

**LAPORAN HASIL PRAKTIKUM
STRUKTUR DATA**



NAMA : INDRA FIQI RIPANI
NIM : 213010503002
KELAS : F
MODUL : III (*LINKED LIST*)

Program Studi S1 Teknik Informatika
Fakultas Teknik
Universitas Palangka Raya
Palangka Raya, Kalimantan Tengah
2022

BAB I

TUJUAN DAN LANDASAN TEORI

I. TUJUAN PRAKTIKUM

1. Mahasiswa memahami struktur data *linked list*.
2. Mahasiswa mampu menggunakan struktur data *linked list* di dalam menyelesaikan masalah pemrograman.

II. LANDASAN TEORI

Linked list adalah suatu cara untuk menyimpan data dengan struktur sehingga *programmer* dapat secara otomatis menciptakan suatu tempat baru untuk menyimpan data kapan saja diperlukan. *Linked list* dikenal juga dengan sebutan senarai berantai adalah stuktur data yang terdiri dari urutan *record* data dimana setiap *record* memiliki *field* yang menyimpan alamat/referensi dari *record* selanjutnya (di dalam urutan). Elemen data yang dihubungkan dengan *link* pada *linked list* disebut Node. Biasanya dalam suatu *linked list*, terdapat istilah *head* dan *tail*.

Jenis *linked list* (yang akan dipelajari) adalah:

1. *Single Linked List*
2. *Double Linked List*
3. *Circular Linked List*
4. *Multiple Linked List*

Ada 5 proses dasar dalam *linked list*:

1. Proses Inisialisasi
 - Proses awal → menyatakan *linked list* belum ada
 - Algoritma:

```
First = Null;  
Last = Null;
```
2. Proses Simpul Baru
 - Instruksi:

```
P = (simpul*) malloc(sizeof(simpul));
```

- Algoritma:

```
void Buat_Simpul (int x){  
    P = (simpul*) malloc(sizeof(simpul));  
    if (P!=NULL){  
        P → Info = x;  
    }  
    else cout << “simpul gagal dibuat”;  
}
```

3. Membuat Simpul Awal

Syarat:

- 1) *Linked list* belum ada
- 2) Sudah ada simpul yang akan dijadikan simpul awal

Algoritma:

```
void Awal (){  
    First = P;  
    Last = P;  
    P → Link = NULL;  
}
```

4. Menambahkan Simpul Baru ke dalam *Linked List* (*Insert*)

Syarat:

- 1) *Linked list* sudah ada.
- 2) Sudah ada simpul yang akan ditambahkan *linked list*.

- a. *Insert Kanan/Akhir*

Algoritma:

```
void Ins_Akhir (){  
    Last → Link = P;  
    Last = P;  
    P → Link = NULL;  
}
```

- b. *Insert Kiri/Awal*

Algoritma:

```
void Ins_Awal (){  
    P → Link = First;  
    First = P;  
}
```

c. *Insert Tengah*

Algoritma:

```
void Ins_Tengah (){  
    P → Link = Q → Link;  
    Q → Link = P;  
}
```

5. Menghapus Sebuah Simpul dari *Linked List* (Delete)

Syarat:

1) *Linked list* sudah ada.

a. *Delete Kanan/Akhir*

Algoritma:

```
void Del_Akhir (){  
    Free (Last);  
    Last = Q;  
    Last → Link = NULL;  
}
```

b. *Delete Kiri/Awal*

Algoritma:

```
void Del_Awal (){  
    Q = First;  
    First = Q → Link;  
    Free (Q);  
}
```

c. *Delete Tengah*

Algoritma:

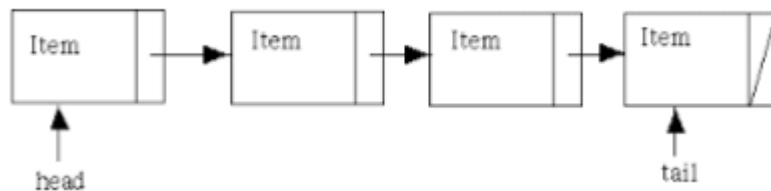
```

void Del_Tengah (){
    R = Q → Link;
    Q → Link = R → Link;
    Free (R);
}

```

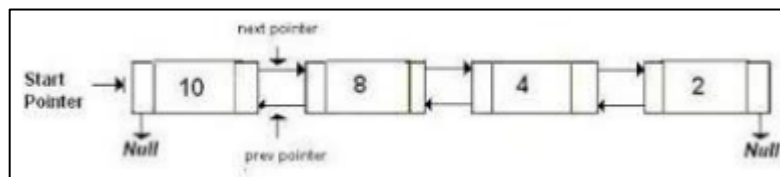
Single Linked List

Single Linked List merupakan suatu *linked list* yang hanya memiliki satu variabel *pointer* saja. Di mana *pointer* tersebut menunjuk ke node selanjutnya. Biasanya *field* pada *tail* menunjuk ke NULL.



Double Linked List

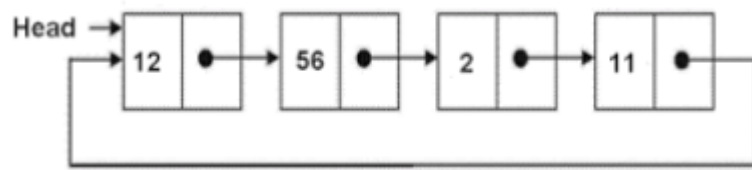
Double Linked List merupakan suatu *linked list* yang memiliki dua variabel *pointer* yaitu *pointer* yang menunjuk ke node selanjutnya dan *pointer* yang menunjuk ke node sebelumnya. Setiap *head* dan *tail*nya juga menunjuk ke NULL.



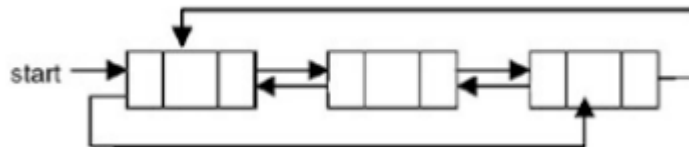
Circular Linked List

Circular Linked List merupakan suatu *linked list* dimana *tail* (node terakhir) menunjuk ke *head* (node pertama). Jadi tidak ada *pointer* yang menunjuk NULL. Ada 2 jenis *Circular Linked List*, yaitu:

- *Circular Single Linked List*

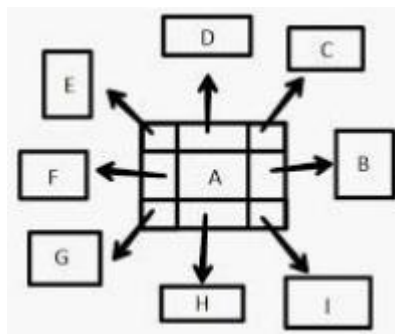


- *Circular Double Linked List*



Multiple Linked List

Multiple Linked List merupakan suatu *linked list* yang memiliki lebih dari 2 buah variabel pointer.



BAB II

PROSEDUR DAN PEMBAHASAN

I. PROSEDUR

1. Buatlah sebuah *linked list non circular* yang berisi nama lengkap dan nim anda!
2. Buatlah sebuah program dengan menggunakan *single linked list non circular* dengan fungsi-fungsi (buat dengan menggunakan menu):
 - Menambah data (dari depan dan dari belakang)
 - Menghapus data (dari depan dan dari belakang)
 - Mencetak data
3. Buat sebuah program *double linked list* dengan fungsi-fungsi (buat dengan menggunakan menu):
 - Menambahkan data (dari depan sesudah simpul dan dari belakang sesudah simpul)
 - Menghapus data (dari tengah)
 - Data yang dimasukkan langsung ke tampil dalam menu

II. PEMBAHASAN

Program pertama adalah pengaplikasian *single link list non circular* dalam pengelolaan data mahasiswa. Pada program harus terdapat perintah atau fungsi untuk menambahkan data dari depan dan belakang, menghapus data dari depan dan belakang, dan mencetak keseluruhan data. Berikut merupakan *source code* program.

```
#include <iostream>
#include <limits>
#include <conio.h>

using namespace std;
```

```
struct node{
    string nama, nim;
    node *next;
};

node *head, *tail;
```

Penjelasan:

Dideklarasikan “#include <iostream>”, “#include <limits>”, dan “#include <conio.h>” pada bagian *header*. Fungsinya adalah agar program dapat mengeksekusi perintah-perintah yang terdapat di dalam *library* tersebut, seperti “cin”, “cout”, “getch”, dan lain-lain. Lalu “using namespace std;” berfungsi agar program dapat menggunakan semua fungsi yang akan digunakan di dalam program. Selanjutnya dideklarasikan suatu struct dengan nama ‘node’ yang di dalamnya terdapat variabel ‘nama’ dan ‘nim’ yang bertipe data *string* dan ‘next’ yang bertipe struct itu sendiri. Fungsi dari ‘next’ ini adalah sebagai pointer di dalam *linked list* yang akan kita buat nanti. Terakhir dideklarasikan variabel ‘*head’ dan ‘*tail’ bertipe ‘node’ yang merupakan pangkal dan ujung dari data di dalam suatu *linked list*.

```
void printAll(){
    node *bantu;
    if(head == NULL){
        cout << "\nBelum ada data.\n";
    } else {
        bantu = head;
        while(bantu != NULL){
            cout << "\nNAMA \t: " << bantu -> nama << endl;
            cout << "NIM \t: " << bantu -> nim << endl;
            bantu = bantu -> next;
        }
    }
}
```



```
        cout << "\nPress Any Button to Continue...";
        getch();
    }
```

Penjelasan:

Agar mempermudah *programmer* dalam membuat suatu program, maka dibuatlah prosedur-prosedur. Prosedur ‘printAll’ adalah prosedur yang apabila dipanggil di dalam program utama, dia akan mencetak semua data yang sudah diinputkan oleh *user*. Pada prosedur ini dideklarasikan ‘*bantu’ yang bertipe ‘node’. Jika pangkal data masih kosong, maka program mencetak “Belum ada data.” pada layar. Jika pangkal data tidak kosong, maka program akan mencetak data dari pangkal data sampai ujung data. Algoritma program tersebut dapat dilakukan dengan kombinasi percabangan ‘if - else’ dan perulangan ‘while’. Lalu ada fungsi ‘getch’ agar layar menahan program pada tampilan yang ada sampai *user* menekan tombol apa pun pada *keyboardnya*.

```
void inputDepan(){
    node *baru;
    baru = new node;
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
    cout << "\n=== PROSES MEMASUKKAN DATA ===\n\n";
    cout << "Masukkan Nama\t : "; getline(cin, baru -> nama);
    cout << "Masukkan NIM \t : "; getline(cin, baru -> nim);
    baru -> next = NULL;
    if(head == NULL){
        head = tail = baru;
        head -> next = NULL;
    } else {
        baru -> next = head; //pointer baru ke head sebelumnya
        head = baru;
    }
    cout << "\nData telah dimasukkan\n\n" << "=== DATA TERBARU
```

```
==";  
    printAll();  
}
```

