doublelinkedlist.h | doublelinkedlist.cpp | main.cpp

```h
#ifndef DOUBLELINKEDLIST_H_INCLUDED
#define DOUBLELINKEDLIST_H_INCLUDED
#include <iostream>
#define first(L) L.first
#define last(L) L.last
#define next(p) p->next
#define info(p) p->info

/*
Name : Muhamad Dwiki Riswanda
NIM : 1302194015
*/

using namespace std;

typedef int infotype;
typedef struct elmList *address;

struct elmList {
    infotype info;
    address next;
    address prev;
};

struct List {
    address first;
    address last;
};


bool isEmpty(List L);
void createList(List &L);
void createNewElmt(infotype x, address &p);
void insertFirst(List &L, address p);
void insertAfter(List &L, address p, address &prec);
void insertLast(List &L, address p);
void deleteFirst(List &L, address &p);
void deleteAfter(List &L, address prec, address &p);
void deleteLast(List &L, address &p);
void concat(List L1, List L2, List &L3);
float median(List L);
int jumlahList(List L);
void show(List L);


#endif // DOUBLELINKEDLIST_H_INCLUDED

```

doublelinkedlist.h | doublelinkedlist.cpp | main.cpp

```cpp
#include "doublelinkedlist.h"

/*
Name : Muhamad Dwiki Riswanda
NIM : 1302194015
*/

using namespace std;

bool isEmpty(List L) {
    if (first(L)==NULL) {
        return true;
    } else {
        return false;
    }
}

void createList(List &L) {
    first(L) = NULL;
    last(L) = NULL;
}

void createNewElmt(infotype x, address &p) {
    p = new elmList;
    info(p) = x;
    next(p) = NULL;
    p->prev = NULL;
}

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

void insertAfter(List &L, address p, address &prec) {
    next(p) = prec->next;
    p->prev = prec;
    prec->next = p;

    p->next->prev = p;
}

void insertLast(List &L, address p) {
    if (isEmpty(L)) {
        first(L) = p;
        last(L) = p;
    } else {
        p->prev = last(L);
        (last(L))->next = p;
        last(L) = p;
```

doublelinkedlist.h | doublelinkedlist.cpp | main.cpp

```
55          (last(L))->next = p;
56          last(L) = p;
57      }
58  }
59
60  void deleteFirst(List &L, address &p) {
61      if (isEmpty(L)){
62          cout << "List Kosong" << endl;
63      } else if (first(L) == last(L)){
64          p = first(L);
65          first(L) = NULL;
66          last(L) = NULL;
67          delete p;
68      } else {
69          p = first(L);
70          first(L) = next(p);
71          next(p) = NULL;
72          (first(L))->prev = NULL;
73
74          delete p;
75      }
76  }
77
78  void deleteAfter(List &L, address prec, address &p){
79      p = (first(L))->next;
80      prec = first(L);
81      prec->next = next(p);
82      next(p)->prev = prec;
83
84      p->prev = NULL;
85      next(p) = NULL;
86
87      delete p;
88  }
89
90  void deleteLast(List &L, address &p){
91      if (isEmpty(L)){
92          cout << "List Kosong" << endl;
93      } else if (first(L) == last(L)){
94          p = first(L);
95          first(L) = NULL;
96          last(L) = NULL;
97          delete p;
98      } else {
99          p = last(L);
100         last(L) = p->prev;
101         (last(L))->next = NULL;
102         p->prev = NULL;
103
104         delete p;
105     }
106 }
107
```

```
107
108 void concat(List L1, List L2, List &L3){
109     L3 = L1;
110
111     L3.last->next = L2.first;
112     L3.last = L2.last;
113 }
114
115 float median (List L){
116     address p;
117     address Q;
118     float mid;
119     float tot;
120     int i;
121
122     tot = 0;
123     i = 0;
124     p = first(L);
125
126     while (p != NULL){
127         i = i + 1;
128         tot = info(p) + tot;
129         p = next(p);
130     }
131     if (first(L) == NULL){
```

doublelinkedlist.h | doublelinkedlist.cpp | main.cpp

```
131     if (first(L) == NULL){
132         return 0;
133     } else if (next(first(L)) == NULL ){
134             return info(next(first(L)));
135     } else {
136         mid = tot / i;
137     }
138     return mid;
139
140 }
141
142 int jumlahList(List L){
143     int i = 1;
144     address p = first(L);
145     while (next(p) != NULL) {
146         i = i + 1;
147         p = next(p);
148     }
149     return i;
150 }
151
152 void show(List L){
153     address p = first(L);
154     while (p != NULL) {
155         cout << info(p) << " ";
156         p = next(p);
157     }
158     cout << endl;
159 }
```

doublelinkedlist.h | doublelinkedlist.cpp | main.cpp

```cpp
#include "doublelinkedlist.h"

/*
Name : Muhamad Dwiki Riswanda
NIM : 1302194015
*/

using namespace std;

int main()
{
    List L;
    address p;

    createList(L);
    createNewElmt(11,p);
    insertFirst(L,p);
    show(L);
    createNewElmt(22,p);
    insertFirst(L,p);
    show(L);
    createNewElmt(33,p);
    insertFirst(L,p);
    show(L);

    cout << "Jumlah elemen ada " << jumlahList(L) << endl;
    cout << "Nilai tengah dari list tersebut adalah " << median(L);
}
```

```
"D:\Telkom University\SE-43-03\Kelas\Semester 2\Struktur Data\TP\TP 4\doublelinkedlist\bin\De
11
22 11
33 22 11
Jumlah elemen ada 3
Nilai tengah dari list tersebut adalah 22
Process returned 0 (0x0)   execution time : 0.023 s
Press any key to continue.
```