

**LAPORAN PRAKTIKUM
ALGORITMA & STRUKTUR DATA
MODUL 4**



Double Linked List

Oleh:

Rizki Adhitiya Maulana

NIM. 2410817110014

**PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2025**

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM ALGORITMA & STRUKTUR DATA
MODUL 4

Laporan Praktikum Algoritma & Struktur Data Modul 4 : Double Linked List ini disusun sebagai syarat lulus mata kuliah Praktikum Algoritma & Struktur Data. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Rizki Adhitiya Maulana
NIM : 2410817110014

Menyetujui,
Asisten Praktikum

Mengetahui,
Dosen Penanggung Jawab Praktikum

Muhammad Fauzan Ahsani
NIM. 2310817310009

Muti'a Maulida, S.Kom., M.TI.
NIP. 198810272019032013

DAFTAR ISI

LEMBAR PENGESAHAN	i
DAFTAR ISI.....	ii
DAFTAR TABEL.....	iii
DAFTAR GAMBAR	iv
SOAL 1	1
A Source Code	6
B Output Program.....	14
C Pembahasan.....	22
SOAL 2	27
A Pembahasan.....	27
SOAL 3	28
A Pembahasan.....	28
SOAL 4	29
A Output Program.....	29
B Pembahasan.....	31
TAUTAN GIT HUB	32

DAFTAR TABEL

Tabel 1 Source Code Linked List	6
---------------------------------------	---

DAFTAR GAMBAR

Gambar 1 Tampilan Menu Double Linked List Non Circular	14
Gambar 2 Masuk Ke Tampilan Menu Head DLLNC	14
Gambar 3 Tambah Data Dari Depan.....	14
Gambar 4 Tampilan Data Setelah Dilakukan Tambah Depan	15
Gambar 5 Tambah Data Dari Belakang	15
Gambar 6 Tampilan Data Setelah Dilakukan Tambah Belakang	15
Gambar 7 Hapus Data Dari Depan.....	16
Gambar 8 Tampilan Data Setelah Dilakukan Hapus Depan	16
Gambar 9 Hapus Data Dari Belakang	16
Gambar 10 Tampilan Data Setelah Dilakukan Hapus Belakang	17
Gambar 11 Reset Data Yang Ada	17
Gambar 12 Tampilan Data Setelah Dilakukan Reset Data	17
Gambar 13 Masuk Ke Tampilan Menu Head Dan Tail DLLNC.....	18
Gambar 14 Tambah Data Dari Depan.....	18
Gambar 15 Tampilan Data Setelah Dilakukan Tambah Depan	18
Gambar 16 Tambah Data Dari Belakang	19
Gambar 17 Tampilan Data Setelah Dilakukan Tambah Belakang	19
Gambar 18 Hapus Data Dari Depan.....	19
Gambar 19 Tampilan Data Setelah Dilakukan Hapus Depan	20
Gambar 20 Hapus Data Dari Belakang	20
Gambar 21 Tampilan Data Setelah Dilakukan Hapus Belakang	20
Gambar 22 Reset Data Yang Ada	21
Gambar 23 Tampilan Data Setelah Dilakukan Reset Data	21
Gambar 24 Tampilan Keluar dari Program.....	21
Gambar 25 Fungsi Next Di Single Linked List	27
Gambar 26 Fungsi Prev Di Double Linked List	28
Gambar 27 Tampilan Tambah Depan Head.....	29
Gambar 28 Tampilan Tambah Belakang Head.....	29
Gambar 29 Tampilan Data Head.....	30

Gambar 30 Tampilan Tambah Depan Head Dan Tail	30
Gambar 31 Tampilan Tambah Belakang Head Dan Tail.....	30
Gambar 32 Tampilan Data Head Dan Tail	31

SOAL 1

Lengkapi coding pada function tambahDepanH() agar bisa berjalan dengan lancar. running, simpan program, dan screenshoot hasil running !

```
1  #include <conio.h>
2  #include <iostream>
3  #include <stdlib.h>
4
5  using namespace std;
6
7  typedef struct TNode {
8      string data;
9      TNode *next;
10     TNode *prev;
11 };
12
13 TNode *head, *tail;
14
15 int pil, menu;
16 char pilihan[1];
17 string dataBaru;
18
19 void inith();
20 void inithT();
21 int isEmptyH();
22 int isEmptyHT();
23
24 void tambahDepanH();
25 void tambahDepanHT();
26 void tambahBelakangH();
27 void tambahBelakangHT();
28 void hapusDepanH();
29 void hapusDepanHT();
30 void hapusBelakangH();
31 void hapusBelakangHT();
32 void tampilkanH();
33 void tampilkanHT();
34 void clearH();
35 void clearHT();
36
37 int main()
38 {
39     menu:
40     cout<<"Double Linked List Non Circular (DLLNC)"<<endl;
41     cout<<"-----"<<endl;
42     cout<<"Silahkan pilih program DLLNC yang ingin dijalankan!"<<endl;
43     cout<<"1. DLLNC dengan Head"<<endl;
44     cout<<"2. DLLNC dengan Head dan Tail"<<endl;
45     cout<<"3. Quit"<<endl;
46     cout<<"Pilihan : ";
47     cin>>menu;
48     system("cls");
```

```
49     if(menu==1){
50         do {
51             cout<<"Double Linked List Non Circular (DLLNC) (Head)"<<endl;
52             cout<<"-----"<<endl;
53             cout<<"1. Tambah Depan"<<endl;
54             cout<<"2. Tambah Belakang"<<endl;
55             cout<<"3. Tampilkan Data"<<endl;
56             cout<<"4. Hapus Depan"<<endl;
57             cout<<"5. Hapus Belakang"<<endl;
58             cout<<"6. Reset"<<endl;
59             cout<<"7. Kembali ke Menu"<<endl;
60             cout<<"Pilihan : ";
61             cin>>pilihan;
62             pil=atoi(pilihan);
```

```

63
64     switch(pil) {
65     case 1:
66         tambahDepanH();
67         break;
68     case 2:
69         tambahBelakangH();
70         break;
71     case 3:
72         tampilkanH();
73         break;
74     case 4:
75         hapusDepanH();
76         break;
77     case 5:
78         hapusBelakangH();
79         break;
80     case 6:
81         clearH();
82         break;
83     default:
84         system("cls");
85         goto menu;
86     }
87
88     cout<<"\npress any key to continue"<<endl;
89     getch();
90     system("cls");
91
92     } while (pil<7);
93 } else if(menu==2){
94     do {

