

**LAPORAN PRAKTIKUM
STRUKTUR DATA**

**MODUL V
DOUBLE LINKED LIST**



Disusun Oleh :

Nama : Besthian Guido Rafael Simbolon
NIM : 103112430258

Dosen

FAHRUDIN MUKTI WIBOWO

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

A. Dasar Teori

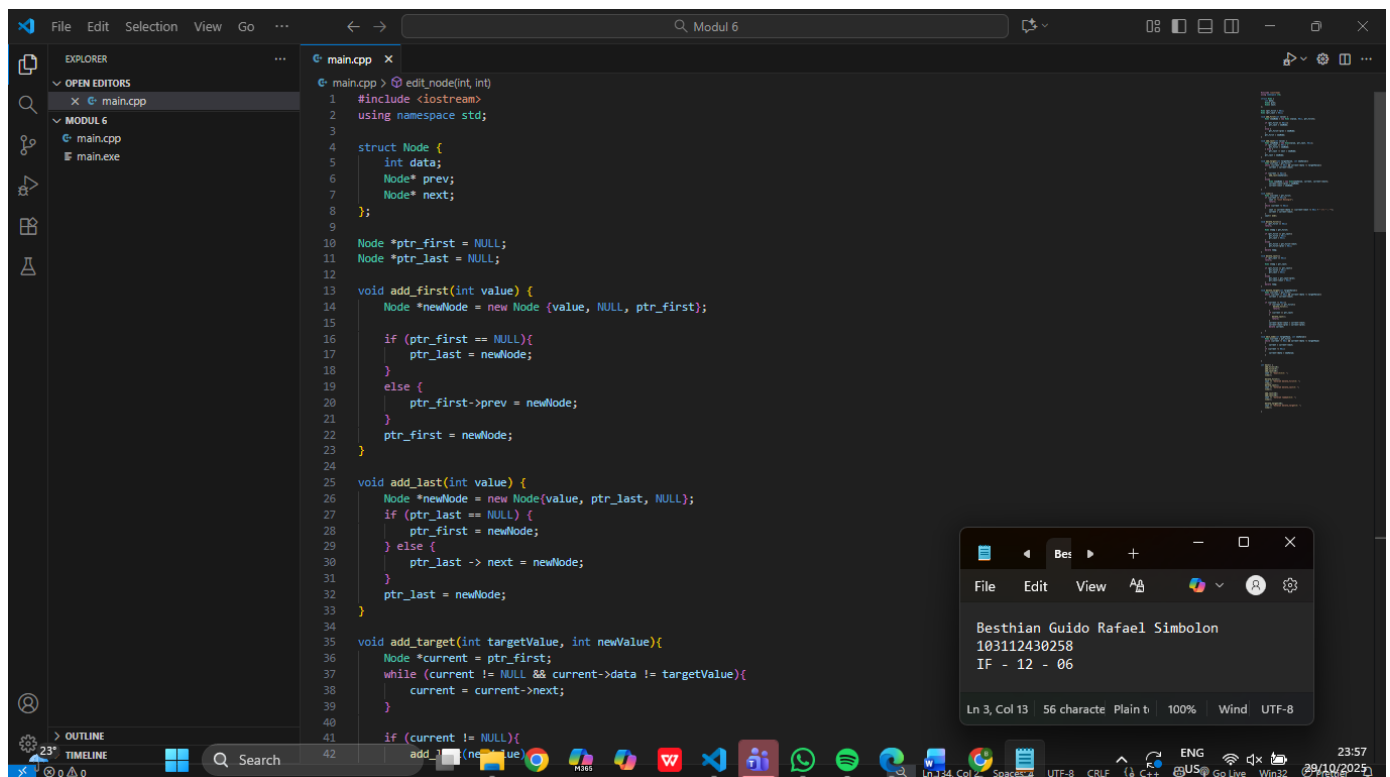
Doubly Linked List adalah salah satu bentuk struktur data berantai, di mana setiap elemen (node) tidak hanya menyimpan data, tapi juga punya dua penunjuk (pointer): *next* untuk menunjuk elemen setelahnya dan *prev* untuk menunjuk elemen sebelumnya. Jadi berbeda dengan Singly Linked List yang hanya bisa bergerak maju, Doubly Linked List bisa ditelusuri bolak-balik, dari depan ke belakang atau dari belakang ke depan. Dalam list ini juga ada pointer khusus bernama *first* untuk menunjuk elemen paling awal dan *last* untuk menunjuk elemen paling akhir.

Keuntungan memakai Doubly Linked List adalah proses mencari data, menghapus, atau menambah elemen lebih fleksibel karena bisa bergerak ke dua arah. Misalnya jika ingin menambahkan data di belakang, kita bisa langsung menuju elemen terakhir lewat pointer *last* tanpa harus menelusuri dari awal. Namun, karena setiap elemen butuh dua pointer, penggunaan memorinya jadi sedikit lebih besar dibanding Singly Linked List.

