak
PS C:\parktikum struktur data>

```

Deskripsi:

Program ini adalah implementasi dari Linked List Circular. Program ini menyediakan beberapa fungsi seperti menambahkan data, mengubah data, menghapus data, menghapus list, menghitung jumlah list, dan menampilkan data.

C. 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 <string>
using namespace std;

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

class CircularLinkedList {
private:
    dataMahasiswa* head;

public:
    CircularLinkedList() : head(nullptr) {}

    void tambahDepan(string nama, string nim) {
        dataMahasiswa* newNode = new dataMahasiswa();
        newNode->nama = nama;
        newNode->nim = nim;
        newNode->next = nullptr;

        if (head == nullptr) {
            head = newNode;
            newNode->next = head;
        } else {
            dataMahasiswa* temp = head;
            while (temp->next != head)
                temp = temp->next;
            temp->next = newNode;
            newNode->next = head;
            head = newNode;
        }
        cout << "\nData telah ditambahkan" << endl;
    }
}
```

```

void tambahBelakang(string nama, string nim) {
    dataMahasiswa* newNode = new dataMahasiswa();
    newNode->nama = nama;
    newNode->nim = nim;
    newNode->next = nullptr;

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

void tambahTengah(string nama, string nim, int posisi) {
    dataMahasiswa* newNode = new dataMahasiswa();
    newNode->nama = nama;
    newNode->nim = nim;
    newNode->next = nullptr;

    if (posisi == 1) {
        tambahDepan(nama, nim);
        return;
    }

    dataMahasiswa* temp = head;
    for (int i = 1; i < posisi - 1; ++i) {
        if (temp == nullptr || temp->next == head) {
            cout << "\nPosisi tidak valid" << endl;
            return;
        }
        temp = temp->next;
    }

    newNode->next = temp->next;
    temp->next = newNode;
    cout << "\nData telah ditambahkan" << endl;
}

```

```

    }

    void ubahDepan(string namaBaru, string nimBaru) {
        if (head == nullptr) {
            cout << "\nLinked list kosong" << endl;
            return;
        }
        head->nama = namaBaru;
        head->nim = nimBaru;
        cout << "\nData depan berhasil diubah" << endl;
    }

    void ubahBelakang(string namaBaru, string nimBaru) {
        if (head == nullptr) {
            cout << "\nLinked list kosong" << endl;
            return;
        }

        dataMahasiswa* temp = head;
        do {
            temp = temp->next;
        } while (temp->next != head);
        temp->nama = namaBaru;
        temp->nim = nimBaru;
        cout << "\nData belakang berhasil diubah" << endl;
    }

    void ubahTengah(string namaBaru, string nimBaru, int posisi)
    {
        if (head == nullptr) {
            cout << "\nLinked list kosong" << endl;
            return;
        }

        dataMahasiswa* temp = head;
        for (int i = 1; i < posisi; ++i) {
            temp = temp->next;
            if (temp == head) {
                cout << "\nPosisi tidak valid" << endl;
                return;
            }
        }
    }

```



```

        temp->nama = namaBaru;
        temp->nim = nimBaru;
        cout << "\nData pada posisi ke-" << posisi << " berhasil
diubah" << endl;
    }

    void hapusDepan() {
        if (head == nullptr) {
            cout << "\nLinked list kosong" << endl;
            return;
        }

        dataMahasiswa* temp = head;
        if (temp->next == head) {
            delete temp;
            head = nullptr;
        } else {
            dataMahasiswa* tail = head;
            while (tail->next != head)
                tail = tail->next;
            head = head->next;
            tail->next = head;
            delete temp;
        }
        cout << "\nData di depan berhasil dihapus" << endl;
    }

    void hapusBelakang() {
        if (head == nullptr) {
            cout << "Linked list kosong" << endl;
            return;
        }

        dataMahasiswa* temp = head;
        dataMahasiswa* tail = nullptr;
        while (temp->next != head) {
            tail = temp;
            temp = temp->next;
        }

        if (temp == head) {
            delete temp;
            head = nullptr;

```

```

    } else {
        tail->next = head;
        delete temp;
    }
    cout << "\nData di belakang berhasil dihapus" << endl;
}

void hapusTengah(int posisi) {
    if (head == nullptr) {
        cout << "Linked list kosong" << endl;
        return;
    }

    if (posisi == 1) {
        hapusDepan();
        return;
    }

    dataMahasiswa* temp = head;
    dataMahasiswa* prev = nullptr;
    for (int i = 1; i < posisi; ++i) {
        prev = temp;
        temp = temp->next;
        if (temp == head) {
            cout << "Posisi tidak valid" << endl;
            return;
        }
    }

    prev->next = temp->next;
    delete temp;
    cout << "\nData pada posisi ke-" << posisi << " berhasil
dihapus" << endl;
}

void hapusList() {
    if (head == nullptr) {
        cout << "Linked list kosong" << endl;
        return;
    }

    dataMahasiswa* temp = head;
    dataMahasiswa* next;