Penjelasan:

Selanjutnya dibuat prosedur ‘inputDepan’. Prosedur ini berfungsi untuk membantu *user* agar dapat memasukkan suatu data dari depan. Dideklarasikan ‘*baru’ bertipe ‘node’ yang akan menampung objek-objek ‘node’. Program akan meminta *user* untuk menginputkan nama dan NIM mahasiswa yang akan menjadi pangkal data. Setelah data diinputkan, maka *pointer* ‘baru’ akan mengarahkannya ke *NULL* atau kosong. Jika data masih kosong, maka data yang diinputkan akan menjadi pangkal dan ujung data, sehingga *pointernya* akan mengarah ke *NULL*. Jika sudah ada data yang diinputkan, maka data baru yang dimasukkan akan menjadi pangkal data yang baru dan *pointernya* akan mengarah pada pangkal data sebelumnya. Lalu, program akan langsung mencetak semua data dengan memanggil prosedur ‘printAll’.

```
void inputBelakang(){  
    node *baru;  
    baru = new node;  
    cin.ignore(numeric_limits<streamsize>::max(), '\n');  
    cout << "\n=== PROSES MEMASUKKAN DATA ===\n\n";  
    cout << "Masukkan Nama\t : "; getline(cin, baru -> nama);  
    cout << "Masukkan NIM \t : "; getline(cin, baru -> nim);  
    baru -> next = NULL;  
    if(head == NULL){  
        head = baru;  
    } else {  
        tail -> next = baru; //ekor ke node baru  
    }  
    tail = baru;  
    tail -> next = NULL;
```

```

        cout << "\nData telah dimasukkan\n\n" << "== DATA TERBARU
        ==";
        printAll();
    }

```

Penjelasan:

Prosedur ‘inputBelakang’ berfungsi untuk membantu *user* memasukkan data dari belakang. Dideklarasikan ‘*baru’ yang bertipe ‘node’ dan dibuat objek ‘baru’ dari ‘node’. Program akan meminta *user* untuk memasukkan nama dan NIM mahasiswa. Lalu *pointer* dari ‘baru’ diarahkan ke *NULL*. Jika data masih kosong, maka data yang diinputkan akan menjadi *head*. Jika data sudah ada sebelumnya, maka *pointer* dari *tail* yang ada akan diarahkan ke data yang baru dimasukkan. Sehingga data terbaru akan menjadi *tail* dan *pointer* dari *tail* yang baru akan diarahkan ke *NULL*. Lalu, program akan langsung mencetak semua data dengan memanggil prosedur ‘printAll’.

```

void hapusDepan(){
    node *hapus;
    if (head == NULL){
        cout << "Data masih Kosong\n";
    } else {
        cout<< "\nData telah dihapus\n\n" << "== DATA TERBARU
        ==\n";
        hapus = head;
        head = head -> next;
        delete hapus;
    }
    printAll();
}

```

Penjelasan:

Prosedur ‘hapusDepan’ berfungsi untuk membantu *user* menghapus data dari depan. Dideklarasikan ‘*hapus’ yang bertipe ‘node’. Jika data belum ada, maka program akan mencetak “Data masih kosong.”. Jika data sudah ada, maka

nilai *head* akan dimasukkan ke ‘hapus’ dan *head* yang ada akan mengubah nilainya menjadi data setelah *head* dan data ‘hapus’ akan dihapus. Lalu program akan mencetak semua data dengan memanggil prosedur ‘printAll’.

```
void hapusBelakang(){
    node *bantu, *hapus;
    if(head == NULL){
        cout << "Data masih Kosong\n";
    } else if(head == tail){
        cout << "\nData telah dihapus\n";
        hapus = head;
        head = head -> next;
        delete hapus;
    } else {
        cout<< "\nData telah dihapus\n\n" << "== DATA TERBARU
        ==\n";
        bantu = head;
        hapus = tail;
        while(bantu -> next != tail){
            bantu = bantu -> next;
        }
        tail = bantu;
        tail -> next = NULL;
        delete hapus;
    }
    printAll();
}
```

Penjelasan:

Prosedur ‘hapusBelakang’ berfungsi untuk membantu *user* agar dapat menghapus suatu data dari belakang. Dideklarasikan ‘*bantu’ dan ‘*hapus’ yang bertipe ‘node’. Jika data masih kosong, maka program akan mencetak “Data masih kosong”. Jika *head* sama dengan *tail*, maka *head* akan memasuk-

kan nilainya ke 'hapus' dan *head* yang ada akan merubah nilainya menjadi data setelah *head* awal. Lalu 'hapus' akan dihapus. Selain dari dua kondisi sebelumnya, maka program akan memasukkan nilai *head* ke 'hapus' dan nilai *tail* ke 'hapus'. Selama nilai setelah 'bantu' bukan *tail*, maka program akan terus mengeser nilai 'bantu'. Setelah data di sebelah 'bantu' adalah *tail* maka perulangan pun berhenti. Lalu nilai *tail* akan berubah menjadi 'bantu', dan *pointer* pada *tail* yang baru akan menunjuk pada *NULL*. Setelah itu, 'hapus' akan dihapus dan program akan mencetak semua data.

```
int main(){
    int pil;
    do{
        system("cls");
        cout << "=====\n"
              << "  MENU UTAMA LINKED LIST\n"
              << "=====\n"
              << "==\n"
              << "== 1. Input Data dari Depan  ==\n"
              << "== 2. Input Data dari Belakang ==\n"
              << "== 3. Hapus Data dari Depan  ==\n"
              << "== 4. Hapus Data dari Belakang ==\n"
              << "== 5. Tampilkan Semua Data  ==\n"
              << "== 6. Keluar                ==\n\n"
              << "Pilih Menu \t: ";
        cin >> pil;
        switch (pil){
            case 1:{
                inputDepan();
                break;
            }
            case 2:{
                inputBelakang();
```

```

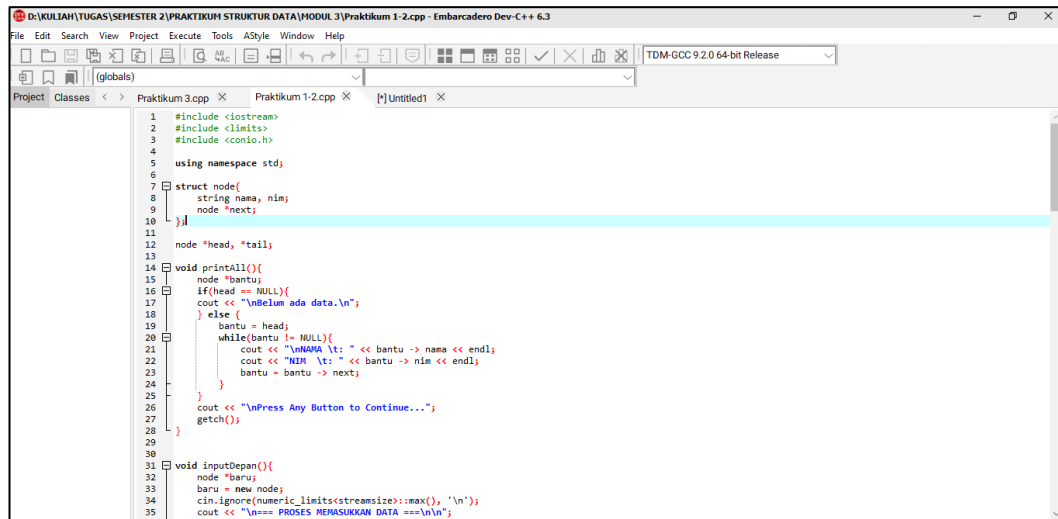
        break;
    }
    case 3:{
        hapusDepan();
        break;
    }
    case 4:{
        hapusBelakang();
        break;
    }
    case 5:{
        printAll();
        break;
    }
    case 6:{
        cout << "\nProgram Selesai";
        break;
    }
    default:{
        cout << "\nMasukan tidak valid.\n" << endl;
        cout << "\nPress Any Button to Continue...";
        getch();
    }
}
} while (pil != 6);
return 0;
}

```

Penjelasan:

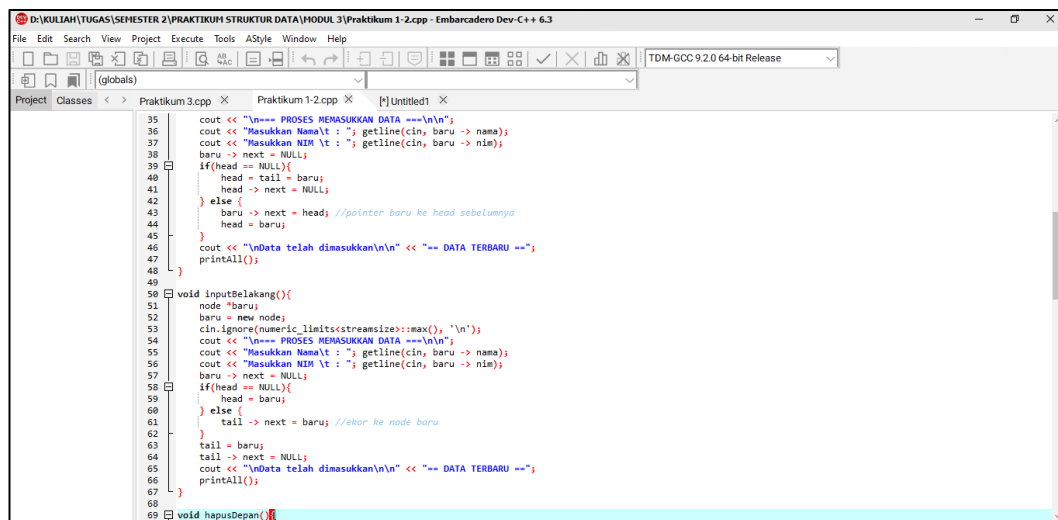
Bagian ini disebut *method main* atau program utama. Dideklarasikan variabel ‘pil’ yang bertipe data *integer* untuk menampung nilai dari inputan *user* saat memilih menu nantinya. Dilakukan perulangan menggunakan ‘do-while’

dengan kondisi perulangan akan berhenti saat *user* memasukkan ‘6’ saat memilih menu. Program akan mencetakkan ke layar seperti yang ada di dalam *source code* dan meminta masukkan pengguna. Percabangan yang digunakan adalah ‘switch(case)’. Karena telah dibuat prosedur-prosedur sebelum *method main*, maka pada program utama cukup memanggil prosedur-prosedurnya saja sesuai dengan fungsinya.



```
1 #include <iostream>
2 #include <limits>
3 #include <conio.h>
4
5 using namespace std;
6
7 struct node{
8     string nama, nim;
9     node *next;
10 };
11
12 node *head, *tail;
13
14 void printAll(){
15     node *bantu;
16     if(head == NULL){
17         cout << "\nBelum ada data.\n";
18     } else {
19         bantu = head;
20         while(bantu != NULL){
21             cout << "\nNama \t: " << bantu->nama << endl;
22             cout << "NIM \t: " << bantu->nim << endl;
23             bantu = bantu->next;
24         }
25     }
26     cout << "\nPress Any Button to Continue...";
27     getch();
28 }
29
30 void inputDepan(){
31     node *baru;
32     baru = new node;
33     cin.ignore(numeric_limits<streamsize>::max(), '\n');
34     cout << "\n== PROSES MEMASUKKAN DATA ==\n\n";
35 }
```

