LAPORAN PRAKTIKUM

- **Identitas Praktikum**

  Nama MK : Struktur Data
  Kode MK : CCK2AAB4
  Bobot SKS : 4 SKS
  Tempat : L-Program, Gedung DC, lantai 3
  Hari, tanggal : Selasa, 29 Oktober 2024
  Jam : 12:30-15:30 WIB
  Topik praktikum : Modul-6 DOUBLE LINKED LIST (BAGIAN PERTAMA)

- **Identitas Mahasiswa**

  Nama lengkap : Afad Fath Musyarof Halim
  NIM : 2211104030
  Program Studi : S-1 Software Engineering

- **Hasil Praktikum**

6.1. Double Linked List

1. Insert

A. Insert First

```
void insertFirst(int data) {
    Node* newNode = new Node(data);
    if (head == nullptr) {
        head = newNode;
    } else {
        newNode->next = head;
        head->prev = newNode;
        head = newNode;
    }
}
```

B. Insert Last

```
void insertLast(int data) {
    Node* newNode = new Node(data);
    if (head == nullptr) {
        head = newNode;
    } else {
        Node* temp = head;
        while (temp->next != nullptr) {
            temp = temp->next;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }
}
```

C. Insert After

```
void insertAfter(int key, int data) {
    Node* temp = head;
    while (temp != nullptr && temp->data != key) {
        temp = temp->next;
    }
    if (temp != nullptr) {
        Node* newNode = new Node(data);
        newNode->next = temp->next;
        newNode->prev = temp;
        if (temp->next != nullptr) {
            temp->next->prev = newNode;
        }
        temp->next = newNode;
    }
}
```

D. Insert Before

```
void insertBefore(int key, int data) {
    Node* temp = head;
    while (temp != nullptr && temp->data != key) {
        temp = temp->next;
    }
    if (temp != nullptr) {
        Node* newNode = new Node(data);
        newNode->next = temp;
        newNode->prev = temp->prev;
        if (temp->prev != nullptr) {
            temp->prev->next = newNode;
        } else {
            head = newNode;
        }
        temp->prev = newNode;
    }
}
```

2. Delete

A. Delete First

```
void deleteFirst() {
    if (head == nullptr) return;
    Node* temp = head;
    head = head->next;
    if (head != nullptr) head->prev = nullptr;
    delete temp;
}
```

B. Delete Last

```
void deleteLast() {
    if (head == nullptr) return;
    Node* temp = head;
    while (temp->next != nullptr) {
        temp = temp->next;
    }
    if (temp->prev != nullptr) {
        temp->prev->next = nullptr;
    } else {
        head = nullptr;
    }
    delete temp;
}
```

C. Delete After

```
void deleteAfter(int key) {
    Node* temp = head;
    while (temp != nullptr && temp->data != key) {
        temp = temp->next;
    }
    if (temp != nullptr && temp->next != nullptr) {
        Node* delNode = temp->next;
        temp->next = delNode->next;
        if (delNode->next != nullptr) {
            delNode->next->prev = temp;
        }
        delete delNode;
    }
}
```

D. Delete Before

```
void deleteBefore(int key) {
    Node* temp = head;
    while (temp != nullptr && temp->data != key) {
        temp = temp->next;
    }
    if (temp != nullptr && temp->prev != nullptr) {
        Node* delNode = temp->prev;
        if (delNode->prev != nullptr) {
            delNode->prev->next = temp;
        } else {
            head = temp;
        }
        temp->prev = delNode->prev;
        delete delNode;
    }
}
```

E. Update

```
void update(int oldData, int newData) {
    Node* temp = head;
    while (temp != nullptr && temp->data != oldData) {
        temp = temp->next;
    }
    if (temp != nullptr) {
        temp->data = newData;
    }
}
```

F. Search

```
Node* search(int key) {
    Node* temp = head;
    while (temp != nullptr && temp->data != key) {
        temp = temp->next;
    }
    return temp;
}
```

G. View

```
void view() {
    Node* temp = head;
    while (temp != nullptr) {
        cout << temp->data << " ";
        temp = temp->next;
    }
    cout << endl;
}
```