B. Guided

Guided 1

main.cpp



```
1 #include <iostream>
2 using namespace std;
3
4 struct Node {
5     int data;
6     Node* prev;
7     Node* next;
8 };
9
10 Node *ptr_first = NULL;
11 Node *ptr_last = NULL;
12
13 void add_first(int value) {
14     Node *newNode = new Node (value, NULL, ptr_first);
15
16     if (ptr_first == NULL){
17         ptr_last = newNode;
18     }
19     else {
20         ptr_first->prev = newNode;
21     }
22     ptr_first = newNode;
23 }
24
25 void add_last(int value) {
26     Node *newNode = new Node(value, ptr_last, NULL);
27     if (ptr_last == NULL) {
28         ptr_first = newNode;
29     } else {
30         ptr_last->next = newNode;
31     }
32     ptr_last = newNode;
33 }
34
35 void add_target(int targetValue, int newValue){
36     Node *current = ptr_first;
37     while (current != NULL && current->data != targetValue){
38         current = current->next;
39     }
40
41     if (current != NULL){
42         add_
```

Code:

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

struct Node {
    int data;
    Node* prev;
    Node* next;
};

Node *ptr_first = NULL;
Node *ptr_last = NULL;

void add_first(int value) {
    Node *newNode = new Node {value, NULL, ptr_first};

    if (ptr_first == NULL){
        ptr_last = newNode;
    }
    else {
        ptr_first->prev = newNode;
    }
    ptr_first = newNode;
}

void add_last(int value) {
    Node *newNode = new Node{value, ptr_last, NULL};
    if (ptr_last == NULL) {
        ptr_first = newNode;
    } else {
        ptr_last -> next = newNode;
    }
    ptr_last = newNode;
}

void add_target(int targetValue, int newValue){
    Node *current = ptr_first;
    while (current != NULL && current->data != targetValue){
        current = current->next;
    }

    if (current != NULL){
        add_last(newValue);
    }
    else{
```

```

        Node *newNode = new Node{newValue, current, current-
>next};
        current->next->prev = newNode;
        current->next = newNode;
    }
}

void view(){
    Node *current = ptr_first;
    if (current == NULL){
        cout << "List Kosong\n";
        return;
    }
    while (current != NULL)
    {
        cout << current->data << (current->next != NULL ? " <->
" : "");
        current = current->next;
    }
    cout<< endl;
}

void delete_first(){
    if (ptr_first == NULL)
        return;

    Node *temp = ptr_first;

    if (ptr_first == ptr_last){
        ptr_first = NULL;
        ptr_last = NULL;
    }
    else{
        ptr_first = ptr_first->next;
        ptr_first->prev = NULL;
    }
    delete temp;
}

void delete_last(){
    if (ptr_last == NULL)
        return;

    Node *temp = ptr_last;

    if (ptr_first == ptr_last){
        ptr_first = NULL;

```

```

        ptr_last = NULL;
    }
    else{
        ptr_last = ptr_last->prev;
        ptr_last->next = NULL;
    }
    delete temp;
}

void delete_target(int targetValue){
    Node *current = ptr_first;
    while (current != NULL && current->data != targetValue){
        current = current->next;
    }

    if (current != NULL){
        if (current == ptr_first){
            delete_first();
            return;
        }
        if (current == ptr_last)
        {
            delete_last();
            return;
        }
        current->prev->next = current->next;
        current->next->prev = current->prev;
        delete current;
    }
}

void edit_node(int targetVaue, int newValue){
    Node *current = ptr_first;
    while (current != NULL && current->data != targetVaue)
    {
        current = current->next;
    }
    if (current != NULL)
    {
        current->data = newValue;
    }
}

int main() {

```

```

add_first(10);
add_first(5);
add_last(20);
cout << "Awal\t\t\t: ";
view();

delete_first();
cout << "Setelah delete_first\t: ";
view();
delete_last();
cout << "Setelah delete_last\t: ";
view();

add_last(30);
add_last(40);
cout << "Setelah tambah\t\t: ";
view();

delete_target(30);
cout << "Setelah delete_target\t: ";
view();
}

```

Screenshoot Output :

Warning: PowerShell detected that you might be using a screen reader and has disabled PSReadLine for compatibility purposes. If you want to re-enable it, run 'Import-Module PSReadLine'.

