

**LAPORAN PRAKTIKUM STRUKTUR DATA
DAN ALGORITMA MODUL 4
“Linked List Circular Dan Noncircular”**



DISUSUN OLEH:
Farrell Edric Kelvianto
2311102079
S1 IF-11-B

DOSEN:

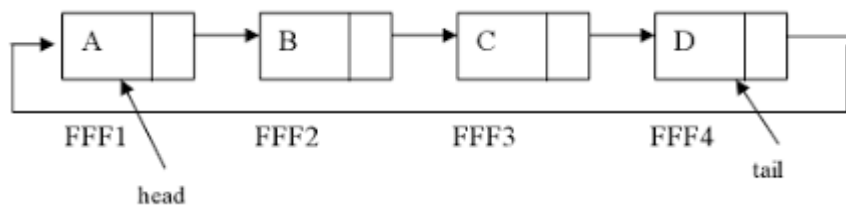
Pak Wahyu Andi Saputra, S.Pd., M.Eng.

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

A. Dasar Teori

a. Linked List Non Circular

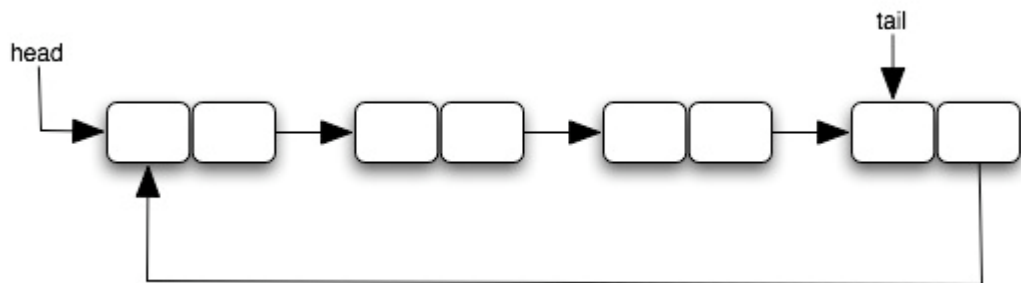
Linked list non circular merupakan linked list dengan node pertama biasa disebut dengan head dan node terakhir yaitu tail. Pointer dibagian akhir atau tail ini bernilai Null atau kosong dibagian akhir ini. Ini adalah cara kerja dari Linked List Non Circular



Beginilah cara kerja dari Linked List Non Circular hanya saja di sebelah D dia bersifat null dikarenakan tidak diisi dengan nilai itu sendiri. Ini adalah gambaran untuk stack nanti.

b. Linked List Circular

Linked list circular adalah linked list yang tidak memiliki akhir karena node tidak bernilai Null, namun tetapi dia terhubung dengan node pertama yaitu head itu sendiri. Jadi gambarannya adalah dia tail dan head itu saling berhubungan jadi bisa berulang – ulang untuk menyimpan data. Jadi sebagai berikut gambaran dari Circular Linked List.



Guided

Guided 1

```
#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;
    }
}

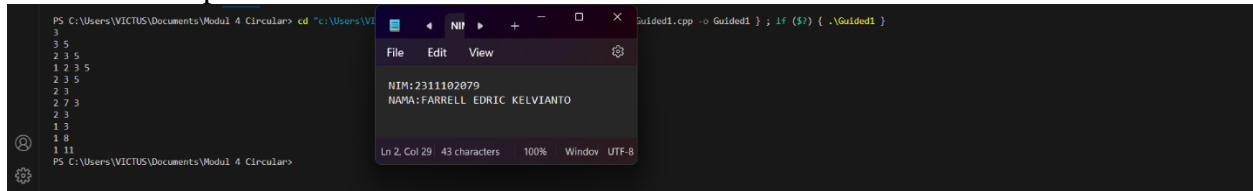
int main() {
    init();
    insertDepan(3);
    tampil();
    insertBelakang(5);
    tampil();
    insertDepan(2);
    tampil();
    insertDepan(1);
    tampil();
    hapusDepan();
    tampil();
    hapusBelakang();
    tampil();
    insertTengah(7, 2);
    tampil();
    hapusTengah(2);
    tampil();
    ubahDepan(1);
    tampil();
    ubahBelakang(8);
    tampil();
    ubahTengah(11, 2);
    tampil();

    return 0;
}