6.2. Latihan

1. Buat Double Linked List

- Code

  - Doublelist.h

```
#ifndef DOUBLELIST_H
#define DOUBLELIST_H

#include <iostream>
#include <string>

using namespace std;

struct kendaraan {
    string nopol;
    string warna;
    int thnBuat;
};

typedef kendaraan infotype;

struct ElmList {
    infotype info;
    ElmList* next;
    ElmList* prev;
};

typedef ElmList* address;

struct List {
    address First;
    address Last;
};

void CreateList(List &L);

address alokasi(infotype x);

void dealokasi(address &P);

void printInfo(const List &L);

void insertLast(List &L, address P);

#endif
```

- Doublelist.cpp

```cpp
#include "doublelist.h"
#include <iostream>

using namespace std;

void CreateList(List &L) {
    L.First = nullptr;
    L.Last = nullptr;
}

address alokasi(infotype x) {
    address P = new ElmList;
    P->info = x;
    P->next = nullptr;
    P->prev = nullptr;
    return P;
}

void dealokasi(address &P) {
    delete P;
    P = nullptr;
}

void printInfo(const List &L) {
    address P = L.First;
    while (P != nullptr) {
        cout << endl;
        cout << "Nomor Polisi    : " << P->info.nopol << endl;
        cout << "Warna Mobil     : " << P->info.warna << endl;
        cout << "Tahun Kendaraan : " << P->info.thnBuat << endl;
        P = P->next;
    }
}

void insertLast(List &L, address P) {
    if (L.First == nullptr) {
        L.First = P;
        L.Last = P;
    } else {
        L.Last->next = P;
        P->prev = L.Last;
        L.Last = P;
    }
}
```

- Main.cpp

```cpp
#include "doublelist.cpp"
#include <iostream>

using namespace std;

int main(){
    List L;
    CreateList(L);

    int n;
    cout << "Berapa banyak mobil yang ingin dimasukkan? ";
    cin >> n;

    for (int i = 0; i < n; i++) {
        infotype kendaraan;
        cout << "\nMasukkan nomor polisi    : ";
        cin >> kendaraan.nopol;
        cout << "Masukkan warna Kendaraan : ";
        cin >> kendaraan.warna;
        cout << "Masukkan tahun kendaraan : ";
        cin >> kendaraan.thnBuat;

        address P = alokasi(kendaraan);
        insertLast(L, P);
    }
    cout << "\n==============" << endl;
    cout << "Data Kendaraan" << endl;
    printInfo(L);
}
```

- Output

```
# & .\'main.exe'
Berapa banyak mobil yang ingin dimasukkan? 3

Masukkan nomor polisi    : D001
Masukkan warna Kendaraan : Putih
Masukkan tahun kendaraan : 2000

Masukkan nomor polisi    : D002
Masukkan warna Kendaraan : Hitam
Masukkan tahun kendaraan : 2010

Masukkan nomor polisi    : D003
Masukkan warna Kendaraan : Merah
Masukkan tahun kendaraan : 2020

==============
Data Kendaraan

Nomor Polisi    : D001
Warna Mobil     : Putih
Tahun Kendaraan : 2000

Nomor Polisi    : D002
Warna Mobil     : Hitam
Tahun Kendaraan : 2010

Nomor Polisi    : D003
Warna Mobil     : Merah
Tahun Kendaraan : 2020
Afadfath | output
#
```

- Penjelasan
    - ‘CreateList()’ untuk membuat list dengan pointer kosong
    - ‘alokasi()’ untuk mengalokasikan memori untuk elemen baru
    - ‘dealokasi()’ untuk menghapus alokasi memori
    - ‘printInfo()’ untuk menampilkan isi/info dari list
    - ‘insertLast()’ untuk menambahkan elemen ke list

2. Carilah elemen dengan nomor polisi D001 dengan membuat fungsi baru

- Code
    - Tambahan Doublelist.h

```
address findElm(const List &L, infotype x);
```

