

**LAPORAN PRAKTIKUM  
STRUKTUR DATA**

**MODUL 6  
DOUBLY LINKED LIST (1)**



**Disusun Oleh :**

NAMA : Faris Walid Awwal Aidi

NIM : 103112430133

**Dosen**

FAHRUDIN MUKTI WIBOWO

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

## A. Dasar Teori

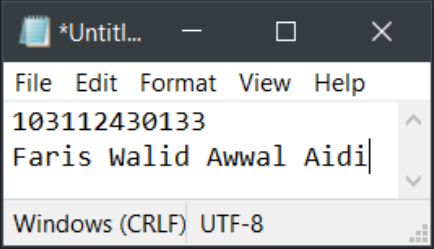
Doubly Linked List adalah sebuah linked list di mana setiap elemennya memiliki dua successor (penerus). Kedua successor tersebut adalah prev, yang merupakan pointer yang menunjuk ke elemen sebelumnya, dan next, yang merupakan pointer yang menunjuk ke elemen sesudahnya. Struktur list ini juga menggunakan dua pointer utama, yaitu first yang menunjuk ke elemen pertama list, dan last yang menunjuk ke elemen terakhir list.

Keunggulan utama dari doubly linked list dibandingkan dengan singly linked list adalah kemudahan dalam melakukan proses akses elemen. Karena setiap elemen memiliki penunjuk ke elemen sebelum (prev) dan sesudahnya (next), proses iterasi atau penelusuran data dapat dilakukan dari dua arah, baik maju (menggunakan next) maupun mundur (menggunakan prev). Proses pencarian, pembaruan (update), dan penampilan (view) data pada dasarnya sama dengan singly linked list, namun menjadi lebih mudah berkat kemampuan iterasi dua arah ini.

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

Guided

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



```

guided_DLL > main.cpp > view()
30 void add_last (int value)
34     if (ptr_last == NULL)
37     }
38     else
39     {
40         ptr_last->next = newNode;
41     }
42     ptr_last = newNode;
43 }
44
45 void add_target(int targetValue, int newValue)
46 {
47     Node *current = ptr_first;
48     while (current != NULL && current->data != targetValue)
49     {
50         current = current->next;
51     }
52
53     if (current != NULL)
54     {
55         if (current == ptr_last)
56         {
57             add_last(newValue);
58         }
59         else
60         {
61             Node *newNode = new Node(newValue, current, current->next);
62             current->next->prev = newNode;
63             current->next = newNode;
64         }
65     }
66 }
67

```

\*Untitl...

File	Edit	Format	View	Help
103112430133				
Faris Walid Awwal Aidi				

Windows (CRLF) UTF-8

```

68 void view()
69 {
70     Node *current = ptr_first;
71     if (current == NULL)
72     {
73         cout << "List kosong\n";
74         return;
75     }
76     while (current != NULL)
77     {
78         cout << current->data << (current->next != NULL ? " <-> " : "");
79         current = current->next;
80     }
81     cout << endl;
82 }
83
84 void delete_first()
85 {
86     if (ptr_first == NULL)
87         return;
88
89     Node *temp = ptr_first;
90
91     if (ptr_first == ptr_last)
92     {
93         ptr_first = NULL;
94         ptr_last = NULL;
95     }
96     else
97     {
98         ptr_first = ptr_first->next;
99         ptr_first->prev = NULL;
100     }
101     delete temp;
102 }

```

\*Untitl...

File	Edit	Format	View	Help
103112430133				
Faris Walid Awwal Aidi				

Windows (CRLF) UTF-8

```

102 }
103
104 void delete_last()
105 {
106     if (ptr_last == NULL)
107         return;
108
109     Node *temp = ptr_last;
110
111     if (ptr_first == ptr_last)
112     {
113         ptr_first = NULL;
114         ptr_last = NULL;
115     }
116     else
117     {
118         ptr_last = ptr_last->prev;
119         ptr_last->next = NULL;
120     }
121     delete temp;
122 }
123
124 void delete_target(int targetValue)
125 {
126     Node *current = ptr_first;
127     while (current != NULL && current->data != targetValue)
128     {
129         current = current->next;
130     }
131
132     if (current != NULL)
133     {
134         if (current == ptr_first)
135         {
136             delete_first();
137             return;
138         }