Gambar 1.1 Source Code Program (1)



```
35     cout << "\n== PROSES MEMASUKKAN DATA ==\n\n";
36     cout << "Masukkan Nama\t : "; getline(cin, baru->nama);
37     cout << "Masukkan NIM \t : "; getline(cin, baru->nim);
38     baru->next = NULL;
39     if(head == NULL){
40         head = tail = baru;
41         head->next = NULL;
42     } else {
43         baru->next = head; //pointer baru ke head sebelumnya
44         head = baru;
45     }
46     cout << "\nData telah dimasukkan\n\n" << "== DATA TERBARU ==";
47     printAll();
48 }
49
50 void inputBelakang(){
51     node *baru;
52     baru = new node;
53     cin.ignore(numeric_limits<streamsize>::max(), '\n');
54     cout << "\n== PROSES MEMASUKKAN DATA ==\n\n";
55     cout << "Masukkan Nama\t : "; getline(cin, baru->nama);
56     cout << "Masukkan NIM \t : "; getline(cin, baru->nim);
57     baru->next = NULL;
58     if(head == NULL){
59         head = baru;
60     } else {
61         tail->next = baru; //ekor ke node baru
62     }
63     tail = baru;
64     tail->next = NULL;
65     cout << "\nData telah dimasukkan\n\n" << "== DATA TERBARU ==";
66     printAll();
67 }
68
69 void hapusDepan(){
```

Gambar 1.2 Source Code Program (2)

```

69 void hapusDepan(){
70     node *hapus;
71     if (head == NULL){
72         cout << "Data masih Kosong\n";
73     }
74     else {
75         cout<< "\nData telah dihapus\n\n" << "== DATA TERBARU ==\n";
76         hapus = head;
77         head = head -> next;
78         delete hapus;
79     }
80     printAll();
81 }
82
83 void hapusBelakang(){
84     node *bantu, *hapus;
85     if(head == NULL){
86         cout << "Data masih Kosong\n";
87     }
88     else if(head == tail){
89         cout << "\nData telah dihapus\n";
90         hapus = head;
91         head = head -> next;
92         delete hapus;
93     }
94     else {
95         cout<< "\nData telah dihapus\n\n" << "== DATA TERBARU ==\n";
96         bantu = head;
97         hapus = tail;
98         while(bantu -> next != tail){
99             bantu = bantu -> next;
100         }
101         delete hapus;
102         tail = bantu;
103         tail -> next = NULL;
104     }
105     printAll();
106 }
107
108 }
109
110 int main(){
111     int pil;
112     do{
113         system("cls");
114         cout << "=====n"
115             << "  MENU UTAMA LINKED LIST\n"
116             << "=====n"
117             << "== 1. Input Data dari Depan ==\n"
118             << "== 2. Input Data dari Belakang ==\n"
119             << "== 3. Hapus Data dari Depan ==\n"
120             << "== 4. Hapus Data dari Belakang ==\n"
121             << "== 5. Tampilkan Semua Data ==\n"
122             << "== 6. Keluar ==\n\n";
123         cin >> pil;
124         switch (pil){
125             case 1:{
126                 inputDepan();
127                 break;
128             }
129             case 2:{
130                 inputBelakang();
131                 break;
132             }
133             case 3:{
134                 hapusDepan();
135                 break;
136             }
137             case 4:{
138                 hapusBelakang();
139                 break;
140             }
141             case 5:{
142                 printAll();
143                 break;
144             }
145             case 6:{
146                 cout << "\nProgram Selesai";
147                 break;
148             }
149             default:{
150                 cout << "\nMasukan tidak valid.\n" << endl;
151                 cout << "\nPress Any Button to Continue...";
152                 getch();
153             }
154         }
155     } while (pil != 6);
156     return 0;
157 }

```

Gambar 1.3 Source Code Program (3)

```

103 }
104
105 int main(){
106     int pil;
107     do{
108         system("cls");
109         cout << "=====n"
110             << "  MENU UTAMA LINKED LIST\n"
111             << "=====n"
112             << "== 1. Input Data dari Depan ==\n"
113             << "== 2. Input Data dari Belakang ==\n"
114             << "== 3. Hapus Data dari Depan ==\n"
115             << "== 4. Hapus Data dari Belakang ==\n"
116             << "== 5. Tampilkan Semua Data ==\n"
117             << "== 6. Keluar ==\n\n";
118         cin >> pil;
119         switch (pil){
120             case 1:{
121                 inputDepan();
122                 break;
123             }
124             case 2:{
125                 inputBelakang();
126                 break;
127             }
128             case 3:{
129                 hapusDepan();
130                 break;
131             }
132             case 4:{
133                 hapusBelakang();
134                 break;
135             }
136             case 5:{
137                 printAll();
138                 break;
139             }
140             case 6:{
141                 cout << "\nProgram Selesai";
142                 break;
143             }
144             default:{
145                 cout << "\nMasukan tidak valid.\n" << endl;
146                 cout << "\nPress Any Button to Continue...";
147                 getch();
148             }
149         }
150     } while (pil != 6);
151     return 0;
152 }

```

Gambar 1.4 Source Code Program (4)

```

119     cin >> pil;
120     switch (pil){
121         case 1:{
122             inputDepan();
123             break;
124         }
125         case 2:{
126             inputBelakang();
127             break;
128         }
129         case 3:{
130             hapusDepan();
131             break;
132         }
133         case 4:{
134             hapusBelakang();
135             break;
136         }
137         case 5:{
138             printAll();
139             break;
140         }
141         case 6:{
142             cout << "\nProgram Selesai";
143             break;
144         }
145         default:{
146             cout << "\nMasukan tidak valid.\n" << endl;
147             cout << "\nPress Any Button to Continue...";
148             getch();
149         }
150     } while (pil != 6);
151     return 0;
152 }

```

Gambar 1.5 Source Code Program (5)


```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 1

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Indra
Masukkan NIM : 123

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Indra
NIM : 123

Press Any Button to Continue...

```

Gambar 2.1 *Output Menu 1 (1)*

```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 1

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Fiqi
Masukkan NIM : 456

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Fiqi
NIM : 456
NAMA : Indra
NIM : 123

Press Any Button to Continue...

```

Gambar 2.2 *Output Menu 1 (2)*

```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 2

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Ripani
Masukkan NIM : 789

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Fiqi
NIM : 456
NAMA : Indra
NIM : 123
NAMA : Ripani
NIM : 789

Press Any Button to Continue...

```

Gambar 2.3 *Output Menu 2 (1)*

```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 2

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Mahasiswa
Masukkan NIM : 000

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Fiqi
NIM : 456
NAMA : Indra
NIM : 123
NAMA : Ripani
NIM : 789
NAMA : Mahasiswa
NIM : 000

Press Any Button to Continue...

```

Gambar 2.4 *Output Menu 2 (2)*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 3

Data telah dihapus

== DATA TERBARU ==

NAMA : Indna
NIM : 123

NAMA : Ripani
NIM : 789

NAMA : Mahasiswa
NIM : 000

Press Any Button to Continue...
```

Gambar 2.5 *Output* Menu 3

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 4

Data telah dihapus

== DATA TERBARU ==

NAMA : Indna
NIM : 123

NAMA : Ripani
NIM : 789

Press Any Button to Continue...
```

Gambar 2.6 *Output* Menu 4

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 5

NAMA : Indna
NIM : 123

NAMA : Ripani
NIM : 789

Press Any Button to Continue...
```

Gambar 2.7 *Output* Menu 5

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 6

Program Selesai
=====
Process exited after 243.1 seconds with return value 0
Press any key to continue . . .
```

Gambar 2.8 *Output* Menu 6

Program kedua adalah pengaplikasian *double linked list* dalam pengelolaan data. *Double linked list* merupakan suatu *linked list* yang memiliki dua variabel *pointer* yaitu *pointer* yang menunjuk ke node selanjutnya dan juga *pointer* yang menunjuk ke node sebelumnya. Program yang dibuat memiliki fungsi menamb-

ahkan data dari depan sesudah simpul dan dari belakang sesudah simpul, menghapus data tengah, dan data langsung tampil pada menu. Berikut merupakan *source code* program.

```
#include<iostream>
#include <conio.h>

using namespace std;

struct node{
    int x;
    node* next;
    node* prev;
};

node *head, *tail;

void init(){
    head = NULL;
    tail = NULL;
}
```

Penjelasan:

Pada bagian *header* dideklarasikan ‘#include<iostream>’ dan ‘#include <conio.h>’ agar program dapat mengeksekusi fungsi-fungsi di dalam *library*-nya. “using namespace std;” berfungsi agar program dapat menggunakan semua fungsi yang akan digunakan di dalam program. Selanjutnya dideklarasikan suatu struct dengan nama ‘node’ yang di dalamnya terdapat variabel ‘x’ yang bertipe data *integer* dan ‘*prev’ dan ‘*next’ yang bertipe ‘node’ itu sendiri sebagai *pointer* penunjuk pada *linked list* yang akan dibuat. Lalu dideklarasikan ‘*head’ dan ‘*tail’ yang bertipe ‘node’ sebagai pangkal dan ujung data. Setelah itu dibuat prosedur ‘init’ untuk memberi nilai *NULL* pada ‘head’ dan ‘tail’.

```
void insertFirst(int data){
    node *baru;
    baru = new node;
    if(head == NULL){
        head = baru;
        baru -> prev = NULL;
        baru -> x = data;
        baru -> next = NULL;
        tail = baru;
    } else {
        baru -> next = head;
        baru -> x = data;
        baru -> prev = NULL;
        head -> prev = baru;
        head = baru;
    }
}
```

Penjelasan:

