LAPORAN PRAKTIKUM

MODUL 4
LINKED LIST CIRCULAR DAN NON CIRCULAR

Disusun oleh:
Rakha Yudhistira
NIM: 2311102010

Dosen Pengampu:
Wahyu Andi Saputra, S. Pd., M.Eng.

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

BAB I

TUJUAN PRAKTIKUM

1. Praktikan dapat mengetahui dan memahami linked list circular dan non circular.
2. Praktikan dapat membuat linked list circular dan non circular.
3. Praktikan dapat mengaplikasikan atau menerapkan linked list circular dan non circular pada program yang dibuat.

BAB II

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.

2. Linked List Circular

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

BAB III

GUIDED

1. Guided 1

Source code

```
#include <iostream>

using namespace std;

// PROGRAM SINGLE LINKED LIST NON-CIRCULAR

// Deklarasi struct node
struct Node
{
    int data;
    Node *next;
};

Node *head; // Deklarasi head
Node *tail; // Deklarasi tail

// Inisialisasi Node
void init()
{
    head = NULL;
    tail = NULL;
}

// Pengecekkan apakah linked list kosong
bool isEmpty()
{
    if (head == NULL)
    {
        return true;
```

```
    }
    else
    {
        return false;
    }
}
// Tambah depan
void insertDepan(int nilai)
{

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

// Tambah belakang
void insertBelakang(int nilai)
{
    // buat node baru
    Node *baru = new Node();
    baru->data = nilai;
    baru->next = NULL;
    if (isEmpty() == true)
```

```
    {
        head = tail = baru;
        head->next = NULL;
    }
    else
    {
        tail->next = baru;
        tail = baru;
    }
}

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

// Tambah tengah
void insertTengah(int data, int posisi)
{
    if (posisi < 1 || posisi > hitungList())
    {
        cout << "Posisi di luar jangkauan" << endl;
    }
    else if (posisi == 1)
    {
```

```
        cout << "Posisi bukan posisi tengah" << endl;
    }
    else
    {
        Node *baru, *bantu;
        baru = new Node();
        baru->data = data;

        // tranversing
        bantu = head;
        int nomor = 1;
        while (nomor < posisi - 1)
        {
            bantu = bantu->next;
            nomor++;
        }

        baru->next = bantu->next;
        bantu->next = baru;
    }
}

// Hapus depan
void hapusDepan()
{
    Node *hapus;
    if (isEmpty() == false)
    {
        if (head->next != NULL)
        {
            hapus = head;
            head = head->next;
            delete hapus;
        }
```

```
        else
        {
            head = tail = NULL;
        }
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// Hapus belakang
void hapusBelakang()
{
    Node *hapus;
    Node *bantu;
    if (isEmpty() == false)
    {
        if (head != tail)
        {
            hapus = tail;
            bantu = head;
            while (bantu->next != tail)
            {
                bantu = bantu->next;
            }
            tail = bantu;
            tail->next = NULL;
            delete hapus;
        }
        else
        {
            head = tail = NULL;
        }
```

```
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}
// Hapus tengah
void hapusTengah(int posisi)
{
    Node *hapus, *bantu, *sebelum;
    if (posisi < 1 || posisi > hitungList())
    {
        cout << "Posisi di luar jangkauan" << endl;
    }
    else if (posisi == 1)
    {
        cout << "Posisi bukan posisi tengah" << endl;
    }
    else
    {
        int nomor = 1;
        bantu = head;
        while (nomor <= posisi)
        {
            if (nomor == posisi - 1)
            {
                sebelum = bantu;
            }
            if (nomor == posisi)
            {
                hapus = bantu;
            }
            bantu = bantu->next;
            nomor++;
```