```

```
}
}
```

Screenshot Output



The screenshot shows a terminal window on the left displaying the output of a program. The output consists of a vertical list of numbers: 3, 3, 5, 2, 3, 5, 1, 2, 3, 5, 2, 3, 5, 2, 3, 2, 3, 1, 3, 1, 8, 1, 11. To the right of the terminal is a code editor window titled 'Guided1.cpp'. The editor shows a menu bar with 'File', 'Edit', and 'View'. Below the menu bar, the text 'NIM: 2311102079' and 'NAMA: FARRELL EDRIC KELVIAN TO' is visible. At the bottom of the editor, it says 'Ln 2, Col 29 43 characters 100% Window UTF-8'. The code in the editor is: `Guided1.cpp : 0 Guided1 ; if ($?) { .\Guided1 }`

Deskripsi Program

Cara kerja program ini adalah seperti Modul 3 kemarin hanya single linked list biasa dan nampak di bagian struct hanya terdapat next saja. Jadi untuk pertama terdapat sebuah struct yang berisikan untuk data tersebut dan next untuk berlanjut ke node selanjutnya. Terdapat inisialisasi dari head dan tail untuk menyatakan bahwa head dan tail itu adalah kosong. Terdapat deklarasi kembali ada isEmpty untuk mereturn jika head itu adalah Null (Opsional bisa menggunakan secara langsung bahwa head itu adalah null itu sendiri menggunakan percabangan). Lalu ada insertDepan ini adalah untuk menambahkan dari depan yaitu head itu sendiri jadi konsepnya adalah dia akan bergerak kebelakang setelah mendapatkan data baru sehingga terbentuklah tail. Lalu di selanjutnya ada insertBelakang disini untuk menambahkan dibagian tail nah disini adalah sebagai bentuk implementasi dari circular linked list itu sendiri dan termasuk insertTengah. HapusDepan berfungsi node awal yaitu head itu sendiri dan HapusBelakang untuk menghapus bagian belakang atau tail. Untuk hapus tengah dia berbeda dari kedua yang dijabarkan tadi untuk hapusTengah seharusnya sama saja bentuk implementasi dari circular linked list termasuk tail tadi hanya saja dia lebih dinamis untuk menghapus jadi intinya tidak hanya menghapus tengah pada sesungguhnya dapat menghapus semua antrian atau proses dalam linked list itu sendiri. Lalu ada sebuah tampil untuk menampilkan semua data dari head dan tail dan untuk menampilkannya adalah menggunakan fungsi perulangan untuk menampilkan semua datanya didalam.

Guided 2

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

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

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

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

int isEmpty() {
```



```

        return head == NULL;
    }

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

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

    void insertDepan(string data) {
        buatNode(data);
        if (isEmpty()) {
            head = baru;
            tail = head;
            baru->next = head;
        } else {
            while (tail->next != head) {
                tail = tail->next;
            }
            baru->next = head;
            head = baru;
            tail->next = head;
        }
    }

    void insertBelakang(string data) {
        buatNode(data);
        if (isEmpty()) {
            head = baru;
            tail = head;
            baru->next = head;
        } else {
            while (tail->next != head) {
                tail = tail->next;
            }
        }
    }

```

```

        tail->next = baru;
        baru->next = head;
    }
}

void insertTengah(string data, int posisi) {
    if (isEmpty()) {
        head = baru;
        tail = head;
        baru->next = head;
    } else {
        baru->data = data;
        int nomor = 1;
        bantu = head;
        while (nomor < posisi - 1) {
            bantu = bantu->next;
            nomor++;
        }
        baru->next = bantu->next;
        bantu->next = baru;
    }
}