```

PS C:\StrukDat\Modul 6> cd "c:\StrukDat\Modul 6\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Awal          : 5 <-> 10 <-> 20
Setelah delete_first : 10 <-> 20
Setelah delete_last  : 10
Setelah tambah      : 10 <-> 30 <-> 40
Setelah delete_target : 10 <-> 40
PS C:\StrukDat\Modul 6>

```

Besthian Guido Rafael Simbolon
103112430258
IF - 12 - 06

Ln 3, Col 13 56 character Plain text 100% Wind UTF-8

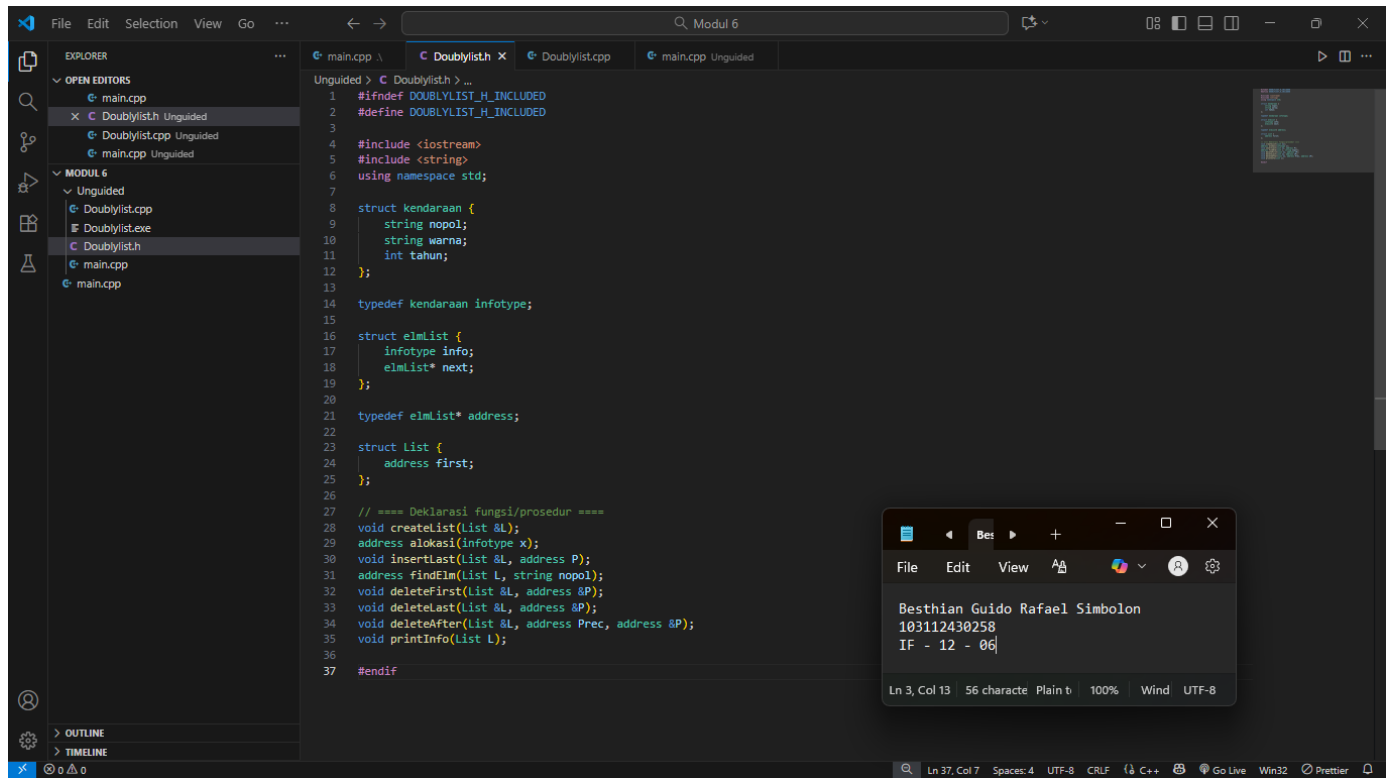
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF C++ Go Live Win32 Prettier

Deskripsi : Program ini menggunakan Doubly Linked List untuk menyimpan data berupa angka. Setiap data disimpan dalam *node* yang memiliki pointer ke elemen sebelumnya (prev) dan pointer ke elemen berikutnya (next), sehingga data dapat ditelusuri bolak-balik.

C. Unguided/Latihan

Unguided 1

Doublylis.h



```
#ifndef DOUBLYLIST_H_INCLUDED
#define DOUBLYLIST_H_INCLUDED

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

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

typedef kendaraan infotype;

struct elmList {
```

```
        infotype info;
        elmList* next;
};

