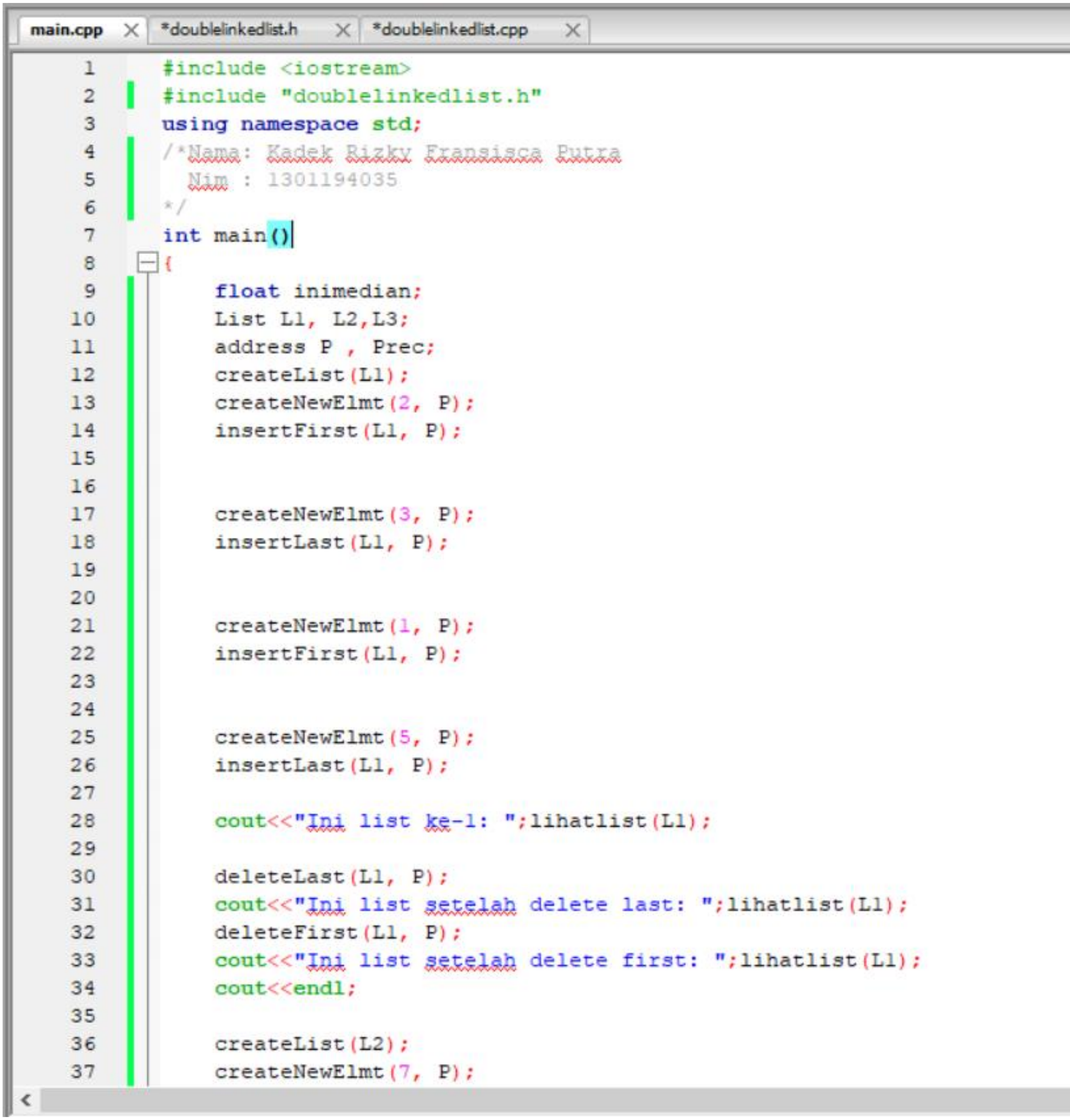
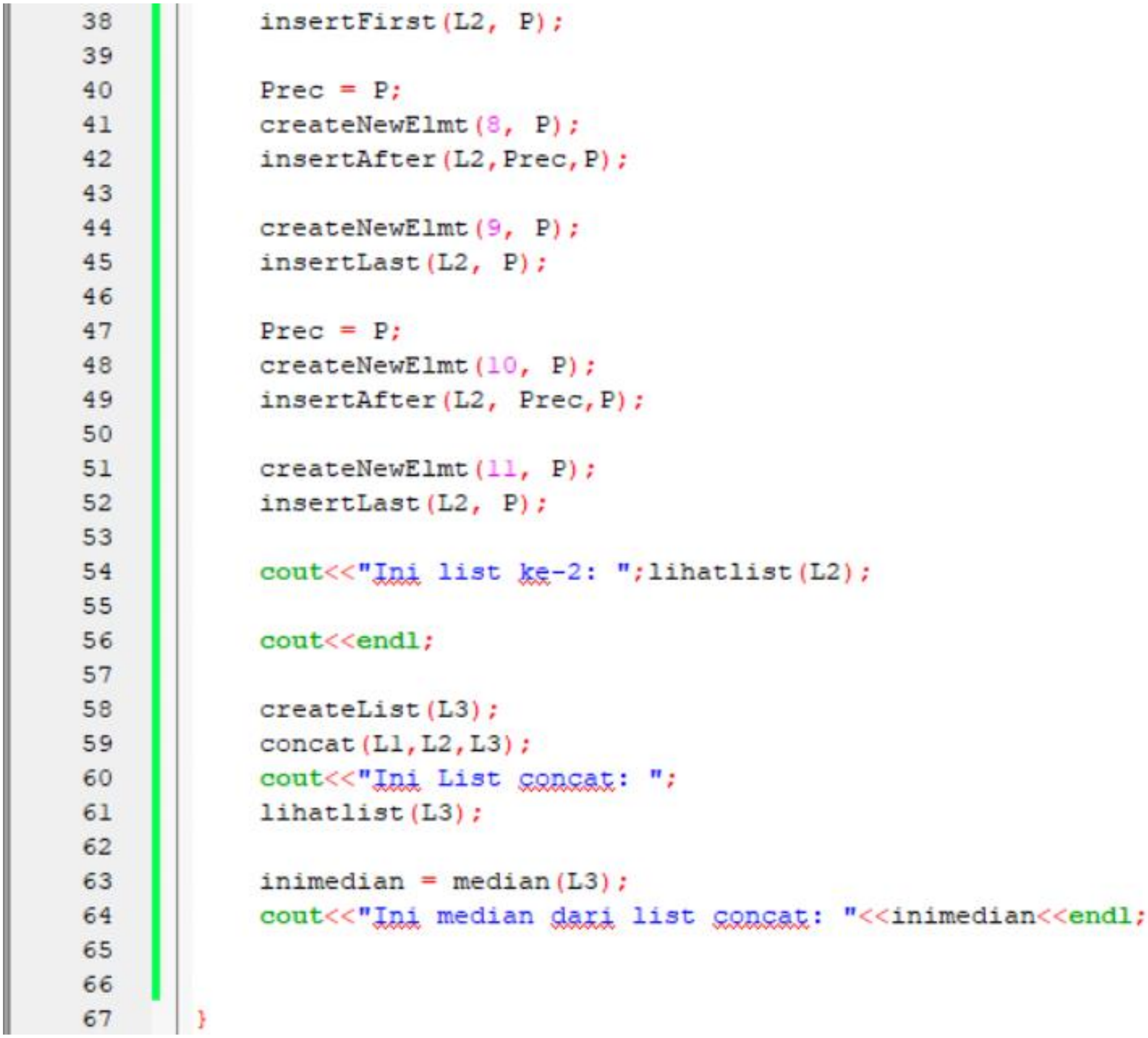
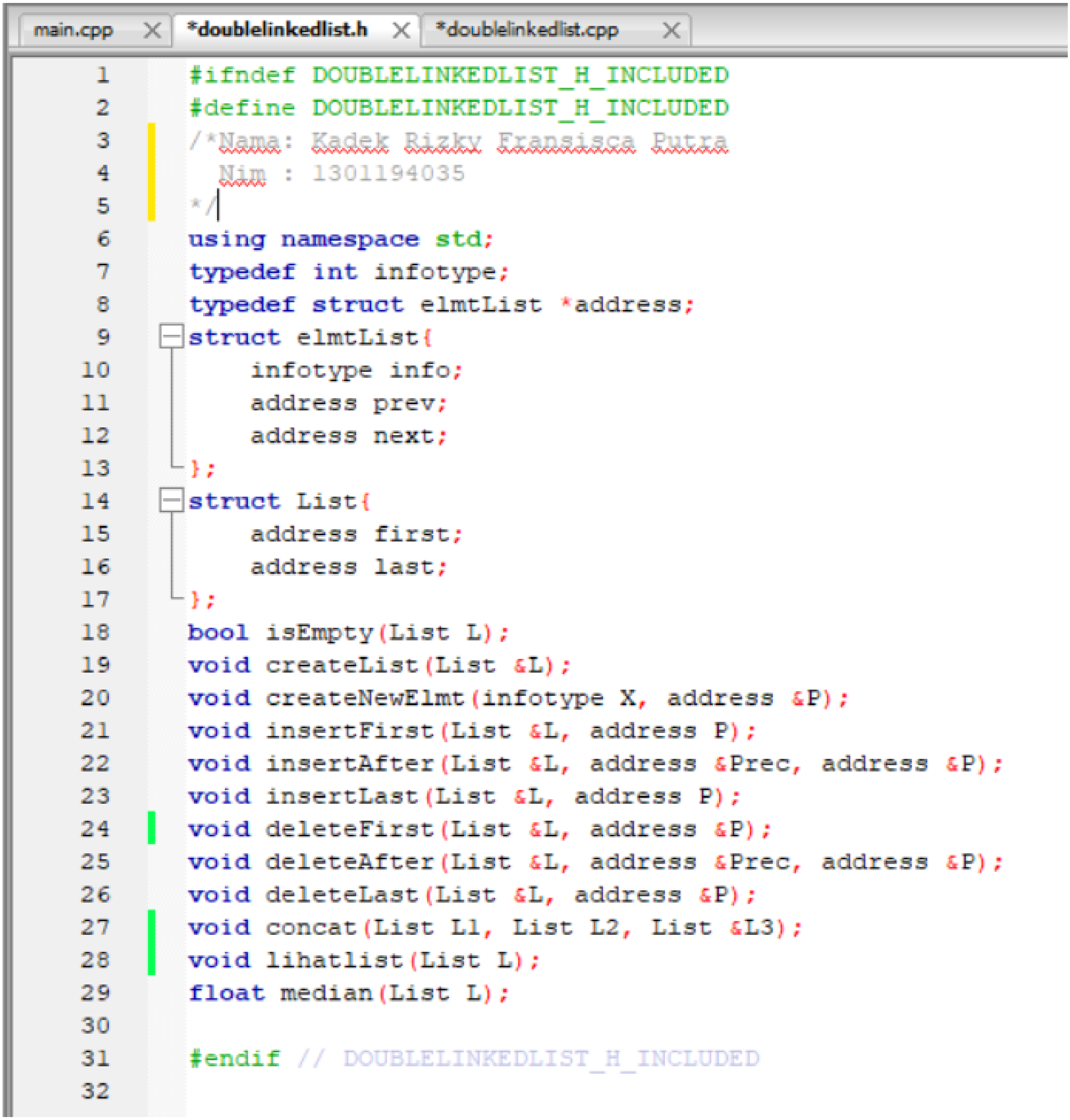
NAMA: Kadek Rizky Fransisca Putra / 1301194035



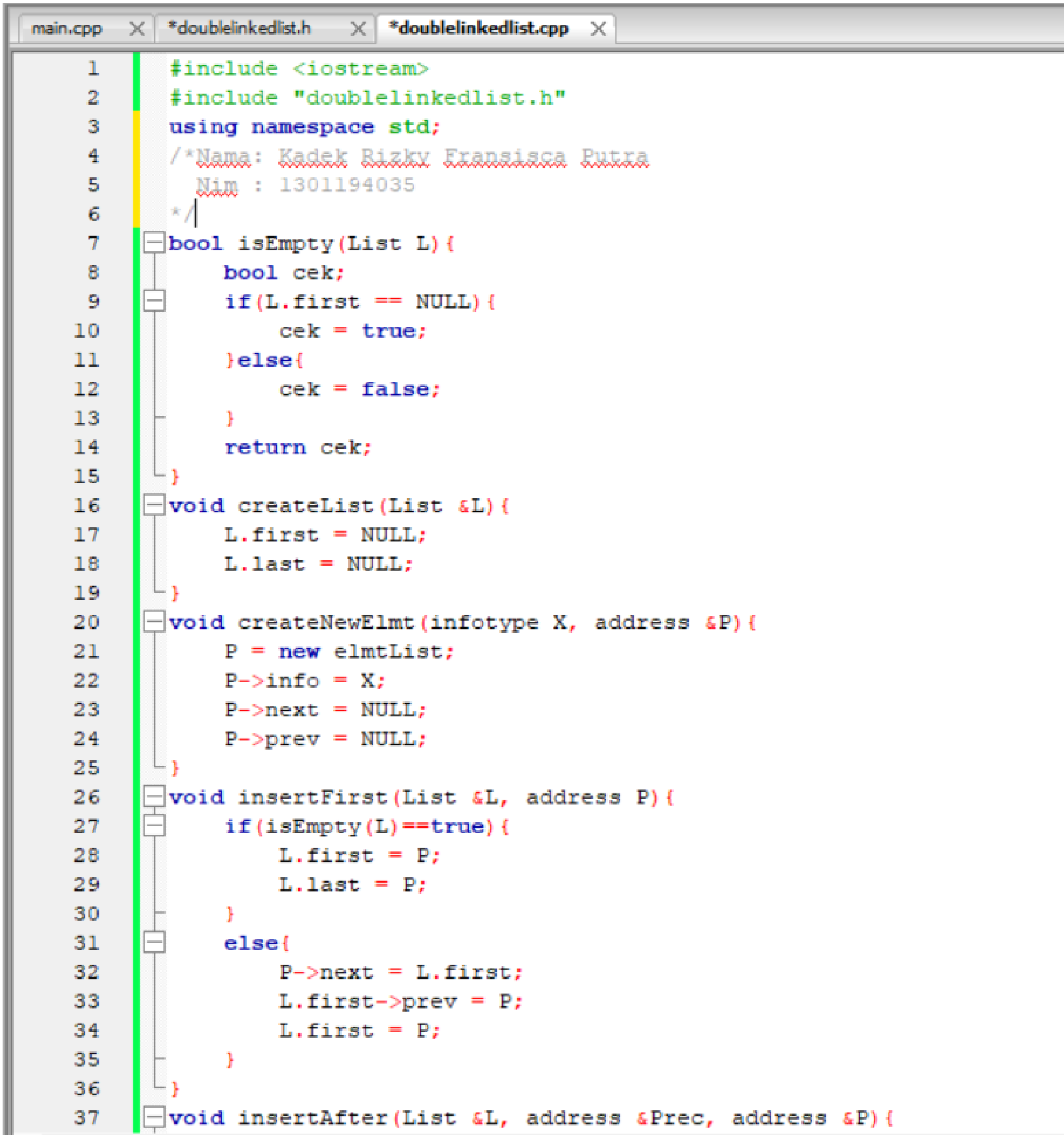
```
#include <iostream>
#include "doublelinkedlist.h"
using namespace std;
