

## **TP MODUL 6**

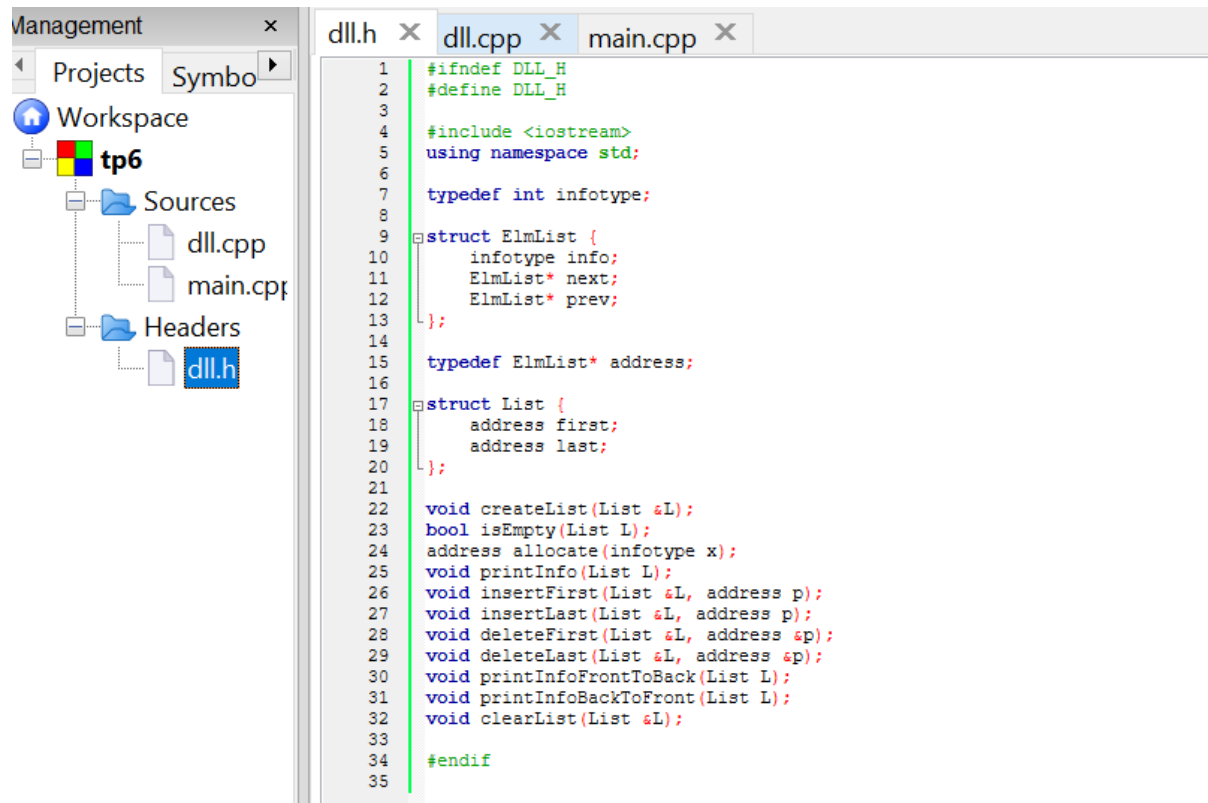
**NAMA : THEODORE ELVIS ESTRADA**

**NIM : 103032400006**

**KELAS : IT 48 01**

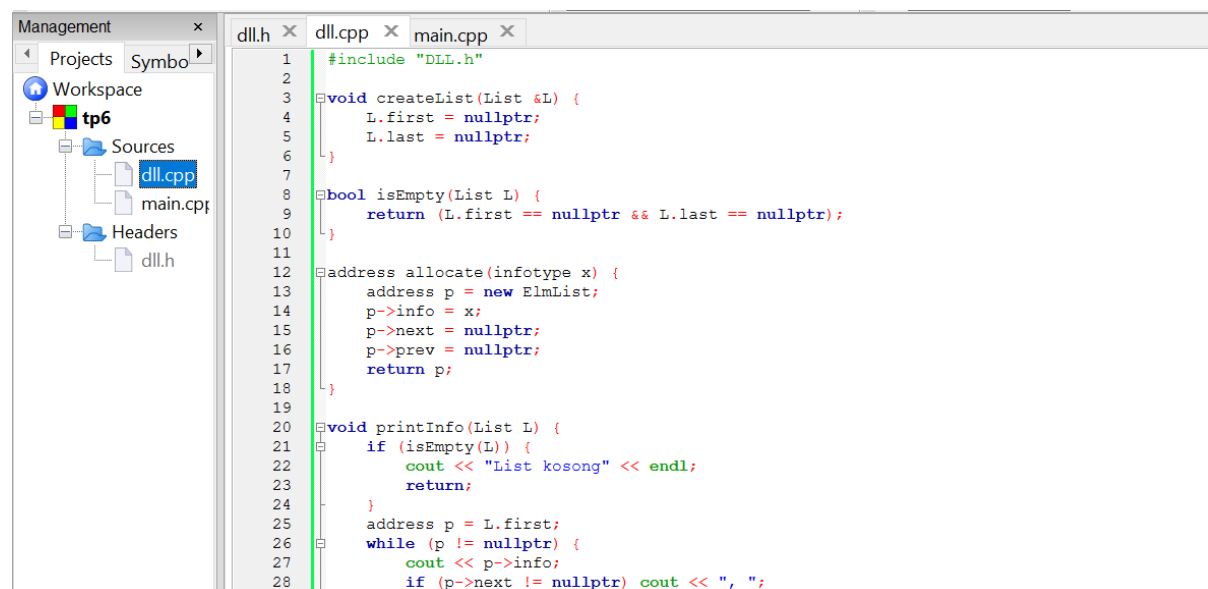
**KODE ASPRAK : MAS**

## DLL.H



```
1  #ifndef DLL_H
2  #define DLL_H
3
4  #include <iostream>
5  using namespace std;
6
7  typedef int infotype;
8
9  struct ElmList {
10     infotype info;
11     ElmList* next;
12     ElmList* prev;
13 };
14
15 typedef ElmList* address;
16
17 struct List {
18     address first;
19     address last;
20 };
21
22 void createList(List &L);
23 bool isEmpty(List L);
24 address allocate(infotype x);
25 void printInfo(List L);
26 void insertFirst(List &L, address p);
27 void insertLast(List &L, address p);
28 void deleteFirst(List &L, address &p);
29 void deleteLast(List &L, address &p);
30 void printInfoFrontToBack(List L);
31 void printInfoBackToFront(List L);
32 void clearList(List &L);
33
34 #endif
35
```

## DLL.CPP



```
1  #include "DLL.h"
2
3  void createList(List &L) {
4      L.first = nullptr;
5      L.last = nullptr;
6  }
7
8  bool isEmpty(List L) {
9      return (L.first == nullptr && L.last == nullptr);
10 }
11
12 address allocate(infotype x) {
13     address p = new ElmList;
14     p->info = x;
15     p->next = nullptr;
16     p->prev = nullptr;
17     return p;
18 }
19
20 void printInfo(List L) {
21     if (isEmpty(L)) {
22         cout << "List kosong" << endl;
23         return;
24     }
25     address p = L.first;
26     while (p != nullptr) {
27         cout << p->info;
28         if (p->next != nullptr) cout << ", ";
29     }
30 }
```