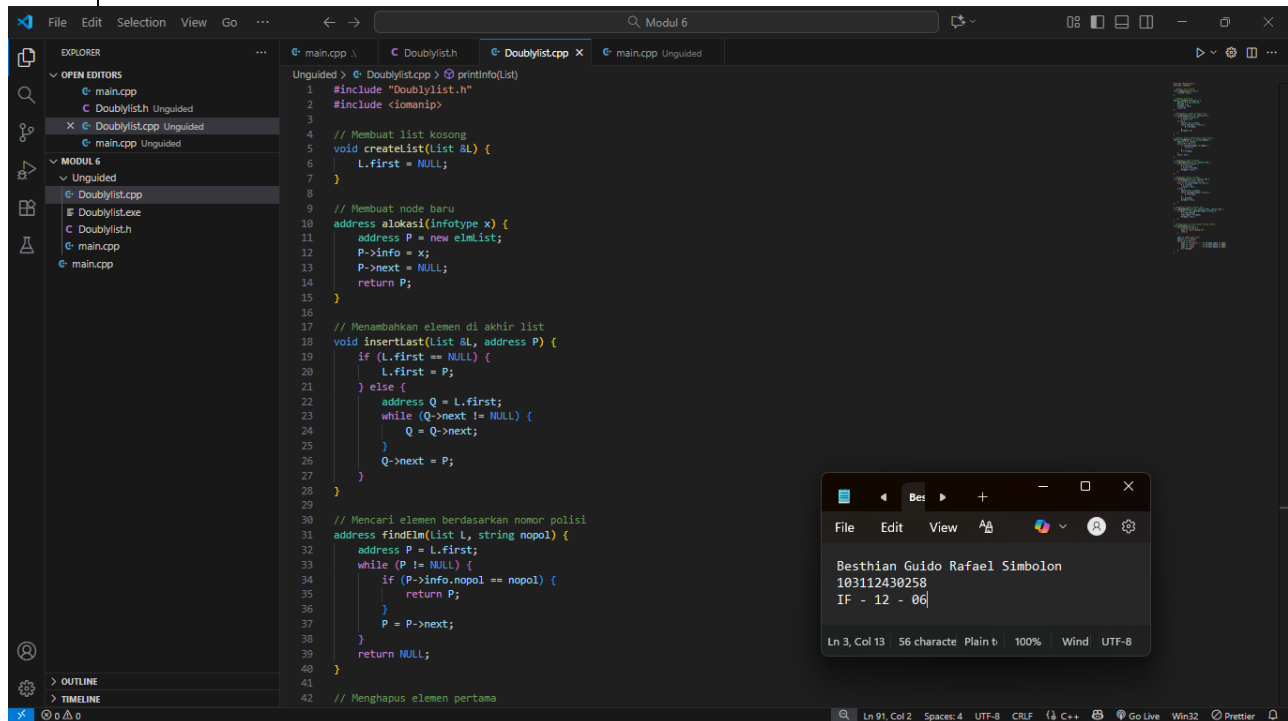
typedef elmList* address;

struct List {
    address first;
};

// ==== Deklarasi fungsi/prosedur ====
void createList(List &L);
address alokasi(infotype x);
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);
void printInfo(List L);

#endif
```


Doublylist.cpp



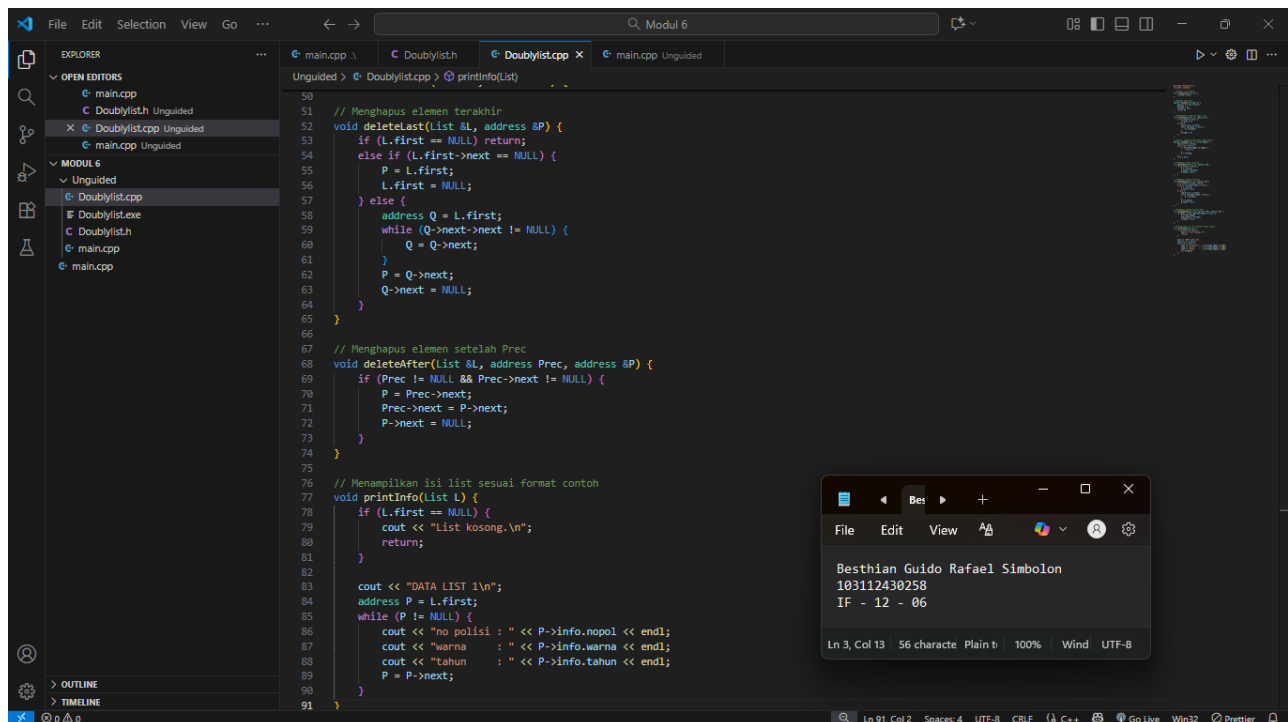
```
1 #include "Doublylist.h"
2 #include <iomanip>
3
4 // Membuat list kosong
5 void createlist(List &L) {
6     L.first = NULL;
7 }
8
9 // Membuat node baru
10 address alokasi(infotype x) {
11     address P = new elmList;
12     P->info = x;
13     P->next = NULL;
14     return P;
15 }
16
17 // Menambahkan elemen di akhir list
18 void insertLast(List &L, address P) {
19     if (L.first == NULL) {
20         L.first = P;
21     } else {
22         address Q = L.first;
23         while (Q->next != NULL) {
24             Q = Q->next;
25         }
26         Q->next = P;
27     }
28 }
29
30 // Mencari elemen berdasarkan nomor polisi
31 address findElm(List L, string nopol) {
32     address P = L.first;
33     while (P != NULL) {
34         if (P->info.nopol == nopol) {
35             return P;
36         }
37         P = P->next;
38     }
39     return NULL;
40 }
41
42 // Menghapus elemen pertama
```

Best

File Edit View A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Besthian Guido Rafael Simbolon
103112430258
IF - 12 - 06

Ln 3, Col 13 56 character Plain text 100% Wind UTF-8