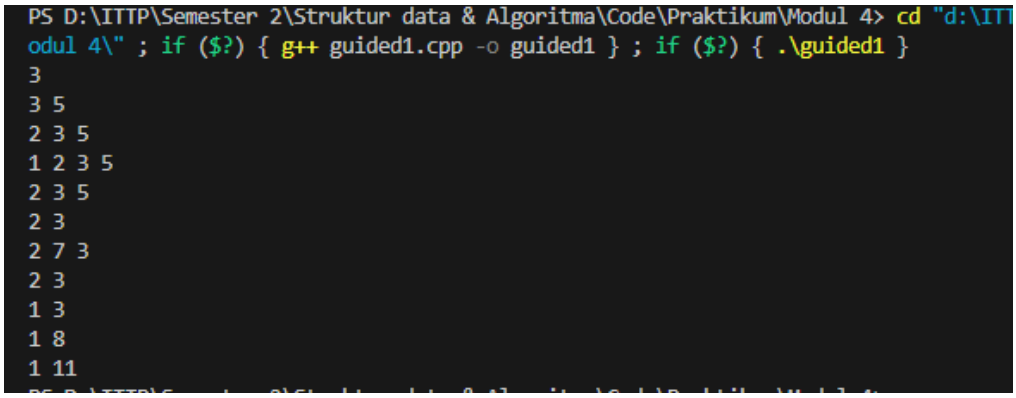
```
        }
        sebelum->next = bantu;
        delete hapus;
    }
}

// ubah depan
void ubahDepan(int data)
{
    if (isEmpty() == 0)
    {
        head->data = data;
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// ubah tengah
void ubahTengah(int data, int posisi)
{
    Node *bantu;
    if (isEmpty() == 0)
    {
        if (posisi < 1 || posisi > hitungList())
        {
            cout << "Posisi di luar jangkauan" << endl;
        }
        else if (posisi == 1)
        {
            cout << "Posisi bukan posisi tengah" << endl;
        }
        else
```

```
        {
            int nomor = 1;
            bantu = head;
            while (nomor < posisi)
            {
                bantu = bantu->next;
                nomor++;
            }
            bantu->data = data;
        }
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// ubah belakang
void ubahBelakang(int data)
{
    if (isEmpty() == 0)
    {
        tail->data = data;
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

// Hapus list
void clearList()
{
    Node *bantu, *hapus;
```

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

// Tampilkan list
void tampilList()
{
    Node *bantu;
    bantu = head;
    if (isEmpty() == false)
    {
        while (bantu != NULL)
        {
            cout << bantu->data << " ";
            bantu = bantu->next;
        }
        cout << endl;
    }
    else
    {
        cout << "Linked list masih kosong" << endl;
    }
}

int main()
{
    init();
```

```
    insertDepan(3);
    tampilList();
    insertBelakang(5);
    tampilList();
    insertDepan(2);
    tampilList();
    insertDepan(1);
    tampilList();
    hapusDepan();
    tampilList();
    hapusBelakang();
    tampilList();
    insertTengah(7, 2);
    tampilList();
    hapusTengah(2);
    tampilList();
    ubahDepan(1);
    tampilList();
    ubahBelakang(8);
    tampilList();
    ubahTengah(11, 2);
    tampilList();

    return 0;
}
```

Screenshoot program



Deskripsi program

Program ini menggunakan linked list non-circular dan memiliki fitur untuk menambah, menghapus, mengubah. Program tersebut juga menggunakan beberapa fungsi if, else if, dan while do. Selain itu, program tersebut juga menggunakan Struct.

2. Guided 2

Source code

```
#include <iostream>

using namespace std;

// Deklarasi Struct Node

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

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

// Inisialisasi node head & tail
```

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

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

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

// Hitung List
int hitungList()
{
    bantu = head;
    int jumlah = 0;
    while (bantu != NULL)
    {
```

```
        jumlah++;
        bantu = bantu->next;
    }
    return jumlah;
}

// 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;
    }
}

// Tambah Belakang
void insertBelakang(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;
        }
        tail->next = baru;
        baru->next = head;
    }
}

// Tambah Tengah
void insertTengah(string data, int posisi)
{
    if (isEmpty() == 1)
    {
        head = baru;
        tail = head;
        baru->next = head;
    }
    else
    {
        baru->data = data;
        // transversing
        int nomor = 1;
        bantu = head;
```

```
        while (nomor < posisi - 1)
        {
            bantu = bantu->next;
            nomor++;
        }
        baru->next = bantu->next;
        bantu->next = baru;
    }
}

// Hapus Depan
void hapusDepan()
{
    if (isEmpty() == 0)
    {
        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;
        }
```

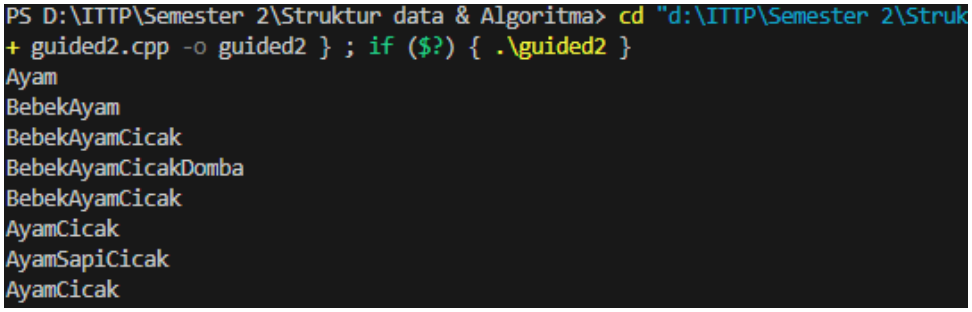
```
    }
    else
    {
        cout << "List masih kosong!" << endl;
    }
}

// Hapus Belakang
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;
        }
```