Management x

Projects Symbol

Workspace

tp6

Sources

dll.cpp

main.cpp

Headers

dll.h

```

28     if (p->next != nullptr) cout << ", ";
29     p = p->next;
30 }
31 cout << endl;
32 }
33
34 void insertFirst(List &L, address p) {
35     if (isEmpty(L)) {
36         L.first = p;
37         L.last = p;
38     } else {
39         p->next = L.first;
40         L.first->prev = p;
41         L.first = p;
42     }
43 }
44
45 void insertLast(List &L, address p) {
46     if (isEmpty(L)) {
47         L.first = p;
48         L.last = p;
49     } else {
50         p->prev = L.last;
51         L.last->next = p;
52         L.last = p;
53     }
54 }
55

```

Management x

Projects Symbol

Workspace

tp6

Sources

dll.cpp

main.cpp

Headers

dll.h

```

55
56 void deleteFirst(List &L, address &p) {
57     if (isEmpty(L)) {
58         p = nullptr;
59         return;
60     }
61     if (L.first == L.last) {
62         p = L.first;
63         L.first = nullptr;
64         L.last = nullptr;
65     } else {
66         p = L.first;
67         L.first = L.first->next;
68         L.first->prev = nullptr;
69         p->next = nullptr;
70     }
71 }
72
73 void deleteLast(List &L, address &p) {
74     if (isEmpty(L)) {
75         p = nullptr;
76         return;
77     }
78     if (L.first == L.last) {
79         p = L.last;
80         L.first = nullptr;
81         L.last = nullptr;
82     } else {