```

```

95         cout<<"Double Linked List Non Circular (DLLNC) (Head dan Tail)"<<endl;
96         cout<<"===== "<<endl;
97         cout<<"1. Tambah Depan"<<endl;
98         cout<<"2. Tambah Belakang"<<endl;
99         cout<<"3. Tampilkan Data"<<endl;
100        cout<<"4. Hapus Depan"<<endl;
101        cout<<"5. Hapus Belakang"<<endl;
102        cout<<"6. Reset"<<endl;
103        cout<<"7. Kembali ke Menu"<<endl;
104        cout<<"Pilihan : ";
105        cin>>pilihan;
106        pil=atoi(pilihan);
107
108        switch(pil) {
109        case 1:
110            tambahDepanHT();
111            break;
112        case 2:
113            tambahBelakangHT();
114            break;
115        case 3:
116            tampilkanHT();
117            break;
118        case 4:
119            hapusDepanHT();
120            break;
121        case 5:
122            hapusBelakangHT();
123            break;
124        case 6:
125            clearHT();
126            break;
127        default:
128            system("cls");
129            goto menu;
130        }
131
132        cout<<"\npress any key to continue"<<endl;
133        getch();
134        system("cls");
135
136    } while (pil<7);
137 } else {
138     cout<<"\nTERIMA KASIH"<<endl;
139     cout<<"Program was made by Nama (NIM)."<<endl;
140 }
141 }
142

```



```

143 void inith(){
144     head = NULL;
145 }
146
147 void inithT(){
148     head = NULL;
149     tail = NULL;
150 }
151
152 int isEmptyH(){
153     if(head == NULL) return 1;
154     else return 0;
155 }
156
157 int isEmptyHT(){
158     if(tail == NULL) return 1;
159     else return 0;
160 }
161
162 void tambahDepanH() {
163     cout<<"Masukkan data : ";
164
165
166
167
168
169
170
171
172
173
174
175
176
177     cout << "Data \""<<dataBaru<<"\" berhasil dimasukkan di bagian depan.";
178 }
179
180 void tambahDepanHT() {
181     cout<<"Masukkan data : ";
182     cin>>dataBaru;
183     TNode *baru;
184     baru = new TNode;
185     baru->data = dataBaru;
186     baru->next = NULL;
187     baru->prev = NULL;
188     if(isEmptyHT() == 1) {
189         head = baru;
190         tail = baru;
191
192     } else {
193         baru->next = head;
194         head->prev = baru;
195         head = baru;
196     }
197     cout << "Data \""<<dataBaru<<"\" berhasil dimasukkan di bagian depan.";
198 }
199
200 void tambahBelakangH() {
201     cout<<"Masukkan data : ";
202     cin>>dataBaru;
203     TNode *baru, *bantu;
204     baru = new TNode;
205     baru->data = dataBaru;
206     baru->next = NULL;
207     baru->prev = NULL;
208     if(isEmptyH() == 1) {
209         head = baru;
210     } else {
211         bantu = head;
212         while(bantu->next != NULL){
213             bantu = bantu->next;
214         }
215         bantu->next = baru;
216         baru->prev = bantu;
217     }
218     cout << "Data \""<<dataBaru<<"\" berhasil dimasukkan di bagian belakang.";
219 }

```

```

220 void tambahBelakangHT() {
221     cout<<"Masukkan data : ";
222     cin>>dataBaru;
223     TNode *baru;
224     baru = new TNode;
225     baru->data = dataBaru;
226     baru->next = NULL;
227     baru->prev = NULL;
228     if(isEmptyHT() == 1) {
229         head = baru;
230         tail = baru;
231     } else {
232         tail->next = baru;
233         baru->prev = tail;
234         tail = baru;
235     }
236     cout << "Data \""<<dataBaru<<"\" berhasil dimasukkan di bagian belakang.";
237 }

```

```

238
239 void tampilkanH() {
240     TNode *bantu;
241     bantu = head;
242     if(isEmptyH() == 0) {
243         while(bantu != NULL) {
244             cout<<bantu->data<<' ';
245             bantu = bantu->next;
246         }
247         cout<<endl;
248     } else cout<<"Tidak terdapat data pada Linked List";
249 }
250
251 void tampilkanHT() {
252     TNode *bantu;
253     bantu = head;
254     if(isEmptyHT() == 0) {
255         while(bantu != tail->next) {
256             cout<<bantu->data<<' ';
257             bantu = bantu->next;
258         }
259         cout<<endl;
260     } else cout<<"Tidak terdapat data pada Linked List";
261 }
262
263 void hapusDepanH() {
264     TNode *hapus;
265     string data;
266     if(isEmptyH() == 0) {
267         hapus = head;
268         data = hapus->data;
269         if(head->next != NULL) {
270             head = head->next;
271             head->prev = NULL;
272         } else {
273             inith();
274         }
275         delete hapus;
276         cout<<"Data \""<<data<<"\" yang berada di depan telah berhasil dihapus.";
277     } else cout<<"Tidak terdapat data pada Linked List";
278 }
279
280 void hapusDepanHT() {
281     TNode *hapus;
282     string data;
283     if(isEmptyHT() == 0) {
284         hapus = head;
285         data = hapus->data;

```

```

286         if(head->next != NULL) {
287             head = head->next;
288             head->prev = NULL;
289         } else {
290             inithT();
291         }
292         delete hapus;
293         cout<<"Data \\"<<data<<" yang berada di depan telah berhasil dihapus.";
294     } else cout<<"Tidak terdapat data pada Linked List";
295 }
296
297 void hapusBelakangH() {
298     TNode *hapus;
299     string data;
300     if(isEmptyH() == 0) {
301         hapus = head;
302         while(hapus->next != NULL){
303             hapus = hapus->next;
304         }
305         data = hapus->data;
306         if(head->next != NULL) {
307             hapus->prev->next = NULL;
308         } else {
309             inith();
310         }
311         delete hapus;
312         cout<<"Data \\"<<data<<" yang berada di belakang telah berhasil dihapus.";
313     } else cout<<"Tidak terdapat data pada Linked List";
314 }
315
316 void hapusBelakangHT() {
317     TNode *hapus;
318     string data;
319     if(isEmptyHT() == 0) {
320         hapus = tail;
321         data = hapus->data;
322         if(head->next != NULL) {
323             tail = tail->prev;
324             tail->next = NULL;
325         } else {
326             inithT();
327         }
328         delete hapus;
329         cout<<"Data \\"<<data<<" yang berada di belakang telah berhasil dihapus.";
330     } else cout<<"Tidak terdapat data pada Linked List";
331 }
332
333 void clearH() {
334     TNode *bantu, *hapus;
335     bantu = head;
336     while(bantu != NULL) {
337         hapus = bantu;
338         bantu = bantu->next;
339         delete hapus;
340     }
341     inith();
342     cout<<"Seluruh data pada Linked List telah dibersihkan.";
343 }
344
345 void clearHT() {
346     TNode *bantu, *hapus;
347     bantu = head;
348     while(bantu != NULL) {
349         hapus = bantu;
350         bantu = bantu->next;
351         delete hapus;
352     }
353     inithT();
354     cout<<"Seluruh data pada Linked List telah dibersihkan.";
355 }