```
50
51 // Menghapus elemen terakhir
52 void deletelast(List &L, address &P) {
53     if (L.first == NULL) return;
54     else if (L.first->next == NULL) {
55         P = L.first;
56         L.first = NULL;
57     } else {
58         address Q = L.first;
59         while (Q->next->next != NULL) {
60             Q = Q->next;
61         }
62         P = Q->next;
63         Q->next = NULL;
64     }
65 }
66
67 // Menghapus elemen setelah Prec
68 void deleteAfter(List &L, address Prec, address &P) {
69     if (Prec != NULL && Prec->next != NULL) {
70         P = Prec->next;
71         Prec->next = P->next;
72         P->next = NULL;
73     }
74 }
75
76 // Menampilkan isi list sesuai format contoh
77 void printInfo(List L) {
78     if (L.first == NULL) {
79         cout << "List kosong.\n";
80         return;
81     }
82     cout << "DATA LIST 1\n";
83     address P = L.first;
84     while (P != NULL) {
85         cout << "no polisi : " << P->info.nopol << endl;
86         cout << "warna : " << P->info.warna << endl;
87         cout << "tahun : " << P->info.tahun << endl;
88         P = P->next;
89     }
90 }
91
```

Best

File Edit View A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Besthian Guido Rafael Simbolon
103112430258
IF - 12 - 06

Ln 3, Col 13 56 character Plain text 100% Wind UTF-8

```

#include "Doublylist.h"
#include <iomanip>

// Membuat list kosong
void createList(List &L) {
    L.first = NULL;
}

// Membuat node baru
address alokasi(infotype x) {
    address P = new elmList;
    P->info = x;
    P->next = NULL;
    return P;
}

// Menambahkan elemen di akhir list
void insertLast(List &L, address P) {
    if (L.first == NULL) {
        L.first = P;
    } else {
        address Q = L.first;
        while (Q->next != NULL) {
            Q = Q->next;
        }
        Q->next = P;
    }
}

// Mencari elemen berdasarkan nomor polisi
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;
}

// Menghapus elemen pertama
void deleteFirst(List &L, address &P) {
    if (L.first != NULL) {
        P = L.first;
        L.first = P->next;
        P->next = NULL;
    }
}

```

```

    }
}

// Menghapus elemen terakhir
void deleteLast(List &L, address &P) {
    if (L.first == NULL) return;
    else if (L.first->next == NULL) {
        P = L.first;
        L.first = NULL;
    } else {
        address Q = L.first;
        while (Q->next->next != NULL) {
            Q = Q->next;
        }
        P = Q->next;
        Q->next = NULL;
    }
}