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

```

```

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

```

```

void hapusTengah(int posisi) {
    if (!isEmpty()) {
        int nomor = 1;
        bantu = head;
        while (nomor < posisi - 1) {
            bantu = bantu->next;
            nomor++;
        }
        hapus = bantu->next;
        bantu->next = hapus->next;
        delete hapus;
    } else {
        cout << "List masih kosong!" << endl;
    }
}

```

```

void clearList() {
    if (head != NULL) {
        hapus = head->next;
        while (hapus != head) {
            bantu = hapus->next;

```

```

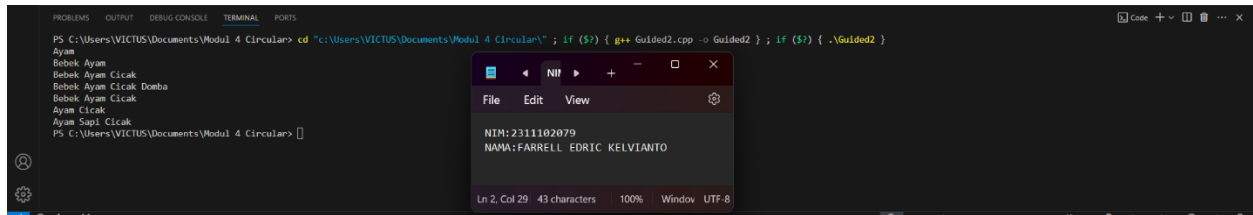
        delete hapus;
        hapus = bantu;
    }
    delete head;
    head = NULL;
}
cout << "List berhasil terhapus!" << endl;
}

void tampil() {
    if (!isEmpty()) {
        tail = head;
        do {
            cout << tail->data << " ";
            tail = tail->next;
        } while (tail != head);
        cout << endl;
    } else {
        cout << "List masih kosong!" << endl;
    }
}

int main() {
    init();
    insertDepan("Ayam");
    tampil();
    insertDepan("Bebek");
    tampil();
    insertBelakang("Cicak");
    tampil();
    insertBelakang("Domba");
    tampil();
    hapusBelakang();
    tampil();
    hapusDepan();
    tampil();
    insertTengah("Sapi", 2);
    tampil();
    hapusTengah(2);
    tampil();
    return 0;
}

```

Screenshot Output



Deskripsi Program

Untuk program ini secara fungsi sama seperti program sebelumnya dan ini adalah circular linked list karena terdapat tail=head dan untuk pada Guided 2 ini adalah Circular Single Linked List karena mempunyai ciri – ciri yang memiliki satu arah dan tail dan head terhubung, hanya saja beda dibagian bagian deklarasi struct node terdapat deklarasi baru , bantu dan hapus ini hanya untuk mempersingkat codingan untuk keseluruhan tanpa mendeklarasi ulang disetiap prosedurnya. Fungsi dimulai dari buatnode berfungsi untuk membuat node baru. Buat node ini kita hiraukan saja karena ini untuk membantu setiap codingan didalam prosedurnya nanti. Nah untuk bagian insertDepan kita dapat memanggil buatNode yang berisikan sebuah data untuk mengisi nilainya tersebut dan fungsinya didalam insertDepan dan insertBelakang berfungsi sama namun kedua nya sama – sama berfungsi menambahkan hanya saja untuk insertDepan untuk menambahkan dibagian head dan untuk InsertBelakang untuk mengisi tail. Untuk hapusDepan ini hampir serupa dengan metode searching namun ini hanya untuk menghapus depan saja termasuk belakang hanya untuk menghapus belakang untuk secara deskripsi program nya dia mencari menggunakan while setelah dia ketemu di buntutnya atau tailnya akan dihapus jika dia tidak lanjut lagi atau tidak dihapus maka dia akan lanjut terus sampai ke ujung buntut untuk dihapus. HapusTengah ini untuk menghapus tetapi lebih dinamis dapat menghapus posisi manapun maupun head dan tail sekalipun untuk cara kerjanya dia seperti searching dia akan mencari sebuah posisi jika dia ketemu maka dia akan berhenti dan menempatkan data tersebut kedalam posisinya. Tampil berfungsi menampilkan semua isi dari data yang telah tersimpan didalam node – node tadi.

B. Tugas Unguided 1

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

struct Node {
    double NIM;
    string namaMhs;
    Node* next;
```

```

};

Node* head = nullptr;
Node* tail = nullptr;

void tambahDepan(string mhs, double NIM) {
    Node* baru = new Node;
    baru->namaMhs = mhs;
    baru->NIM = NIM;
    baru->next = nullptr;

    if (head == nullptr) {
        head = tail = baru;
    } else {
        baru->next = head;
        head = baru;
    }
}

void tambahBelakang(string mhs, double NIM) {
    Node* baru = new Node;
    baru->namaMhs = mhs;
    baru->NIM = NIM;
    baru->next = nullptr;

    if (head == nullptr) {
        head = tail = baru;
    } else {
        tail->next = baru;
        tail = baru;
    }
}

void tambahTengah(string mhs, double NIM, int posisi) {
    if (posisi <= 0) {
        cout << "Posisi tidak valid." << endl;
        return;
    }

    Node* baru = new Node;