Apabila data masih kosong, maka *pointer* 'next' dan 'prev' data akan menunjuk ke *NULL*. Sehingga data yang baru diinputkan akan menjadi *head* dan *tail* dari *linked list* tersebut. Apabila sebelumnya sudah ada data, maka *pointer* 'next' data terbaru akan menunjuk pada *head* dan *pointer* 'prev' *head* akan menunjuk data terbaru. Sehingga *head* terbaru adalah data yang baru dimasukkan. Prosedur ini berfungsi untuk menambahkan data dari depan.

```
void insertLast(int data){
    node *baru;
    baru = new node;
    baru -> x = data;
    if(head == NULL){
        head = baru;
```

```

    baru -> prev = NULL;
    baru -> next = NULL;
    tail = baru;
} else {
    baru -> prev = tail;
    tail -> next = baru;
    baru -> next = NULL;
    tail = baru;
}
}

```

Penjelasan:

Jika data masih kosong, maka *pointer* ‘next’ dan ‘prev’ data akan menunjuk ke *NULL*. Sehingga data yang baru diinputkan akan menjadi *head* dan *tail* dari *linked list* tersebut. Namun, jika sudah ada data sebelumnya, maka *pointer* ‘prev’ data terbaru akan menunjuk pada *tail* dan *pointer* ‘next’ *tail* akan menunjuk data terbaru. Lalu dideklarasikan nilai *tail* sama dengan data terbaru. Prosedur ini berfungsi untuk menambahkan data dari belakang.

```

void printList(){
    cout << endl;
    node *temp;
    if(head == NULL){
        cout << "Belum ada data.";
    } else {
        temp = head;
        while(temp != NULL){
            cout << temp -> x << " -> ";
            temp = temp -> next;
        }
    }
}

```

Penjelasan:

Jika data belum ada, maka program akan mencetak “Belum ada data.”. Jika sudah ada data sebelumnya, maka selama *pointer* ‘next’ data tidak menunjuk pada *NULL* data akan terus dicetak. Prosedur ini berfungsi untuk menampilkan semua data pada program.

```
void insertAfter(int old, int data){
    node *baru;
    baru = new node;
    node *temp;
    temp = head;
    if(head == NULL){
        cout << "Tidak bisa memasukkan data." << endl;
    }
    if(head == tail){
        if(head -> x != old){
            cout << "Tidak bisa memasukkan data." << endl;
        }
        baru -> x = data;
        head -> next = baru;
        baru -> next = NULL;
        head -> prev = NULL;
        baru -> prev = head;
        tail = baru;
    }
    if(tail -> x == data){
        baru -> next = NULL;
        baru -> prev = tail;
        tail -> next = baru;
        tail = baru;
    }
    while(temp -> x != old){
        temp = temp -> next;
```

```

        if(temp == NULL){
            cout << "Tidak bisa memasukkan data." << endl;
            cout << "Tidak ada data." << endl;
        }
    }
    baru -> next = temp -> next;
    baru -> prev = temp;
    baru -> x = data;
    temp -> next -> prev = baru;
    temp -> next = baru;
}

```

Penjelasan:

Prosedur ini berfungsi untuk membantu *user* untuk dapat memasukkan data setelah atau di depan suatu data yang telah ada berdasarkan inputan *user*. Prosedur tidak dapat dieksekusi jika data masih kosong atau hanya ada satu ada. Ini merupakan pengembangan dari pengelolaan data *linked list* di tengah (bukan *head* dan *tail*). Sehingga dia juga tidak bisa mengeksekusi penambahan data setelah *tail*. Penambahan data setelah *tail* sudah ada prosedurnya sendiri. Prosedur dilakukan dengan mengganti *pointer* 'next' data yang di sebelah kirinya diarahkan pada data terbaru, *pointer* 'prev' data terbaru diarahkan pada data di sebelah kirinya, *pointer* 'next' data terbaru diarahkan ke data di sebelah kanannya, dan *pointer* 'prev' data di sebelah kanannya diarahkan pada data terbaru.

```

void insertBefore(int old, int data){
    node *baru;
    baru = new node;
    node *temp;
    temp = head;
    if(head == NULL){
        cout<< "Tidak bisa memasukkan data." << endl;
    }
}

```

```

if(head == tail){
    if(head -> x != old){
        cout << "Tidak bisa memasukkan data." << endl;
    }
    baru -> x = data;
    head -> next = baru;
    baru -> next = NULL;
    head -> prev = NULL;
    baru -> prev = head;
    tail = baru;
}
if(tail -> x == data){
    baru -> next = tail;
    baru -> prev = NULL;
    tail -> next = baru;
    tail = baru;
}
while(temp -> x != old){
    temp = temp -> next;
    if(temp == NULL){
        cout << "Tidak bisa memasukkan data." << endl;
        cout << "Tidak ada data." << endl;
    }
}
baru -> prev = temp -> prev;
baru -> next = temp;
baru -> x = data;
temp -> prev -> next = baru;
temp -> prev = baru;
}

```

Penjelasan:

Prosedur ini berfungsi untuk membantu *user* untuk dapat memasukkan data sebelum atau di belakang suatu data yang telah ada berdasarkan inputan *user*. Prosedur tidak dapat dieksekusi jika data masih kosong atau hanya ada satu ada. Ini merupakan pengembangan dari pengelolaan data *linked list* di tengah (bukan *head* dan *tail*). Sehingga dia juga tidak bisa mengeksekusi penambahan data sebelum *head*. Penambahan data sebelum *head* sudah ada prosedurnya sendiri. Prosedur dilakukan dengan mengganti *pointer* 'next' data yang di sebelah kirinya diarahkan pada data terbaru, *pointer* 'prev' data terbaru diarahkan pada data di sebelah kirinya, *pointer* 'next' data terbaru diarahkan ke data di sebelah kanannya, dan *pointer* 'prev' data di sebelah kanannya diarahkan kepada data terbaru.

```
void deleteFirst(){
    if(head == NULL){
        return;
    }
    if(head == tail){
        node *hapus;
        hapus = head;
        head = NULL;
        tail = NULL;
        delete hapus;
    } else {
        node *hapus;
        hapus = head;
        head = head -> next;
        head -> prev = NULL;
        delete hapus;
    }
}
```

Penjelasan:

Prosedur ini berfungsi untuk membantu *user* menghapus suatu data dari depan. Jika belum ada data, maka tidak akan ada yang dihapus. Jika sudah ada data, maka *head* akan dihapus dengan *head* terbaru adalah data setelah *head* sebelumnya. Sehingga *pointer* 'prev' dari *head* terbaru akan menunjuk pada *NULL*. Dalam penghapusan data ini, kita memerlukan variabel bantu, yaitu 'hapus'.

```
void deleteLast(){
    if(head==NULL){
        return;
    }
    if(head==tail){
        node *hapus;
        hapus = head;
        head = NULL;
        tail = NULL;
        delete hapus;
    } else {
        node *hapus;
        hapus = tail;
        tail = tail -> prev;
        tail -> next = NULL;
        delete hapus;
    }
}
```

Penjelasan:

Jika data masih kosong, maka tidak akan ada yang dihapus. Jika *tail* sama dengan *head* (hanya terdapat 1 data) nilai *tail* dan *head* akan dirubah menjadi *NULL* dan akan terhapus data awalnya. Jika bukan kedua kondisi sebelumnya, maka *tail* akan dirubah menjadi data sebelum *tail* dan *pointer* 'next' *tail* terbaru akan diarahkan ke *NULL*. Lalu *tail* sebelumnya akan dihapus.

```
void deleteMid(int data){
    node *temp;
    temp = head;
    if(head == tail)
    {
        if(head -> x != data){
            cout << "Tidak dapat menghapus data." << endl;
        }
        head = NULL;
        tail = NULL;
        delete temp;
    }
    if(head -> x == data){
        head = head -> next;
        head -> prev = NULL;
        delete temp;
    } else if(tail -> x == data){
        temp = tail;
        tail = tail -> prev;
        tail -> next = NULL;
        delete temp;
    }
    while(temp -> x != data){
        temp = temp -> next;
        if(temp == NULL){
            cout << "data not found" << endl;

        }
    }
    temp -> next -> prev = temp -> prev;
    temp -> prev -> next = temp -> next;
```

```
delete temp;
}
```

Penjelasan:

Prosedur ini berfungsi untuk menghapus suatu data di dalam *linked list* yang berada di antara *head* dan *tail*. Apabila ingin menghapus *head* atau *tail*, maka sudah ada prosedurnya sendiri. Jika belum ada data, maka tidak akan ada yang dihapus. Jika sudah ada data, maka data yang diinginkan *user* untuk dihapus akan terhapus. Caranya adalah *pointer* 'next' data sebelum data yang ingin dihapus langsung menunjuk data setelah data yang ingin dihapus dan *pointer* 'prev' data setelah data yang ingin dihapus langsung menunjuk data sebelum data yang ingin dihapus.

```
int main(){
    init();
    int baru, data, old, pil;
    do{
        system("cls");
        cout << "=====\n"
             << "  MENU UTAMA LINKED LIST  =\n"
             <<
             "=====\n"
             << "== 1. Input Data dari Depan   ==\n"
             << "== 2. Input Data dari Belakang ==\n"
             << "== 3. Input Data Setelah Data  ==\n"
             << "== 4. Input Data Sebelum Data  ==\n"
             << "== 5. Hapus Data dari Depan   ==\n"
             << "== 6. Hapus Data dari Belakang ==\n"
             << "== 7. Hapus Data dari Tengah  ==\n"
             << "== 8. Keluar                  ==\n"
             <<
             "=====\n";
    }
```

```
        printList();

        cout << "\n\nPilih Menu\t: ";
        cin >> pil;
        switch (pil){
            case 1:
                cout << "\nMasukkan data : ";
                cin >> data;
                insertFirst(data);
                printList();
                getch();
                break;
            case 2:
                cout << "\nMasukkan data : ";
                cin >> data;
                insertLast(data);
                printList();
                getch();
                break;
            case 3:
                printList();
                cout << endl;
                cout << "\nMasukkan setelah  : ";
                cin >> old;
                cout << "Masukkan data baru : ";
                cin >> baru;
                insertAfter(old, baru);
                printList();
                getch();
                break;
            case 4:
                printList();
```

```
        cout << endl;
        cout << "\nMasukkan sebelum  : ";
        cin >> old;
        cout << "Masukkan data baru : ";
        cin >> baru;
        insertBefore(old, baru);
        printList();
        getch();
        break;
    case 5:
        deleteFirst();
        printList();
        getch();
        break;
    case 6:
        deleteLast();
        printList();
        getch();
        break;
    case 7:
        printList();
        cout << endl;
        int data;
        cout << "\nMasukkan data yang ingin dihapus: ";
                cin >> data;
        deleteMid(data);
        printList();
        getch();
        break;
    case 8:
        cout << "\nProgram Selesai";
```

```

        break;

    default:

        cout << "\nMasukan tidak valid.\n" << endl;
        cout << "\nPress Any Button to Continue...";
        getch();

    }

} while (pil != 8);

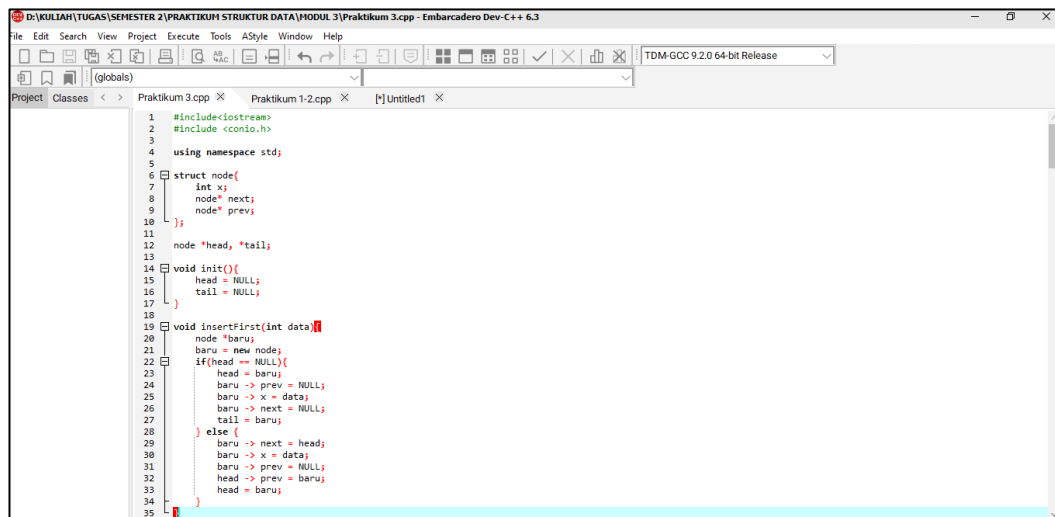
return 0;

}