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

// Hapus Tengah
void hapusTengah(int posisi)
{
    if (isEmpty() == 0)
    {
        // transversing
        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;
    }
}

// Hapus List
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;
}

// Tampilkan List
void tampil()
{
    if (isEmpty() == 0)
    {
        tail = head;
        do
        {
            cout << tail->data << ends;
            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;
}
```

Screenshoot program



Deskripsi Program

Program ini dibuat dengan menggunakan Linked List Circular, program tersebut dapat menambah dan menghapus pada depan dan belakang. Struktur data tersebut terdiri dari Struct Node yang berisi data dan pointer ke Node selanjutnya

LATIHAN KELAS - UNGUIDED

1. **Unguided 1**

Source code

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

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

Node *head = NULL;
Node *tail = NULL;

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

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

void insertDepan(string nama, string nim)
{
    Node *baru = new Node;
    baru->nama = nama;
    baru->nim = nim;
```

```
    baru->next = NULL;
    if (isEmpty())
    {
        head = tail = baru;
    }
    else
    {
        baru->next = head;
        head = baru;
    }
}

void insertBelakang(string nama, string nim)
{
    Node *baru = new Node;
    baru->nama = nama;
    baru->nim = nim;
    baru->next = NULL;
    if (isEmpty())
    {
        head = tail = baru;
    }
    else
    {
        tail->next = baru;
        tail = baru;
    }
}

int hitungList()
{
    Node *current = head;
    int count = 0;
    while (current != NULL)
```

```
    {
        count++;
        current = current->next;
    }
    return count;
}

void insertTengah(string nama, string nim, int posisi)
{
    if (posisi < 1 || posisi > hitungList() + 1)
    {
        cout << "Posisi diluar jangkauan" << endl;
    }
    else if (posisi == 1)
    {
        insertDepan(nama, nim);
    }
    else if (posisi == hitungList() + 1)
    {
        insertBelakang(nama, nim);
    }
    else
    {
        Node *baru = new Node();
        baru->nama = nama;
        baru->nim = nim;
        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;
        head = head->next;
        delete hapus;
        if (head == NULL)
            tail = NULL;
    }
    else
    {
        cout << "List kosong!" << endl;
    }
}

void hapusBelakang()
{
    if (!isEmpty())
    {
        if (head == tail)
        {
            delete head;
            head = tail = NULL;
        }
        else
        {
            Node *bantu = head;
            while (bantu->next != tail)
            {
```

```
                bantu = bantu->next;
            }
            delete tail;
            tail = bantu;
            tail->next = NULL;
        }
    }
    else
    {
        cout << "List kosong!" << endl;
    }
}

void hapusTengah(int posisi)
{
    if (isEmpty() || posisi < 1 || posisi > hitungList())
    {
        cout << "Posisi di luar jangkauan atau list kosong" <<
endl;
    }
    else if (posisi == 1)
    {
        hapusDepan();
    }
    else if (posisi == hitungList())
    {
        hapusBelakang();
    }
    else
    {
        Node *bantu = head;
        for (int nomor = 1; nomor < posisi - 1; ++nomor)
        {
            bantu = bantu->next;
```

```
        }
        Node *hapus = bantu->next;
        bantu->next = hapus->next;
        delete hapus;
    }
}

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

void ubahDepan(string nama, string nim)
{
    if (!isEmpty())
    {
        head->nama = nama;
        head->nim = nim;
    }
    else
    {
        cout << "List masih kosong!" << endl;
    }
}

void ubahTengah(string nama, string nim, int posisi)
```

```
{
    if (isEmpty() || posisi < 1 || posisi > hitungList())
    {
        cout << "Posisi di luar jangkauan atau list kosong" <<
endl;
    }
    else
    {
        Node *bantu = head;
        for (int nomor = 1; nomor < posisi; ++nomor)
        {
            bantu = bantu->next;
        }
        bantu->nama = nama;
        bantu->nim = nim;
    }
}

void ubahBelakang(string nama, string nim)
{
    if (!isEmpty())
    {
        tail->nama = nama;
        tail->nim = nim;
    }
    else
    {
        cout << "List masih kosong!" << endl;
    }
}
void tampil()
{
    if (!isEmpty())
    {
```

```
        Node *current = head;
        cout << "\nDATA MAHASISWA\n\nNama\t\tNIM\n";
        while (current != NULL)
        {
            cout << current->nama << "\t\t" << current->nim <<
endl;
            current = current->next;
        }
    }
    else
    {
        cout << "List masih kosong!" << endl;
    }
}
int main()
{
    while (true)
    {
        cout << "\nPROGRAM SINGLE LINKED LIST NON-CIRCULAR\n\n";
        cout << "1. Tambah Depan" << endl;
        cout << "2. Tambah Belakang" << endl;
        cout << "3. Tambah Tengah" << endl;
        cout << "4. Ubah Depan" << endl;
        cout << "5. Ubah Belakang" << endl;
        cout << "6. Ubah Tengah" << endl;
        cout << "7. Hapus Depan" << endl;
        cout << "8. Hapus Belakang" << endl;
        cout << "9. Hapus Tengah" << endl;
        cout << "10. Hapus List" << endl;
        cout << "11. TAMPILKAN" << endl;
        cout << "0. KELUAR" << endl;
        int pilih;
        cout << "\nPilih Operasi : ";
        cin >> pilih;
```

```
        switch (pilih)
        {
        case 1:
        {
            string nama;
            string nim;
            cout << "\n-Tambah Depan-\n\n";
            cout << "Masukan Nama : ";
            cin >> nama;
            cout << "Masukan Nim  : ";
            cin >> nim;
            insertDepan(nama, nim);
            cout << "\nData telah ditambahkan " << endl;
            break;
        }
        case 2:
        {
            string nama;
            string nim;
            cout << "\n-Tambah Belakang-";
            cout << "\n\nMasukan Nama : ";
            cin >> nama;
            cout << "Masukan Nim  : ";
            cin >> nim;
            insertBelakang(nama, nim);
            cout << "\nData telah ditambahkan " << endl;
            break;
        }
        case 3:
        {
            string nama;
            string nim;
            int posisi;
            cout << "\n-Tambah Tengah-\n\n";
```