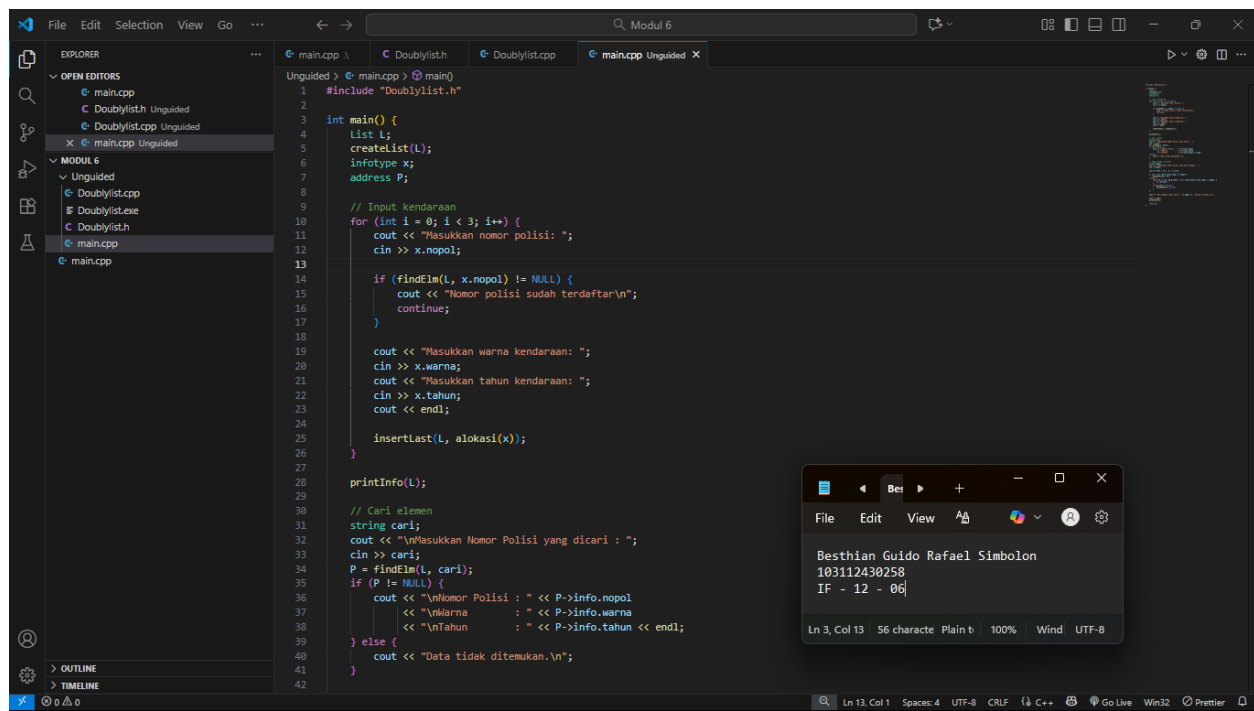
// Menghapus elemen setelah Prec
void deleteAfter(List &L, address Prec, address &P) {
    if (Prec != NULL && Prec->next != NULL) {
        P = Prec->next;
        Prec->next = P->next;
        P->next = NULL;
    }
}

// Menampilkan isi list sesuai format contoh
void printInfo(List L) {
    if (L.first == NULL) {
        cout << "List kosong.\n";
        return;
    }

    cout << "DATA LIST 1\n";
    address P = L.first;
    while (P != NULL) {
        cout << "no polisi : " << P->info.nopol << endl;
        cout << "warna      : " << P->info.warna << endl;
        cout << "tahun       : " << P->info.tahun << endl;
        P = P->next;
    }
}

```

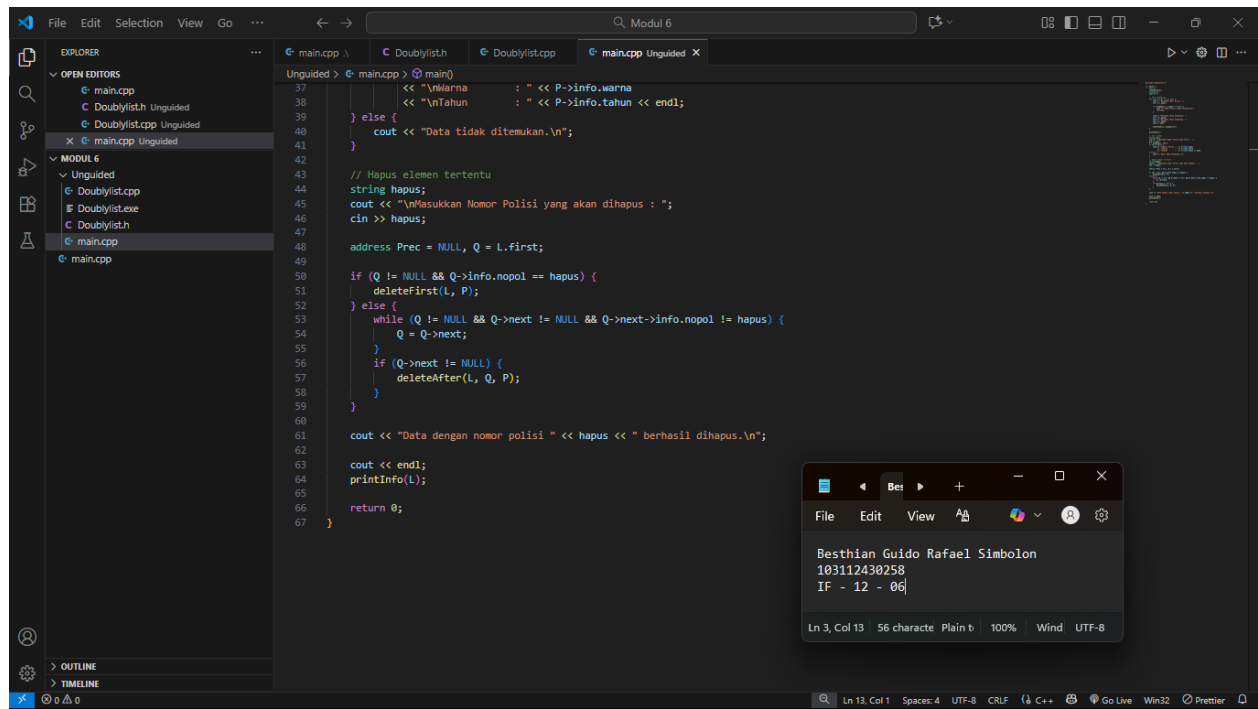
Main.cpp