```

Management x

Projects Symbol

Workspace

tp6

Sources

dll.cpp

main.cpp

Headers

dll.h

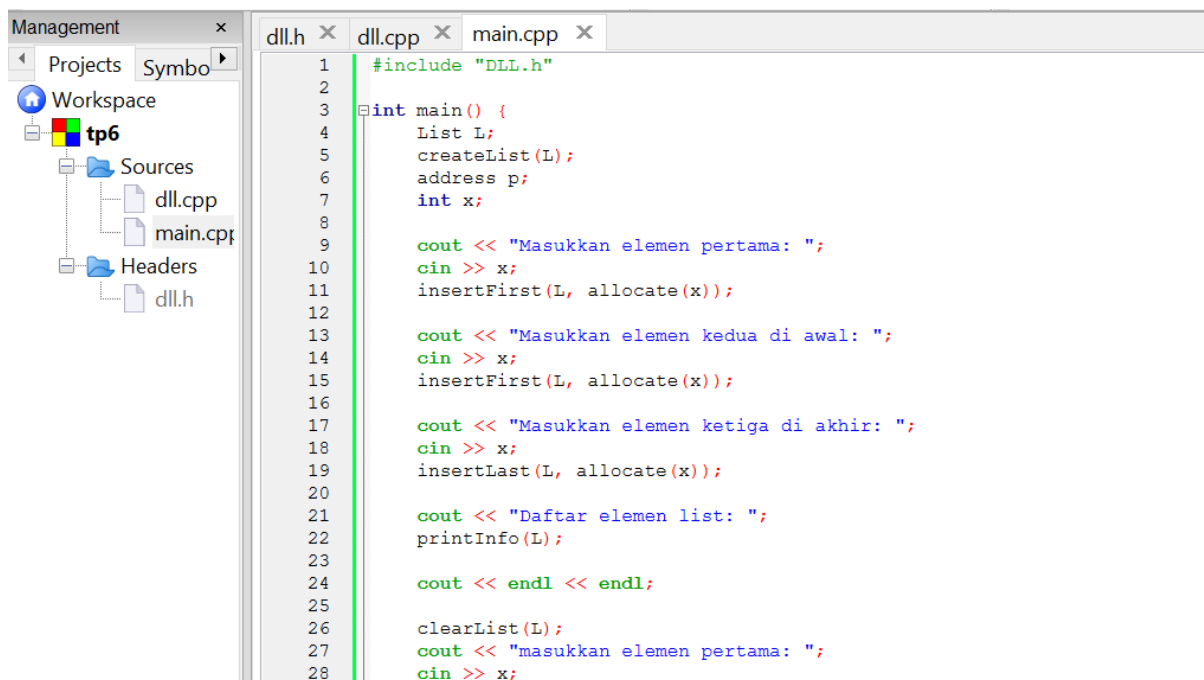
```

82     } else {
83         p = L.last;
84         L.last = L.last->prev;
85         L.last->next = nullptr;
86         p->prev = nullptr;
87     }
88 }
89
90 void printInfoFrontToBack(List L) {
91     address p = L.first;
92     while (p != nullptr) {
93         cout << p->info;
94         if (p->next != nullptr) cout << ", ";
95         p = p->next;
96     }
97     cout << endl;
