LAPORAN PRAKTIKUM STRUKTUR DATA DAN ALGORITMA

MODUL IV
LINKED LIST CIRCULAR DAN NON CIRCULAR

Disusun Oleh :
Muhammad Arsyad Zaidan (2311102058)

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

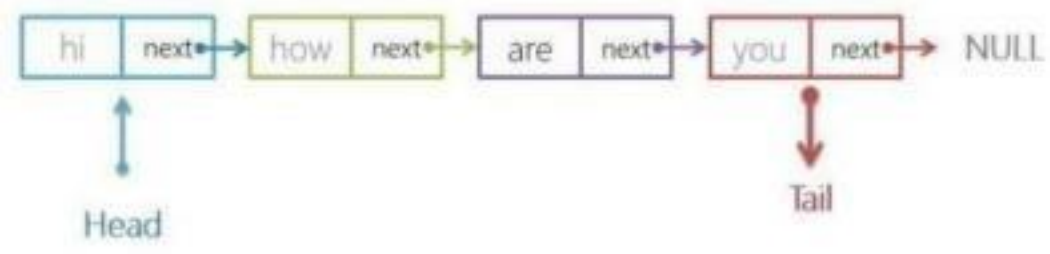
PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

B. 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. *Linkedlist non circular* dapat digambarkan sebagai berikut.



Gambar 1 *Single Linked List Non Circular*

Operasi Linked List Non Circular

1. Deklarasi Simpul (Node)

```
{
    int data;
    Node* next;
};
```

2. Membuat dan Menginisialisasi Pointer Head dan Tail

```
node *head, *tail;

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

3. Pengecekan Kondisi Linked List

```
bool isEmpty() {
    if (head == NULL && tail == NULL) {
        return true;
    } else {
        return false;
    }
}
```

4. Penambahan Simpul (Node)

```
void insertBelakang(string dataUser) {
    if (isEmpty()) {
        node *baru = new node;
        baru->data = dataUser;
        head = baru;
        tail = baru;
        baru->next = NULL;
```

```
    } else {
        node *baru = new node;
        baru->data = dataUser;
        baru->next = NULL;
        tail->next = baru;
        tail = baru;
    }
}
```

5. Penghapusan Simpul (Node)

```
if (isEmpty()) {
    cout << "List kosong!" << endl;
} else {
    node *helper;
    helper = head;
    if (head == tail) {
        head = NULL;
        tail = NULL;
        delete helper;
    } else {
        head = head->next;
        helper->next = NULL;
        delete helper;
    }
}
```

6. Tampil Data Linked List

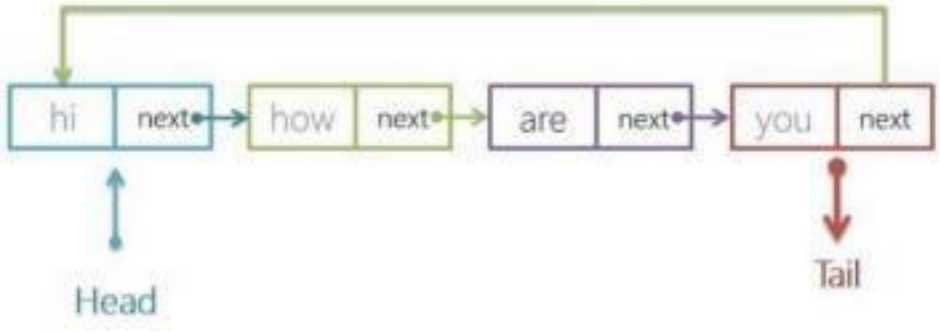
```
void tampil() {
    if (isEmpty()) {
        cout << "List kosong!" << endl;
    } else {
        node *helper;
        helper = head;
        while (helper != NULL) {
            cout << helper->data << ends;
            helper = helper->next;
        }
    }
}
```

2. Linked List Circular

Linked list circular merupakan *linked list* yang tidak memiliki akhir karena node terakhir (tail) tidak bernilai '***NULL***', tetapi terhubung dengan node pertama (head). Saat menggunakan *linked list circular* kita membutuhkan *dummy node* atau node pengecoh yang biasanya dinamakan dengan node *current* supaya program dapat berhenti menghitung data ketika node *current* mencapai node pertama (head).

Linked list circular dapat digunakan untuk menyimpan data yang perlu diakses secara berulang, seperti daftar putar lagu, daftar pesan dalam antrian, atau penggunaan

memori berulang dalam suatu aplikasi. *Linked list circular* dapat digambarkan sebagai berikut.



Gambar 2 *Single Linked List Circular*

Operasi Pada Linked List Circular

1. Deklarasi Node (Simpul)

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

2. Membuat dan Menginialisasi Pointer Head dan Tail

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

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

3. Pengecekan Kondisi Linked List

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

4. Pembuatan Simpul (Node)

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

5. Penambahan Simpul (Node)