```

\*Untitl... — □ ×

File Edit Format View Help

103112430133

Faris Walid Awwal Aidi

Windows (CRLF) UTF-8

```

124 void delete_target(int targetValue)
132     if (current != NULL)
134         if (current == ptr_first)
138         {
139             if (current == ptr_last)
140             {
141                 delete_last();
142                 return;
143             }
144
145             current->prev->next = current->next;
146             current->next->prev = current->prev;
147             delete current;
148         }
149     }
150
151 void edit_node(int targetValue, int newValue)
152 {
153     Node *current = ptr_first;
154     while (current != NULL && current->data != targetValue)
155     {
156         current = current->next;
157     }
158
159     if (current != NULL)
160     {
161         current->data = newValue;
162     }
163 }
164
165 int main()
166 {
167     add_first(10);
168     add_first(5);
169     add_last(20);
170     cout << "Awwal\t\t\t: ";
171     view();

```

\*Untitl... — □ ×

File Edit Format View Help

103112430133

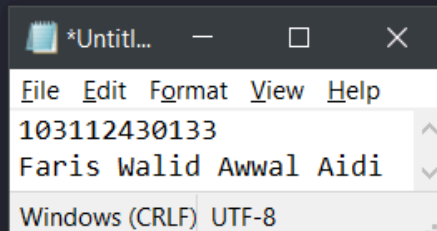
Faris Walid Awwal Aidi

Windows (CRLF) UTF-8

```

171     view();
172
173     delete_first();
174     cout << "Setelah delete_first\t: ";
175     view();
176     delete_last();
177     cout << "Setelah delete_last\t: ";
178     view();
179
180     add_last(30);
181     add_last(40);
182     cout << "Setelah tambah\t\t: ";
183     view();
184
185     delete_target(30);
186     cout << "Setelah delete_target\t: ";
187     view();
188
189     return 0;
190 }

```



## Screenshots Output

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

PS D:\C++\modul6_23okt25> cd .\guided_DLL\
PS D:\C++\modul6_23okt25\guided_DLL> .\main.exe
Awal : 5 <-> 10 <-> 20
Setelah delete_first : 10 <-> 20
Setelah delete_last : 10
Setelah tambah : 10 <-> 30 <-> 40
Setelah delete_target : 10 <-> 40
PS D:\C++\modul6_23okt25\guided_DLL>

```

Deskripsi:

Program ini merupakan implementasi dari struktur data Doubly Linked List dalam bahasa C++. Struktur dasarnya menggunakan struct Node yang memiliki tiga komponen: data untuk menyimpan nilai integer, serta dua pointer (prev dan next) yang memungkinkan navigasi dua arah (maju dan mundur). Program ini menggunakan dua pointer global, ptr\_first dan ptr\_last, untuk secara efisien melacak elemen pertama dan terakhir dalam list, yang juga mempermudah operasi penambahan dan penghapusan di kedua ujung.

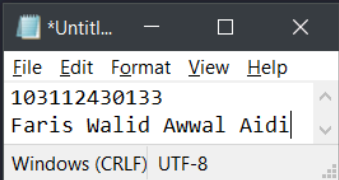
Fungsionalitas program ini mencakup operasi dasar manipulasi list. Terdapat fungsi untuk menambah data di awal (add\_first), di akhir (add\_last), dan menyisipkan data baru setelah nilai target tertentu (add\_target). Selain itu, program ini juga dilengkapi fungsi untuk menghapus data, baik elemen pertama (delete\_first), elemen terakhir (delete\_last), maupun elemen dengan nilai spesifik (delete\_target). Fungsi edit\_node disediakan untuk mengubah nilai data yang sudah ada, dan fungsi view digunakan untuk mencetak seluruh isi list dari ptr\_first hingga ptr\_last. Fungsi main() mendemonstrasikan penggunaan praktis dari fungsi-fungsi ini secara berurutan.

C. Unguided/Tugas (berisi screenshot source code & output program disertai penjelasannya)

Unguided

- doublyList.h

