**LAPORAN PRAKTIKUM**
**STRUKTUR DATA**

**MODUL 6**
**DOUBLY LINKED LIST**

**Disusun Oleh :**
NAMA : FANDIKA PRIMADANI
NIM : 103112400231

**Dosen**
FAHRUDIN MUKTI WIBOWO

**PROGRAM STUDI STRUKTUR DATA**
**FAKULTAS INFORMATIKA**
**TELKOM UNIVERSITY PURWOKERTO**
**2025**

A. Dasar Teori

Dasar teori dari program manajemen data kendaraan ini adalah penerapan **struktur data Double Linked List**, yaitu struktur data dinamis yang terdiri dari rangkaian elemen (node) di mana setiap node memiliki dua pointer, yaitu next yang menunjuk ke node berikutnya dan prev yang menunjuk ke node sebelumnya. Dengan dua arah hubungan tersebut, Double Linked List memungkinkan proses penambahan, pencarian, dan penghapusan data dilakukan lebih mudah dan efisien tanpa perlu menggeser elemen lain seperti pada array. Dalam program ini, konsep tersebut digunakan untuk mengelola data kendaraan berupa nomor polisi, warna, dan tahun pembuatan secara dinamis dan terstruktur.

B. SOAL 1

main.cpp

```cpp
#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) {
    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 insertLast(List &L, address P) {
    if (L.first == NULL) {
        L.first = P;
        L.last = P;
    } else {
        P->prev = L.last;
        L.last->next = P;
        L.last = P;
    }
}

bool isExist(List L, string nopol) {
    address Q = L.first;
    while (Q != NULL) {
        if (Q->info.nopol == nopol) {
            return true;
        }
        Q = Q->next;
    }
    return false;
}

void printInfo(List L) {
    cout << "\nDATA LIST 1" << endl;
    address P = L.first;
    while (P != NULL) {
        cout << "Nomor Polisi : " << P->info.nopol << endl;
        cout << "Warna        : " << P->info.warna << endl;
        cout << "Tahun        : " << P->info.thnBuat << endl;
```

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

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

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

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

```
}

void deleteAfter(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;
        }
        P->next = NULL;
        P->prev = NULL;
    }
}

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

    infotype x;
    char lagi;

    do {
        cout << "\nMasukkan Nomor Polisi: ";
        cin >> x.nopol;

        if (isExist(L, x.nopol)) {
            cout << "Nomor polisi sudah terdaftar!" << endl;
        } else {
            cout << "Masukkan Warna Kendaraan: ";
            cin >> x.warna;
            cout << "Masukkan Tahun Kendaraan: ";
            cin >> x.thnBuat;

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

        cout << "Tambah data lagi? (y/n): ";
        cin >> lagi;
    } while (lagi == 'y' || lagi == 'Y');

    printInfo(L);
```

```
    cout << "\nMasukkan Nomor Polisi yang dicari: ";
    string cari;
    cin >> cari;
    address found = findElm(L, cari);
    if (found != NULL) {
        cout << "Nomor Polisi : " << found->info.nopol <<
endl;
        cout << "Warna        : " << found->info.warna <<
endl;
        cout << "Tahun        : " << found->info.thnBuat <<
endl;
    } else {
        cout << "Data dengan nomor polisi " << cari << "
tidak ditemukan." << endl;
    }

    cout << "\nMasukkan Nomor Polisi yang akan dihapus: ";
    string hapus;
    cin >> hapus;

    address Q = findElm(L, hapus);
    if (Q == NULL) {
        cout << "Data dengan nomor polisi " << hapus << "
tidak ditemukan." << endl;
    } else {
        if (Q == L.first) {
            deleteFirst(L, Q);
        } else if (Q == L.last) {
            deleteLast(L, Q);
        } else {
            address Prec = Q->prev;
            deleteAfter(Prec, Q);
        }
        dealokasi(Q);
        cout << "Data dengan nomor polisi " << hapus << "
berhasil dihapus." << endl;
    }

    printInfo(L);

    return 0;
}
```