```

A Source Code

Tabel 1 Source Code Linked List

1	#include <conio.h>
2	#include <iostream>
3	#include <stdlib.h>
4	
5	using namespace std;
6	
7	typedef struct Tnode {
8	string data;
9	Tnode *next;
10	Tnode *prev;
11	};
12	
13	Tnode *head, *tail;
14	
15	int pil, menu;
16	char pilihan[1];
17	string dataBaru;
18	
19	void initH();
20	void initHT();
21	int isEmptyH();
22	int isEmptyHT();
23	
24	void tambahDepanH();
25	void tambahDepanHT();
26	void tambahBelakangH();
27	void tambahBelakangHT();
28	void hapusDepanH();
29	void hapusDepanHT();
30	void hapusBelakangH();
31	void hapusBelakangHT();
32	void tampilkanH();
33	void tampilkanHT();
34	void clearH();
35	void clearHT();
36	
37	int main()
38	{
39	menu:
40	cout<<"Double Linked List Non Circular (DLLNC)"<<endl;
41	cout<<"===== "<< endl;
42	cout<<"Silahkan pilihan program DLLNC yang ingin dijalankan!"<<endl;
43	cout<<"1. DLLNC dengan head"<<endl;
44	cout<<"2. DLLNC dengan Head dan Tail"<<endl;

```

45     cout<<"3. Quit"<<endl;
46     cout<<"Pilihan"<<endl;
47     cin>>menu;
48     system("cls");
49     if(menu==1){
50         do {
51             cout<<"Double Linked List Non Circular
(DLLNC) (head) "<<endl;
52             cout<<"=====
===== "<<endl;
53             cout<<"1. Tambah Depan"<<endl;
54             cout<<"2. Tambah Belakang"<<endl;
55             cout<<"3. Tampilkan Data"<<endl;
56             cout<<"4. Hapus Depan"<<endl;
57             cout<<"5. Hapus Belakang"<<endl;
58             cout<<"6. Reset"<<endl;
59             cout<<"7. Kembali Ke Menu"<<endl;
60             cout<<"Pilihan : "<<endl;
61             cin>>pilihan;
62             pil=atoi(pilihan);
63
64             switch (pil){
65             case 1:
66                 tambahDepanH();
67                 break;
68             case 2:
69                 tambahBelakangH();
70                 break;
71             case 3:
72                 tampilkanH();
73                 break;
74             case 4:
75                 hapusDepanH();
76                 break;
77             case 5:
78                 hapusBelakangH();
79                 break;
80             case 6:
81                 clearH();
82                 break;
83             default:
84                 system("cls");
85                 goto menu;
86             }
87
88             cout<<"\npress any key to continue"<<endl;
89             getch();
90             system("cls");
91

```

```

92         } while (pil<7);
93     } else if(menu==2){
94         do {
95             cout<<"Double Linked List Non Circular
(DLLNC) (head dan Tail)"<<endl;
96             cout<<"=====
===== "<<endl;
97             cout<<"1. Tambah Depan"<<endl;
98             cout<<"2. Tambah Belakang"<<endl;
99             cout<<"3. Tampilkan Data"<<endl;
100            cout<<"4. Hapus Depan"<<endl;
101            cout<<"5. Hapus Belakang"<<endl;
102            cout<<"6. Reset"<<endl;
103            cout<<"7. Kembali Ke Menu"<<endl;
104            cout<<"Pilihan : "<<endl;
105            cin>>pilihan;
106            pil=atoi(pilihan);
107
108            switch (pil){
109            case 1:
110                tambahDepanHT();
111                break;
112            case 2:
113                tambahBelakangHT();
114                break;
115            case 3:
116                tampilkanHT();
117                break;
118            case 4:
119                hapusDepanHT();
120                break;
121            case 5:
122                hapusBelakangHT();
123                break;
124            case 6:
125                clearHT();
126                break;
127            default:
128                system("cls");
129                goto menu;
130            }
131
132            cout<<"\npress any key to continue"<<endl;
133            getch();
134            system("cls");
135
136        } while (pil<7);
137    } else {
138        cout<<"\nTERIMA KASIH"<<endl;

```

```

139         cout<<"Program was made by Rizki Adhitiya
Maulana (2410817110014)"<<endl;
140     }
141 }
142
143 void initH(){
144     head=NULL;
145 }
146
147 void initHT(){
148     head=NULL;
149     tail=NULL;
150 }
151
152 int isEmptyH(){
153     if(head == NULL) return 1;
154     else return 0;
155 }
156
157 int isEmptyHT(){
158     if(tail == NULL) return 1;
159     else return 0;
160 }
161
162 void tambahDepanH(){
163     cout<<"Masukan Data : ";
164     cin>>dataBaru;
165     Tnode *baru;
166     baru = new Tnode;
167     baru->data = dataBaru;
168     baru->next = NULL;
169     baru->prev = NULL;
170     if(isEmptyH() == 1) {
171         head = baru;
172     } else {
173         baru->next = head;
174         head->prev = baru;
175         head = baru;
176     }
177     cout<<"Data \""<<dataBaru<<"\" berhasil dimasukan
di bagian depan.";
178 }
179
180 void tambahDepanHT() {
181     cout<<"Masukan Data : ";
182     cin>>dataBaru;
183     Tnode *baru;
184     baru = new Tnode;
185     baru->data = dataBaru;

```

```

186     baru->next = NULL;
187     baru->prev = NULL;
188     if(isEmptyHT() == 1) {
189         head = baru;
190         tail = baru;
191     } else {
192         baru->next = head;
193         head->prev = baru;
194         head = baru;
195     }
196     cout<<"Data \""<<dataBaru<<"\" berhasil dimasukan
di bagian depan.";
197 }
198
199 void tambahBelakangH() {
200     cout<<"Masukan Data : ";
201     cin>>dataBaru;
202     Tnode *baru, *bantu;
203     baru = new Tnode;
204     baru->data = dataBaru;
205     baru->next = NULL;
206     baru->prev = NULL;
207     if(isEmptyH() == 1) {
208         head = baru;
209     } else {
210         bantu = head;
211         while(bantu->next != NULL) {
212             bantu = bantu->next;
213         }
214         bantu->next = baru;
215         baru->prev = bantu;
216     }
217     cout <<"Data \""<<dataBaru<<"\" berhasil dimasukan
di bagian belakang.";
218 }
219
220 void tambahBelakangHT() {
221     cout<<"Masukan Data : ";
222     cin>>dataBaru;
223     Tnode *baru;
224     baru = new Tnode;
225     baru->data = dataBaru;
226     baru->next = NULL;
227     baru->prev = NULL;
228     if(isEmptyHT() == 1) {
229         head = baru;
230         tail = baru;
231     } else {
232         tail->next = baru;