```

```

baru->namaMhs = mhs;
baru->NIM = NIM;
baru->next = nullptr;

if (posisi == 1) {
    tambahDepan(mhs, NIM);
    return;
}

Node* current = head;
Node* prev = nullptr;
int count = 1;

while (current != nullptr && count < posisi) {
    prev = current;
    current = current->next;
    count++;
}

if (current == nullptr && count != posisi) {
    cout << "Posisi tidak valid." << endl;
    return;
}

prev->next = baru;
baru->next = current;
}

void ubahDepan(string mhs, double NIM) {
    if (head == nullptr) {
        cout << "List kosong." << endl;
        return;
    }
    head->namaMhs = mhs;
    head->NIM = NIM;
    cout << "Data mahasiswa pertama berhasil diubah." << endl;
}

void ubahBelakang(string mhs, double NIM) {
    if (head == nullptr) {
        cout << "List kosong." << endl;

```

```

        return;
    }
    Node* current = tail;

    tail->namaMhs = mhs;
    tail->NIM = NIM;
    cout << "Data Berhasil diubah" << endl;
}

void ubahTengah(double nim, string mhsBaru, double nimBaru) {
    if (head == nullptr) {
        cout << "List kosong." << endl;
        return;
    }

    Node* current = head;

    while (current != nullptr && current->NIM != nim) {
        current = current->next;
    }
    if (current == nullptr) {
        cout << "Data dengan NIM tersebut tidak ditemukan." << endl;
        return;
    }

    current->namaMhs = mhsBaru;
    current->NIM = nimBaru;
    cout << "Data mahasiswa dengan NIM " << nim << " berhasil diubah." << endl;
}

void hapusDepan() {
    if (head == nullptr) {
        cout << "List kosong." << endl;
        return;
    }

    Node* temp = head;
    head = head->next;
    delete temp;
}

void hapusBelakang() {

```



```

    if (head == nullptr) {
        cout << "List kosong." << endl;
        return;
    }

    if (head == tail) {
        delete head;
        head = tail = nullptr;
        cout << "Data mahasiswa terakhir berhasil dihapus." << endl;
        return;
    }

    Node* current = head;
    while (current->next != tail) {
        current = current->next;
    }

    delete tail;
    tail = current;
    tail->next = nullptr;
}

void hapusTengah(double nim) {
    if (head == nullptr) {
        cout << "List kosong." << endl;
        return;
    }

    Node* current = head;
    Node* prev = nullptr;

    while (current != nullptr && current->NIM != nim) {
        prev = current;
        current = current->next;
    }

    if (current == nullptr) {
        cout << "Data dengan NIM tersebut tidak ada dalam daftar" << endl;
        return;
    }

    if (prev == nullptr) {

