Nama : Helmi Efendi Lubis

NIM : 1301223338

DLL.h

#ifndef DLL\_H\_INCLUDED

#define DLL\_H\_INCLUDED

#include <iostream>

using namespace std;

*// typedef int infotype;*

struct infotype {

string **Band**;

string **judul**;

};

typedef struct elmList \*address;

struct elmList {

infotype **info**;

address **prev**;

address **next**;

};

struct List {

address **first**;

address **last**;

};

bool isEmpty\_1301223338(List L);

address createNewElmt\_1301223338(infotype X, address P);

void insertFirst\_1301223338(List &L, address P);

void insertAfter\_130122333(List &L, address &Prec, address &P);

void insertLast\_1301223338(List &L, address P);

void deleteFirst\_1301223338(List &L, address &P);

void deleteAfter\_1301223338(List &L, address Prec, address &P);

void deleteLast\_1301223338(List &L, address &P);

void concat\_1301223338(List L1, List L2, List &L3);

address findLagu\_1301223338(string judul, List L);

void removeLagu\_1301223338(string judul, List &L);

void selectMenu\_1301223338();

void show\_1301223338(List L);

#endif *// DLL\_H\_INCLUDED*

DLL.cpp

#include "DLL.h"

bool isEmpty\_1301223338(List L){

return L.**first** == NULL;

}

address createNewElmt\_1301223338(infotype X, address P) {

P = new elmList;

P->**info** = X;

P->**next** = NULL;

P->**prev** = NULL;

return P;

}

void insertFirst\_1301223338(List &L, address P) {

if(isEmpty\_1301223338(L)){

L.**first** = P;

L.**last** = P;

}else {

P->**next** = L.**first**;

L.**first**->**prev** = P;

L.**first** = P;

}

}

void insertAfter\_130122333(List &L, address &Prec, address &P) {

if(Prec == L.**last**) {

insertLast\_1301223338(L, P);

}else {

P->**next** = Prec->**next**;

P->**prev** = Prec;

Prec->**next**->**prev** = P;

Prec->**next** = P;

}

}

void insertLast\_1301223338(List &L, address P) {

if(isEmpty\_1301223338(L)){

L.**first** = P;

L.**last** = P;

}else {

P->**prev** = L.**last**;

L.**last**->**next** = P;

L.**last** = P;

}

}

void deleteFirst\_1301223338(List &L, address &P) {

P = L.**first**;

L.**first** = P->**next**;

P->**next** = NULL;

L.**first**->**prev** = NULL;

}

void deleteAfter\_1301223338(List &L, address Prec, address &P) {

if(Prec->**next** == L.**last**){

deleteLast\_1301223338(L, P);

}else {

P = Prec->**next**;

Prec->**next** = P->**next**;

P->**next**->**prev** = Prec;

P->**next** = NULL;

P->**prev** = NULL;

}

}

void deleteLast\_1301223338(List &L, address &P) {

P = L.**last**;

L.**last** = P->**prev**;

P->**prev** = NULL;

L.**last**->**next** = NULL;

}

void concat\_1301223338(List L1, List L2, List &L3) {

L3.**first** = NULL; L3.**last** = NULL;

address p = L1.**first**;

address q = L2.**first**;

while(p != NULL){

insertLast\_1301223338(L3, p);

p = p->**next**;

}

while(q != NULL){

insertLast\_1301223338(L3, q);

q = q->**next**;

}

}

address findLagu\_1301223338(string judul, List L){

address p, q, find;

if (isEmpty\_1301223338(L)){

find = NULL;

}

else{

p = L.**first**;

q = L.**last**;

while(p->**info**.**judul** != judul && q->**info**.**judul** != judul && p != q && p->**next** != q){

p = p->**next**;

q = q->**next**;

}

if(p->**info**.**judul** == judul) {

find = p;

}else if(q->**info**.**judul** == judul){

find = q;

}else{

find = NULL;

}

}

return find;

}

void removeLagu\_1301223338(string judul, List &L) {

address check = findLagu\_1301223338(judul, L);

address temp;

if (isEmpty\_1301223338(L))

{

cout << "List Kosong" << endl;

}

else

{

if(L.**first** == check && L.**last** == check){

L.**first** = NULL;

L.**last** = NULL;

}else {

if(L.**first** == check) {

deleteFirst\_1301223338(L, check);

}else if(L.**last** == check){

deleteLast\_1301223338(L, check);

}else {

deleteAfter\_1301223338(L, check, temp);