```



```

233     baru->prev = tail;
234     tail = baru;
235 }
236     cout<<"Data \""<<dataBaru<<"\" berhasil dimasukan
di bagian belakang.";
237 }
238
239 void tampilkanH() {
240     Tnode *bantu;
241     bantu = head;
242     if(isEmptyH() == 0) {
243         while(bantu != NULL) {
244             cout<<bantu->data<<' ';
245             bantu = bantu->next;
246         }
247         cout<<endl;
248     } else cout<<"Tidak terdapat data pada Linked
List";
249 }
250
251 void tampilkanHT() {
252     Tnode *bantu;
253     bantu = head;
254     if(isEmptyHT() == 0) {
255         while(bantu != tail->next) {
256             cout<<bantu->data<<' ';
257             bantu = bantu->next;
258         }
259         cout<<endl;
260     } else cout<<"Tidak terdapat data pada Linked
List";
261 }
262
263 void hapusDepanH() {
264     Tnode *hapus;
265     string data;
266     if(isEmptyH() == 0) {
267         hapus = head;
268         data = hapus->data;
269         if(head->next != NULL) {
270             head = head->next;
271             head->prev = NULL;
272         } else {
273             initH();
274         }
275         delete hapus;
276         cout<<"Data \""<<data<<"\" berhasil dihapus
dari bagian depan."<<endl;

```

```

277     } else cout<<"Tidak ada data pada Linked
List"<<endl;
278 }
279
280 void hapusDepanHT() {
281     Tnode *hapus;
282     string data;
283     if(isEmptyHT() == 0) {
284         hapus = head;
285         data = hapus->data;
286         if(head->next != NULL) {
287             head = head->next;
288             head->prev = NULL;
289         } else {
290             initHT();
291         }
292         delete hapus;
293         cout<<"Data \""<<data<<"\" berhasil dihapus
dari bagian depan."<<endl;
294     } else cout<<"Tidak ada data pada Linked
List"<<endl;
295 }
296
297 void hapusBelakangH() {
298     Tnode *hapus;
299     string data;
300     if(isEmptyH() == 0) {
301         hapus = head;
302         while(hapus->next != NULL) {
303             hapus = hapus->next;
304         }
305         data = hapus->data;
306         if(head->next != NULL) {
307             hapus->prev->next = NULL;
308         } else {
309             initH();
310         }
311         delete hapus;
312         cout<<"Data \""<<data<<"\" berhasil dihapus
dari bagian belakang."<<endl;
313     } else cout<<"Tidak ada data pada Linked
List"<<endl;
314 }
315
316 void hapusBelakangHT() {
317     Tnode *hapus;
318     string data;
319     if(isEmptyHT() == 0) {
320         hapus = tail;

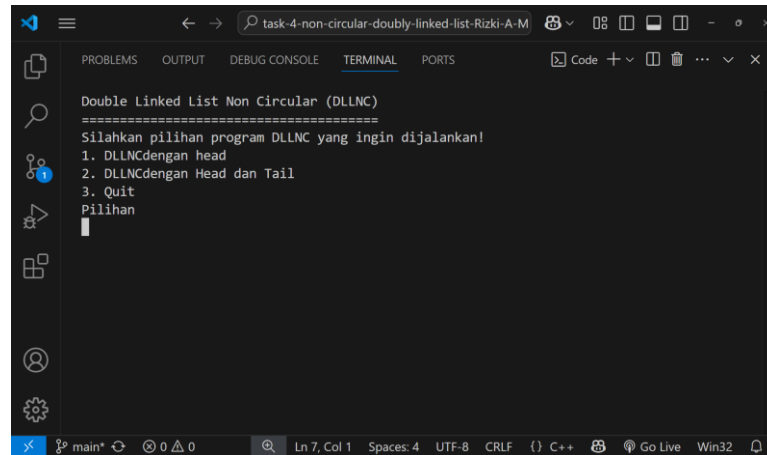
```

```

321         data = hapus->data;
322         if(head->next != NULL) {
323             tail = tail->prev;
324             tail->next = NULL;
325         } else {
326             initHT();
327         }
328         delete hapus;
329         cout<<"Data \""<<data<<"\" berhasil dihapus
dari bagian belakang."<<endl;
330     } else cout<<"Tidak ada data pada Linked
List"<<endl;
331 }
332
333 void clearH() {
334     Tnode *bantu, *hapus;
335     bantu = head;
336     while(bantu != NULL) {
337         hapus = bantu;
338         bantu = bantu->next;
339         delete hapus;
340     }
341     initH();
342     cout<<"Seluruh data pada Linked List telah
dibersihkan"<<endl;
343 }
344
345 void clearHT() {
346     Tnode *bantu, *hapus;
347     bantu = head;
348     while(bantu != NULL) {
349         hapus = bantu;
350         bantu = bantu->next;
351         delete hapus;
352     }
353     initHT();
354     cout<<"Seluruh data pada Linked List telah
dibersihkan"<<endl;
355 }

```

B Output Program

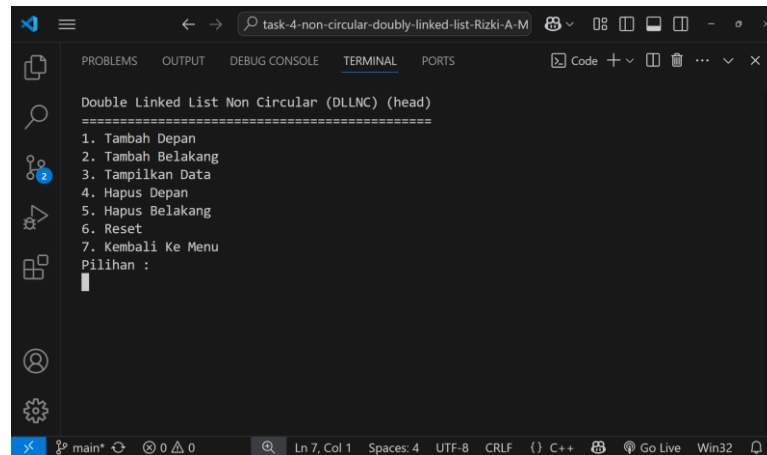


```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC)
=====
Silahkan pilihan program DLLNC yang ingin dijalankan!
1. DLLNC dengan head
2. DLLNC dengan Head dan Tail
3. Quit
Pilihan