```

Penjelasan:

Ini merupakan program utama yang ditandai dengan “int main()”. Pada program utama ini diberi perintah untuk mencetak sederet keluaran tampilan menu utama program. *User* akan diminta untuk memasukkan input pilihan menu yang ingin diakses. Percabangan yang digunakan dalam proses ini adalah ‘switch (case)’. Pada program utama pun dideklarasikan perulangan. Selama *user* tidak memilih ‘8’ saat memilih menu, maka program akan terus diulang. Dalam setiap *case* pilihan oleh *user*, program hanya perlu memanggil prosedur-prosedur yang sudah dideklarasikan sebelumnya.



```

1  #include <iostream>
2  #include <conio.h>
3
4  using namespace std;
5
6  struct node{
7      int x;
8      node* next;
9      node* prev;
10 };
11
12 node *head, *tail;
13
14 void init(){
15     head = NULL;
16     tail = NULL;
17 }
18
19 void insertFirst(int data){
20     node *baru;
21     baru = new node;
22     if(head == NULL){
23         head = baru;
24         baru->prev = NULL;
25         baru->x = data;
26         baru->next = NULL;
27         tail = baru;
28     } else {
29         baru->next = head;
30         baru->x = data;
31         baru->prev = NULL;
32         head->prev = baru;
33         head = baru;
34     }
35 }

```

Gambar 3.1 Source Code Program (1)

```
37 void insertLast(int data){
38     node *baru;
39     baru = new node;
40     baru->x = data;
41     if(head == NULL){
42         head = baru;
43         baru->prev = NULL;
44         baru->next = NULL;
45         tail = baru;
46     } else {
47         baru->prev = tail;
48         tail->next = baru;
49         baru->next = NULL;
50         tail = baru;
51     }
52 }
53
54 void printList(){
55     cout << endl;
56     node *temp;
57     if(head == NULL){
58         cout << "Belum ada data.";
59     } else {
60         temp = head;
61         while(temp != NULL){
62             cout << temp->x << " -> ";
63             temp = temp->next;
64         }
65     }
66 }
67
68 void insertAfter(int old, int data){
69     node *baru;
70     baru = new node;
71     node *temp;
```

Gambar 3.2 Source Code Program (2)

```
72     temp = head;
73     if(head == NULL){
74         cout << "Tidak bisa memasukkan data." << endl;
75     }
76     if(head == tail){
77         if(head->x != old){
78             cout << "Tidak bisa memasukkan data." << endl;
79         }
80         baru->x = data;
81         head->next = baru;
82         baru->next = NULL;
83         head->prev = NULL;
84         baru->prev = head;
85         tail = baru;
86     }
87     if(tail->x == data){
88         baru->next = NULL;
89         baru->prev = tail;
90         tail->next = baru;
91         tail = baru;
92     }
93     while(temp->x != old){
94         temp = temp->next;
95         if(temp == NULL){
96             cout << "Tidak bisa memasukkan data." << endl;
97             cout << "Tidak ada data." << endl;
98         }
99     }
100     baru->next = temp->next;
101     baru->prev = temp;
102     baru->x = data;
103     temp->next->prev = baru;
104     temp->next = baru;
105 }
```

Gambar 3.3 Source Code Program (3)

```
107 void insertBefore(int old, int data){
108     node *baru;
109     baru = new node;
110     node *temp;
111     temp = head;
112     if(head == NULL){
113         cout << "Tidak bisa memasukkan data." << endl;
114     }
115     if(head == tail){
116         if(head->x != old){
117             cout << "Tidak bisa memasukkan data." << endl;
118         }
119         baru->x = data;
120         head->next = baru;
121         baru->next = NULL;
122         head->prev = NULL;
123         baru->prev = head;
124         tail = baru;
125     }
126     if(tail->x == data){
127         baru->next = tail;
128         baru->prev = NULL;
129         tail->next = baru;
130         tail = baru;
131     }
132     while(temp->x != old){
133         temp = temp->next;
134         if(temp == NULL){
135             cout << "Tidak bisa memasukkan data." << endl;
136             cout << "Tidak ada data." << endl;
137         }
138     }
139     baru->prev = temp->prev;
140     baru->next = temp;
141     baru->x = data;
```

Gambar 3.4 Source Code Program (4)


```

142     temp->prev->next = baru;
143     temp->prev = baru;
144 }
145
146 void deleteFirst(){
147     if(head == NULL){
148         return;
149     }
150     if(head == tail){
151         node *hapus;
152         hapus = head;
153         head = NULL;
154         tail = NULL;
155         delete hapus;
156     } else {
157         node *hapus;
158         hapus = head;
159         head = head->next;
160         head->prev = NULL;
161         delete hapus;
162     }
163 }
164
165 void deleteLast(){
166     if(head==NULL){
167         return;
168     }
169     if(head==tail){
170         node *hapus;
171         hapus = head;
172         head = NULL;
173         tail = NULL;
174         delete hapus;
175     } else {
176         node *hapus;

```

Gambar 3.5 Source Code Program (5)

```

177     hapus = tail;
178     tail->tail->prev;
179     tail->next = NULL;
180     delete hapus;
181 }
182 }
183
184 void deleteId(int data){
185     node *temp;
186     temp = head;
187     if(head == tail){
188         if(head->x != data){
189             cout << "Tidak dapat menghapus data." << endl;
190         }
191         head = NULL;
192         tail = NULL;
193         delete temp;
194     }
195     if(head->x == data){
196         head = head->next;
197         head->prev = NULL;
198         delete temp;
199     } else if(tail->x == data){
200         temp = tail;
201         tail = tail->prev;
202         tail->next = NULL;
203         delete temp;
204     }
205     while(temp->x != data){
206         temp = temp->next;
207         if(temp == NULL){
208             cout << "data not found" << endl;
209         }
210     }
211 }

```

Gambar 3.6 Source Code Program (6)

```

212     }
213     temp->next->prev = temp->prev;
214     temp->prev->next = temp->next;
215     delete temp;
216 }
217
218 int main(){
219     init();
220     int baru, data, old, pil;
221     do{
222         system("cls");
223         cout << "=====\\n"
224             << "  MENU UTAMA LINKED LIST  \\n"
225             << "=====\\n"
226             << "== 1. Input Data dari Depan  ==\\n"
227             << "== 2. Input Data dari Belakang ==\\n"
228             << "== 3. Input Data Setelah Data ==\\n"
229             << "== 4. Input Data Sebelum Data ==\\n"
230             << "== 5. Hapus Data dari Depan  ==\\n"
231             << "== 6. Hapus Data dari Belakang ==\\n"
232             << "== 7. Hapus Data dari Tengah ==\\n"
233             << "== 8. Keluar                  ==\\n"
234             << "=====\\n";
235         printList();
236         cout << "\\n\\nPilih Menu/t: ";
237         cin >> pil;
238         switch (pil){
239             case 1:
240                 cout << "\\nMasukkan data : ";
241                 cin >> data;
242                 insertFirst(data);
243                 printList();
244                 getch();
245                 break;
246             case 2:

```

Gambar 3.7 Source Code Program (7)

```

246         case 2:
247             cout << "\nMasukkan data : ";
248             cin >> data;
249             insertLast(data);
250             printList();
251             getch();
252             break;
253         case 3:
254             printList();
255             cout << endl;
256             cout << "\nMasukkan setelah : ";
257             cin >> old;
258             cout << "\nMasukkan data baru : ";
259             cin >> baru;
260             insertAfter(old, baru);
261             printList();
262             getch();
263             break;
264         case 4:
265             printList();
266             cout << endl;
267             cout << "\nMasukkan sebelum : ";
268             cin >> old;
269             cout << "\nMasukkan data baru : ";
270             cin >> baru;
271             insertBefore(old, baru);
272             printList();
273             getch();
274             break;
275         case 5:
276             deleteFirst();
277             printList();
278             getch();
279             break;
280         case 6:

```

Gambar 3.8 Source Code Program (8)

```

271             insertBefore(old, baru);
272             printList();
273             getch();
274             break;
275         case 5:
276             deleteFirst();
277             printList();
278             getch();
279             break;
280         case 6:
281             deleteLast();
282             printList();
283             getch();
284             break;
285         case 7:
286             printList();
287             cout << endl;
288             int data;
289             cout << "\nMasukkan data yang ingin dihapus : ";
290             cin >> data;
291             deleteId(data);
292             printList();
293             getch();
294             break;
295         case 8:
296             cout << "\nProgram Selesai";
297             break;
298         default:
299             cout << "\nMasukan tidak valid.\n" << endl;
300             cout << "\nPress Any Button to Continue...";
301             getch();
302     }
303 } while (pil != 8);
304 return 0;
305

```

Gambar 3.9 Source Code Program (9)

```

=====
==  MENU UTAMA LINKED LIST  ==
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

Belum ada data.

Pilih Menu : 1

Masukkan data : 1

1 ->