}

}

}

}

void selectMenu\_1301223338() {

cout << "\n===== MENU =====" << endl;

cout << "1. Menambah N data baru" << endl;

cout << "2. Menampilkan semua data" << endl;

cout << "3. Menggabungkan list1 dan list2" << endl;

cout << "4. Mencari Lagu di dalam List" << endl;

cout << "5. Menghapus lagu di dalam list" << endl;

cout << "0. Exit" << endl;

cout << "Pilihan menu: ";

}

void show\_1301223338(List L){

address p = L.**first**;

if(isEmpty\_1301223338(L)){

cout << "List Kosong" << endl;

}else {

while(p != NULL){

cout << "Nama Band : " << p->**info**.**Band** << " "

<< "Judul Lagu : " << p->**info**.**judul** << endl;

p = p->**next**;

}

}

}

Main.cpp

#include <iostream>

#include "DLL.h"

#include "DLL.cpp"

using namespace std;

int main()

{

List L1, L2, L3;

int n, inpList, inpLoc, pilihan, i = 1;

address p,q,r;

infotype x;

L1.**first** = NULL; L1.**last** = NULL;

L2.**first** = NULL; L2.**last** = NULL;

selectMenu\_1301223338();

cin >> pilihan;

while (pilihan != 0)

{

switch (pilihan)

{

case 1:

cout << "\nMau masukin berapa lagu : ";

cin >> n;

while (i <= n)

{

cout << "\nMau input ke-List 1 atau List-2: ";

cin >> inpList;

cout << "Input ke-" << i << endl;

cout << "Masukkan nama band : ";

cin >> x.**Band**;

cout << "Masukkan judul lagu : ";

cin >> x.**judul**;

q = createNewElmt\_1301223338(x, p);

cout << "\nInput di:\n 1.Awal \n 2.Tengah \n 3.Akhir";

cout << "\nSilahkan pilih: ";

cin >> inpLoc;

if (inpList == 1)

{

if(inpLoc == 1){

insertFirst\_1301223338(L1, q);

}else if(inpLoc == 2){

show\_1301223338(L1);

cout << "Mau di-input setelah lagu apa: ";

cin >> x.**judul**;

r = findLagu\_1301223338(x.**judul**, L1);

insertAfter\_130122333(L1, r, q);

}else {

insertLast\_1301223338(L1, q);

}

}else if (inpList == 2){

if(inpLoc == 1){

insertFirst\_1301223338(L2, q);

}else if(inpLoc == 2){

show\_1301223338(L2);

cout << "Mau di-input setelah lagu apa: ";

cin >> x.**judul**;

r = findLagu\_1301223338(x.**judul**, L2);

insertAfter\_130122333(L2, r, q);

}else {

insertLast\_1301223338(L2, q);

}

}

i = i + 1;

}

i = 1;

break;

case 2:

cout << "\nIsi List 1 : " << endl;

show\_1301223338(L1);

cout << "\nIsi List 2 : " << endl;

show\_1301223338(L2);

cout << "\nIsi List 3 : " << endl;

show\_1301223338(L3);

break;

case 3:

concat\_1301223338(L1, L2, L3);

cout << "\nIsi List 3 : " << endl;

show\_1301223338(L3);

break;

case 4:

cout << "\nMau cari lagu di: \n1.List-1 \n2.List-2 \n3.List-3";

cout << "\nSilahkan pilih: ";

cin >> inpList;

cout << "\nMasukkan judul lagu : ";

cin >> x.**judul**;

if(inpList == 1){

p = findLagu\_1301223338(x.**judul**, L1);

}else if(inpList == 2){

p = findLagu\_1301223338(x.**judul**, L2);

}else {

p = findLagu\_1301223338(x.**judul**, L3);

}

if (p != NULL) {

cout << "Nama Band : " << p->info.**Band** << " "

<< "Judul Lagu : " << p->info.**judul** << endl;

}else {

cout << "Lagu tidak ditemukan" << endl;

}

break;

case 5:

cout << "\nMau hapus lagu di: \n1.List-1 \n2.List-2 \n3.List-3";

cout << "\nSilahkan pilih: ";

cin >> inpList;

cout << "\nMasukkan judul lagu : ";

cin >> x.**judul**;

if(inpList == 1){

removeLagu\_1301223338(x.**judul**, L1);

}else if(inpList == 2){

removeLagu\_1301223338(x.**judul**, L2);

}else {

removeLagu\_1301223338(x.**judul**, L3);

}

break;

default:

cout << "\nInput tidak valid" << endl;

break;

}

selectMenu\_1301223338();

cin >>pilihan;