```
// Tambah Depan
void insertDepan(string data) {
    // Buat Node baru
    buatNode(data);
```

```
    if (isEmpty() == 1) {
        head = baru;
        tail = head;
        baru->next = head;
    } else {
        while (tail->next != head) {
            tail = tail->next;
        }
        baru->next = head;
        head = baru;
        tail->next = head;
    }
}
```

6. Penghapusan Simpul (Node)

```
void hapusBelakang() {
    if (isEmpty() == 0) {
        hapus = head;
        tail = head;

        if (hapus->next == head) {
            head = NULL;
            tail = NULL;

            delete hapus;
        } else {
            while (hapus->next != head) {
                hapus = hapus->next;
            }

            while (tail->next != hapus) {
                tail = tail->next;
            }

            tail->next = head;
            hapus->next = NULL;

            delete hapus;
        }
    }
}
```

7. Menampilkan Data Linked List

```
void tampil() {
    if (isEmpty() == 0) {
        tail = head;
        do {
            cout << tail->data << ends;
            tail = tail->next;
        } while (tail != head);
        cout << endl;
    }
}
```

1. **Latihan Guided dan Unguided**

1) Guided

Guided 1 (Linked List Non Circular)

Source Code

```
#include <iostream>
using namespace std;
/// PROGRAM SINGLE LINKED LIST NON-CIRCULAR
// Deklarasi Struct Node
struct Node
{
    int data;
    Node *next;
};
Node *head;
Node *tail;
// Inisialisasi Node
void init()
{
    head = NULL;
    tail = NULL;
}
// Pengecekan
bool isEmpty()
{
    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;
        tail->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;
        tail->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 diluar 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;
        }
        else
        {
        }
    }
    else
    {
        delete hapus;
        head = tail =
            NULL;
        cout << "List 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;
        }
        else
        {
        }
    }
    else
    {
```

```
        delete hapus;
        head = tail = NULL;
        cout << "List kosong!" << endl;
    }
}
// Hapus Tengah
void hapusTengah(int posisi)
{
    Node *bantu, *hapus, *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 << "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
        {
            bantu = head;
            int nomor = 1;
            while (nomor < posisi)
            {
                bantu = bantu->next;
                nomor++;
            }
            bantu->data = data;
        }
    }
    else
    {
        cout << "List masih kosong!" << endl;
    }
}
// Ubah Belakang
void ubahBelakang(int data)
{
    if (isEmpty() == 0)
    {
        tail->data = data;
    }
    else
    {
        cout << "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 tampil()
{
```

```
    Node *bantu;
    bantu = head;
    if (isEmpty() == false)
    {
        while (bantu != NULL)
        {
            cout << bantu->data << ends;
            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;
}
```

Output Code