```

```

        head = head->next;
        delete current;
        cout << "Data mahasiswa dengan NIM " << setprecision(0)<< fixed << nim
<< " Dengan atas nama" << current->namaMhs << " berhasil dihapus." << endl;
        return;
    }

    if (current == tail) {
        tail = prev;
    }

    prev->next = current->next;
    delete current;
    cout << "Data mahasiswa dengan NIM " << setprecision(0)<< fixed << nim << "
Dengan atas nama " << current->namaMhs << " berhasil dihapus." << endl;
}

void hapusList() {
    while (head != nullptr) {
        Node* temp = head;
        head = head->next;
        delete temp;
    }
    tail = nullptr;
    cout << "Semua data mahasiswa berhasil dihapus." << endl;
}

void tampilkan() {
    if (head == nullptr) {
        cout << "List kosong." << endl;
        return;
    }

    int maxNamaLength = 0;
    Node* current = head;
    while (current != nullptr) {
        int namaLength = current->namaMhs.length();
        if (namaLength > maxNamaLength) {
            maxNamaLength = namaLength;

```

```

    }
    current = current->next;
}

cout << "\n-----" << endl;
cout << "| No |          NIM          | " << setw(maxNamaLength + 4) << "Nama
Mahasiswa" << " |" << endl;
cout << "-----" << endl;

int count = 1;
current = head;
while (current != nullptr) {
    cout << "| " << setw(2) << count << " | " << setw(17) << fixed <<
    setprecision(0) << current->NIM
    << " | " << setw(maxNamaLength + 4) << current->namaMhs << " |" <<
    endl;
    current = current->next;
    count++;
}
cout << "-----" << endl;
}

int main() {
    tambahDepan ("Jawad",23300001);
    tambahBelakang("Farrel",23300003);
    tambahBelakang("Denis",23300005);
    tambahBelakang("Anis",23300008);
    tambahBelakang("Bowo",23300015);
    tambahBelakang("Gahar",23300040);
    tambahBelakang("Udin",23300048);
    tambahBelakang("Ucok",23300050);
    tambahBelakang("Budi",23300099);
    tambahTengah("Farrell Edric Kelvianto",2311102079,2);

    cout << "PROGRAM SINGLE LINKED LIST NON CIRCULAR" << endl;
    int choice;
    while (true) {

        cout << "\nMenu:" << endl;
        cout << "1. Tambah Data di Depan" << endl;

