LAPORAN PRAKTIKUM STRUKTUR DATA

MODUL VI
DOUBLY LINKED LIST (BAGIAN PERTAMA)

Disusun Oleh :

NAMA : IVAN RAMADHAN

NIM : 103112400186

Dosen

FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

A. Dasar Teori

C++ adalah pengembangan dari dari bahasa c yang dibuat oleh Bjarne Stroustrup sekitar tahun 1980-an. C++ disebut bahasa multi-paradigma, artinya bisa dipakai dengan gaya prosedural (pakai fungsi biasa), beriorientasi objek (pakai class dan object), atau bahkan gabungan keduanya. C++ punya dasar-dasar seperti variabel, operator percabangan (if, switch), perulangan (for, while), dan bisa memakai class untuk membuat objek.

B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1
doublylist.h

```
#ifndef DOUBLYLIST_H
#define DOUBLYLIST_H

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

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

typedef kendaraan infotype;

struct ElmList;
typedef ElmList* address;

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

struct List {
   address first;
   address last;
};

void createList(List &L);
address alokasi(infotype x);
void dealokasi(address &P);
void printInfo(List L);
```

```
void insertLast(List &L, address P);
address findElm(List L, string nopol);
void deleteFirst(List &L, address &P);
void deleteLast(List &L, address &P);
void deleteAfter(List &L, address Prec, address &P);

#endif
```

Doublylist.cpp

```
#include "Doublylist.h"

void createList(List &L) {
   L.first = NULL;
   L.last = NULL;
}

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

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

void printInfo(List L) {
   address P = L.last;
   cout << "\nDATA LIST\n";
   while (P != NULL) {
      cout << "No Polisi : " << P->info.nopol << endl;
      cout << "Warna     : " << P->info.warna << endl;
      cout << "Tahun     : " << P->info.thnbuat << endl;
      cout << endl;
      P = P->prev;
   }
}

void insertLast(List &L, address P) {
   if (L.first == NULL) {
      L.first = P;
      L.last = P;
```

```
    } else {
        L.last->next = P;
        P->prev = L.last;
        L.last = P;
    }
}

address findElm(List L, string nopol) {
    address P = L.first;
    while (P != NULL && P->info.nopol != nopol) {
        P = P->next;
    }
    return P;
}

void deleteFirst(List &L, address &P) {
    if (L.first != NULL) {
        P = L.first;
        if (L.first == L.last) {
            L.first = NULL;
            L.last = NULL;
        } else {
            L.first = P->next;
            L.first->prev = NULL;
        }
        P->next = NULL;
    }
}

void deleteLast(List &L, address &P) {
    if (L.last != NULL) {
        P = L.last;
        if (L.first == L.last) {
            L.first = NULL;
            L.last = NULL;
        } else {
            L.last = P->prev;
            L.last->next = NULL;
        }
        P->prev = NULL;
    }
}

void deleteAfter(List &L, address Prec, address &P) {
    if (Prec != NULL && Prec->next != NULL) {
        P = Prec->next;
```

```
        Prec->next = P->next;
        if (P->next != NULL)
            P->next->prev = Prec;
        else
            L.last = Prec;

        P->next = NULL;
        P->prev = NULL;
    }
}
```

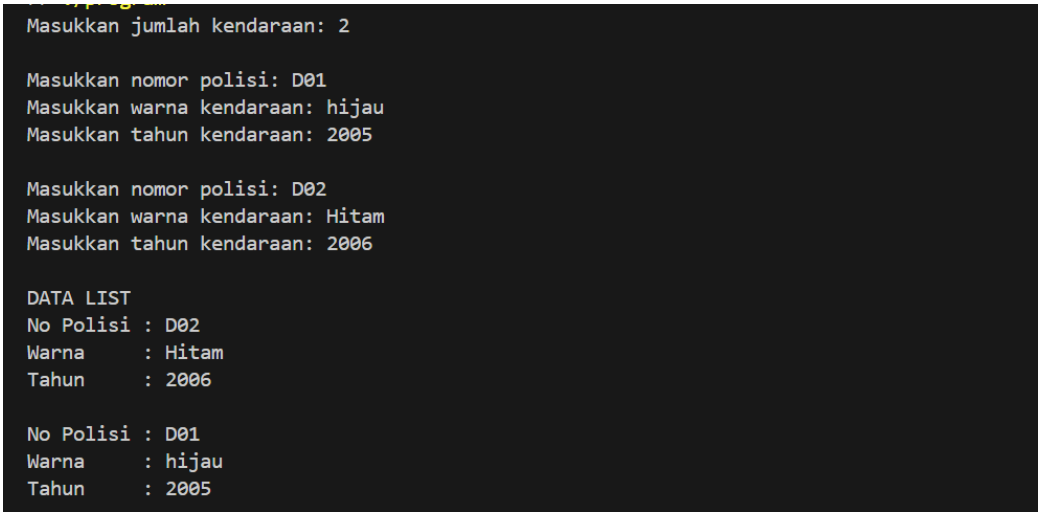
Main.cpp

```
#include "Doublylist.h"

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

    infotype x;
    address P;
    string cari;
    int n;

    cout << "Masukkan jumlah kendaraan: ";
    cin >> n;

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

        if (findElm(L, x.nopol) != NULL) {
            cout << "Nomor polisi sudah terdaftar\n";
            continue;
        }

        P = alokasi(x);
        insertLast(L, P);
    }

    printInfo(L);
```