```
PS C:\Belajar coding\Strukdat\Learning Code\STRUKDAT (PRAKTEK)\circular2>  & 'c:\Users\ASUS
\.vscode\extensions\ms-vscode.cpptools-1.19.9-win32-x64\debugAdapters\bin\WindowsDebugLaunc
her.exe' '--stdin=Microsoft-MIEngine-In-f0c2wqzi.qmt' '--stdout=Microsoft-MIEngine-Out-2cv4
evkx.gan' '--stderr=Microsoft-MIEngine-Error-ggdsn0a0.xrk' '--pid=Microsoft-MIEngine-Pid-ss
pijetj.3rt' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
3
35
235
1235
235
23
273
23
13
18
111
PS C:\Belajar coding\Strukdat\Learning Code\STRUKDAT (PRAKTEK)\circular2>
```

Deskripsi Code

Single linked list pada pemrograman tersebut pada tiap node memiliki data pointer yang menunjukkan ke node berikutnya. Interaksi tersebut pada fungsi main yang memiliki beberapa fungsi untuk menambah, menghapus, mengubah dan menampilkan data pada pemrograman tersebut.

Guided 2 (Linked List Circular)

Source Code

```
#include <iostream>
using namespace std;
/// PROGRAM SINGLE LINKED LIST CIRCULAR
// Deklarasi Struct Node
struct Node
{
    string data;
    Node *next;
};
Node *head, *tail, *baru, *bantu, *hapus;
void init()
{
    head = NULL;
    tail = head;
}
// Pengecekan
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)
```

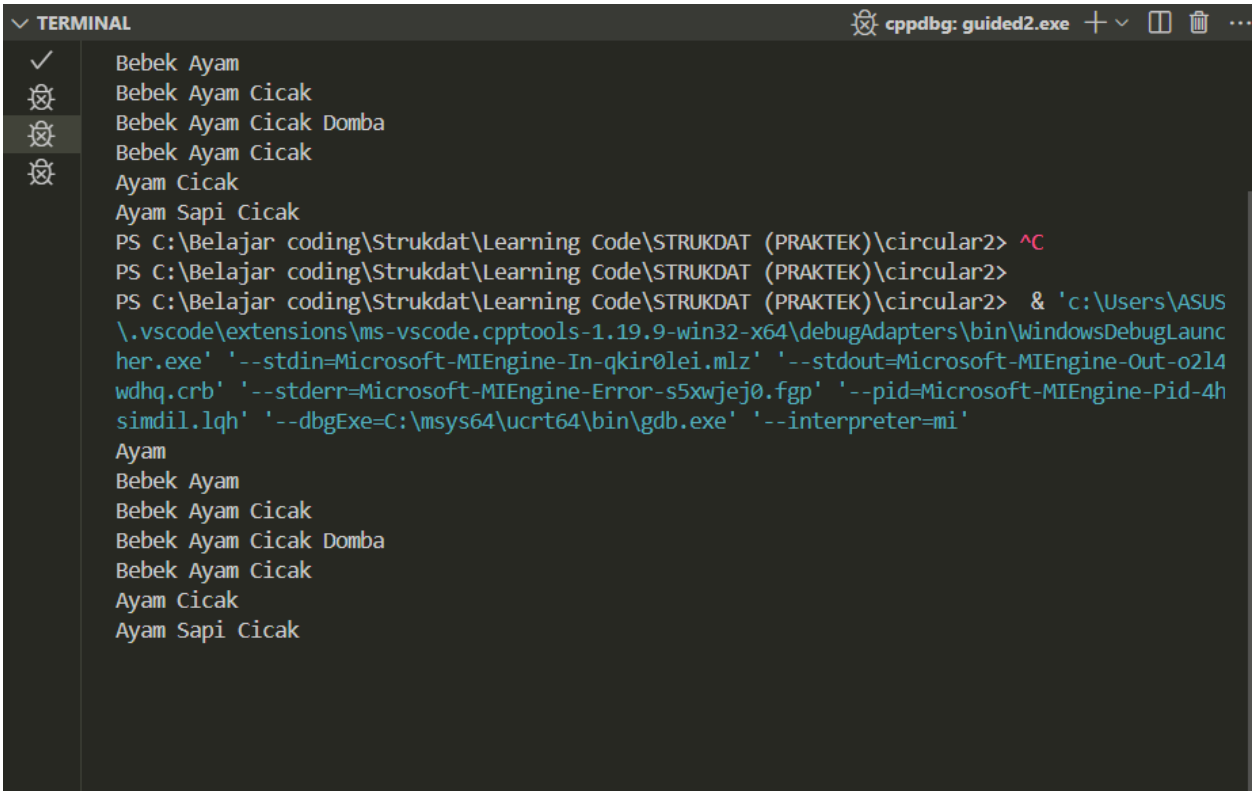
```
    {
        head = baru;
        tail = head;
        baru->next = head;
    }
    else
    {
        while (tail->next !=
               head)
        {
            tail = tail->next;
        }
        tail->next = baru;
        baru->next = head;
    }
}
// Tambah Tengah
void insertTengah(string 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 << " ";
            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;
}
```

Output Code



Deskripsi Code

Mengimplementasikan single linked list circular yang mana terdiri dari serangkaian node yang setiap nodenya memiliki dua bagian : data dan pointer ke node berikutnya dalam urutan.

2) Unguided

1. Buatlah program menu Single Linked List Non-Circular untuk menyimpan Nama dan NIM mahasiswa dengan menggunakan inputan dari user. Lakukan operasi berikut. 1. Buatlah menu untuk menambahkan, mengubah, menghapus, dan melihat Nama dan NIM mahasiswa, berikut contoh tampilan output dari nomor 1:

- Tampilan Menu :