98 }
99
100 void printInfoBackToFront(List L) {
101     address p = L.last;
102     while (p != nullptr) {
103         cout << p->info;
104         if (p->prev != nullptr) cout << ", ";
105         p = p->prev;
106     }
107     cout << endl;
108 }
109

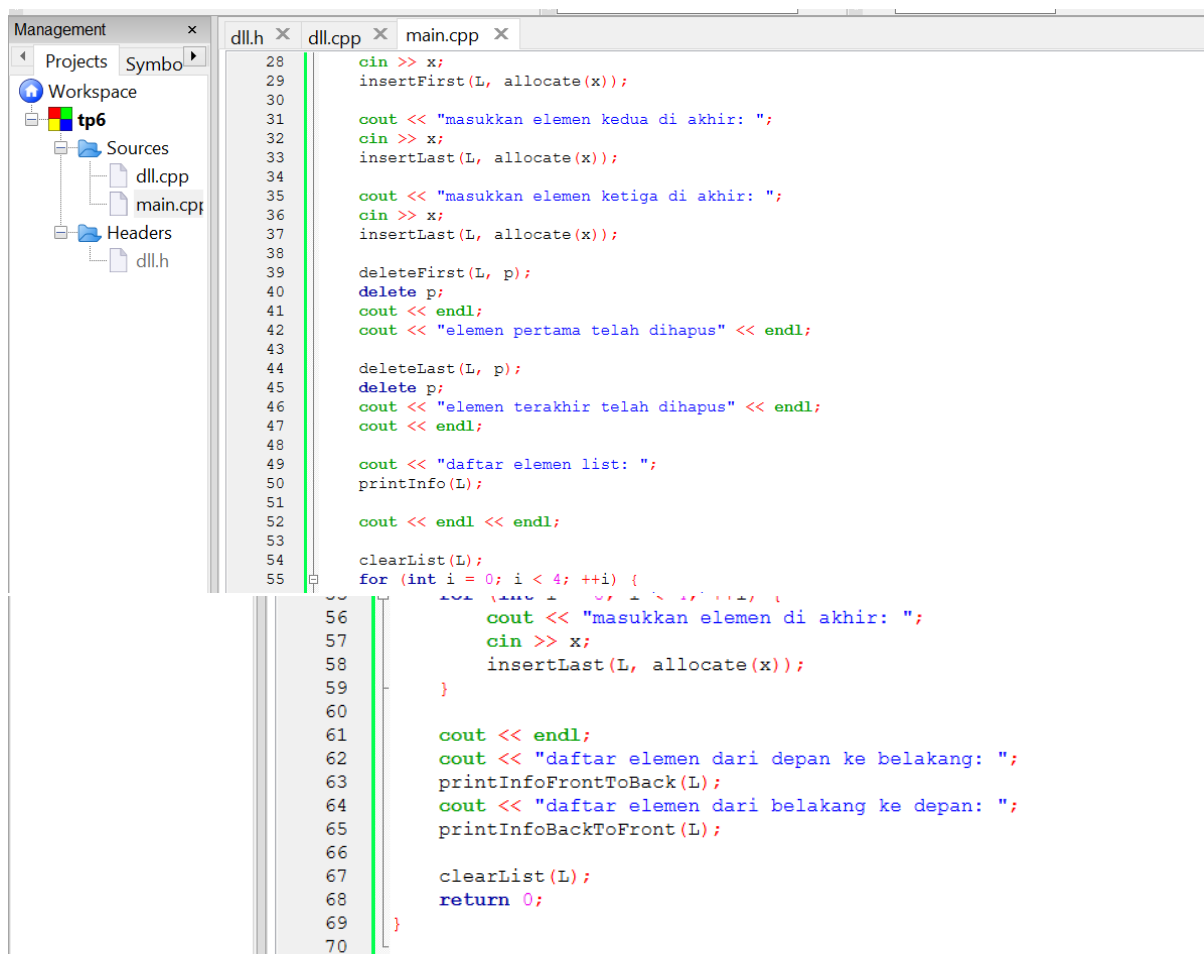
```

```
108     }  
109  
110     void clearList(List &L) {  
111         address p;  
112         while (!isEmpty(L)) {  
113             deleteFirst(L, p);  
114             delete p;  
115         }  
116     }  
117 }
```