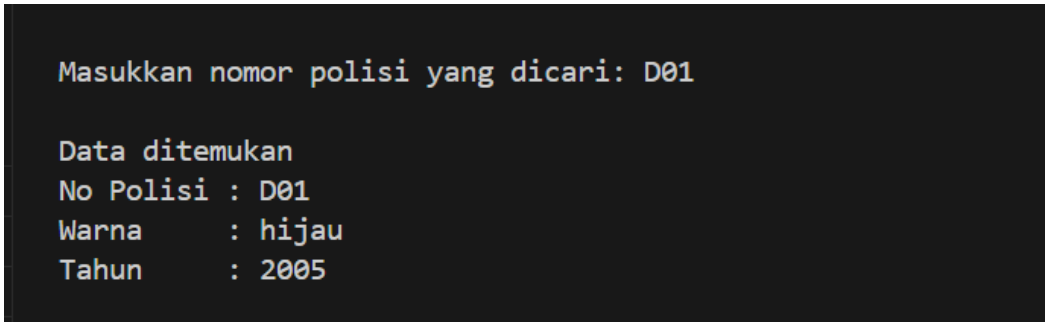
```
    // Cari data
    cout << "Masukkan nomor polisi yang dicari: ";
    cin >> cari;
    P = findElm(L, cari);

    if (P != NULL) {
        cout << "\nData ditemukan\n";
        cout << "No Polisi : " << P->info.nopol << endl;
        cout << "Warna     : " << P->info.warna << endl;
        cout << "Tahun     : " << P->info.thnbuat << endl;
    } else {
        cout << "Data tidak ditemukan.\n";
    }

    // Hapus data
    cout << "\nMasukkan nomor polisi yang akan dihapus: ";
    cin >> cari;
    P = findElm(L, cari);

    if (P == NULL) {
        cout << "Data tidak ditemukan.\n";
    } else {
        if (P == L.first)
            deleteFirst(L, P);
        else if (P == L.last)
            deleteLast(L, P);
        else
            deleteAfter(L, P->prev, P);

        dealokasi(P);
        cout << "Data berhasil dihapus.\n";
    }

    printInfo(L);
    return 0;
}
```

Screenshots Output:

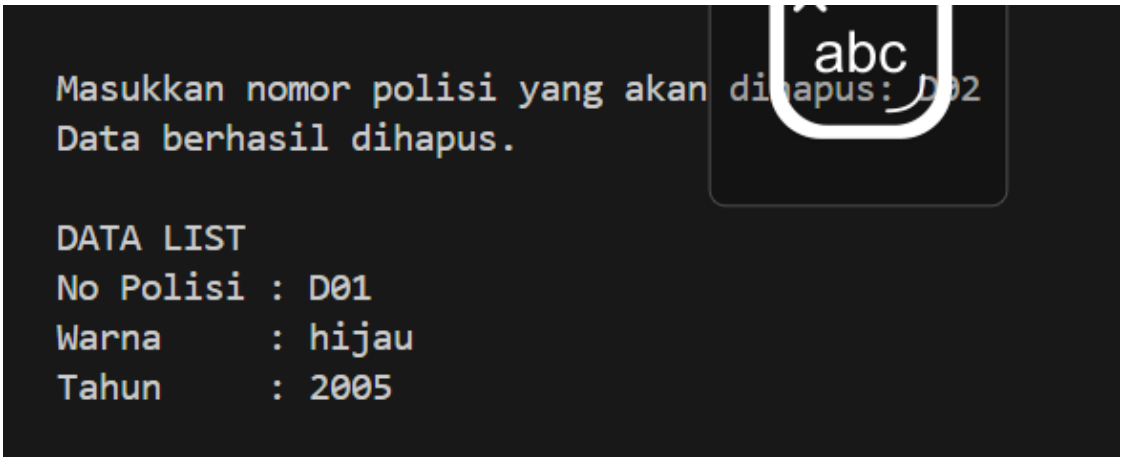
1.



2.



3.



Deskripsi:

Program ini mengimplementasikan Doubly Linked List untuk mengelola data kendaraan yang terdiri dari nomor polisi, warna, dan tahun pembuatan. Program menyediakan operasi dasar seperti menambah data kendaraan, menampilkan data dari belakang ke depan, mencari kendaraan berdasarkan nomor polisi, serta menghapus data kendaraan baik di awal, di akhir, maupun di tengah list. Dengan penggunaan pointer next dan prev, program memungkinkan penelusuran data secara dua arah dan pengelolaan data yang lebih fleksibel serta terstruktur.

C. Kesimpulan

Program Doubly Linked List kendaraan ini menunjukkan penerapan struktur data dinamis yang memungkinkan penyimpanan, pencarian, penampilan, dan penghapusan data secara efisien. Dengan adanya pointer next dan prev, program dapat menelusuri data maju dan mundur, serta menghapus data dari awal, akhir, maupun tengah list dengan tepat. Program ini juga menerapkan validasi data unik (nomor polisi) dan pengelolaan memori yang baik menggunakan fungsi alokasi dan dealokasi, sehingga mencerminkan penggunaan struktur data yang terorganisir, fleksibel, dan efektif untuk pengolahan data berurutan.

D. Referensi

- W3Resource. (2020). *C++ String Exercises: Convert digit/number to words*.
- GeeksforGeeks. (2020). *Loops in C++ (for, while, do-while).*