```

Gambar 4.1 Output Menu 1

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 ->

Pilih Menu      : 2
Masukkan data : 9
1 -> 9 -> _
```

Gambar 4.2 *Output Menu 2*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 -> 9 ->

Pilih Menu      : 3
1 -> 9 ->

Masukkan setelah : 1
Masukkan data baru : 5
1 -> 5 -> 9 -> _
```

Gambar 4.3 *Output Menu 3*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 -> 5 -> 9 ->

Pilih Menu      : 4
1 -> 5 -> 9 ->

Masukkan sebelum : 9
Masukkan data baru : 3
1 -> 5 -> 3 -> 9 -> _
```

Gambar 4.4 *Output Menu 4*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 -> 5 -> 3 -> 9 ->

Pilih Menu      : 5
5 -> 3 -> 9 ->
```

Gambar 4.5 *Output Menu 5*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
+ MENU UTAMA LINKED LIST +
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

5 -> 3 -> 9 ->

Pilih Menu : 6

5 -> 3 ->
```

Gambar 4.6 *Output* Menu 6

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
+ MENU UTAMA LINKED LIST +
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

5 -> 3 ->

Pilih Menu : 1

Masukkan data : 2

2 -> 5 -> 3 ->
```

Gambar 4.7 *Output* Menu 1 (2)

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
+ MENU UTAMA LINKED LIST +
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

2 -> 5 -> 3 ->

Pilih Menu : 7

2 -> 5 -> 3 ->

Masukkan data yang ingin dihapus: 5

2 -> 3 -> _
```

Gambar 4.8 *Output* Menu 7

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
+ MENU UTAMA LINKED LIST +
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

2 -> 3 ->

Pilih Menu : 8

Program Selesai
-----
Process exited after 311 seconds with return value 0
Press any key to continue . . .
```

Gambar 4.9 *Output* Menu 8

BAB III

KESIMPULAN

I. KESIMPULAN

Linked list adalah suatu cara untuk menyimpan data dengan suatu struktur sehingga *programmer* dapat secara otomatis menciptakan suatu tempat baru untuk menyimpan data kapan saja diperlukan. *Linked list* dikenal juga dengan sebutan senarai berantai adalah struktur data yang terdiri dari urutan *record* data dimana setiap *record* memiliki *field* yang menyimpan alamat/referensi dari *record* selanjutnya (di dalam urutan). Elemen data yang dihubungkan dengan *link* pada *linked list* disebut Node. Biasanya dalam suatu *linked list*, terdapat istilah *head* dan *tail*.

Ada 5 proses dasar di dalam *linked list*, yaitu:

1. Proses inisialisasi
2. Proses simpul baru
3. Membuat simpul awal
4. Menambah simpul baru ke dalam *linked list*
5. Menghapus sebuah simpul dalam *linked list*

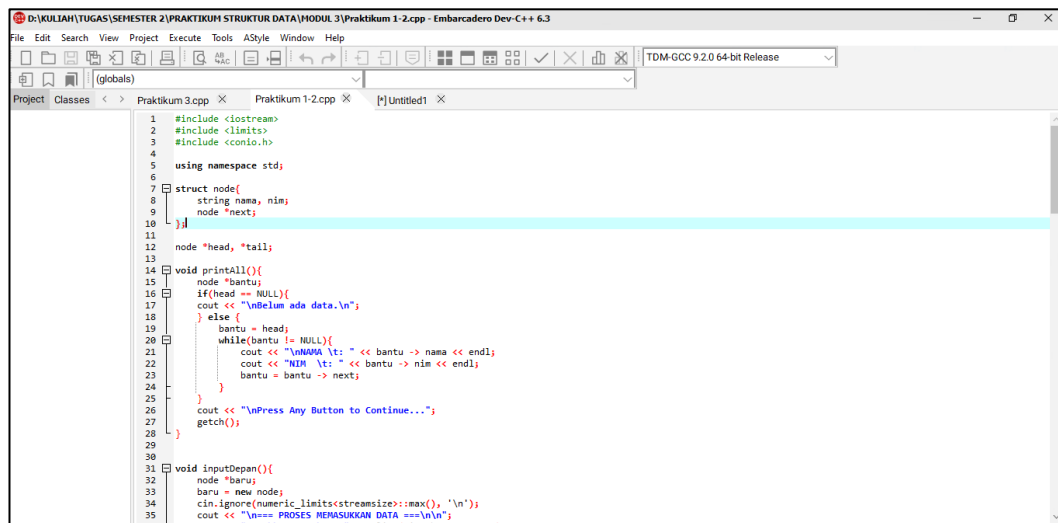
Ada beberapa jenis *linked list*, yaitu sebagai berikut.

1. *Single linked list*
2. *Double linked list*
3. *Circular linked list*
4. *Multiple linked list*

DAFTAR PUSTAKA

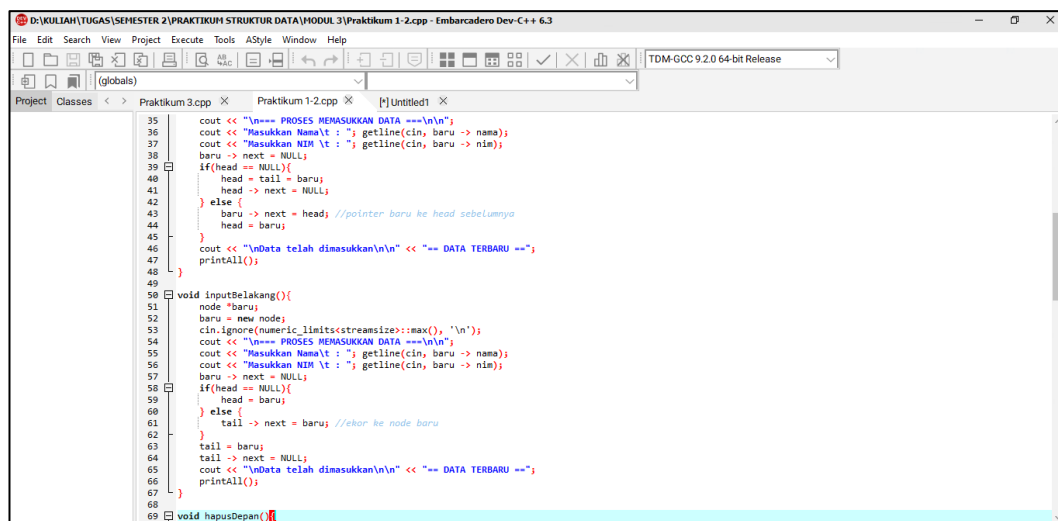
Dosen Teknik Informatika. 2020. *Modul Praktikum Struktur Data*. Palangka Raya.
Jurusan Teknik Informatika Fakultas Teknik Universitas Palangka Raya.

LAMPIRAN



```
1 #include <iostream>
2 #include <limits>
3 #include <conio.h>
4
5 using namespace std;
6
7 struct node{
8     string nama, nim;
9     node *next;
10 }
11
12 node *head, *tail;
13
14 void printAll(){
15     node *bantu;
16     if(head == NULL){
17         cout << "\nBelum ada data.\n";
18     } else {
19         bantu = head;
20         while(bantu != NULL){
21             cout << "\nNama \t: " << bantu->nama << endl;
22             cout << "\nim \t: " << bantu->nim << endl;
23             bantu = bantu->next;
24         }
25     }
26     cout << "\nPress Any Button to Continue...";
27     getch();
28 }
29
30 void inputDepan(){
31     node *baru;
32     baru = new node;
33     cin.ignore(numeric_limits<streamsize>::max(), '\n');
34     cout << "\n== PROSES MEMASUKKAN DATA ==\n\n";
35 }
```

Gambar 1.1 Source Code Program (1)



```
35 cout << "\n== PROSES MEMASUKKAN DATA ==\n\n";
36 cout << "Masukkan Nama\t : "; getline(cin, baru->nama);
37 cout << "Masukkan NIM \t : "; getline(cin, baru->nim);
38 baru->next = NULL;
39 if(head == NULL){
40     head = tail = baru;
41     head->next = NULL;
42 } else {
43     baru->next = head; //pointer baru ke head sebelumnya
44     head = baru;
45 }
46 cout << "\nData telah dimasukkan\n\n" << "== DATA TERBARU ==";
47 printAll();
48
49 void inputBelakang(){
50     node *baru;
51     baru = new node;
52     cin.ignore(numeric_limits<streamsize>::max(), '\n');
53     cout << "\n== PROSES MEMASUKKAN DATA ==\n\n";
54     cout << "Masukkan Nama\t : "; getline(cin, baru->nama);
55     cout << "Masukkan NIM \t : "; getline(cin, baru->nim);
56     baru->next = NULL;
57     if(head == NULL){
58         head = baru;
59     } else {
60         tail->next = baru; //ekor ke node baru
61     }
62     tail = baru;
63     tail->next = NULL;
64     cout << "\nData telah dimasukkan\n\n" << "== DATA TERBARU ==";
65     printAll();
66 }
67
68 void hapusDepan(){
69 }
```

Gambar 1.2 Source Code Program (2)

```

69 void hapusDepan(){
70     node *hapus;
71     if (head == NULL){
72         cout << "Data masih Kosong\n";
73     }
74     else {
75         cout<< "\nData telah dihapus\n\n" << "==" DATA TERBARU ==\n";
76         hapus = head;
77         head = head -> next;
78         delete hapus;
79     }
80     printAll();
81 }
82
83 void hapusBelakang(){
84     node *bantu, *hapus;
85     if(head == NULL){
86         cout << "Data masih Kosong\n";
87     }
88     else if(head == tail){
89         cout << "\nData telah dihapus\n";
90         hapus = head;
91         head = head -> next;
92         delete hapus;
93     }
94     else {
95         cout<< "\nData telah dihapus\n\n" << "==" DATA TERBARU ==\n";
96         bantu = head;
97         hapus = tail;
98         while(bantu -> next != tail){
99             bantu = bantu -> next;
100         }
101         tail = bantu;
102         tail -> next = NULL;
103         delete hapus;
104     }
105     printAll();
106 }
107
108 }
109
110 int main(){
111     int pil;
112     do{
113         system("cls");
114         cout << "=====n"
115             << "  MENU UTAMA LINKED LIST\n"
116             << "=====n"
117             << "== 1. Input Data dari Depan ==\n"
118             << "== 2. Input Data dari Belakang ==\n"
119             << "== 3. Hapus Data dari Depan ==\n"
120             << "== 4. Hapus Data dari Belakang ==\n"
121             << "== 5. Tampilkan Semua Data ==\n"
122             << "== 6. Keluar ==\n\n"
123             << "Pilih Menu : ";
124         cin >> pil;
125         switch (pil){
126             case 1:{
127                 inputDepan();
128                 break;
129             }
130             case 2:{
131                 inputBelakang();
132                 break;
133             }
134             case 3:{
135                 hapusDepan();
136                 break;
137             }
138             case 4:{
139                 hapusBelakang();
140                 break;
141             }
142             case 5:{
143                 printAll();
144                 break;
145             }
146             case 6:{
147                 cout << "\nProgram Selesai";
148                 break;
149             }
150             default:{
151                 cout << "\nMasukan tidak valid.\n" << endl;
152                 cout << "\nPress Any Button to Continue...";
153                 getch();
154             }
155         }
156     } while (pil != 6);
157     return 0;
158 }