```
PROGRAM SINGLE LINKED LIST NON-CIRCULAR

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

Pilih Operasi :
```

- Tampilan Operasi Tambah :

```
-Tambah Depan

Masukkan Nama :
Masukkan NIM :

Data telah ditambahkan
```

```
-Tambah Tengah

Masukkan Nama :
Masukkan NIM :
Masukkan Posisi :

Data telah ditambahkan
```

```
-Hapus Belakang

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

- Tampilan operasi ubah

```
-Ubah Belakang

Masukkan nama :
Masukkan NIM :

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

```
-Ubah Belakang

Masukkan nama :
Masukkan NIM :
Masukkan posisi :

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

Tampilan Operasi Data

```
DATA MAHASISWA

NAMA NIM
Nama1 NIM1
Nama2 NIM2
```

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

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

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

3. Lakukan perintah berikut:
    a. Tambahkan data berikut diantara Farrel dan Denis: Wati 2330004
    b. Hapus data Denis
    c. Tambahkan data berikut di awal: Owi 2330000
    d. Tambahkan data berikut di akhir: David 23300100
    e. Ubah data Udin menjadi data berikut: Idin 23300045
    f. Ubah data terkahir menjadi berikut: Lucy 23300101
    g. Hapus data awal
    h. Ubah data awal menjadi berikut: Bagas 2330002
    i. Hapus data akhir
    j. Tampilkan seluruh data