```

```

        do {
            next = temp->next;
            delete temp;
            temp = next;
        } while (temp != head);
        head = nullptr;
        cout << "\nLinked list berhasil dihapus" << endl;
    }

    void tampilkanData() {
        if (head == nullptr) {
            cout << "Linked list kosong" << endl;
            return;
        }
        dataMahasiswa* temp = head;
        cout << "_____ " << endl;
        cout << "| \t" << " DATA MAHASISWA " << " \t|" << endl;
        cout << "_____ " << endl;
        cout << "| NAMA \t\t | NIM \t\t|" << endl;
        do {
            cout << "_____ " << endl;
            cout << "|" << temp-> nama << " \t\t" << " | " << temp-
> nim << " \t" << "|" << endl;
            temp = temp->next;
        } while (temp != head);
        cout << "_____ " << endl;
    }

    ~CircularLinkedList() {
        if (head == nullptr)
            return;
        dataMahasiswa* curr = head;
        dataMahasiswa* next;
        do {
            next = curr->next;
            delete curr;
            curr = next;
        } while (curr != head);
    }
};

int main() {
    CircularLinkedList dita;

```

```

int choice, posisi;
string nama, nim;

do {
    cout << endl;
    cout << "\t      " << " M E N U " << "\t" << endl;
    cout << "_____ " << endl;
    cout << endl;
    cout << "1. Tambah Depan" << endl;
    cout << "2. Tambah Belakang" << endl;
    cout << "3. Tambah Tengah" << endl;
    cout << "4. Ubah Depan" << endl;
    cout << "5. Ubah Belakang" << endl;
    cout << "6. Ubah Tengah" << endl;
    cout << "7. Hapus Depan" << endl;
    cout << "8. Hapus Belakang" << endl;
    cout << "9. Hapus Tengah" << endl;
    cout << "10. Hapus List" << endl;
    cout << "11. Tampilkan Data" << endl;
    cout << "0. Keluar" << endl;
    cout << "_____ " << endl;
    cout << endl;
    cout << "\nPilih Operasi : ";
    cin >> choice;
    cout << endl;

    switch (choice) {
        case 1:
            cout << "_____ " <<
endl;
            cout << endl;
            cout << "\t " << " Tambah Depan " << "\t" << endl;
            cout << endl;
            cout << "Masukkan Nama : ";
            cin >> nama;
            cout << "Masukkan NIM : ";
            cin >> nim;
            dita.tambahDepan(nama, nim);
            cout << "_____ " <<
endl;
            break;
        case 2:
            cout << "_____ " <<

```

```

endl;
        cout << endl;
        cout << "\t" << " Tambah Belakang " << "\t" <<
endl;
        cout << endl;
        cout << "Masukkan Nama : ";
        cin >> nama;
        cout << "Masukkan NIM : ";
        cin >> nim;
        dita.tambahBelakang(nama, nim);
        cout << "_____ " <<
endl;
        break;
    case 3:
        cout << "_____ " <<
endl;
        cout << endl;
        cout << "\t " << " Tambah Tengah " << "\t" <<
endl;
        cout << endl;
        cout << "Masukkan Nama : ";
        cin >> nama;
        cout << "Masukkan NIM : ";
        cin >> nim;
        cout << "Masukkan Posisi : ";
        cin >> posisi;
        dita.tambahTengah(nama, nim, posisi);
        cout << "_____ " <<
endl;
        break;
    case 4:
        cout << "_____ " <<
endl;
        cout << endl;
        cout << "\t " << " Ubah Depan " << "\t" << endl;
        cout << endl;
        cout << "Masukkan Nama Baru : ";
        cin >> nama;
        cout << "Masukkan NIM Baru : ";
        cin >> nim;
        dita.ubahDepan(nama, nim);
        cout << "_____ " <<
endl;

```

```

        break;
    case 5:
        cout << "_____ " <<
endl;

        cout << endl;
        cout << "\t " << " Ubah Belakang " << "\t" <<
endl;

        cout << endl;
        cout << "Masukkan Nama Baru : ";
        cin >> nama;
        cout << "Masukkan NIM Baru : ";
        cin >> nim;
        dita.ubahBelakang(nama, nim);
        cout << "_____ " <<
endl;

        break;
    case 6:
        cout << "_____ " <<
endl;

        cout << endl;
        cout << "\t " << " Ubah Tengah " << "\t" << endl;
        cout << endl;
        cout << "Masukkan Nama Baru : ";
        cin >> nama;
        cout << "Masukkan NIM Baru : ";
        cin >> nim;
        cout << "Masukkan Posisi : ";
        cin >> posisi;
        dita.ubahTengah(nama, nim, posisi);
        cout << "_____ " <<
endl;

        break;
    case 7:
        cout << "_____ " <<
endl;

        cout << endl;
        cout << "\t " << " Hapus Depan " << "\t" << endl;
        cout << endl;
        dita.hapusDepan();
        cout << "_____ " <<
endl;

        break;
    case 8:

```

```

        cout << "_____ " <<
endl;
        cout << endl;
        cout << "\t " << " Hapus Belakang " << "\t" <<
endl;
        cout << endl;
        dita.hapusBelakang();
        cout << "_____ " <<
endl;
        break;
    case 9:
        cout << "_____ " <<
endl;
        cout << endl;
        cout << "\t " << " Hapus Tengah " << "\t" <<
endl;
        cout << endl;
        cout << "Masukkan Posisi : ";
        cin >> posisi;
        dita.hapusTengah(posisi);
        cout << "_____ " <<
endl;
        break;
    case 10:
        cout << "_____ " <<
endl;
        cout << endl;
        cout << "\t " << " Hapus List " << "\t" << endl;
        cout << endl;
        cout << "List anda telah dihapus!" << endl;
        dita.hapusList();
        cout << "_____ " <<
endl;
        break;
    case 11:
        cout << "_____ " <<
endl;
        cout << endl;
        cout << "\t" << " Menampilkan Data " << "\t" <<
endl;
        cout << endl;
        dita.tampilkanData();

```

```

        break;
    case 0:
        cout << "_____ " <<
endl;

        cout << endl;
        cout << "Terima kasih!" << endl;
        cout << "_____ " <<
endl;

        break;
    default:
        cout << "_____ " <<
endl;

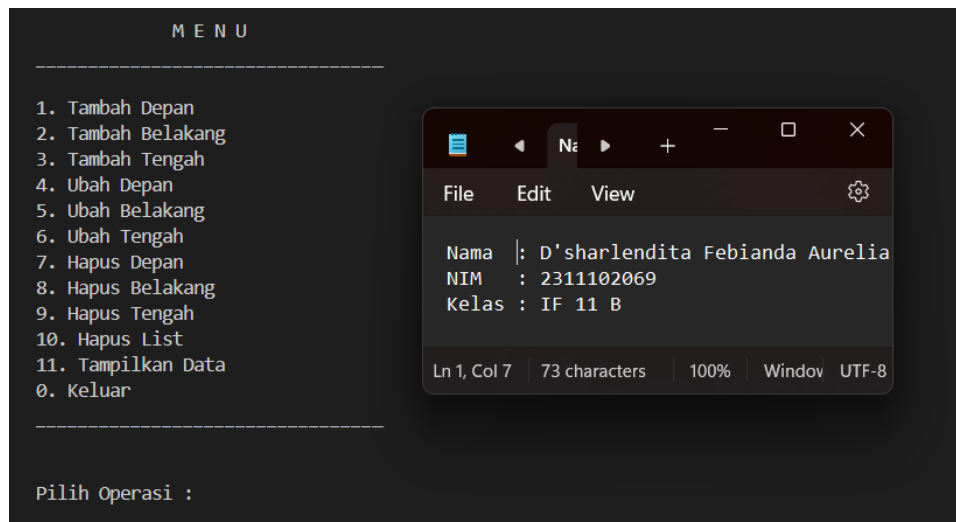
        cout << endl;
        cout << "Pilihan anda tidak valid!" << endl;
        cout << "_____ " <<
endl;
    }
} while (choice != 0);

return 0;
}

```

Screenshots Output

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



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

```
M E N U
-----
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 Data
0. Keluar
-----

Pilih Operasi : 1

-----

          Tambah Depan

Masukkan Nama : Jawad
Masukkan NIM : 23300001

Data telah ditambahkan
-----

M E N U
-----
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 Data
0. Keluar
-----

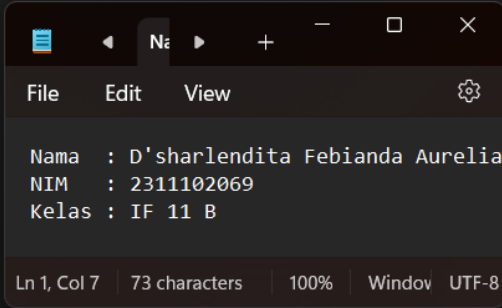
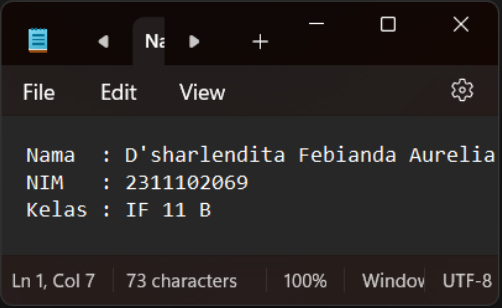
Pilih Operasi : 2

-----

          Tambah Belakang

Masukkan Nama : Budi
Masukkan NIM : 23300099

Data telah ditambahkan
-----
```



