Tugas Pendahuluan Struktur Data Modul 4

Nama : Kurniadi Ahmad Wijaya

NIM : 1301194024

Kelas : IF-43-09

File doublelinkedlist.h

```
/*
    Nama : Kurniadi Ahmad Wijaya
    NIM : 1301194024
*/
#include <iostream>
#define first(L) L.first
#define last(L) L.last
#define next(P) P->next
#define prev(P) P->prev
#define info(P) P->info

using namespace std;
typedef int infotype;
typedef struct elmlist *address;

struct List{
    address first;
    address last;
};

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

bool isEmpty(List L);
address searchElm(List L, int data);
void createList(List &L);
void createNewElmt(infotype x, address &P);
void insertFirst(List &L, address P);
void insertAfter(List &L, address &Prec, address &P);
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);
void show(List L);
float median(List L);
```

File doublelinkedlist.cpp

```
void show(List L){
    address Q = first(L);
    while(Q != NULL){
        cout << info(Q) << ", ";
        Q = next(Q);
    }
}
```

```
/*
    Nama : Kurniadi Ahmad Wijaya
    NIM : 1301194024
*/

#include "doublelinkedlist.h"

bool isEmpty(List L){
    return first(L) == NULL;
}

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;
    prev(P) = NULL;
}

address searchElm(List L, int data){
    address Q = first(L);
    bool found;
    while(Q != NULL && !found){
        if(info(Q) == data){
            found = true;
        } else {
            Q = next(Q);
        }
    }
    return Q;
}
void deleteFirst(List &L, address &P){
    if(isEmpty(L)){
        cout << "List Is Empty" << endl;
    } else {
        P = first(L);
        first(L) = next(first(L));
        next(P) = NULL;
        prev(first(L)) = NULL;
        next(P) = NULL;
    }
}

void deleteLast(List &L, address &P){
    if(isEmpty(L)){
        cout << "List Is Empty" << endl;
    } else {
        P = last(L);
        last(L) = prev(last(L));
        prev(P) = NULL;
        next(last(L)) = NULL;
    }
}

void deleteAfter(List &L, address Prec, address &P){
    if(next(Prec) == last(L)){
        deleteLast(L, P);
    }else if(!isEmpty(L)){
        P = next(Prec);
        address Q = next(next(Prec));
        prev(Q) = Prec;
        next(Prec) = Q;
        next(P) = NULL;
        prev(P) = NULL;
    }else {
        cout << "No Data After Prec" << endl << endl;
    }
}

void concat(List L1, List L2, List &L3){
    next(last(L1)) = first(L2);
    prev(first(L2)) = last(L1);

    first(L3) = first(L1);
    last(L3) = first(L1);
}

float median(List L){
    int i = 0;
    int mid = 0;
    float data;
    bool found;

    address Q = first(L);
    while(Q != NULL){
        i++;
        Q = next(Q);
    }

    address P = first(L);
    mid = i/2;

    for(int j=0; j<mid; j++){
        P = next(P);
    }

    if (i % 2 == 0){
        data = (info(P) + info(prev(P))) / 2.0;
    } else {
        data = info(P);
    }

    return data;
}
void insertFirst(List &L, address P){
    if(isEmpty(L)){
        first(L) = P;
        last(L) = P;
    } else {
        next(P) = first(L);
        prev(first(L)) = P;
        first(L) = P;
    }
}

void insertAfter(List &L, address &Prec, address &P){
    if(isEmpty(L)){
        first(L) = P;
        last(L) = P;
    } else {
        next(P) = next(Prec);
        prev(P) = Prec;
        prev(next(Prec)) = P;
        next(Prec) = P;
    }
}

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

File main.cpp

```
/*
    Nama : Kurniadi Ahmad Wijaya
    NIM : 1301194024
*/
#include "doublelinkedlist.h"

int main()
{
    List L, L1, L2, L3;
    address P;
    infotype x;
    createList(L);

    cout << "Insert First (10, 20)" << endl;
    x = 10;
    createNewElmt(x, P);
    insertFirst(L, P);
    x = 20;
    createNewElmt(x, P);
    insertFirst(L, P);
    show(L);

    cout << endl << endl << "Insert Last (30, 40, 50)" << endl;
    x = 30;
    createNewElmt(x, P);
    insertLast(L, P);
    x = 40;
    createNewElmt(x, P);
    insertLast(L, P);
    x = 50;
    createNewElmt(x, P);
    insertLast(L, P);
    show(L);

    cout << endl << endl << "Insert After 10 (60)" << endl;
    address Q = searchElm(L, 10);
    x = 60;
    createNewElmt(x, P);
    insertAfter(L, Q, P);
    show(L);

    cout << endl << endl << "Delete First" << endl;
    deleteFirst(L, P);
    show(L);

    cout << endl << endl << "Delete Last" << endl;
    deleteLast(L, P);
    show(L);

    cout << endl << endl << "Delete After (60)" << endl;
    address A = searchElm(L, 60);
    deleteAfter(L, A, P);
    show(L);

    cout << endl << endl;

    createList(L1);
    x = 0;
    for(int i=0; i<4; i++){
        x = x + 2;
        createNewElmt(x, P);
        insertFirst(L1, P);
    }
    cout << "List L1 :" << endl;
    show(L1);

    createList(L2);
    x = 0;
    for(int i=0; i<5; i++){
        x = x + 1;
        createNewElmt(x, P);
        insertFirst(L2, P);
    }
    cout << "List L2 :" << endl;
    show(L2);

    cout << endl << endl;
    createList(L3);
    concat(L1, L2, L3);
    cout << "List L3 :" << endl;
    show(L3);

    cout << endl << endl;
    cout << "Median L3 : " << median(L3);

    cout << endl;
}
```

Output Kode

```
"C:\Users\ShinyQ\Desktop\Strukdat Path\TP4\b...
Insert First (10, 20)
20, 10,

Insert Last (30, 40, 50)
20, 10, 30, 40, 50,

Insert After 10 (60)
20, 10, 60, 30, 40, 50,

Delete First
10, 60, 30, 40, 50,

Delete Last
10, 60, 30, 40,

Delete After (60)
10, 60, 40,

List L1 :
8, 6, 4, 2,

List L2 :
5, 4, 3, 2, 1,

List L3 :
8, 6, 4, 2, 5, 4, 3, 2, 1,

Median L3 : 5
```