```

Gambar 1 Tampilan Menu Double Linked List Non Circular

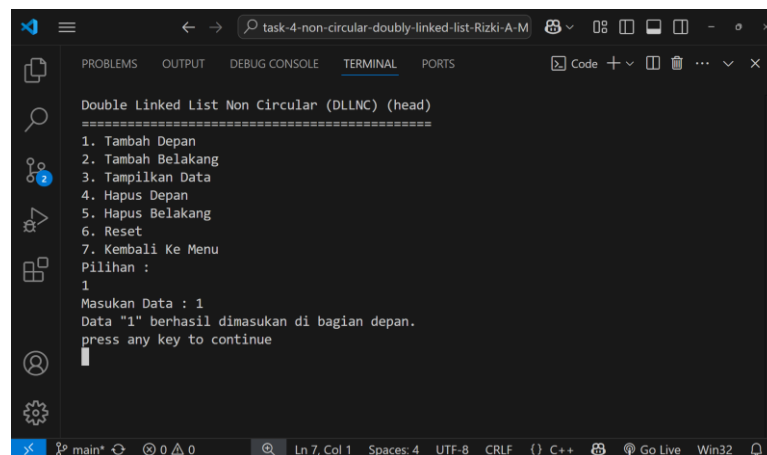


```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :

```

Gambar 2 Masuk Ke Tampilan Menu Head DLLNC



```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
1
Masukan Data : 1
Data "1" berhasil dimasukkan di bagian depan.
press any key to continue

```

Gambar 3 Tambah Data Dari Depan

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - x

Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
3 2 1

press any key to continue

```

Gambar 4 Tampilan Data Setelah Dilakukan Tambah Depan

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - x

Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
2
Masukan Data : 2
Data "2" berhasil dimasukan di bagian belakang.
press any key to continue

```

Gambar 5 Tambah Data Dari Belakang

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - x

Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
3 2 1 2 3

press any key to continue

```

Gambar 6 Tampilan Data Setelah Dilakukan Tambah Belakang

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
4
Data "3" berhasil dihapus dari bagian depan.

press any key to continue

```

Gambar 7 Hapus Data Dari Depan

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
2 1 2 3

press any key to continue

```

Gambar 8 Tampilan Data Setelah Dilakukan Hapus Depan

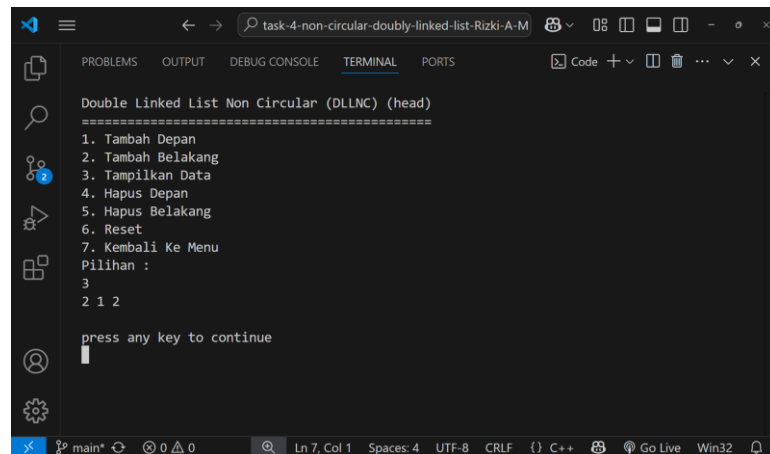
```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
5
Data "3" berhasil dihapus dari bagian belakang.

press any key to continue

```

Gambar 9 Hapus Data Dari Belakang



```

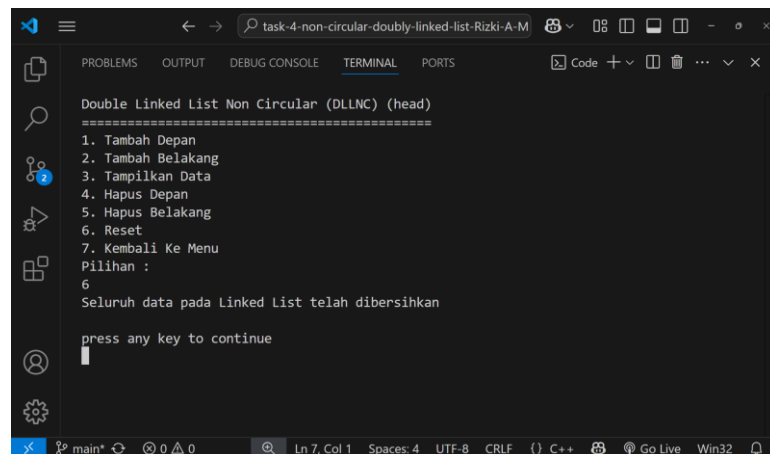
task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - x

Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
2 1 2

press any key to continue

```

Gambar 10 Tampilan Data Setelah Dilakukan Hapus Belakang



```

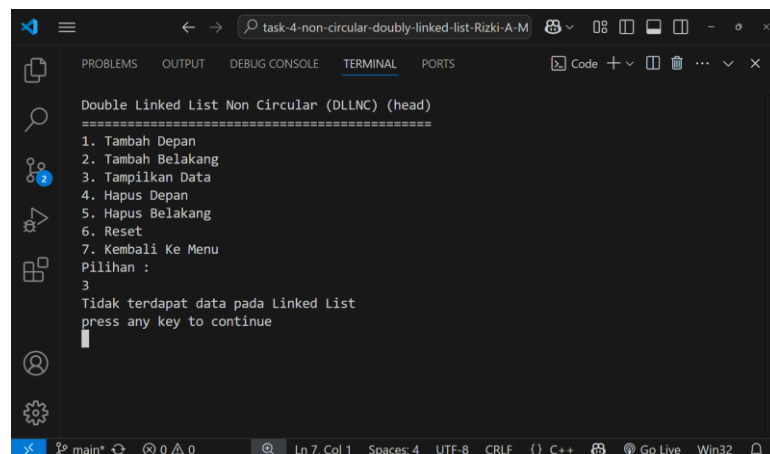
task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - x

Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
6
Seluruh data pada Linked List telah dibersihkan

press any key to continue

```

Gambar 11 Reset Data Yang Ada



```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - x

Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
Tidak terdapat data pada Linked List

press any key to continue

```

Gambar 12 Tampilan Data Setelah Dilakukan Reset Data

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
  
```

Gambar 13 Masuk Ke Tampilan Menu Head Dan Tail DLLNC

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
1
Masukan Data : 3
Data "3" berhasil dimasukan di bagian depan.
press any key to continue
  
```

Gambar 14 Tambah Data Dari Depan

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
1 2 3
press any key to continue
  
```

Gambar 15 Tampilan Data Setelah Dilakukan Tambah Depan


```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - x

Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
2
Masukan Data : 2
Data "2" berhasil dimasukkan di bagian belakang.
press any key to continue

main* 0 0 0 Ln 39, Col 10 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32

```

Gambar 16 Tambah Data Dari Belakang

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - x

Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
1 2 3 2 1

press any key to continue

main* 0 0 0 Ln 39, Col 10 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32

```

Gambar 17 Tampilan Data Setelah Dilakukan Tambah Belakang

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - x

Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
4
Data "1" berhasil dihapus dari bagian depan.

press any key to continue

main* 0 0 0 Ln 39, Col 10 Spaces: 4 UTF-8 CRLF {} C++ Go Live Win32