Source Code

```
#include <iostream>
#include <iomanip>
```

```
using namespace std;
//Membuat struct dengan variabel mahasiswa terdapat 2 field
struct mahasiswa{
    string nama;//Field nama berOpe data string
    string nim;//Field nim berOpe data integer
};

//Struct node terdapat 2 field
struct node{
    mahasiswa ITTP;// field ITTP berOpe inisialisasi struct mahasiswa
    node *next;// Field next berOpe pointer node
};

//inisialisasi
node *head, *tail, *bantu, *hapus, *before, *baru;
//Menginisialisasikan nilai head & tail dengan nilai NULL
void init(){
    head = NULL;
    tail = NULL;
}

//Pengecekan Nilai
bool isEmpty(){
    if (head == NULL)
    {
        return true;
    }

else{
    return false;
    }
}
mahasiswa Pendataan(){
    mahasiswa ITTP;
    cout << "\nMasukkan Nama\t: ";
    cin.ignore();
    getline(cin, ITTP.nama);
    cout << "Masukkan NIM\t: ";
    cin >> ITTP.nim;
    return ITTP;
}

// Fungsi Tambah depan
void insertDepan(mahasiswa ITTP){
    node *baru = new node;
    baru->ITTP.nama = ITTP.nama;
    baru->ITTP.nim = ITTP.nim;
    baru->next = NULL;
    if (isEmpty() == true){
```

```
                            head = tail = baru;
                            tail->next = NULL;
                        }
else{
    baru->next = head;
    head = baru;
}
cout << "Data " << ITTP.nama << " berhasil diinput!\n";
}

//Fungsi Tambah Belakang
void insertBelakang(mahasiswa ITTP){
    node *baru = new node;
    baru->ITTP.nama = ITTP.nama;
    baru->ITTP.nim = ITTP.nim;
    baru->next = NULL;
    if (isEmpty() == true){
                            head = tail = baru;
                            tail->next = NULL;
                        }
else{
    tail->next = baru;
    tail = baru;
    }
}

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

//Fungsi Tambah tengah
void insertTengah(mahasiswa idenOtas, int posisi){
    node *baru = new node;
    baru->ITTP.nama = idenOtas.nama;
    baru->ITTP.nim = idenOtas.nim;
    node *bantu;
    if (posisi < 1 || posisi > hitungList()){
        cout << "posisi diluar jangakauan";
    }
else if (posisi == 1){
    cout << "INi bukan posisi tengah\n";
```

```
    }
else{
    bantu = head;
    int penghitung = 1;
    while (penghitung != posisi - 1){
        penghitung++;
        bantu = bantu->next;
    }
    baru->next = bantu->next;
    bantu->next = baru;
    }
}

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

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

//Fungsi Ubah Tengah
void ubahTengah(mahasiswa data){
    int posisi;
    cout << "\nMasukkan posisi data yang akan diubah : ";
    cin >> posisi;

if (posisi < 1 || posisi > hitungList()){
    cout << "\nPosisi diluar jangkauan\n";
}
else if (posisi == 1){
    cout << "\nBukan posisi tengah\n";
}
else{
        bantu = head;
        int penghitung = 1;
        while (penghitung != posisi){
            penghitung++;
            bantu = bantu->next;
```

```
        }
    bantu->ITTP.nama = data.nama;
    bantu->ITTP.nim = data.nim;
    }
}

//Fungsi Menampilkan List
void tampil(){
    node *bantu = head;
    cout << "Nama " << " Nim\n";
    while (bantu != NULL){
        cout << bantu->ITTP.nama << " " << bantu->ITTP.nim << endl;
        bantu = bantu->next;
    }
}

//Fungsi Hapus Depan
void hapusDepan(){
    string dataBefore = head->ITTP.nama;
    hapus = head;
    if (head != tail){
        head = head->next;
        delete hapus;
    }
    else{
            head = tail = NULL;
        }
    cout << "Data " << dataBefore << " berhasil dihapus\n";
}

//Fungsi Hapus Belakang
void hapusBelakang(){
    string dataBefore = head->ITTP.nama;
    if (head != tail){
        hapus = tail;
        bantu = head;
        while (bantu->next != tail){
            bantu = bantu->next;
        }
        tail = bantu;
        tail->next = NULL;
    delete hapus;
    }
else{
        head = tail = NULL;
    }
    cout << "Data " << dataBefore << " berhasil dihapus\n";