}

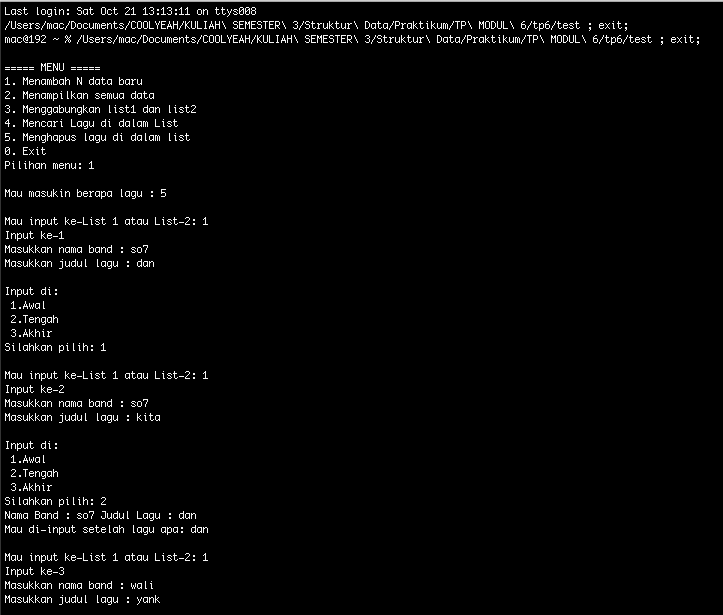
cout << "\nTerima kasih" << endl;

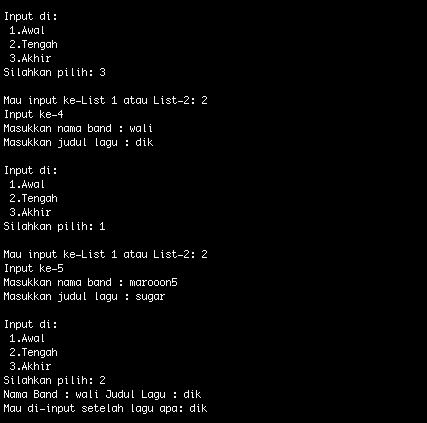
*//*

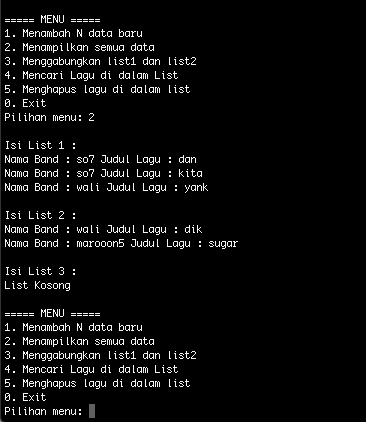
}

**Hasil:**

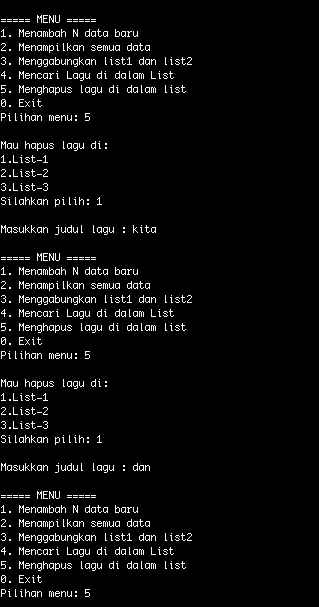
1. **Coba procedure insert.**

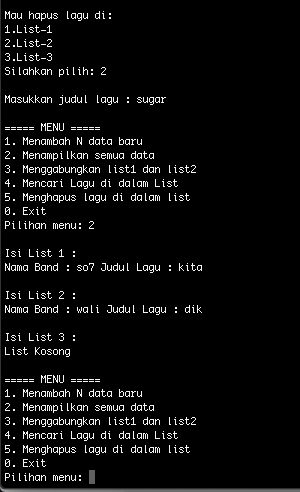




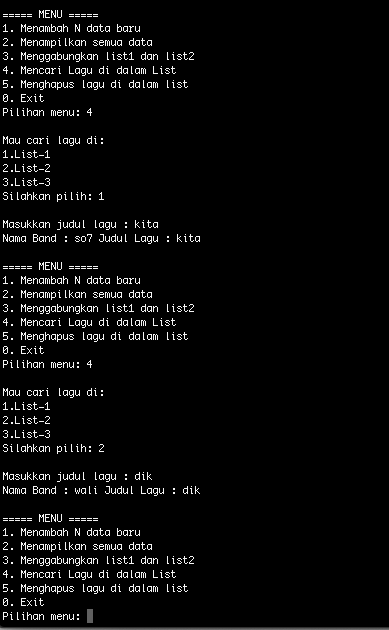


1. **Coba procedure delete**





1. **Coba function find**



1. **Coba procedure concat**