```

Gambar 18 Hapus Data Dari Depan

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
2 3 2 1

press any key to continue

```

Gambar 19 Tampilan Data Setelah Dilakukan Hapus Depan

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
5
Data "1" berhasil dihapus dari bagian belakang.

press any key to continue

```

Gambar 20 Hapus Data Dari Belakang

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
2 3 2

press any key to continue

```

Gambar 21 Tampilan Data Setelah Dilakukan Hapus Belakang

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
6
Seluruh data pada Linked List telah dibersihkan

press any key to continue

```

Gambar 22 Reset Data Yang Ada

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
Tidak terdapat data pada Linked List

press any key to continue

```

Gambar 23 Tampilan Data Setelah Dilakukan Reset Data

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
TERIMA KASIH
Program was made by Rizki Adhitiya Maulana (2410817110014)
PS D:\task-4-non-circular-doubly-linked-list-Rizki-A-M>

```

Gambar 24 Tampilan Keluar dari Program

C Pembahasan

Pada baris [1] sampai [3] terdapat *#include* yang mana digunakan untuk mengakses sebuah file yang diinginkan. *<iostream>* yang ada digunakan untuk input dan output. Kemudian *<conio.h>* digunakan agar menyediakan fungsi fungsi yang berguna ketika ada interaksi langsung dengan keyboard, tanpa perlu menekan Enter. Terus *<stdlib.h>* digunakan untuk fungsi fungsi manajemen memori, konversi angka, kontrol proses, dan lingkungan program.

Pada baris [5] terdapat using *namespace std;* yang mana digunakan untuk menghindari penulisan std.

Pada baris [7] terdapat [11] terdapat *struct TNode* yang mana digunakan untuk menyimpan elemen-elemen dari linked list, dimana variabel *string data* digunakan untuk menyimpan isi atau informasi dari node tersebut seperti angka yang di input ke dalam list. Terus *Tnode *next* digunakan untuk menunjuk ke node berikutnya dalam urutan linked list, node terakhir yang ada dalam urutan linked list akan menunjuk kembali ke node pertama. Kemudian, *Tnode *prev* digunakan untuk menunjuk ke node sebelumnya dalam urutan linked list, sehingga memungkinkan dilakukannya penelusuran dua arah, baik maju (menggunakan next) maupun mundur (menggunakan prev).

Pada baris [13] terdapat *Tnode *head, *tail* yang mana *TNode *head* menunjuk pointer node pertama dan *TNode *tail* menunjuk pointer node terakhir.

Pada baris [15] sampai [17] terdapat *int pil dan menu* yang mana digunakan untuk menyimpan variabel Integer atau bilangan bulat. Terus *char pilihan [1]* yang mana digunakan untuk menyimpan variabel character, ditambah array sebagai batasan input dari user, *string dataBaru* yang digunakan untuk menyimpan variabel string atau katakter.

Pada baris [19] sampai [35] terdapat penamaan fungsi yang akan dimasukkan ke dalam program Linked list.

Pada baris [37] sampai [141] terdapat *int main()* yang mana digunakan untuk menjalankan dan menampilkan menu SLLC. Menu yang akan ditampilkan di dalam sistem ada sebanyak 3 buah, terdiri dari menu pertama untuk DLLNC

dengan head, menu kedua DLLNC dengan head dan tail, menu terakhir untuk keluar. Setiap pilihan yang ada akan menampilkan tampilan berbeda sesuai dengan fungsi yang ada di dalam *switch case* yang dimasukkan pada program yang akan dijalankan. Terdapat *getch()* untuk menunggu tombol yang ditekan oleh pengguna dan membersihkan layar menggunakan *system("cls")*. Terus program akan terus berjalan selama user tidak memilih pilihan tiga (3) untuk keluar atau menghentikan program yang ada yang terletak di menu pilihan paling awal.

Pada baris [143] sampai [145] terdapat *void initH()* yang mana digunakan untuk menginisialisasikan kondisi awal dari linked list. *InitH()* akan berfokus dalam mengatur pointer atau variabel head ke dalam keadaan NULL.

Pada baris [147] sampai [150] terdapat *void initHT()* yang mana digunakan untuk menginisialisasikan kondisi awal dari linked list. *initHT()* akan berfokus dalam mengatur pointer atau variabel head ke dalam keadaan NULL, begitu juga untuk pointer atau variabel tail yang akan disetting atau diatur kedalam keadaan NULL seperti head.

Pada baris [152] sampai [155] terdapat *isEmptyH()* yang mana digunakan untuk melakukan pengecekan pada linked list, apakah head dalam keadaan kosong atau tidak. Fungsi ini akan mengembalikan nilai 1 apabila linked list dalam keadaan kosong dan 0 apabila tidak dalam keadaan kosong.

Pada baris [157] sampai [160] terdapat *isEmptyHT()* yang mana digunakan untuk melakukan pengecekan pada linked list, apakah head atau tail dalam keadaan kosong atau tidak. Fungsi ini akan mengembalikan nilai 1 apabila linked list dalam keadaan kosong dan 0 apabila tidak dalam keadaan kosong.

Pada baris [162] sampai [178] terdapat *void tambahDepanH()* yang mana digunakan untuk menambahkan node baru ke bagian depan dari linked list, dimana pointer head yang menjadi patokannya. Apabila linked list dalam keadaan kosong, node yang baru ditambahkan akan menjadi head atau node baru. Kemudian, apabila linked list dalam keadaan tidak kosong, node yang baru ditambahkan akan menjadi head atau node pertama dan node yang sudah ada sebelumnya akan menjadi node kedua atau seterusnya.

Pada baris [180] sampai [197] terdapat *void tambahDepanHT()* yang mana digunakan untuk menambahkan node baru ke bagian depan dari linked list, dimana menggunakan head dan tail. Apabila linked list dalam keadaan kosong, head dan tail yang ada akan menunjuk ke node baru. Kemudian, apabila linked list dalam keadaan tidak kosong, node yang baru ditambahkan akan menjadi head atau node pertama dan node yang sudah ada sebelumnya akan menjadi node kedua atau seterusnya.

Pada baris [199] sampai [218] terdapat *void tambahBelakangH()* yang mana digunakan untuk menambahkan node baru ke bagian belakang dari linked list, tanpa bantuan tail yang membuat diperlukannya traversal dari head ke node terakhir, sehingga berkurangan efisiensi yang ada. Apabila linked list dalam keadaan kosong, node yang baru ditambahkan akan menjadi tail atau node terakhir. Kemudian, apabila linked list dalam keadaan tidak kosong, node yang baru ditambahkan akan menjadi tail atau node terakhir dan node yang sudah ada sebelumnya akan menjadi node sebelum node akhir.

