Nama : Abiyoso Danar Panji Yudhanto

Kelas = IF 47 05

NIM = 103012300006

Header :

DLL.h DLL.cpp *main.cpp tes.h

```cpp
#include "DLL.h"
#include <iostream>

using namespace std;

int main()
{
    List L, L1, L2, L3;
    infotype x;
    bool tes_isEmpty, tes_findLagu;
    address p, prec;
    string judul;

    tes_isEmpty = isEmpty_103012300006(L);

    createNewElmt_103012300006(x);

    insertFirst_103012300006(L, p);

    insertAfter_103012300006(L,prec,p);

    insertLast_103012300006(L, p);

    deleteFirst_103012300006(L, p);

    deleteAfter_103012300006(L, prec, p);

    deleteLast_103012300006(L, p);
```

DLL.h DLL.cpp *main.cpp tes.h

```cpp
    infotype x;
    bool tes_isEmpty, tes_findLagu;
    address p, prec;
    string judul;

    tes_isEmpty = isEmpty_103012300006(L);

    createNewElmt_103012300006(x);

    insertFirst_103012300006(L, p);

    insertAfter_103012300006(L,prec,p);

    insertLast_103012300006(L, p);

    deleteFirst_103012300006(L, p);

    deleteAfter_103012300006(L, prec, p);

    deleteLast_103012300006(L, p);

    concat_103012300006(L1, L2, L3);

    tes_findLagu = findLagu_103012300006(judul, L);

    removeLagu_103012300006(judul, L);
}
```

Cpp :

DLLh DLL.cpp *main.cpp tes.h

```cpp
#include "DLL.h"
#include <iostream>

using namespace std;

bool isEmpty_103012300006(List L){
    return (L.first == NULL && L.last == NULL);
}

address createNewElmt_103012300006(infotype x){
    address p = new elmlist;
    p -> info.judul = x.judul;
    p ->info.namaBand = x.namaBand;
    next(p) = NULL;
    prev(p) = NULL;
    return p;
}

void insertLast_103012300006(List &L, address p) {
    if (isEmpty_103012300006(L) == true) {
        first(L) = p;
        last(L) = p;
    } else {
        prev(p) = last(L);
        last(L) -> next = p;
        last(L) = p;
    }
}

void insertAfter_103012300006(List &L, address &prec, address &p) {
    if (prec != NULL) {
        next(p) = next(prec);
        prev(p) = prec;

        if (next(prec) != NULL) {
            prev(next(prec)) = p;
```

DLLh DLL.cpp *main.cpp tes.h

```cpp
void insertAfter_103012300006(List &L, address &prec, address &p) {
    if (prec != NULL) {
        next(p) = next(prec);
        prev(p) = prec;

        if (next(prec) != NULL) {
            prev(next(prec)) = p;
        } else {
            last(L) = p;
        }

        next(prec) = p;
    }
}

void insertFirst_103012300006(List &L, address p) {
    if (isEmpty_103012300006(L) == true) {
        first(L) = p;
        last(L) = p;
    } else {
        next(p) = first(L);
        first(L) -> prev = p;
        first(L) = p;
    }
}

void deleteFirst_103012300006(List &L, address p) {
    if (isEmpty_103012300006(L) == true) {
        p = NULL;
    }else if (first(L) -> next == NULL) {
        p = first(L);
        first(L) = NULL;
        last(L) = NULL;
    }else {
        p = first(L);
        first(L) = next(p);
```

DLLh DLL.cpp *main.cpp tes.h

```cpp
void deleteFirst_103012300006(List &L, address p) {
    if (isEmpty_103012300006(L) == true) {
        p = NULL;
    }else if (first(L) -> next == NULL) {
        p = first(L);
        first(L) = NULL;
        last(L) = NULL;
    }else {
        p = first(L);
        first(L) = next(p);
        next(p) = NULL;
        first(L) -> prev = NULL;
    }
}

void deleteAfter_103012300006(List &L, address &prec, address p) {
    if (prec != NULL && p != NULL) {
        if (next(prec) == p) {
            next(prec) = next(p);

            if (next(p) != NULL) {
                prev(next(p)) = prec;
            } else {
                last(L) = prec;
            }
            p->next = NULL;
            p->prev = NULL;
            p = NULL;
        }
    }
}

void deleteLast_103012300006(List &L, address p) {
    if (p != NULL) {
        if (p == first(L)) {
            first(L) = NULL;
```

DLLh | DLL.cpp | *main.cpp | tes.h

```
88  void deleteLast_103012300006(List &L, address p) {
89      if (p != NULL) {
90          if (p == first(L)) {
91              first(L) = NULL;
92              last(L) = NULL;
93          } else {
94              last(L) = prev(p);
95              next(last(L)) = NULL;
96          }
97          p->next = NULL;
98          p->prev = NULL;
99      }
100 }
101
102 void concat_103012300006(List L1, List L2, List L3) {
103     if (isEmpty_103012300006(L1) || isEmpty_103012300006(L2)) {
104         if (isEmpty_103012300006(L1)) {
105             L3 = L2;
106         } else {
107             L3 = L1;
108         }
109     } else {
110         last(L1) -> next = first(L2);
111         first(L2) -> prev = last(L1);
112         first(L3) = first(L1);
113         last(L3) = last(L2);
114     }
115
116 }
117
118 address findLagu_103012300006(string judul, List L) {
119     address cek;
120     address ketemu;
121     ketemu = NULL;
122     cek = first(L);
123     while (cek != NULL && ketemu == NULL) {
```

DLLh | *DLL.cpp | *main.cpp | tes.h

```
118 address findLagu_103012300006(string judul, List L) {
119     address cek;
120     address ketemu;
121     ketemu = NULL;
122     cek = first(L);
123     while (cek != NULL && ketemu == NULL) {
124         if (cek -> info.judul == judul) {
125             ketemu = cek;
126         }else {
127             cek = cek -> next;
128         }
129     }
130     return ketemu;
131 }
132 void removeLagu_103012300006(string judul, List &L) {
133     address p = findLagu_103012300006(judul, L);
134     if (p != NULL) {
135         if (p == first(L)) {
136             if (next(p) != NULL) {
137                 first(L) = next(p);
138                 prev(first(L)) = NULL;
139             } else {
140                 first(L) = NULL;
141                 last(L) = NULL;
142             }
143         } else if (p == last(L)) {
144             last(L) = prev(p);
145             next(last(L)) = NULL;
146         } else {
147             next(prev(p)) = next(p);
148             prev(next(p)) = prev(p);
149         }
150         p->next = NULL;
151         p->prev = NULL;
152     }
153 }
```

Main :

DLL.h | *DLL.cpp | *main.cpp | tes.h

```cpp
#include "DLL.h"
#include <iostream>

using namespace std;

int main()
{
    List L, L1, L2, L3;
    infotype x;
    bool tes_isEmpty, tes_findLagu;
    address p, prec;
    string judul;

    tes_isEmpty = isEmpty_103012300006(L);

    createNewElmt_103012300006(x);

    insertFirst_103012300006(L, p);

    insertAfter_103012300006(L,prec,p);

    insertLast_103012300006(L, p);

    deleteFirst_103012300006(L, p);

    deleteAfter_103012300006(L, prec, p);

    deleteLast_103012300006(L, p);

    concat_103012300006(L1, L2, L3);

    tes_findLagu = findLagu_103012300006(judul, L);

    removeLagu_103012300006(judul, L);
}

```