```

Gambar 1.3 Source Code Program (3)

```

103 }
104
105 int main(){
106     int pil;
107     do{
108         system("cls");
109         cout << "=====n"
110             << "  MENU UTAMA LINKED LIST\n"
111             << "=====n"
112             << "== 1. Input Data dari Depan ==\n"
113             << "== 2. Input Data dari Belakang ==\n"
114             << "== 3. Hapus Data dari Depan ==\n"
115             << "== 4. Hapus Data dari Belakang ==\n"
116             << "== 5. Tampilkan Semua Data ==\n"
117             << "== 6. Keluar ==\n\n"
118             << "Pilih Menu : ";
119         cin >> pil;
120         switch (pil){
121             case 1:{
122                 inputDepan();
123                 break;
124             }
125             case 2:{
126                 inputBelakang();
127                 break;
128             }
129             case 3:{
130                 hapusDepan();
131                 break;
132             }
133             case 4:{
134                 hapusBelakang();
135                 break;
136             }
137             case 5:{
138                 printAll();
139                 break;
140             }
141             case 6:{
142                 cout << "\nProgram Selesai";
143                 break;
144             }
145             default:{
146                 cout << "\nMasukan tidak valid.\n" << endl;
147                 cout << "\nPress Any Button to Continue...";
148                 getch();
149             }
150         }
151     } while (pil != 6);
152     return 0;
153 }

```

Gambar 1.4 Source Code Program (4)

```

119     cin >> pil;
120     switch (pil){
121         case 1:{
122             inputDepan();
123             break;
124         }
125         case 2:{
126             inputBelakang();
127             break;
128         }
129         case 3:{
130             hapusDepan();
131             break;
132         }
133         case 4:{
134             hapusBelakang();
135             break;
136         }
137         case 5:{
138             printAll();
139             break;
140         }
141         case 6:{
142             cout << "\nProgram Selesai";
143             break;
144         }
145         default:{
146             cout << "\nMasukan tidak valid.\n" << endl;
147             cout << "\nPress Any Button to Continue...";
148             getch();
149         }
150     }
151     while (pil != 6);
152     return 0;
153 }

```

Gambar 1.5 Source Code Program (5)


```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 1

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Indra
Masukkan NIM : 123

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Indra
NIM : 123

Press Any Button to Continue...

```

Gambar 2.1 *Output Menu 1 (1)*

```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 1

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Fiqi
Masukkan NIM : 456

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Fiqi
NIM : 456
NAMA : Indra
NIM : 123

Press Any Button to Continue...

```

Gambar 2.2 *Output Menu 1 (2)*

```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 2

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Ripani
Masukkan NIM : 789

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Fiqi
NIM : 456
NAMA : Indra
NIM : 123
NAMA : Ripani
NIM : 789

Press Any Button to Continue...

```

Gambar 2.3 *Output Menu 2 (1)*

```

D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praктиkum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
-- 1. Input Data dari Depan --
-- 2. Input Data dari Belakang --
-- 3. Hapus Data dari Depan --
-- 4. Hapus Data dari Belakang --
-- 5. Tampilkan Semua Data --
-- 6. Keluar --

Pilih Menu : 2

=== PROSES MEMASUKKAN DATA ===

Masukkan Nama : Mahasiswa
Masukkan NIM : 000

Data telah dimasukkan

== DATA TERBARU ==
NAMA : Fiqi
NIM : 456
NAMA : Indra
NIM : 123
NAMA : Ripani
NIM : 789
NAMA : Mahasiswa
NIM : 000

Press Any Button to Continue...

```

Gambar 2.4 *Output Menu 2 (2)*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 3

Data telah dihapus

== DATA TERBARU ==

NAMA : Indna
NIM : 123

NAMA : Ripani
NIM : 789

NAMA : Mahasiswa
NIM : 000

Press Any Button to Continue...
```

Gambar 2.5 *Output* Menu 3

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 4

Data telah dihapus

== DATA TERBARU ==

NAMA : Indna
NIM : 123

NAMA : Ripani
NIM : 789

Press Any Button to Continue...
```

Gambar 2.6 *Output* Menu 4

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 5

NAMA : Indna
NIM : 123

NAMA : Ripani
NIM : 789

Press Any Button to Continue...
```

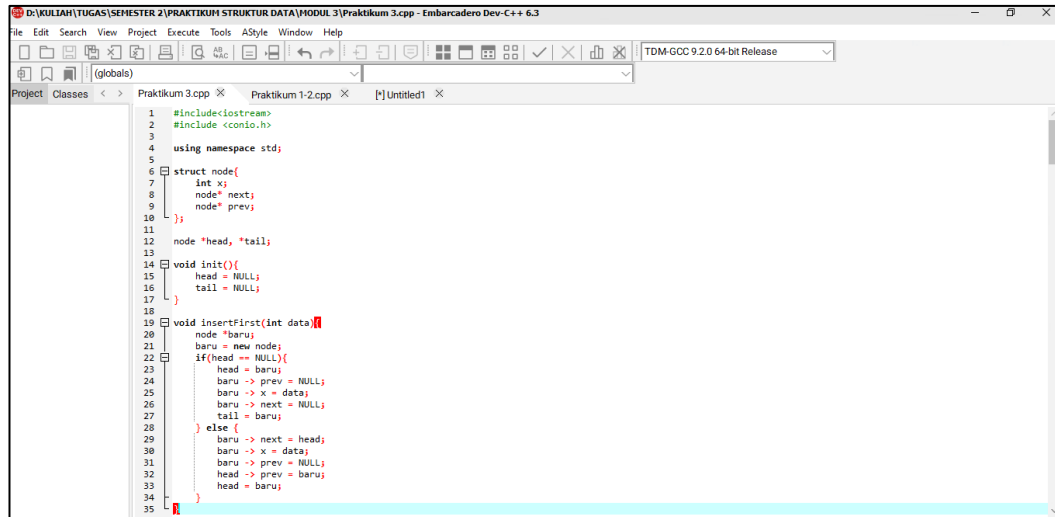
Gambar 2.7 *Output* Menu 5

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praikum 1-2.exe
=====
MENU UTAMA LINKED LIST
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Hapus Data dari Depan ==
== 4. Hapus Data dari Belakang ==
== 5. Tampilkan Semua Data ==
== 6. Keluar ==

Pilih Menu : 6

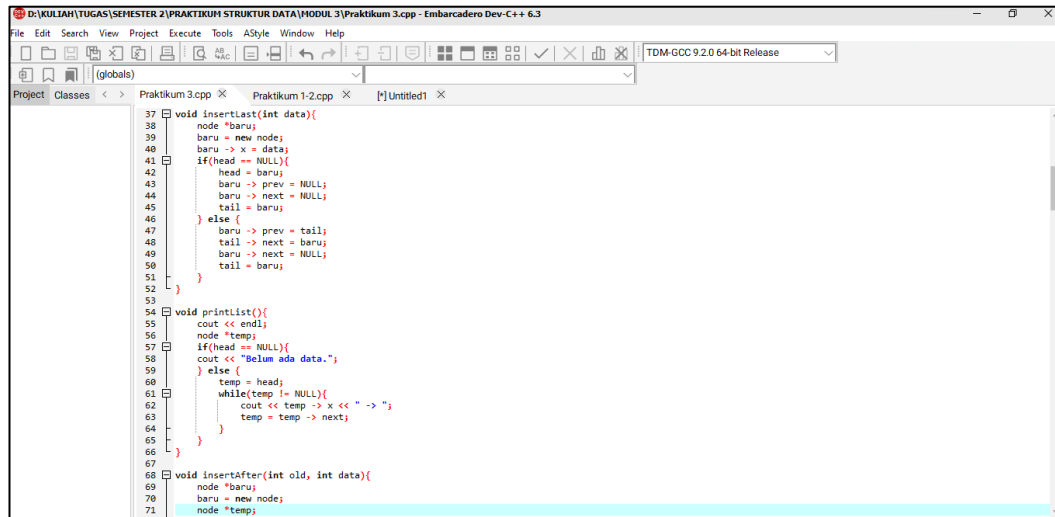
Program Selesai
=====
Process exited after 243.1 seconds with return value 0
Press any key to continue . . .
```

Gambar 2.8 *Output* Menu 6



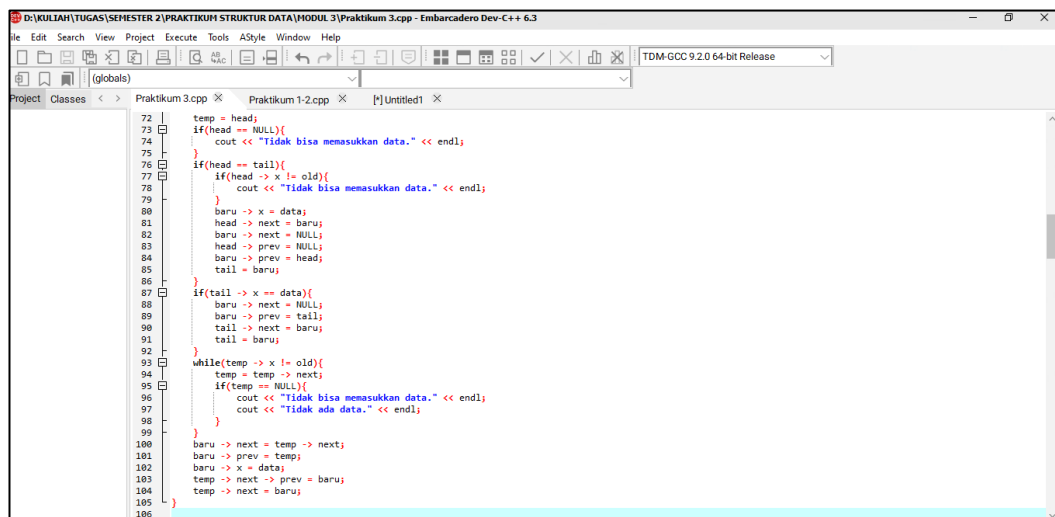
```
1 #include <iostream>
2 #include <conio.h>
3
4 using namespace std;
5
6 struct node{
7     int x;
8     node* next;
9     node* prev;
10 };
11
12 node *head, *tail;
13
14 void init(){
15     head = NULL;
16     tail = NULL;
17 }
18
19 void insertFirst(int data){
20     node *baru;
21     baru = new node;
22     if(head == NULL){
23         head = baru;
24         baru -> prev = NULL;
25         baru -> x = data;
26         baru -> next = NULL;
27         tail = baru;
28     } else {
29         baru -> next = head;
30         baru -> x = data;
31         baru -> prev = NULL;
32         head -> prev = baru;
33         head = baru;
34     }
35 }
```