}
```

```
//Fungsi Hapus Tengah
void hapusTengah(){
    tampil();
    cout << endl;
    if (isEmpty() == false){
        back:
            int posisi;
            cout << "Masukkan Posisi yang dihapus : ";
            cin >> posisi;
        if (posisi < 1 || posisi > hitungList()){
        cout << "\nPosisi di luar jangkauan!\n";
        cout << "Masukkan posisi baru\n";
        goto back;
    }
    else if (posisi == 1 || posisi == hitungList()){
        cout << "\nBukan Posisi tengah\n";
        cout << "Masukkan posisi baru\n";
        goto back;
    }
    else{
            bantu = head;
            int penghitung = 1;
            while (penghitung <= posisi){
            if (penghitung == posisi - 1){
            before = bantu;
        }
        if (penghitung == posisi){
        hapus = bantu;
    }
    bantu = bantu->next;
    penghitung++;
 }
 string dataBefore = hapus->ITTP.nama;
 before->next = bantu;
 delete hapus;
 cout << "\nData " << dataBefore << " berhasil dihapus!\n";
}
    }
    else{
            cout << "\n!!! List Data Kosong !!!\n";
    }
}

//Fungsi Hapus List
void hapusList(){
    bantu = head;
    while (bantu != NULL){
        hapus = bantu;
        delete hapus;
```

```
        bantu = bantu->next;
    }
    init();
cout << "\nsemua data berhasil dihapus\n";
}

//Program Utama
int main(){
    init();
    mahasiswa ITTP;
    back:
    int operasi, posisi;
    cout << " PROGRAM SINGLE LINKED LIST NON-CIRCULAR" << endl;
    cout << " ========================================\n\n" << endl;
    cout << "1. Tambah Depan" << endl;
    cout << "2. Tambah Belakang" << endl;
    cout << "3. Tambah Tengah" << endl;
    cout << "4. Ubah Depan" << endl;
    cout << "5. Ubah Belakang" << endl;
    cout << "6. Ubah Tengah" << endl;
    cout << "7. Hapus depan" << endl;
    cout << "8. Hapus belakang" << endl;
    cout << "9. Hapus Teangah" << endl;
    cout << "10.Hapus list" << endl;
    cout << "11.Tampilkan" << endl;
    cout << "0. Exit" << endl;
    cout << "\nPilih Operasi :> ";
    cin >> operasi;
    switch (operasi){
        case 1:
            cout << "tambah depan\n";
        insertDepan(Pendataan());
            cout << endl;
        goto back;
    break;

        case 2:
            cout << "tambah belakang\n";
        insertBelakang(Pendataan());
            cout << endl;
        goto back;
    break;

        case 3:
            cout << "tambah tengah\n";
            cout << "nama : ";
            cin >> ITTP.nama;
            cout << "NIM : ";
            cin >> ITTP.nim;
```

```
            cout << "Posisi: ";
            cin >> posisi;
        insertTengah(ITTP, posisi);
            cout << endl;
        goto back;
    break;

        case 4:
            cout << "ubah depan\n";
        ubahDepan(Pendataan());
            cout << endl;
        goto back;
    break;

        case 5:
            cout << "ubah belakang\n";
        ubahBelakang(Pendataan());
            cout << endl;
        goto back;
    break;

        case 6:
            cout << "ubah tengah\n";
        ubahTengah(Pendataan());
            cout << endl;
        goto back;
    break;

        case 7:
            cout << "hapus depan\n";
        hapusDepan();
            cout << endl;
        goto back;
    break;

        case 8:
            cout << "hapus belakang\n";
        hapusBelakang();
            cout << endl;
        goto back;
    break;

        case 9:
            cout << "hapus tengah\n";
        hapusTengah();
            cout << endl;
        goto back;
    break;
```

```
        case 10:
            cout << "hapus list\n";
        hapusList();
            cout << endl;
        goto back;
    break;

        case 11:
        tampil();
            cout << endl;
        goto back;
    break;

        case 0:
            cout << "\nEXIT PROGRAM\n";
    break;

default:
    cout << "\nSalah input operasi\n";
    cout << endl;
    goto back;
break;
}

return 0;
}
```