Pada baris [220] sampai [237] terdapat *void tambahBelakangHT()* yang mana digunakan untuk menambahkan node baru ke bagian belakang dari linked list, dibantu dengan tail untuk menambahkan node dibagian terakhir, efisiensi yang ada meningkat. Apabila linked list dalam keadaan kosong, node yang baru ditambahkan akan menjadi tail atau node terakhir. Kemudian, apabila linked list dalam keadaan tidak kosong, node yang baru ditambahkan akan menjadi tail atau node terakhir dan node yang sudah ada sebelumnya akan menjadi node sebelum node akhir.

Pada baris [239] sampai [249] terdapat *void tampilkanH()* yang mana digunakan untuk menampilkan seluruh isi linked list yang ada, dimulai dari depan hingga ke belakang. Data atau node yang akan ditampilkan akan dilakukan secara traversal yaitu dari HEAD (node pertama) hingga NULL (node terakhir).

Pada baris [251] sampai [261] terdapat *void tampilkanHT()* yang mana digunakan untuk menampilkan seluruh isi linked list yang ada, dimulai dari depan

hingga ke belakang. Data atau node yang akan ditampilkan akan dilakukan secara traversal yaitu dari HEAD (node pertama) hingga TAIL (node terakhir).

Pada baris [263] sampai [278] terdapat *void hapusDepanH()* yang mana digunakan untuk menghapus node pertama yang terdapat pada linked list, dimana pointer head yang menjadi patokannya. Apabila terdapat satu node saja pada linked list, maka setelah dilakukan hapus depan linked list head akan diatur menjadi kosong atau NULL menggunakan fungsi *InitH()*. Kemudian, apabila terdapat lebih dari satu node yang terdapat pada linked list, node pertama yang dihapus akan digantikan dengan elemen yang ada di setelah atau node kedua sebagai head terbaru.

Pada baris [280] sampai [295] terdapat *void hapusDepanHT()* yang mana digunakan untuk menghapus node pertama yang terdapat pada linked list, dimana menggunakan head dan tail. Apabila terdapat satu node saja pada linked list, maka setelah dilakukan hapus depan linked list head akan diatur menjadi kosong atau NULL menggunakan fungsi *InitHT()*. Kemudian, apabila terdapat lebih dari satu node yang terdapat pada linked list, node pertama yang dihapus akan digantikan dengan elemen yang ada di setelah atau node kedua sebagai head terbaru.

Pada baris [297] sampai [314] terdapat *void hapusBelakangH()* yang mana digunakan untuk menghapus node terakhir yang terdapat pada linked list, tanpa bantuan tail yang membuat diperlukannya traversal dari head ke node terakhir. Apabila terdapat satu node saja pada linked list, maka setelah dilakukan hapus belakang linked list akan diatur menjadi kosong atau NULL menggunakan fungsi *InitH()*. Kemudian, apabila terdapat lebih dari satu node yang terdapat pada linked list, node terakhir yang dihapus akan digantikan dengan elemen yang ada di sebelumnya atau node sebelum node akhir sebagai tail terbaru.

Pada baris [316] sampai [331] terdapat *void hapusBelakangHT()* yang mana digunakan untuk menghapus node terakhir yang terdapat pada linked list, dibantu dengan tail untuk menghapus node dibagian terakhir,. Apabila terdapat satu node saja pada linked list, maka setelah dilakukan hapus belakang linked list akan diatur menjadi kosong atau NULL menggunakan fungsi *InitHT()*. Kemudian, apabila

terdapat lebih dari satu node yang terdapat pada linked list, node terakhir yang dihapus akan digantikan dengan elemen yang ada di sebelumnya atau node sebelum node akhir sebagai tail terbaru.

Pada baris [333] sampai [343] terdapat *void clearH()* yang mana digunakan untuk menghapus semua node yang ada pada linked list, baik dari node pertama hingga node terakhir. Setelah di lakukan penghapusan untuk semua node yang ada di dalam linked list, kemudian akan dipanggil fungsi *InitH()* untuk mengatur linked list ke dalam kondisi kosong.

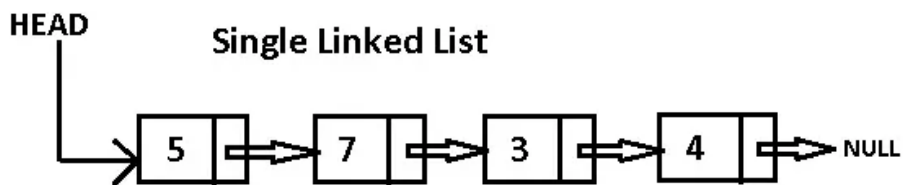
Pada baris [345] sampai [355] terdapat *void clearHT()* yang mana digunakan untuk menghapus semua node yang ada pada linked list, baik dari node pertama hingga node terakhir. Setelah di lakukan penghapusan untuk semua node yang ada di dalam linked list, kemudian akan dipanggil fungsi *InitHT()* untuk mengatur linked list ke dalam kondisi kosong.

SOAL 2

Apa fungsi next pada coding?

A Pembahasan

Fungsi next dalam struktur data linked list dan double linked list digunakan untuk menunjuk ke node berikutnya dalam urutan data. Pada single linked list, setiap node mempunyai satu pointer saja yaitu next yang akan menunjuk ke node berikutnya. Dengan demikian, penelusuran dapat dilakukan dari node pertama (head) ke node terakhir dalam satu arah saja. Pada double linked list, next tetap berfungsi untuk menunjuk ke node berikutnya, tetapi pada struktur ini terdapat penunjuk tambahan yaitu prev yang memungkinkan untuk melakukan penelusuran dua arah.



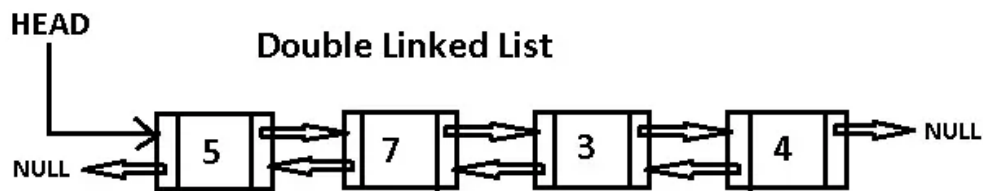
Gambar 25 Fungsi Next Di Single Linked List

SOAL 3

Apa fungsi prev pada coding?

A Pembahasan

Fungsi prev hanya terdapat pada struktur double linked list, yang mana berfungsi untuk menunjuk ke node sebelumnya dari node yang sedang diakses. Dengan menggunakan prev, penjelajahan tidak hanya dapat dilakukan dengan satu arah yaitu maju dengan menggunakan next, tetapi juga dapat dilakukan dengan arah mundur dari node akhir ke node awal. Hal ini akan sangat berguna ketika dalam keadaan dimana kita harus bergerak mundur, menghapus elemen dari belakang, atau membalikkan arah penelusuran. Tanpa menggunakan prev, seperti pada single linked list, operasi-operasi yang ada akan menjadi lebih sulit dan kurang efisien karena kita harus memulai dari awal list setiap kali melakukan penelusuran.