M E N U

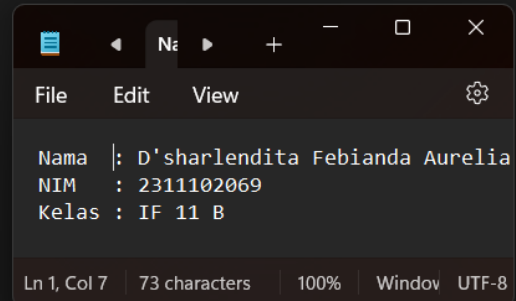
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Hapus List
11. Tampilkan Data
0. Keluar

Pilih Operasi : 3

Tambah Tengah

Masukkan Nama : Dita
Masukkan NIM : 2311102069
Masukkan Posisi : 2

Data telah ditambahkan



M E N U

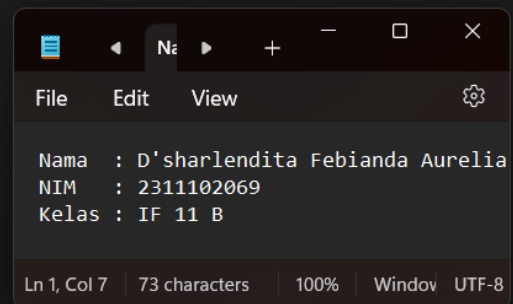
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Hapus List
11. Tampilkan Data
0. Keluar