Output Code

1. Menampilkan semua menu

```
PS C:\Belajar coding\Strukdat\Learning Code\STRUKDAT (PRAKTEK)\circular2>  & 'c:\Users\ASUS\.vs
code\extensions\ms-vscode.cpptools-1.19.9-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe'
 '--stdin=Microsoft-MIEngine-In-acimgvs4.jsu' '--stdout=Microsoft-MIEngine-Out-4vgesria.wgh' '-
-stderr=Microsoft-MIEngine-Error-tttipnic.j4b' '--pid=Microsoft-MIEngine-Pid-4amb2zdh.dwe' '--d
bgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
 PROGRAM SINGLE LINKED LIST NON-CIRCULAR
 ========================================


1. Tambah Depan
2. Tambah Belakang
3. Tambah Tengah
4. Ubah Depan
5. Ubah Belakang
6. Ubah Tengah
7. Hapus depan
8. Hapus belakang
9. Hapus Teangah
10.Hapus list
11.Tampilkan
0. Exit
```

2. Menampilkan tambah depan

```
Pilih Operasi :> 1
tambah depan

Masukkan Nama    : zaidan
Masukkan NIM     : 2311102058
Data zaidan data telah terinput !
```

3. Menampilkan tambah belakang

```
Pilih Operasi :> 2
tambah belakang

Masukkan Nama    : tegar
Masukkan NIM     : 2311102041
```

4. Menampilkan tambah tengah

```
Pilih Operasi :> 3
tambah tengah
nama : pandia
NIM : 2311102076
Posisi: 2
```

5. Menampilkan ubah depan (ubah data zaidan -> data syifa)

```
Pilih Operasi :> 4
ubah depan

Masukkan Nama   : syifa
Masukkan NIM    : 2311102080
data zaidan telah diganθ dengan data syifa
```

6. Menampilkan ubah belakang (ubah data tegar -> data farel)

```
Pilih Menu :5
ubah belakang

Masukkan Nama   : farel
Masukkan NIM    : 2311102079
data tegar telah diganti dengan data farel
```

7. Menampilkan ubah tengah (ubah data pandia -> syarief)

```
Pilih Menu :6
ubah tengah

Masukkan Nama   : syarief
Masukkan NIM    : 2311102072

Masukkan posisi data yang akan diubah : 2
```

8. Menampilkan hapus depan

```
Pilih Menu :7
hapus depan
Data syifa berhasil dihapus
```

9. Menampilkan hapus belakang

```
Pilih Menu :8
hapus belakang
Data farel berhasil dihapus
```

10. Menampilkan hapus tengah

```
Pilih Menu :9
hapus tengah
Nama  Nim
syifa 2311102080
syarief 2311102072
farel 2311102079

Masukkan Posisi yang dihapus : 2

Data syarief berhasil dihapus!
```

11. Menampilkan Keseluruhan Data

```
Pilih Menu :11
Nama  Nim
syifa 2311102080
syarief 2311102072
farel 2311102079
```

Soal nomer dua

1) Memasukan data jawad

```
Pilih Menu :1
tambah depan

Masukkan Nama   : Jawad
Masukkan NIM    : 23300001
Data Jawad data telah terinput !
```

2) Memasukan data nama sendiri

```
Pilih Menu :3
tambah tengah
nama : Zaidan
NIM : 2311102058
Posisi: 2
```

3) Memasukan data farel

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Farrel
Masukkan NIM    : 23300003
```

4) Memasukan data denis

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Denis
Masukkan NIM    : 23300005
```

5) Memasukan data anis

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Anis
Masukkan NIM    : 23300008
```

6) Memasukan data bowo

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Bowo
Masukkan NIM    : 23300015
```

7) Memasukan data gahar

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Gahar
Masukkan NIM    : 23300040
```

8) Memasukan data udin

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Udin
Masukkan NIM    : 23300048
```

9) Memasukan data ucok

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Ucok
Masukkan NIM    : 23300050
```

10) Memasukan data budi

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : Budi
Masukkan NIM    : 23300099
```

11) Menampilkan data sementara

```
Nama  Nim
Jawad 23300001
Zaidan 2311102058
Farrel 23300003
Denis 23300005
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099
```