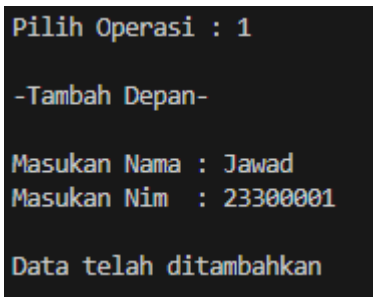
```
            cout << "Masukan Nama   : ";
            cin >> nama;
            cout << "Masukan Nim    : ";
            cin >> nim;
            cout << "Masukan Posisi : ";
            cin >> posisi;
            insertTengah(nama, nim, posisi);
            cout << "\nData telah ditambahkan " << endl;
            break;
        }
        case 4:
        {
            string nama;
            string nim;
            cout << "\n-Ubah Depan-\n\n";
            cout << "Masukan Nama : ";
            cin >> nama;
            cout << "Masukan Nim  : ";
            cin >> nim;
            ubahDepan(nama, nim);
            cout << "\nData telah diubah " << endl;
            ;
            break;
        }
        case 5:
        {
            string nama;
            string nim;
            cout << "\n-Ubah Belakang-\n\n";
            cout << "Masukan Nama : ";
            cin >> nama;
            cout << "Masukan Nim  : ";
            cin >> nim;
            ubahBelakang(nama, nim);
```

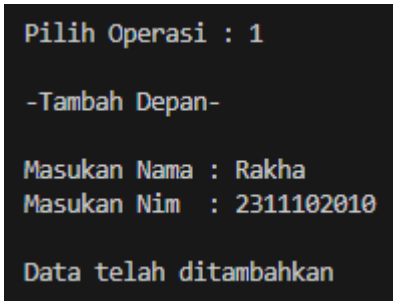
```
            cout << "\nData telah diubah " << endl;
            break;
        }
        case 6:
        {
            string nama;
            string nim;
            int posisi;
            cout << "\n-Ubah Tengah-\n\n";
            cout << "Masukan Nama   : ";
            cin >> nama;
            cout << "Masukan Nim    : ";
            cin >> nim;
            cout << "Masukan Posisi : ";
            cin >> posisi;
            ubahTengah(nama, nim, posisi);
            cout << "\nData telah diubah " << endl;
            break;
        }
        case 7:
            hapusDepan();
            cout << "\nData telah dihapus " << endl;
            break;
        case 8:
            hapusBelakang();
            cout << "\nData telah dihapus " << endl;
            break;
        case 9:
        {
            int posisi;
            cout << "\n-Hapus Tengah-\n\nMasukkan posisi : ";
            cin >> posisi;
            hapusTengah(posisi);
            break;
```

```
        }
        case 10:
            clearList();
            break;
        case 11:
            tampil();
            break;
        case 0:
            return 0;
        default:
            cout << "Pilihan tidak valid!" << endl;
        }
    }
    return 0;
}
```