```

```
cout << "2. Tambah Data di Belakang" << endl;
cout << "3. Tambah Data di Tengah" << endl;
cout << "4. Ubah Data Pertama" << endl;
cout << "5. Ubah Data Terakhir" << endl;
cout << "6. Ubah Data di Tengah" << endl;
cout << "7. Hapus Data Pertama" << endl;
cout << "8. Hapus Data Terakhir" << endl;
cout << "9. Hapus Data di Tengah" << endl;
cout << "10. Hapus Semua Data" << endl;
cout << "11. Tampilkan Data" << endl;
cout << "0. Keluar" << endl;
```

```
cout << "Masukkan pilihan: ";
cin >> choice;
```

```
switch (choice) {
    case 1: {
        string nama;
        double nim;
        cout << "-Tambah Depan" << endl;
        cout << "Masukkan nama mahasiswa: ";
        cin >> nama;
        cout << "Masukkan NIM mahasiswa: ";
        cin >> nim;
        tambahDepan(nama, nim);
        cout << "Data telah ditambahkan";
        break;
    }
    case 2: {
        string nama;
        double nim;
        cout << "-Tambah Belakang" << endl;
        cout << "Masukkan nama mahasiswa: ";
        cin >> nama;
        cout << "Masukkan NIM mahasiswa: ";
        cin >> nim;
        tambahBelakang(nama, nim);
        cout << "Data telah ditambahkan";
        break;
    }
    case 3: {
        string nama;
        double nim;
        int posisi;
```

```

        cout << "-Tambah Tengah" << endl;
        cout << "Masukkan nama mahasiswa: ";
        cin >> nama;
        cout << "Masukkan NIM mahasiswa: ";
        cin >> nim;
        cout << "Masukkan posisi: ";
        cin >> posisi;
        tambahTengah(nama, nim, posisi);
        cout << "Data telah ditambahkan";
        break;
    }
    case 4: {
        string nama;
        double nim;
        cout << "-Ubah Depan" << endl;
        cout << "Masukkan nama baru untuk mahasiswa pertama: ";
        cin >> nama;
        cout << "Masukkan NIM baru untuk mahasiswa pertama: ";
        cin >> nim;
        ubahDepan(nama, nim);
        cout << "Data Telah Diubah" << endl;
        break;
    }
    case 5: {
        string nama;
        double nim;
        cout << "-Ubah Belakang" << endl;
        cout << "Masukkan nama baru untuk mahasiswa terakhir: ";
        cin >> nama;
        cout << "Masukkan NIM baru untuk mahasiswa terakhir: ";
        cin >> nim;
        ubahBelakang(nama, nim);

        break;
    }
    case 6: {
        double NIM;
        string mhs;
        double nimBar;
        cout << "-Mengubah Data Tengah" << endl;
        cout << "Masukkan NIM lama untuk diubah: ";
        cin >> NIM;
        cout << "Masukkan Nama Data Mahasiswa Baru: ";
        cin >> mhs;
        cout << "Masukkan NIM baru: ";

```

```

        cin >> nimBar;
        ubahTengah(NIM, mhs, nimBar);
        break;
    }
    case 7:
        hapusDepan();
        cout << "Data Depan Telah Dihapus";
        break;
    case 8:
        hapusBelakang();
        cout << "Data Belakang Telah Dihapus";
        break;
    case 9: {
        double NIM;
        cout << "-Hapus Tengah" << endl;
        cout << "Masukkan NIM mahasiswa yang ingin dihapus: ";
        cin >> NIM;
        hapusTengah(NIM);
        break;
    }
    case 10:
        hapusList();
        cout << "Semua data telah dihapus " << endl;
        break;
    case 11:
        tampilkan();
        break;
    case 0:
        cout << "=====Program Was Returned======" << endl;
        return 0;
    default:
        cout << "Anda salah input,silahkan pilih kembali." << endl;
    }
}
return 0;
}

```

Screenshot Output

Soal No. 1 Membuat menu untuk menambahkan ,mengubah dan melihat nama dan NIM mahasiswa,sebagai berikut outputnya :

- Tampilan Menu.

```
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 5
-Ubah Belakang
Masukkan nama baru untuk mahasiswa terakhir: F
Masukkan NIM baru untuk mahasiswa terakhir: 23111000
Data Berhasil diubah
```

- Tampilan Operasi Tambah.

```
> OUTLINE 278
> TIMELINE 279
double nim;
cout << "-Tambah Depan" << endl;
```

```
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 3
-Tambah Tengah
Masukkan nama mahasiswa: Edric
Masukkan NIM mahasiswa: 2311102079
Masukkan posisi: 2
Data telah ditambahkan
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
```

```
268 cout << "11. Tampilkan Data" << endl;
269 cout << "0. Keluar" << endl;
270
271 cout << "Masukkan pilihan: ";
272 cin >> choice;
273
274
275 switch (choice) {
276 case 1: {
277     string nama;
278     double nim;
279     cout << "-Tambah Depan" << endl;
```

```
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 1
-Tambah Depan
Masukkan nama mahasiswa: F
Masukkan NIM mahasiswa: 2311102079
Data telah ditambahkan
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
```