Gambar 3.1 Source Code Program (1)



```
37 void insertLast(int data){
38     node *baru;
39     baru = new node;
40     baru -> x = data;
41     if(head == NULL){
42         head = baru;
43         baru -> prev = NULL;
44         baru -> next = NULL;
45         tail = baru;
46     } else {
47         baru -> prev = tail;
48         tail -> next = baru;
49         baru -> next = NULL;
50         tail = baru;
51     }
52 }
53
54 void printList(){
55     cout << endl;
56     node *temp;
57     if(head == NULL){
58         cout << "Belum ada data.";
59     } else {
60         temp = head;
61         while(temp != NULL){
62             cout << temp -> x << " -> ";
63             temp = temp -> next;
64         }
65     }
66 }
67
68 void insertAfter(int old, int data){
69     node *baru;
70     baru = new node;
71     node *temp;
```

Gambar 3.2 Source Code Program (2)



```
72     temp = head;
73     if(head == NULL){
74         cout << "Tidak bisa memasukkan data." << endl;
75     }
76     if(head == tail){
77         if(head -> x != old){
78             cout << "Tidak bisa memasukkan data." << endl;
79         }
80         baru -> x = data;
81         head -> next = baru;
82         baru -> next = NULL;
83         head -> prev = NULL;
84         baru -> prev = head;
85         tail = baru;
86     }
87     if(tail -> x == data){
88         baru -> next = NULL;
89         baru -> prev = tail;
90         tail -> next = baru;
91         tail = baru;
92     }
93     while(temp -> x != old){
94         temp = temp -> next;
95     }
96     if(temp == NULL){
97         cout << "Tidak bisa memasukkan data." << endl;
98         cout << "Tidak ada data." << endl;
99     }
100     baru -> next = temp -> next;
101     baru -> prev = temp;
102     baru -> x = data;
103     temp -> next -> prev = baru;
104     temp -> next = baru;
105 }
106 }
```

Gambar 3.3 Source Code Program (3)

```

107 void insertBefore(int old, int data){
108     node *baru;
109     baru = new node;
110     node *temp;
111     temp = head;
112     if(head == NULL){
113         cout<< "Tidak bisa memasukkan data." << endl;
114     }
115     if(head == tail){
116         if(head -> x != old){
117             cout<< "Tidak bisa memasukkan data." << endl;
118         }
119         baru -> x = data;
120         head -> next = baru;
121         baru -> next = NULL;
122         head -> prev = NULL;
123         baru -> prev = head;
124         tail = baru;
125     }
126     if(tail -> x == data){
127         baru -> next = tail;
128         baru -> prev = NULL;
129         tail -> next = baru;
130         tail = baru;
131     }
132     while(temp -> x != old){
133         temp = temp -> next;
134         if(temp == NULL){
135             cout<< "Tidak bisa memasukkan data." << endl;
136             cout<< "Tidak ada data." << endl;
137         }
138     }
139     baru -> prev = temp -> prev;
140     baru -> next = temp;
141     baru -> x = data;

```

Gambar 3.4 Source Code Program (4)

```

142     temp -> prev -> next = baru;
143     temp -> prev = baru;
144 }
145
146 void deleteFirst(){
147     if(head == NULL){
148         return;
149     }
150     if(head == tail){
151         node *hapus;
152         hapus = head;
153         head = NULL;
154         tail = NULL;
155         delete hapus;
156     } else {
157         node *hapus;
158         hapus = head;
159         head = head -> next;
160         head -> prev = NULL;
161         delete hapus;
162     }
163 }
164
165 void deleteLast(){
166     if(head == NULL){
167         return;
168     }
169     if(head == tail){
170         node *hapus;
171         hapus = head;
172         head = NULL;
173         tail = NULL;
174         delete hapus;
175     } else {
176         node *hapus;

```

Gambar 3.5 Source Code Program (5)

```

177     hapus = tail;
178     tail = tail -> prev;
179     tail -> next = NULL;
180     delete hapus;
181 }
182
183 void deleteMid(int data){
184     node *temp;
185     temp = head;
186     if(head == tail){
187         if(head -> x != data){
188             cout<< "Tidak dapat menghapus data." << endl;
189         }
190         head = NULL;
191         tail = NULL;
192         delete temp;
193     }
194     if(head -> x == data){
195         head = head -> next;
196         head -> prev = NULL;
197         delete temp;
198     } else if(tail -> x == data){
199         temp = tail;
200         tail = tail -> prev;
201         tail -> next = NULL;
202         delete temp;
203     }
204     while(temp -> x != data){
205         temp = temp -> next;
206         if(temp == NULL){
207             cout<< "data not found" << endl;
208         }
209     }
210 }
211

```

Gambar 3.6 Source Code Program (6)

```

212     }
213     temp -> next -> prev = temp -> prev;
214     temp -> prev -> next = temp -> next;
215     delete temp;
216 }
217
218 int main(){
219     init();
220     int baru, data, old, pil;
221     do{
222         system("cls");
223         cout << "=====\\n"
224             << "  MENU UTAMA LINKED LIST  \\n"
225             << "=====\\n"
226             << "  1. Input Data dari Depan  ==\\n"
227             << "  2. Input Data dari Belakang ==\\n"
228             << "  3. Input Data Setelah Data ==\\n"
229             << "  4. Input Data Sebelum Data ==\\n"
230             << "  5. Hapus Data dari Depan  ==\\n"
231             << "  6. Hapus Data dari Belakang ==\\n"
232             << "  7. Hapus Data dari Tengah ==\\n"
233             << "  8. Keluar                  ==\\n"
234             << "=====\\n";
235         printList();
236         cout << "\\n\\nPilih Menu/t: ";
237         cin >> pil;
238         switch (pil){
239             case 1:
240                 cout << "\\nMasukkan data : ";
241                 cin >> data;
242                 insertFirst(data);
243                 printList();
244                 getch();
245                 break;
246             case 2:

```

Gambar 3.7 Source Code Program (7)

```

246             case 2:
247                 cout << "\\nMasukkan data : ";
248                 cin >> data;
249                 insertLast(data);
250                 printList();
251                 getch();
252                 break;
253             case 3:
254                 printList();
255                 cout << endl;
256                 cout << "\\nMasukkan setelah : ";
257                 cin >> old;
258                 cout << "\\nMasukkan data baru : ";
259                 cin >> baru;
260                 insertAfter(old, baru);
261                 printList();
262                 getch();
263                 break;
264             case 4:
265                 printList();
266                 cout << endl;
267                 cout << "\\nMasukkan sebelum : ";
268                 cin >> old;
269                 cout << "\\nMasukkan data baru : ";
270                 cin >> baru;
271                 insertBefore(old, baru);
272                 printList();
273                 getch();
274                 break;
275             case 5:
276                 deleteFirst();
277                 printList();
278                 getch();
279                 break;
280             case 6:

```

Gambar 3.8 Source Code Program (8)

```

271             insertBefore(old, baru);
272             printList();
273             getch();
274             break;
275             case 5:
276                 deleteFirst();
277                 printList();
278                 getch();
279                 break;
280             case 6:
281                 deleteLast();
282                 printList();
283                 getch();
284                 break;
285             case 7:
286                 printList();
287                 cout << endl;
288                 int data;
289                 cout << "\\nMasukkan data yang ingin dihapus : ";
290                 cin >> data;
291                 deleteMid(data);
292                 printList();
293                 getch();
294                 break;
295             case 8:
296                 cout << "\\nProgram Selesai";
297                 break;
298             default:
299                 cout << "\\nMasukan tidak valid.\\n" << endl;
300                 cout << "\\nPress Any Button to Continue...";
301                 getch();
302         }
303     } while (pil != 8);
304     return 0;
305 }

```

Gambar 3.9 *Source Code* Program (9)

```
Select D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
=  MENU UTAMA LINKED LIST  =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

Belum ada data.

Pilih Menu      : 1

Masukkan data : 1

1 -> _
```

Gambar 4.1 *Output* Menu 1

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
=  MENU UTAMA LINKED LIST  =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 ->

Pilih Menu      : 2

Masukkan data : 9

1 -> 9 -> _
```

Gambar 4.2 *Output* Menu 2

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
=  MENU UTAMA LINKED LIST  =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 -> 9 ->

Pilih Menu      : 3

1 -> 9 ->

Masukkan setelah : 1
Masukkan data baru : 5

1 -> 5 -> 9 -> _
```

Gambar 4.3 *Output* Menu 3

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
=  MENU UTAMA LINKED LIST  =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 -> 5 -> 9 ->

Pilih Menu      : 4

1 -> 5 -> 9 ->

Masukkan sebelum : 9
Masukkan data baru : 3

1 -> 5 -> 3 -> 9 -> _
```

Gambar 4.4 *Output* Menu 4

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
= MENU UTAMA LINKED LIST =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

1 -> 5 -> 3 -> 9 ->

Pilih Menu : 5

5 -> 3 -> 9 ->
```

Gambar 4.5 *Output Menu 5*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
= MENU UTAMA LINKED LIST =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

5 -> 3 -> 9 ->

Pilih Menu : 6

5 -> 3 ->
```

Gambar 4.6 *Output Menu 6*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
= MENU UTAMA LINKED LIST =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

5 -> 3 ->

Pilih Menu : 1

Masukkan data : 2

2 -> 5 -> 3 ->
```

Gambar 4.7 *Output Menu 1 (2)*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
= MENU UTAMA LINKED LIST =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

2 -> 5 -> 3 ->

Pilih Menu : 7

2 -> 5 -> 3 ->

Masukkan data yang ingin dihapus: 5

2 -> 3 -> _
```

Gambar 4.8 *Output Menu 7*

```
D:\KULIAH\TUGAS\SEMESTER 2\PRAKTIKUM STRUKTUR DATA\MODUL 3\Praktikum 3.exe
=====
= MENU UTAMA LINKED LIST =
=====
== 1. Input Data dari Depan ==
== 2. Input Data dari Belakang ==
== 3. Input Data Setelah Data ==
== 4. Input Data Sebelum Data ==
== 5. Hapus Data dari Depan ==
== 6. Hapus Data dari Belakang ==
== 7. Hapus Data dari Tengah ==
== 8. Keluar ==
=====

2 -> 3 ->

Pilih Menu : 8

Program Selesai
Process exited after 311 seconds with return value 0
Press any key to continue . . .
```

Gambar 4.9 *Output* Menu 8