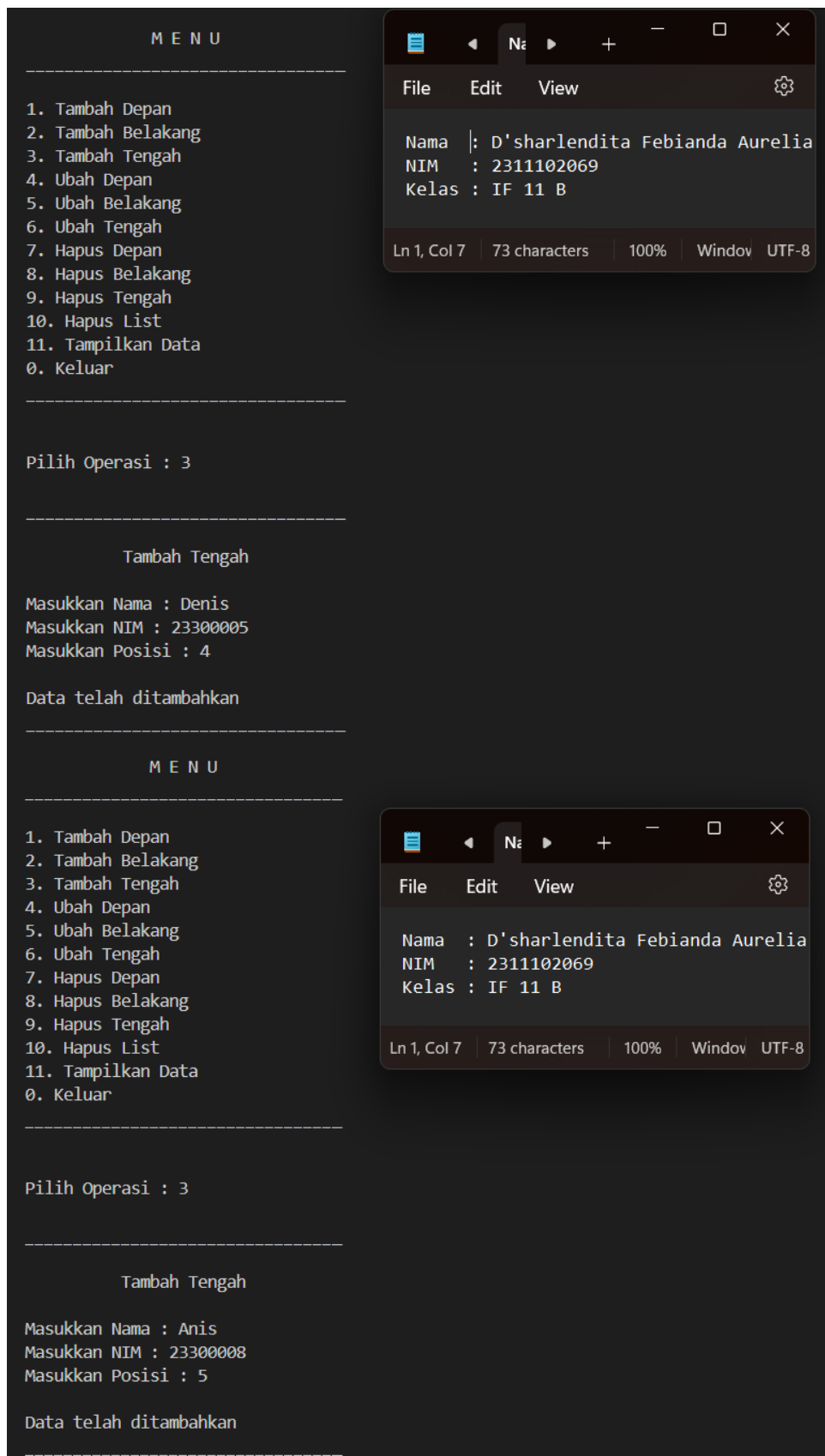
Pilih Operasi : 3

Tambah Tengah

Masukkan Nama : Farrel
Masukkan NIM : 23300003
Masukkan Posisi : 3

Data telah ditambahkan





M E N U

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 Data
0. Keluar

Pilih Operasi : 3

Tambah Tengah

Masukkan Nama : Bowo
Masukkan NIM : 23300015
Masukkan Posisi : 6

Data telah ditambahkan

M E N U

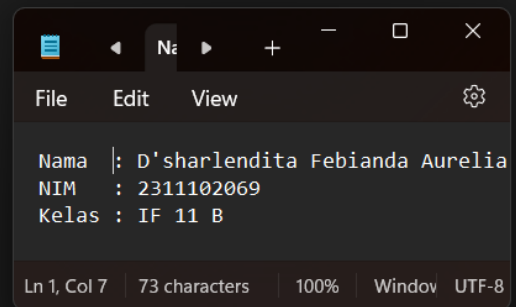
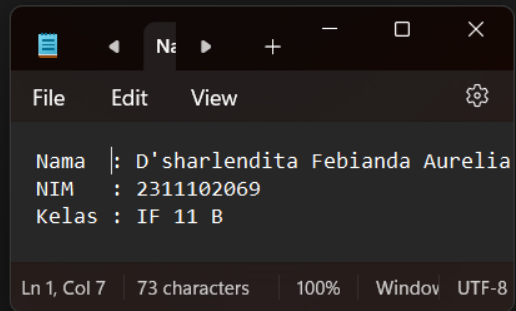
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Hapus List
11. Tampilkan Data
0. Keluar