- Tampilan Operasi Hapus.

```
> OUTLINE 354
> TIMELINE 360
return B;
```

```
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 8
Data Belakang telah dihapus
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
```

```
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 9
-Hapus Tengah
Masukkan NIM mahasiswa yang ingin dihapus: 2311102079
Data mahasiswa dengan NIM 2311102079 Dengan atas nama Farrell Edric Kelvianto berhasil dihapus.
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
```

- Tampilan Operasi Ubah.

```

> TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 5
-Ubah Belakang
Masukkan nama baru untuk mahasiswa terakhir: F
Masukkan NIM baru untuk mahasiswa terakhir: 23111000
Data Berhasil diubah

Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah

```

NIM: 2311102079
NAMA: FARRELL EDRIC KELVIAN TO

- Tampilan Operasi Tampil Data.

No	NIM	Nama Mahasiswa
1	23300001	Jasad
2	2311102079	Farrell Edric Kelvianto
3	23300003	Farrel
4	23300005	Denis
5	23300008	Anis
6	23300015	Bowo
7	23300040	Gahar
8	23300048	Udin
9	23300050	Ukok
10	23111000	Budi

Soal No. 2 Tampilan Output Tabel Nama – Nama yang diinputkan :

No	NIM	Nama Mahasiswa
1	23300001	Jasad
2	2311102079	Farrell Edric Kelvianto
3	23300003	Farrel
4	23300005	Denis
5	23300008	Anis
6	23300015	Bowo
7	23300040	Gahar
8	23300048	Udin
9	23300050	Ukok
10	23111000	Budi

Soal No .3 Melakukan perintah :

- Menambahkan Data Diantara Farrell Dan Denis :

```

Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 3
-Tambah Tengah
Masukkan nama mahasiswa: Meti
Masukkan NIM mahasiswa: 23300004
Masukkan posisi: 4
Data telah ditambahkan
Menu:

```

NIM: 2311102079
NAMA: FARRELL EDRIC KELVIAN TO

- Menghapus Data Denis :

```

Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 9
-Hapus Tengah
Masukkan NIM mahasiswa yang ingin dihapus: 23300005
Data mahasiswa dengan NIM 23300005 Dengan atas nama Denis berhasil dihapus.

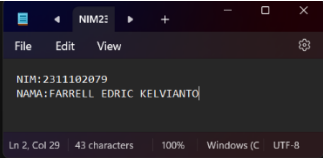
```

NIM: 2311102079
NAMA: FARRELL EDRIC KELVIAN TO

- Menambahkan Data Di Awal :



```
Masukkan NIM mahasiswa yang ingin dihapus: 23300005
Data mahasiswa dengan NIM 23300005 Dengan atas nama Denis berhasil dihapus.

Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 1
-Tambah Depan
Masukkan nama mahasiswa: Odi
Masukkan NIM mahasiswa: 23300000
Data telah ditambahkan
Menu:
```



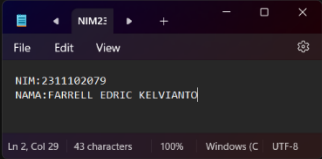
d. Menambahkan data di akhir :

```
Masukkan nama mahasiswa: Odi
Masukkan NIM mahasiswa: 23300000
Data telah ditambahkan
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 2
-Tambah Belakang
Masukkan nama mahasiswa: David
Masukkan NIM mahasiswa: 23300100
Data telah ditambahkan
Menu:
```



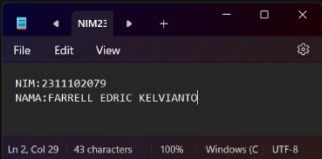
e. Ubah data udin menjadi data berikut :

```
Masukkan nama mahasiswa: David
Masukkan NIM mahasiswa: 23300100
Data telah ditambahkan
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 6
-Mengubah Data Tengah
Masukkan NIM lama untuk diubah: 23300048
Masukkan Nama Data Mahasiswa Baru: Idin
Masukkan NIM baru: 23300045
Data mahasiswa dengan NIM 23300048 berhasil diubah.
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
```