Soal nomer 3

a) Tambahkan data berikut diantara Farrel dan Denis: Wati 2330004

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

```
Pilih Menu :11
Nama  Nim
Jawad 23300001
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Denis 23300005
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099
```

b) Hapus data Denis

```
Masukkan Posisi yang dihapus : 5

Data Denis berhasil dihapus!
```

```
Pilih Menu :11
Nama  Nim
Jawad 23300001
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099
```

c) Tambahkan data berikut di awal: Owi 2330000

```
Pilih Menu :1
tambah depan

Masukkan Nama   : Owi
Masukkan NIM    : 2330000
Data Owi data telah terinput !
```

```
Pilih Menu :11
Nama  Nim
Owi 2330000
Jawad 23300001
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
Udin 23300048
Ucok 23300050
Budi 23300099
```

d) Tambahkan data berikut di akhir: David 23300100

```
Pilih Menu :2
tambah belakang

Masukkan Nama   : David
Masukkan NIM    : 23300100
```

```
Pilih Menu :11
Nama  Nim
Owi 2330000
Jawad 23300001
Zaidan 2311102058
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

```
Pilih Menu :6
ubah tengah

Masukkan Nama   : idin
Masukkan NIM    : 23300045

Masukkan posisi data yang akan diubah : 9
```

```
Pilih Menu :11
Nama  Nim
Owi 2330000
Jawad 23300001
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
idin 23300045
Ucok 23300050
Budi 23300099
David 23300100
```

f) Ubah data terakhir menjadi berikut: Lucy 23300101

```
Pilih Menu :5
ubah belakang

Masukkan Nama   : Lucy
Masukkan NIM    : 23300101
data David telah diganti dengan data Lucy
```

```
Pilih Menu :11
Nama  Nim
Owi 2330000
Jawad 23300001
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
idin 23300045
Ucok 23300050
Budi 23300099
Lucy 23300101
```

g) Hapus data awal

```
Pilih Menu :7
hapus depan
Data Owi berhasil dihapus
```

```
Pilih Menu :11
Nama  Nim
Jawad 23300001
Zaidan 2311102058
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

```
Pilih Menu :4
ubah depan

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

```
Pilih Menu :11
Nama  Nim
Bagas 2330002
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
idin 23300045
Ucok 23300050
Budi 23300099
Lucy 23300101
```

i) Hapus data akhir

```
Pilih Menu :11
Nama  Nim
Bagas 2330002
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
idin 23300045
Ucok 23300050
Budi 23300099
```

Data lucy telah terhapus

j) Tampilkan seluruh data

```
Pilih Menu :11
Nama  Nim
Bagas 2330002
Zaidan 2311102058
Farrel 23300003
Wati 2330004
Anis 23300008
Bowo 23300015
Gahar 23300040
idin 23300045
Ucok 23300050
Budi 23300099
```

Deskripsi Code

Pemrograman ini mengimplementasikan Single Linked List Non Circular yang mana menggunakan beberapa konsep linked list. Program ini berfungsi untuk menyimpan data-data mahasiswa seperti nama dan NIM. Program juga terdapat beberapa fungsi untuk insert, menghapus, mengubah dan menampilkan data. Program mengimplementasikan struct node yang bertipe string dan juga yang berperan menjadi poitner ke node berikutnya.

2. Kesimpulan

Dari pratikum ini, mahasiswa dapat mempelajari bagiaman konsep Single Linked List Circular maupun Non Circular. Circular linked list adalah daftar tertaut yang semua nodenya terhubung membentuk lingkaran. Dalam daftar tertaut melingkar, simpul pertama dan simpul terakhir dihubungkan satu sama lain sehingga membentuk lingkaran. Tidak ada NULL di akhir.

Sedangkan untuk Non Circular itu field pointer-nya hanya satu buah saja dan satu arah serta pada akhir node, pointernya menunjuk NULL yang akan digunakan sebagai kondisi berhenti pada saat pembacaan isi linked list.

3. Referensi

[1] Tim Asisten Asprak, "Linked List Circular Dan Non Circular", Learning Management System, 2024.

[2] Trivusi, "Struktur Data Linked List: Pengertian, Karakteristik, dan Jenis-jenisnya", Struktur Data Linked List: Pengertian, Karakteristik, dan Jenis-jenisnya - Trivusi, terakhir kali diakses pada tanggal 10 Maret 2024.

[3] Geeksforgeeks, "Introduction to Circular Linked List", https://www.geeksforgeeks.org/circular-linked-list/, terakhir kali diakses pada tanggal 10 maret 2024.