```
unguided_DLL > C doublyList.h > ...
1  #ifndef DOUBLYLIST_H
2  #define DOUBLYLIST_H
3
4  #include <iostream>
5  #include <string>
6
7  using namespace std;
8
9  #define Nil NULL
10 #define info(P) (P)->info
11 #define next(P) (P)->next
12 #define prev(P) (P)->prev
13 #define First(L) (L).First
14 #define Last(L) (L).Last
15
16 struct infotype {
17     string nopol;
18     string warna;
19     int thnBuat;
20 };
21
22 typedef struct Elmlist *address;
23
24 struct Elmlist {
25     infotype info;
26     address next;
27     address prev;
28 };
29
30 struct List {
31     address First;
32     address Last;
33 };
34
35 void CreateList(List &L);
36 address alokasi(infotype x);
37 void dealokasi(address P);
38
39 void insertLast(List &L, address P);
40 void printInfo(List L);
41 address findElm(List L, string nopol);
42
43 void deleteFirst(List &L, address &P);
44 void deleteLast(List &L, address &P);
45 void deleteAfter(address Prec, address &P);
46 void deleteElm(List &L, string nopol, address &P);
47
48 #endif
```





- doublyList.cpp

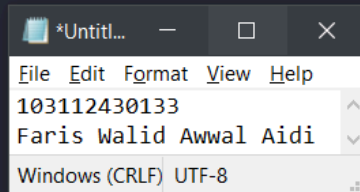
```
unguided_DLL > G doublyList.cpp > ...
1  #include "Doublylist.h"
2
3  void CreateList(List &L) {
4      First(L) = Nil;
5      Last(L) = Nil;
6  }
7
8  address alokasi(infotype x) {
9      address P = new Elmlist;
10     info(P) = x;
11     next(P) = Nil;
12     prev(P) = Nil;
13     return P;
14 }
15
16 void dealokasi(address P) {
17     delete P;
18 }
19
20 void insertLast(List &L, address P) {
21     if (First(L) == Nil) {
22         First(L) = P;
23         Last(L) = P;
24     } else {
25         next(Last(L)) = P;
26         prev(P) = Last(L);
27         Last(L) = P;
28     }
29 }
30
```

```
29 }
30
31 void printInfo(List L) {
32     address P = First(L);
33     if (First(L) == Nil) {
34         cout << "List Kosong" << endl;
35     } else {
36         while (P != Nil) {
37             cout << "Nomor Polisi : " << info(P).nopol << endl;
38             cout << "Warna          : " << info(P).warna << endl;
39             cout << "Tahun            : " << info(P).thnBuat << endl;
40             P = next(P);
41         }
42     }
43 }
44
45 address findElm(List L, string nopol) {
46     address P = First(L);
47     while (P != Nil) {
48         if (info(P).nopol == nopol) {
49             return P;
50         }
51         P = next(P);
52     }
53     return Nil;
54 }
55
56 void deleteFirst(List &L, address &P) {
57     P = First(L);
58     if (First(L) == Last(L)) {
59         First(L) = Nil;
60         Last(L) = Nil;
61     } else {
62         First(L) = next(P);
63         prev(First(L)) = Nil;
64         next(P) = Nil;
65     }
66 }
```

```

66 }
67
68 void deletelast(List &L, address &P) {
69     P = Last(L);
70     if (First(L) == Last(L)) {
71         First(L) = Nil;
72         Last(L) = Nil;
73     } else {
74         Last(L) = prev(P);
75         next(Last(L)) = Nil;
76         prev(P) = Nil;
77     }
78 }
79
80 void deleteAfter(address Prec, address &P) {
81     P = next(Prec);
82     address Q = next(P);
83     next(Prec) = Q;
84     if (Q != Nil) {
85         prev(Q) = Prec;
86     }
87     next(P) = Nil;
88     prev(P) = Nil;
89 }
90
91 void deleteElm(List &L, string nopol, address &P) {
92     P = findElm(L, nopol);
93     if (P == Nil) {
94         cout << "Data dengan nomor polisi " << nopol << " tidak ditemukan." << endl;
95     } else {
96         if (P == First(L)) {
97             deleteFirst(L, P);
98         } else if (P == Last(L)) {
99             deletelast(L, P);
100         } else {
101             address Prec = prev(P);
102             deleteAfter(Prec, P);
103         }
104         cout << "Data dengan nomor polisi " << info(P).nopol << " berhasil dihapus." << endl;
105     }
106 }

```



- main.cpp

```
unguided_DLL > main.cpp > main()
1  #include "Doublylist.h"
2
3  int main() {
4      List L;
5      CreateList(L);
6
7      infotype data;
8      address P, Pdel;
9
10     for (int i = 0; i < 4; i++) {
11         cout << endl;
12         cout << "masukkan nomor polisi: ";
13         cin >> data.nopol;
14
15         if (findElm(L, data.nopol) != Nil) {
16             cout << "nomor polisi sudah terdaftar" << endl;
17         } else {
18             cout << "masukkan warna kendaraan: ";
19             cin >> data.warna;
20             cout << "masukkan tahun kendaraan: ";
21             cin >> data.thnBuat;
22
23             P = alokasi(data);
24             insertLast(L, P);
25         }
26     }
27
28     cout << endl << "DATA LIST 1" << endl;
29     printInfo(L);
30     cout << endl;
31
32     string nopolCari;
33     cout << "Masukkan Nomor Polisi yang dicari : ";
34     cin >> nopolCari;
35
```