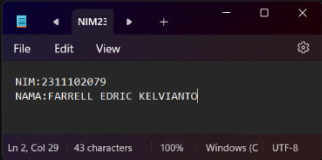
f. Ubah data terakhir menjadi berikut :

```
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 5
-Ubah Belakang
Masukkan nama baru untuk mahasiswa terakhir: Lucy
Masukkan NIM baru untuk mahasiswa terakhir: 23300101
Data Berhasil diubah
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
```



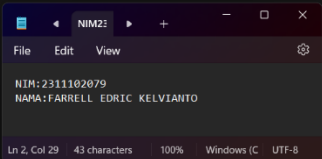
g. Hapus Data awal :

```
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 7
Data Depan Telah Dihapus
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
```



h. Ubah data awal menjadi berikut :

```
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
Masukkan pilihan: 4
-Ubah Depan
Masukkan nama baru untuk mahasiswa pertama: Bagas
Masukkan NIM baru untuk mahasiswa pertama: 2330002
Data mahasiswa pertama berhasil diubah.
Data Telah Diubah
Menu:
1. Tambah Data di Depan
2. Tambah Data di Belakang
3. Tambah Data di Tengah
4. Ubah Data Pertama
5. Ubah Data Terakhir
6. Ubah Data di Tengah
7. Hapus Data Pertama
8. Hapus Data Terakhir
9. Hapus Data di Tengah
10. Hapus Semua Data
11. Tampilkan Data
0. Keluar
```



i. Hapus data akhir :

j. Menampilkan seluruh data :

No	NIM	Nama Mahasiswa
1	23300002	Bagas
2	2311102079	Farrell Edric Kelvianto
3	23300003	Farrel
4	23300004	Neri
5	23300008	Anis
6	23300015	Bowo
7	23300049	Gahar
8	23300045	Edin
9	23300050	Ucok
10	23300099	Budi

Deskripsi Program

Cara kerja program ini adalah

- tambahDepan () berfungsi untuk menambahkan node awal atau head.
- tambahBelakang () berfungsi untuk menambahkan node pada belakang atau tail.
- tambahTengah () ini berfungsi untuk menambahkan pada posisi yang ditentukan oleh user dengan cara memanfaatkan while untuk mencari posisi yang sesuai.
- ubahDepan () ini berfungsi untuk mengubah dibagian depan saja dan terdapat di source code kalau dia hanya mengarahkan pada head saja untuk mengubah depan.
- ubahBelakang () ini berfungsi mengubah bagian tail atau node bagian belakang saja.
- ubahTengah () ini berfungsi mengubah semua bagian tetapi menggunakan NIM untuk mengubah data. Maksudnya adalah user menginputkan NIM mahasiswa yang lama menjadi data yang baru.
- hapusDepan () ini berfungsi untuk menghapus bagian head saja.
- hapusBelakang () berfungsi untuk menghapus bagian tail atau belakang saja.
- hapusTengah () ini berfungsi menghapus data dengan posisi tertentu dengan cara user menginputkan NIM yang ingin dihapus.
- hapusList () berfungsi untuk menghapus semua data/node.
- Tampilkan () berfungsi untuk menampilkan hasil data yang user telah inputkan.

Kesimpulan

Kesimpulannya adalah sebenarnya single linked list ini tidak jauh beda dengan single circular linked list hanya saja berbeda jika single linked list head dan tail nya tidak saling terhubung dan circular single linked list antara head dan tailnya terhubung. Dan untuk cara kerjanya sama untuk mengimplementasikan stack dan queue nanti.

C. Refrensi

Asisten Praktikum. "Modul 4 Linked List Circular Dan Non Circular". Learning Management System. 2024