# MAIN.CPP



```
1  #include "DLL.h"
2
3  int main() {
4      List L;
5      createList(L);
6      address p;
7      int x;
8
9      cout << "Masukkan elemen pertama: ";
10     cin >> x;
11     insertFirst(L, allocate(x));
12
13     cout << "Masukkan elemen kedua di awal: ";
14     cin >> x;
15     insertFirst(L, allocate(x));
16
17     cout << "Masukkan elemen ketiga di akhir: ";
18     cin >> x;
19     insertLast(L, allocate(x));
20
21     cout << "Daftar elemen list: ";
22     printInfo(L);
23
24     cout << endl << endl;
25
26     clearList(L);
27     cout << "masukkan elemen pertama: ";
28     cin >> x;
```



```
28     cin >> x;
29     insertFirst(L, allocate(x));
30
31     cout << "masukkan elemen kedua di akhir: ";
32     cin >> x;
33     insertLast(L, allocate(x));
34
35     cout << "masukkan elemen ketiga di akhir: ";
36     cin >> x;
37     insertLast(L, allocate(x));
38
39     deleteFirst(L, p);
40     delete p;
41     cout << endl;
42     cout << "elemen pertama telah dihapus" << endl;
43
44     deleteLast(L, p);
45     delete p;
46     cout << "elemen terakhir telah dihapus" << endl;
47     cout << endl;
48
49     cout << "daftar elemen list: ";
50     printInfo(L);
51
52     cout << endl << endl;
53
54     clearList(L);
55     for (int i = 0; i < 4; ++i) {
56         cout << "masukkan elemen di akhir: ";
57         cin >> x;
58         insertLast(L, allocate(x));
59     }
60
61     cout << endl;
62     cout << "daftar elemen dari depan ke belakang: ";
63     printInfoFrontToBack(L);
64     cout << "daftar elemen dari belakang ke depan: ";
65     printInfoBackToFront(L);
66
67     clearList(L);
68     return 0;
69 }
70
```

## HASIL

```
"C:\Users\Zephyrus\Documents\KULIAH\Struktur Data\tp6\bin\Debug\tp6.exe"
Masukkan elemen pertama: 10
Masukkan elemen kedua di awal: 5
Masukkan elemen ketiga di akhir: 20
Daftar elemen list: 5, 10, 20

masukkan elemen pertama: 10
masukkan elemen kedua di akhir: 15
masukkan elemen ketiga di akhir: 20

elemen pertama telah dihapus
elemen terakhir telah dihapus

daftar elemen list: 15

masukkan elemen di akhir: 1
masukkan elemen di akhir: 2
masukkan elemen di akhir: 3
masukkan elemen di akhir: 4

daftar elemen dari depan ke belakang: 1, 2, 3, 4
daftar elemen dari belakang ke depan: 4, 3, 2, 1
```