Screenshots Output

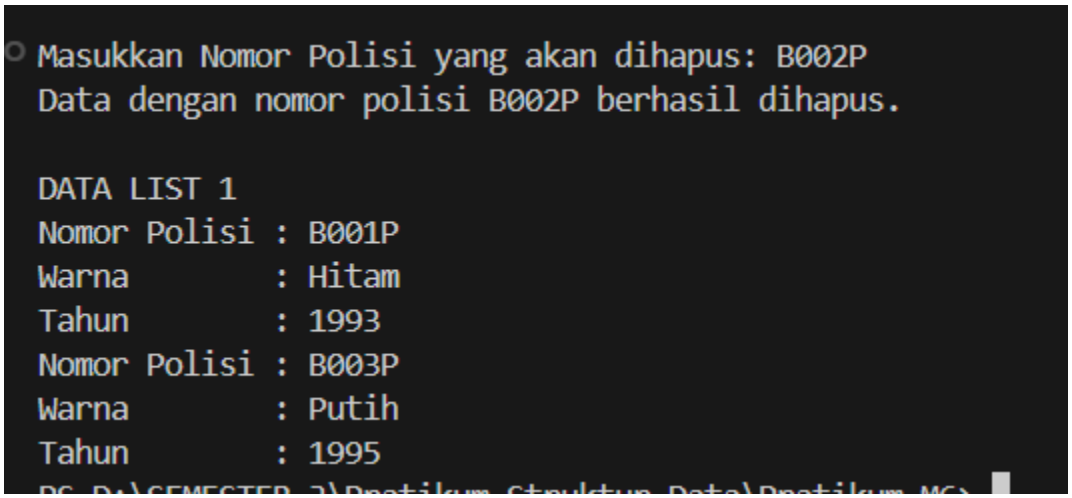
1. Output 1

```
PS D:\SEMESTER 3\Pratikum Struktur Data\Pratikum M6> cd
ain.cpp -o main } ; if ($?) { .\main }

Masukkan Nomor Polisi: B001P
Masukkan Warna Kendaraan: Hitam
Masukkan Tahun Kendaraan: 1993
Tambah data lagi? (y/n): y

Masukkan Nomor Polisi: B002P
Masukkan Warna Kendaraan: Biru
Masukkan Tahun Kendaraan: 1994
Tambah data lagi? (y/n): y

Masukkan Nomor Polisi: B003P
Masukkan Warna Kendaraan: Putih
Masukkan Tahun Kendaraan: 1995
Tambah data lagi? (y/n): n

DATA LIST 1
Nomor Polisi : B001P
Warna        : Hitam
Tahun        : 1993
Nomor Polisi : B002P
Warna        : Biru
Tahun        : 1994
Nomor Polisi : B003P
Warna        : Putih
Tahun        : 1995
```

2. Output 2 Carilah elemen dengan nomor polisi B001P dengan membuat fungsi baru.

```
Masukkan Nomor Polisi yang dicari: B001P
Nomor Polisi : B001P
Warna        : Hitam
Tahun        : 1993
```

3. Output 3 hapus elemen dengan nomor polisi B002P dengan procedure delete.


Masukkan Nomor Polisi yang akan dihapus: B002P
Data dengan nomor polisi B002P berhasil dihapus.
DATA LIST 1
Nomor Polisi : B001P
Warna : Hitam
Tahun : 1993
Nomor Polisi : B003P
Warna : Putih
Tahun : 1995

Deskripsi:

Tujuan dari program ini adalah untuk menyimpan, menampilkan, mencari, dan menghapus data kendaraan berdasarkan nomor polisi.Program dapat melakukan empat operasi utama:Menambah data kendaraan baru ke akhir list, Menampilkan seluruh data kendaraan, Mencari data berdasarkan nomor polisi, Menghapus data kendaraan berdasarkan nomor polisi

C. Kesimpulan

Kesimpulan dari program manajemen data kendaraan ini adalah bahwa penggunaan struktur data Double Linked List memungkinkan penyimpanan dan pengelolaan data kendaraan secara dinamis, di mana setiap elemen saling terhubung dua arah sehingga memudahkan proses penambahan, pencarian, dan penghapusan data. Melalui fungsi findElm() serta prosedur deleteFirst(), deleteLast(), dan deleteAfter(), program dapat mengelola data kendaraan berdasarkan nomor polisi dengan efisien. Dengan demikian, program ini berhasil menerapkan konsep Double Linked List untuk mengolah data kendaraan secara terstruktur dan fleksibel tanpa batasan ukuran tertentu.

D. Referensi

Deitel, H. M., & Deitel, P. J. (2013). *C++ How to Program (9th Edition)*. Pearson Education.
Malik, D. S. (2011). *Data Structures Using C++*. Cengage Learning.
Wibowo, A. (2019). *Struktur Data dengan C++*. Informatika Bandung.