```
1 #include "Doublylist.h"
2
3 int main() {
4     List L;
5     createList(L);
6     infotype X;
7     address P;
8
9     // Input kendaraan
10    for (int i = 0; i < 3; i++) {
11        cout << "Masukkan nomor polisi : ";
12        cin >> x.nopol;
13
14        if (findElm(L, x.nopol) != NULL) {
15            cout << "Nomor polisi sudah terdaftar\n";
16            continue;
17        }
18
19        cout << "Masukkan warna kendaraan : ";
20        cin >> x.warna;
21        cout << "Masukkan tahun kendaraan : ";
22        cin >> x.tahun;
23        cout << endl;
24
25        insertLast(L, alokasi(x));
26    }
27
28    printInfo(L);
29
30    // Cari elemen
31    string cari;
32    cout << "\nMasukkan Nomor Polisi yang dicari : ";
33    cin >> cari;
34    P = findElm(L, cari);
35    if (P != NULL) {
36        cout << "\nNomor Polisi : " << P->info.nopol
37              << "\nWarna      : " << P->info.warna
38              << "\nTahun      : " << P->info.tahun << endl;
39    } else {
40        cout << "Data tidak ditemukan.\n";
41    }
42 }
```

Besthian Guido Rafael Simbolon
103112430258
IF - 12 - 06

Ln 3, Col 13 56 character Plain t 100% Wind UTF-8



Code :

```
#include "Doublylist.h"

int main() {
    List L;
    createList(L);
    infotype x;
    address P;

    // Input kendaraan
    for (int i = 0; i < 3; i++) {
        cout << "Masukkan nomor polisi: ";
        cin >> x.nopol;

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

        cout << "Masukkan warna kendaraan: ";
        cin >> x.warna;
        cout << "Masukkan tahun kendaraan: ";
        cin >> x.tahun;
        cout << endl;

        insertLast(L, alokasi(x));
    }

    printInfo(L);

    // Cari elemen
    string cari;
    cout << "\nMasukkan Nomor Polisi yang dicari : ";
    cin >> cari;
    P = findElm(L, cari);
    if (P != NULL) {
        cout << "\nNomor Polisi : " << P->info.nopol
            << "\nWarna : " << P->info.warna
            << "\nTahun : " << P->info.tahun << endl;
    } else {
        cout << "Data tidak ditemukan.\n";
    }