```
35
36     P = findElm(L, nopolCari);
37     if (P != Nil) {
38         cout << "Nomor Polisi : " << info(P).nopol << endl;
39         cout << "Warna          : " << info(P).warna << endl;
40         cout << "Tahun            : " << info(P).thnBuat << endl;
41     } else {
42         cout << "Data tidak ditemukan" << endl;
43     }
44     cout << endl;
45
46     string nopolHapus;
47     cout << "Masukkan Nomor Polisi yang akan dihapus : ";
48     cin >> nopolHapus;
49
50     deleteElm(L, nopolHapus, Pdel);
51     if (Pdel != Nil) {
52         dealokasi(Pdel);
53     }
54
55     cout << endl << "DATA LIST 1" << endl;
56     printInfo(L);
57
58     return 0;
59 }
```

## Screenshots Output 1

```
PS D:\C++\modul6_23okt25\unguided_DLL> .\doublyList.exe

masukkan nomor polisi: D8880
masukkan warna kendaraan: hitam
masukkan tahun kendaraan: 2013

masukkan nomor polisi: F4431
masukkan warna kendaraan: putih
masukkan tahun kendaraan: 2023

masukkan nomor polisi: D3451
masukkan warna kendaraan: merah
masukkan tahun kendaraan: 2019

masukkan nomor polisi: Z6608
masukkan warna kendaraan: putih
masukkan tahun kendaraan: 2020

DATA LIST 1
Nomor Polisi : D8880
Warna       : hitam
Tahun       : 2013
Nomor Polisi : F4431
Warna       : putih
Tahun       : 2023
Nomor Polisi : D3451
Warna       : merah
Tahun       : 2019
Nomor Polisi : Z6608
Warna       : putih
Tahun       : 2020
```

## Screenshots Output 2

```
Masukkan Nomor Polisi yang dicari : F4431
Nomor Polisi : F4431
Warna       : putih
Tahun       : 2023
```

## Screenshots Output 3

```
Masukkan Nomor Polisi yang akan dihapus : D8880
Data dengan nomor polisi D8880 berhasil dihapus.

DATA LIST 1
Nomor Polisi : F4431
Warna       : putih
Tahun       : 2023
Nomor Polisi : D3451
Warna       : merah
Tahun       : 2019
Nomor Polisi : Z6608
Warna       : putih
Tahun       : 2020
PS D:\C++\modul6_23okt25\unguided_DLL> |
```

Deskripsi:

1. doublylist.h

File header ini adalah definisi ADT Anda. Isinya adalah semua "cetak biru" struktur, seperti infotype (data kendaraan), ElmList (simpul dengan pointer next dan prev), dan List (penanda First dan Last). File ini juga mendaftarkan semua prototipe (deklarasi) fungsi yang akan digunakan.

2. doublylist.cpp

File implementasi ini berisi kode logika dari semua fungsi yang dideklarasikan di .h. Di sinilah saya menulis logika untuk manajemen memori (alokasi, dealokasi), penambahan data di akhir (insertLast), pencarian data (findElm berdasarkan nopol), dan prosedur penghapusan modular (deleteElm yang memanggil deleteFirst/Last/After).

3. main.cpp

File program utama ini menggunakan ADT untuk menjalankan skenario dari soal. Program ini membuat List, meminta input data dalam loop, dan memanggil findElm untuk validasi data duplikat sebelum menambahkannya. Terakhir, program ini menjalankan simulasi pencarian (Tugas 2) dan penghapusan (Tugas 3) lalu mencetak isi list terakhir.

#### D. Kesimpulan

Berdasarkan praktikum yang sudah dilakukan, program diatas secara menyeluruh merupakan implementasi sebuah ADT Doubly Linked List yang fungsional dan terstruktur untuk mengelola data kendaraan, dengan memisahkan secara jelas antara definisi struktur dalam file Doublylist.h, implementasi logika dalam Doublylist.cpp, dan program utama dalam main.cpp. Struktur data ini menggunakan simpul (ElmList) yang memiliki pointer next dan prev, memungkinkan navigasi dua arah dan operasi yang efisien. Implementasi ini berhasil mencakup semua fungsi yang diminta, mulai dari penambahan data (insertLast), pencarian (findElm) yang juga cerdas digunakan untuk validasi data duplikat, hingga mekanisme penghapusan (deleteElm) yang modular. Mekanisme hapus ini secara efektif menangani semua kemungkinan posisi data di awal, akhir, atau tengah yang menunjukkan pemanfaatan penuh dari keunggulan pointer prev.

#### Referensi

- Raharjo, Budi. 2025. *Buku Pemrograman C++ Mudah dan Cepat Menjadi Master C*.
- Wikipedia contributors. (2024, 8 Mei). C++. Wikipedia, Ensiklopedia Bebas. Diakses pada 2 Oktober 2025, dari <https://id.wikipedia.org/wiki/C%2B%2B>