    - Tambahan Doublelist.cpp

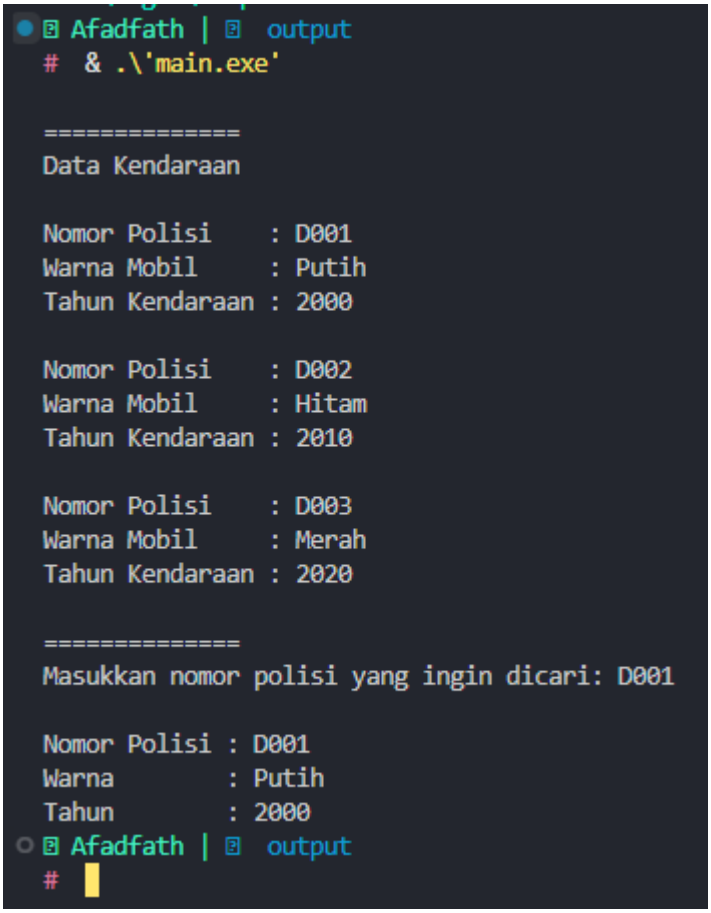
```
address findElm(const List &L, infotype x) {
    address P = L.First;
    while (P != nullptr) {
        if (P->info.nopol == x.nopol) {
            return P;
        }
        P = P->next;
    }
    return nullptr;
}
```

    - Tambahan Main.cpp

```
cout << "\n==============" << endl;
infotype cari;
cout << "Masukkan nomor polisi yang ingin dicari: ";
cin >> cari.nopol;

address P = findElm(L, cari);
if (P != nullptr) {
    cout << endl;
    cout << "Nomor Polisi : " << P->info.nopol << endl;
    cout << "Warna        : " << P->info.warna << endl;
    cout << "Tahun        : " << P->info.thnBuat << endl;
} else {
    cout << "Data tidak ditemukan" << endl;
}
```

- Output



- Penjelasan
  - Memulai dari node pertama
  - Membandingkan nopol dari setiap info dengan nopol yang ingin dicari
  - Jika ada maka node tersebut di kembalikan dan node di tampilkan
  - Jika tidak maka menampilkan data kosong

3. Hapus elemen dengan nomor polisi D003 dengan prosedur delete

- Code
  - Tambahan Doublelist.h

```
void deleteAfter(address Prec, address &P);
```

  - Tambahan Doublelist.cpp

```
void deleteAfter(address Prec, address &P) {
    if (Prec != nullptr && Prec->next != nullptr) {
        P = Prec->next;
        Prec->next = P->next;
        if (P->next != nullptr) {
            P->next->prev = Prec;
        }
        P->next = nullptr;
        P->prev = nullptr;
    }
}
```

- Tambahan Main.cpp

```
cout << "\n==============" << endl;
    infotype hapus;
    cout << "Masukkan nomor polisi yang ingin dihapus: ";
    cin >> hapus.nopol;

    address P = findElm(L, hapus);
    if (P != nullptr) {
        deleteAfter(P->prev, P);
        dealokasi(P);
        cout << "Data berhasil dihapus" << endl;
    } else {
        cout << "Data tidak ditemukan" << endl;
    }

    printInfo(L);
```

- Output

```
Afadfath | output
# & .\'main.exe'

==============
Data Kendaraan

Nomor Polisi    : D001
Warna Mobil     : Putih
Tahun Kendaraan : 2000

Nomor Polisi    : D002
Warna Mobil     : Hitam
Tahun Kendaraan : 2010

Nomor Polisi    : D003
Warna Mobil     : Merah
Tahun Kendaraan : 2020

==============
Masukkan nomor polisi yang ingin dihapus: D003
Data berhasil dihapus

Nomor Polisi    : D001
Warna Mobil     : Putih
Tahun Kendaraan : 2000

Nomor Polisi    : D002
Warna Mobil     : Hitam
Tahun Kendaraan : 2010
Afadfath | output
# Compiled su
```

- Penjelasan
  - Cek apakah list kosong, jika iya lanjutkan, jika tidak berhenti
  - Menghapus elemen setelah prec
  - Mengatur pointer dari prev elemen
  - Mengatur pointer p