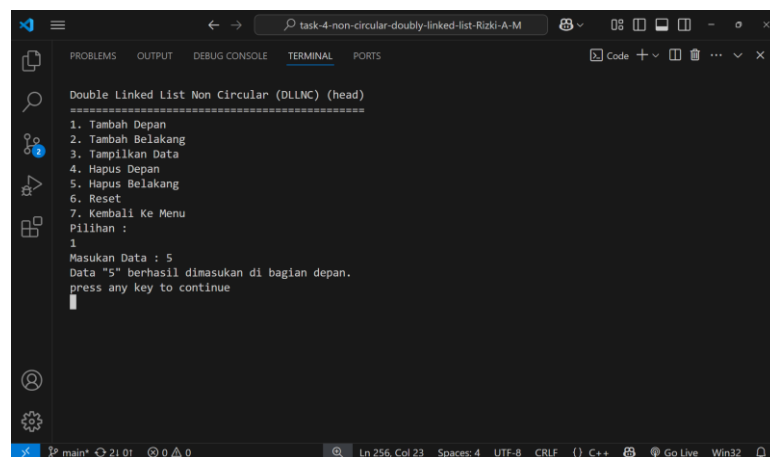
Gambar 26 Fungsi Prev Di Double Linked List

SOAL 4

Gantilah baris 244 dan 256 dari `cout<<"bantu->data<<"`; menjadi `cout<<"head->data<<"`; lalu jawab pertanyaan berikut :

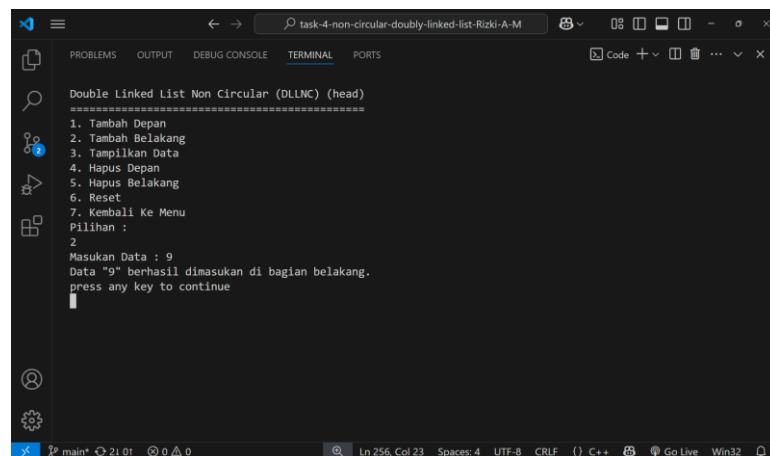
- Apa yang terjadi jika anda menambahkan beberapa data pada program lalu tampilkan datanya, dan screenshoot hasilnya.
- Jelaskan mengapa hal tersebut bisa terjadi dan data apa yang ditampilkan oleh program?

A Output Program



```
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
1
Masukan Data : 5
Data "5" berhasil dimasukan di bagian depan.
press any key to continue
```

Gambar 27 Tampilan Tambah Depan Head



```
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
2
Masukan Data : 9
Data "9" berhasil dimasukan di bagian belakang.
press any key to continue
```

Gambar 28 Tampilan Tambah Belakang Head

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
Masukan Data :
5
press any key to continue

```

Gambar 29 Tampilan Data Head

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
1
Masukan Data : 4
Data "4" berhasil dimasukan di bagian depan.
press any key to continue

```

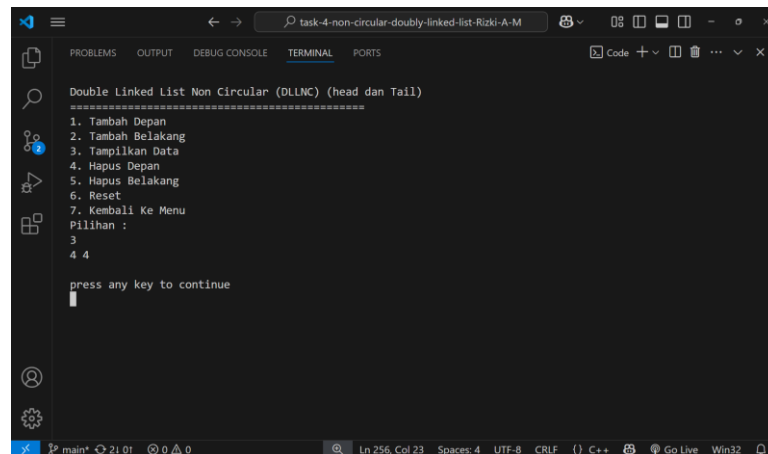
Gambar 30 Tampilan Tambah Depan Head Dan Tail

```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
2
Masukan Data : 8
Data "8" berhasil dimasukan di bagian belakang.
press any key to continue

```

Gambar 31 Tampilan Tambah Belakang Head Dan Tail



```

task-4-non-circular-doubly-linked-list-Rizki-A-M
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code + - - - - - X

Double Linked List Non Circular (DLLNC) (head dan Tail)
=====
1. Tambah Depan
2. Tambah Belakang
3. Tampilkan Data
4. Hapus Depan
5. Hapus Belakang
6. Reset
7. Kembali Ke Menu
Pilihan :
3
4 4

press any key to continue

```

Gambar 32 Tampilan Data Head Dan Tail

B Pembahasan

Setelah dilakukan perubahan atau pergantian kode dari `cout<<bantu->data<<'` '; menjadi `cout<<head->data<<'` ';. Hal yang terjadi adalah program hanya akan mencetak data dari node pertama atau head yang terdapat dari linked list secara berulang-ulang mengikuti banyaknya data atau node yang ada di dalam linked list. Output yang keluar atau ditampilkan tidak menampilkan semua node yang dimasukkan sebelumnya dalam urutan, namun hanya mengulang data atau node pertama dalam output yang keluar. Hal ini berlaku baik di “DLLNC menggunakan head” dan “DLLNC menggunakann head dan tail” dari program yang ada.

Variabel bantu yang ada sebelumnya berguna untuk menunjukkan node perta sampai node terakhir di dalam list. Namun karena varibael bantu yang ada diganti menjadi head, program hanya akan menunjukkan node pertama dan tidak menunjukkan node selanjutnya seperti urutan masuk atau node yang ada hingga tail.

TAUTAN GIT HUB

https://github.com/Rizki-A-M/Rizki-A-M-PRAKTIKUM_ALGORITMA_DAN_STRUKTUR_DATA.git