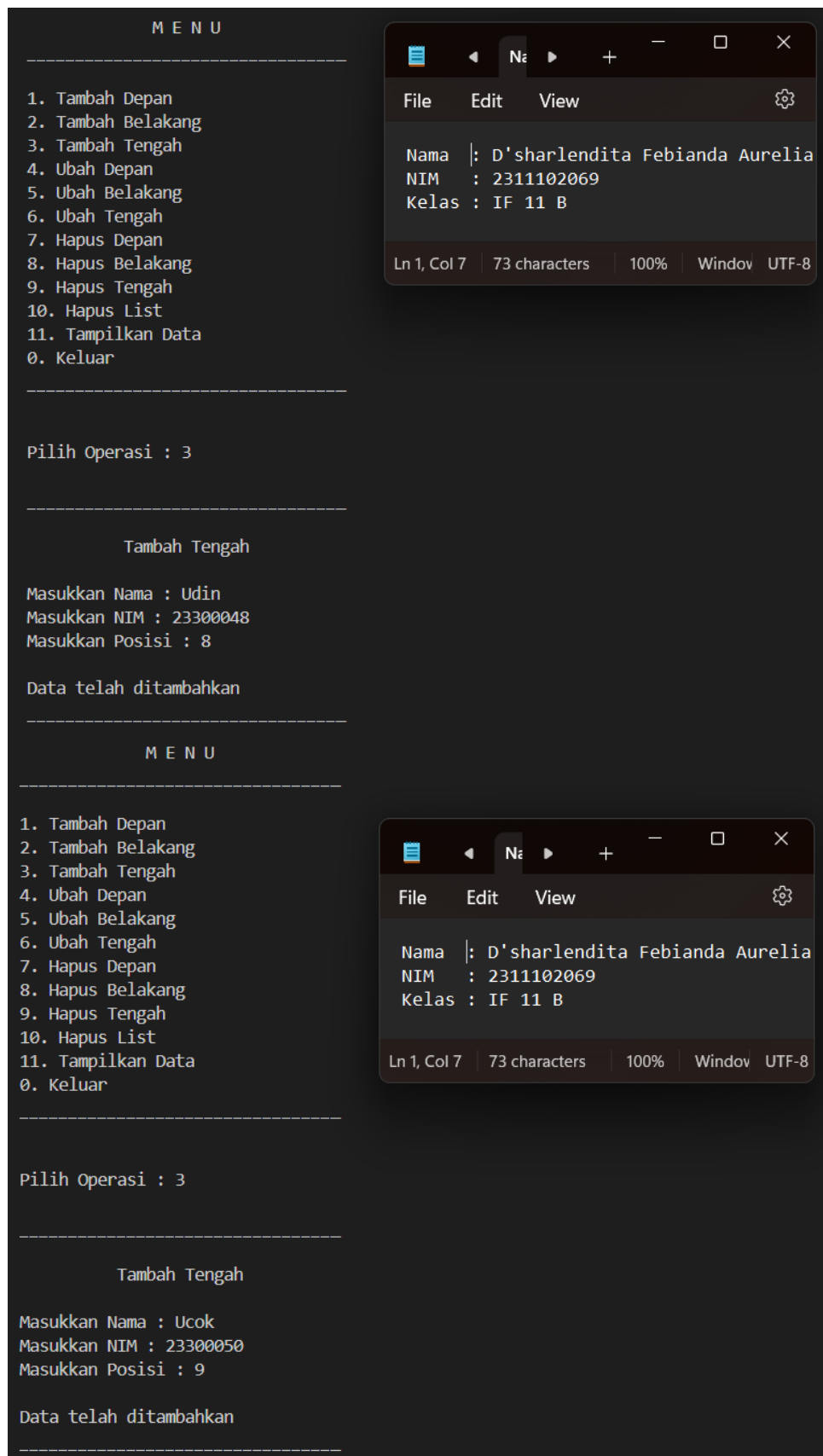
Pilih Operasi : 3

Tambah Tengah

Masukkan Nama : Gahar
Masukkan NIM : 23300040
Masukkan Posisi : 7

Data telah ditambahkan





M E N U

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 Data
0. Keluar

Pilih Operasi : 11

Menampilkan Data

DATA MAHASISWA	
NAMA	NIM
Jawad	23300001
Dita	2311102069
Farrel	23300003
Denis	23300005
Anis	23300008
Bowo	23300015
Gahar	23300040
Udin	23300048
Ucok	23300050
Budi	23300099

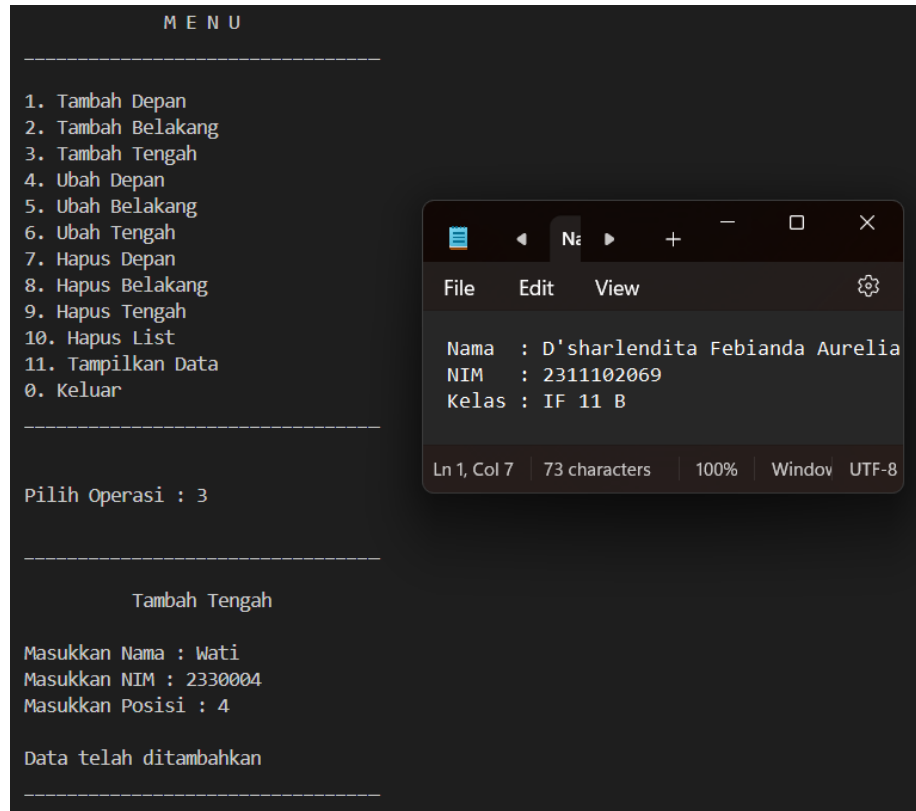
```
File Edit View
Nama |: D'sharlendita Febianda Aurelia
NIM  : 2311102069
Kelas : IF 11 B
Ln 1, Col 7 | 73 characters | 100% | Window UTF-8
```

```
File Edit View
Nama |: D'sharlendita Febianda Aurelia
NIM  : 2311102069
Kelas : IF 11 B
Ln 1, Col 7 | 73 characters | 100% | Window UTF-8
```

3. Lakukan perintah berikut:

a) Tambahkan data berikut diantara Farrel dan Denis:

Wati 2330004



```
M E N U
-----
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 Data
0. Keluar

Pilih Operasi : 3

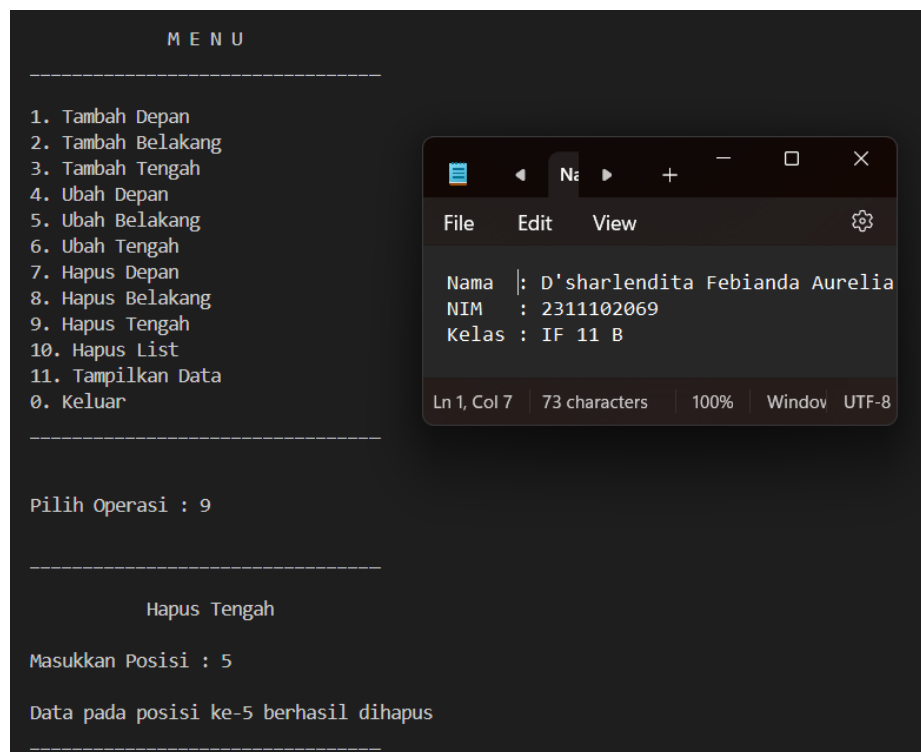
-----

          Tambah Tengah

Masukkan Nama : Wati
Masukkan NIM : 2330004
Masukkan Posisi : 4

Data telah ditambahkan
-----
```

b) Hapus data Denis



```
M E N U
-----
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 Data
0. Keluar

Pilih Operasi : 9

-----

          Hapus Tengah

Masukkan Posisi : 5

Data pada posisi ke-5 berhasil dihapus
-----
```

c) Tambahkan data berikut di awal:

Owi 2330000

```
M E N U
-----
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 Data
0. Keluar

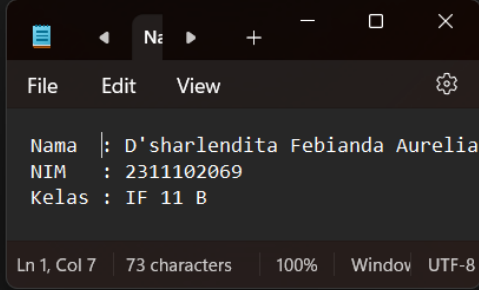
Pilih Operasi : 1

-----

          Tambah Depan

Masukkan Nama : Owi
Masukkan NIM : 2330000

Data telah ditambahkan
-----
```



d) Tambahkan data berikut di akhir:

David 23300100

```
M E N U
-----
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 Data
0. Keluar

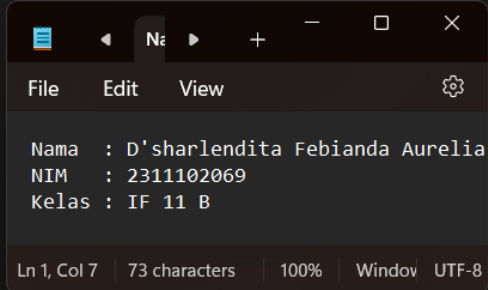
Pilih Operasi : 2

-----

          Tambah Belakang

Masukkan Nama : David
Masukkan NIM : 23300100

Data telah ditambahkan
-----
```



e) Ubah data Udin menjadi data berikut:

Idin 23300045

```
M E N U
-----
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 Data
0. Keluar

Pilih Operasi : 6

-----

Ubah Tengah

Masukkan Nama Baru : Idin
Masukkan NIM Baru : 23300045
Masukkan Posisi : 9

Data pada posisi ke-9 berhasil diubah
-----
```

f) Ubah data terakhir menjadi berikut:

Lucy 23300101

```
M E N U
-----
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 Data
0. Keluar

Pilih Operasi : 5

-----

Ubah Belakang

Masukkan Nama Baru : Lucy
Masukkan NIM Baru : 23300101

Data belakang berhasil diubah
-----
```

g) Hapus data awal

```

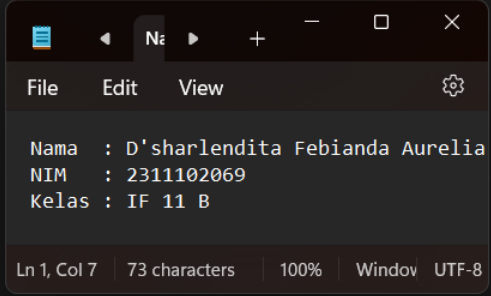
M E N U
-----
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Hapus List
11. Tampilkan Data
0. Keluar

Pilih Operasi : 7

-----

Hapus Depan

Data di depan berhasil dihapus
-----
```



h) Ubah data awal menjadi berikut:

Bagas 2330002

```

M E N U
-----
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 Data
0. Keluar

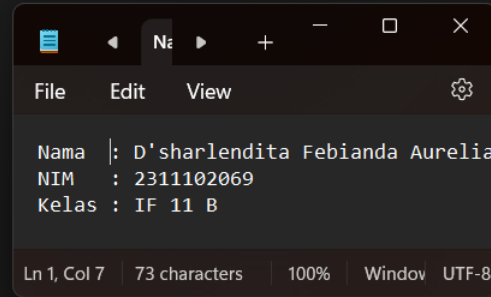
Pilih Operasi : 4

-----

Ubah Depan

Masukkan Nama Baru : Bagas
Masukkan NIM Baru : 2330002

Data depan berhasil diubah
-----
```



i) Hapus data akhir

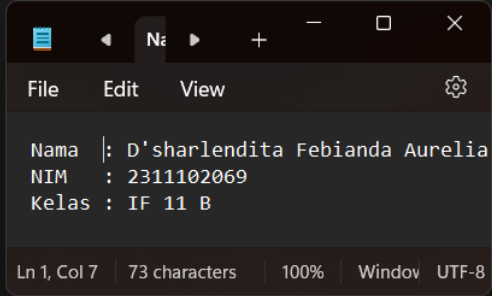
```
M E N U
-----
1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus Depan
8. Hapus Belakang
9. Hapus Tengah
10. Hapus List
11. Tampilkan Data
0. Keluar

Pilih Operasi : 8

-----

Hapus Belakang

Data di belakang berhasil dihapus
-----
```



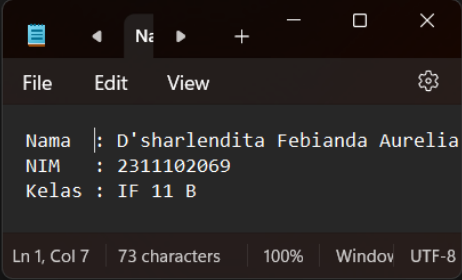
The screenshot shows a terminal window with a menu and a confirmation message. The menu lists 11 options, with '8. Hapus Belakang' selected. Below the menu, the text 'Hapus Belakang' is displayed, followed by 'Data di belakang berhasil dihapus'.

j) Tampilkan seluruh data

Output:

```
-----
Menampilkan Data
-----
```

DATA MAHASISWA	
NAMA	NIM
Bagas	2330002
Dita	2311102069
Farrel	23300003
Wati	2330004
Anis	23300008
Bowo	23300015
Gahar	23300040
Idin	23300045
Ucok	23300050
Budi	23300099



The screenshot shows a terminal window displaying a table of student data. The table has two columns: 'NAMA' and 'NIM'. The data is as follows:

NAMA	NIM
Bagas	2330002
Dita	2311102069
Farrel	23300003
Wati	2330004
Anis	23300008
Bowo	23300015
Gahar	23300040
Idin	23300045
Ucok	23300050
Budi	23300099

Deskripsi:

Program ini adalah implementasi dari Linked List Non Circular yang berisi data mahasiswa. Program ini menyediakan beberapa fungsi seperti menambahkan data mahasiswa, mengubah data mahasiswa, menghapus data mahasiswa, menghapus list, dan menampilkan data mahasiswa yang sudah terdata di dalam Linked List Non Circular. Program ini menyediakan pilihan menu agar dapat memudahkan pengguna dalam memilih fungsi yang diinginkan.

D. Kesimpulan

Setelah mempelajari materi Linked List Circular Dan Non Circular, dapat disimpulkan bahwa Linked List Non-Circular adalah jenis Linked List yang pointer terakhirnya menunjuk ke NULL yang artinya tidak ada simpul yang mengarah ke simpul pertama. Sedangkan Linked List Circular adalah jenis Linked List yang pointer terakhirnya menunjuk ke simpul pertama yang berarti bahwa setiap simpul saling terhubung dalam lingkaran.

E. Referensi

Asisten Praktikum. 3 April 2024. "Modul 4 Linked List Circular dan Non Circular". Diakses pada 8 April 2024, dari Learning Management System. 2024

Trivusi. 16 September 2022. Struktur Data Linked List : Pengertian, Karakteristik, dan Jenis-jenisnya. Diakses pada 9 April 2024, dari <https://www.trivusi.web.id/2022/07/struktur-data-linked-list.html>

Geeksforgeeks.org. 10 April 2024. Linked List Data Structure. Diakses pada 13 April 2024, dari <https://www.geeksforgeeks.org/data-structures/linked-list/>