Screenshoot program

2. Input data





3. 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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 11

DATA MAHASISWA

Nama            NIM
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Denis           23300005
Anis            23300008
Bowo            23300015
Gahar           23300040
Udin            23300048
Ucok            23300050
Budi            23300099
```

4. 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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 3

-Tambah Tengah-

Masukan Nama   : Wati
Masukan Nim    : 2330004
Masukan Posisi : 4

Data telah ditambahkan
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Wati            2330004
Denis           23300005
Anis            23300008
Bowo            23300015
Gahar           23300040
Udin            23300048
Ucok            23300050
Budi            23300099
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 9

-Hapus Tengah-

Masukkan posisi : 5
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Udin            23300048
Ucok            23300050
Budi            23300099
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 1

-Tambah Depan-

Masukan Nama : Owi
Masukan Nim  : 2330000

Data telah ditambahkan
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Owi             2330000
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Udin            23300048
Ucok            23300050
Budi            23300099
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 2

-Tambah Belakang-

Masukan Nama : David
Masukan Nim  : 23300100

Data telah ditambahkan
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Owi             2330000
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Udin            23300048
Ucok            23300050
Budi            23300099
David           23300100
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 6

-Ubah Tengah-

Masukan Nama   : Idin
Masukan Nim    : 23300045
Masukan Posisi : 9

Data telah diubah
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Owi             2330000
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Idin            23300045
Ucok            23300050
Budi            23300099
David           23300100
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 5

-Ubah Belakang-

Masukan Nama : Lucy
Masukan Nim  : 23300101

Data telah diubah
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Owi             2330000
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Idin            23300045
Ucok            23300050
Budi            23300099
Lucy            23300101
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 7

Data telah dihapus
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Jawad           23300001
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Idin            23300045
Ucok            23300050
Budi            23300099
Lucy            23300101
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 4

-Ubah Depan-

Masukan Nama : Bagas
Masukan Nim  : 2330002

Data telah diubah
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Bagas           2330002
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Idin            23300045
Ucok            23300050
Budi            23300099
Lucy            23300101
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 8

Data telah dihapus
```

```
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 : 11

DATA MAHASISWA

Nama            NIM
Bagas           2330002
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Idin            23300045
Ucok            23300050
Budi            23300099
```

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 Tengah
10. Hapus List
11. TAMPILKAN
0. KELUAR

Pilih Operasi : 11

DATA MAHASISWA

Nama            NIM
Bagas           2330002
Rakha           2311102010
Farrel          23300003
Wati            2330004
Anis            23300008
Bowo            23300015
Gahar           23300040
Idin            23300045
Ucok            23300050
Budi            23300099
```

Deskripsi program

Program ini menggunakan struktur data dari Single Linked List non-circular. Program ini dibuat untuk menyimpan data dan alamat dari setiap node. Inputan program ini berupa informasi Mahasiswa yang terdiri dari Nama dan NIM (Nomor Induk Mahasiswa). Dalam Program ini terdapat beberapa operasi yaitu, Menambahkan Node, Menghapus Node, Mengubah Node, dan Menampilkan data-data.

BAB IV

KESIMPULAN

Linked list adalah strukur data linier berbentuk rantai simpul di mana setiap simpul menyimpan 2 item, yaitu nilai data dan pointer ke simpul elemen berikutnya. Berbeda dengan array, elemen linked list tidak ditempatkan dalam alamat memori yang berdekatan melainkan elemen ditautkan menggunakan pointer.

Single linked list circular adalah Single Linked List yang pointer nextnya menunjuk pada dirinya sendiri., double linked list memiliki dua pointer yang menghubungkan node secara berurutan dan dua arah. Meskipun fitur tambahan ini memungkinkan lebih fleksibilitas saat menjalankan operasi seperti menambah atau menghapus node dengan mudah, ini juga memerlukan memori tambahan untuk menyimpan kedua pointer tersebut.

Perbedaan antara circular dengan non circular adalah jika non circular pada node terakhir semulanya menunjuk ke NULL. Jika circular terletak pada node terakhir yang semulanya menunjuk ke kepala atau head.

DAFTAR PUSTAKA

Asisten Praktikum. (2024). MODUL 4 LINKED LIST CIRCULAR DAN NON CIRCULAR, Learning Managament System