    // Hapus elemen tertentu
    string hapus;
```

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

address Prec = NULL, Q = L.first;

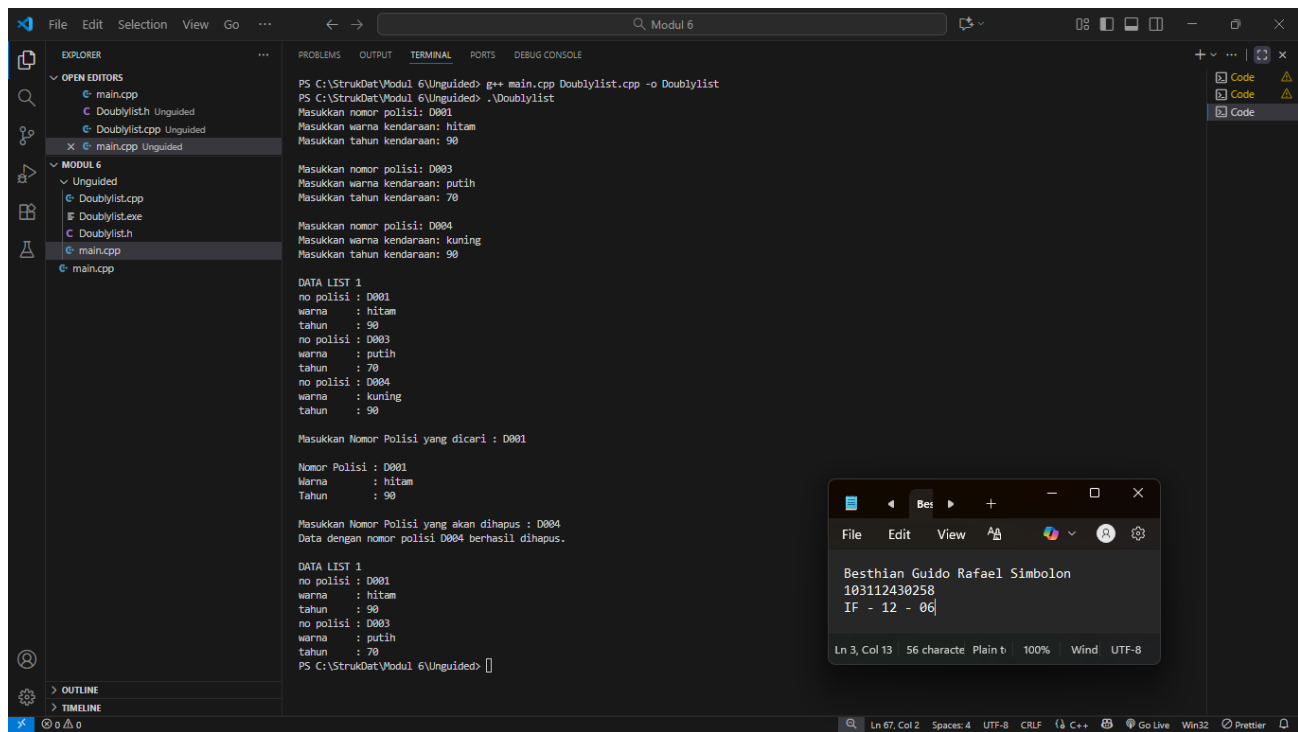
if (Q != NULL && Q->info.nopol == hapus) {
    deleteFirst(L, P);
} else {
    while (Q != NULL && Q->next != NULL && Q->next->info.nopol != hapus) {
        Q = Q->next;
    }
    if (Q->next != NULL) {
        deleteAfter(L, Q, P);
    }
}

cout << "Data dengan nomor polisi " << hapus << " berhasil dihapus.\n";

cout << endl;
printInfo(L);

return 0;
}
```

Screenshot Output



```
PS C:\StrukDat\Modul 6\Unguided> g++ main.cpp Doublylist.cpp -o Doublylist
PS C:\StrukDat\Modul 6\Unguided> .\Doublylist
Masukkan nomor polisi: D001
Masukkan warna kendaraan: hitam
Masukkan tahun kendaraan: 90

Masukkan nomor polisi: D003
Masukkan warna kendaraan: putih
Masukkan tahun kendaraan: 70

Masukkan nomor polisi: D004
Masukkan warna kendaraan: kuning
Masukkan tahun kendaraan: 90

DATA LIST 1
no polisi : D001
warna : hitam
tahun : 90
no polisi : D003
warna : putih
tahun : 70
no polisi : D004
warna : kuning
tahun : 90

Masukkan Nomor Polisi yang dicari : D001

Nomor Polisi : D001
Warna : hitam
Tahun : 90

Masukkan Nomor Polisi yang akan dihapus : D004
Data dengan nomor polisi D004 berhasil dihapus.

DATA LIST 1
no polisi : D001
warna : hitam
tahun : 90
no polisi : D003
warna : putih
tahun : 70
PS C:\StrukDat\Modul 6\Unguided>
```

Deskripsi

Program ini menggunakan struktur data *Linked List* untuk menyimpan data kendaraan yang terdiri dari nomor polisi, warna, dan tahun kendaraan. Pada awal program, list dibuat dalam keadaan kosong. Kemudian, pengguna diminta menginput 3 data kendaraan satu per satu. Jika nomor polisi yang dimasukkan sudah ada dalam list, maka data tersebut tidak akan ditambahkan.

Setiap data kendaraan baru akan disimpan sebagai node dan ditambahkan di bagian akhir List (*insertLast*). Setelah proses input selesai, program menampilkan seluruh data kendaraan yang sudah tersimpan.

Selanjutnya, program meminta pengguna memasukkan nomor polisi yang ingin dicari. Jika ditemukan, maka data kendaraan tersebut akan ditampilkan. Jika tidak ada, program menampilkan pesan bahwa data tidak ditemukan.

D. Referensi

Arifin, Z. (2019). *Struktur data dan algoritma dengan bahasa C++*. Yogyakarta: CV Andi Offset.

Harahap, O. S. (2020). *Struktur data lanjutan: Linked list, stack, dan queue dengan implementasi C++*. Medan: Perdana Publishing.

Kadir, A. (2018). *Pemrograman dasar C++*. Yogyakarta: Andi Publisher.

Khairil, K., & Fadlisyah, F. (2021). *Struktur data dan implementasinya dalam pemrograman*. Banda Aceh: Lembaga Penerbitan Universitas Syiah Kuala.

Suhada, G. (2022). *Struktur data dan algoritma: Konsep dan implementasi*. Bandung: Informatika Bandung.