/*Nama: Kadek Rizky Fransisca Putra
  Nim : 1301194035
*/
int main()
{
    float inimedian;
    List L1, L2,L3;
    address P , Prec;
    createList(L1);
    createNewElmt(2, P);
    insertFirst(L1, P);


    createNewElmt(3, P);
    insertLast(L1, P);


    createNewElmt(1, P);
    insertFirst(L1, P);


    createNewElmt(5, P);
    insertLast(L1, P);

    cout<<"Ini list ke-1: ";lihatlist(L1);

    deleteLast(L1, P);
    cout<<"Ini list setelah delete last: ";lihatlist(L1);
    deleteFirst(L1, P);
    cout<<"Ini list setelah delete first: ";lihatlist(L1);
    cout<<endl;

    createList(L2);
    createNewElmt(7, P);
```



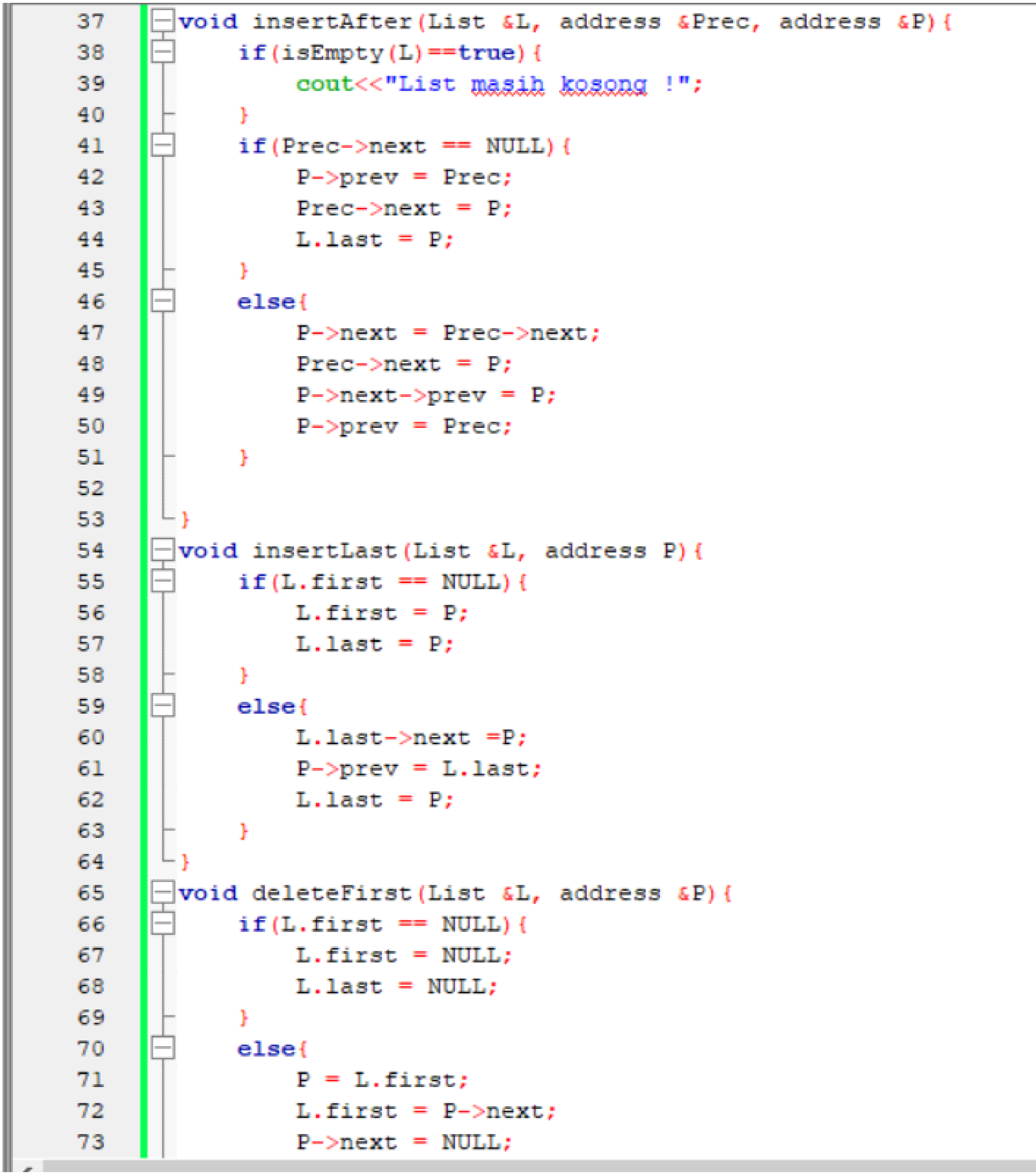
```cpp
    insertFirst(L2, P);

    Prec = P;
    createNewElmt(8, P);
    insertAfter(L2,Prec,P);

    createNewElmt(9, P);
    insertLast(L2, P);

    Prec = P;
    createNewElmt(10, P);
    insertAfter(L2, Prec,P);

    createNewElmt(11, P);
    insertLast(L2, P);

    cout<<"Ini list ke-2: ";lihatlist(L2);

    cout<<endl;

    createList(L3);
    concat(L1,L2,L3);
    cout<<"Ini List concat: ";
    lihatlist(L3);

    inimedian = median(L3);
    cout<<"Ini median dari list concat: "<<inimedian<<endl;


}
```



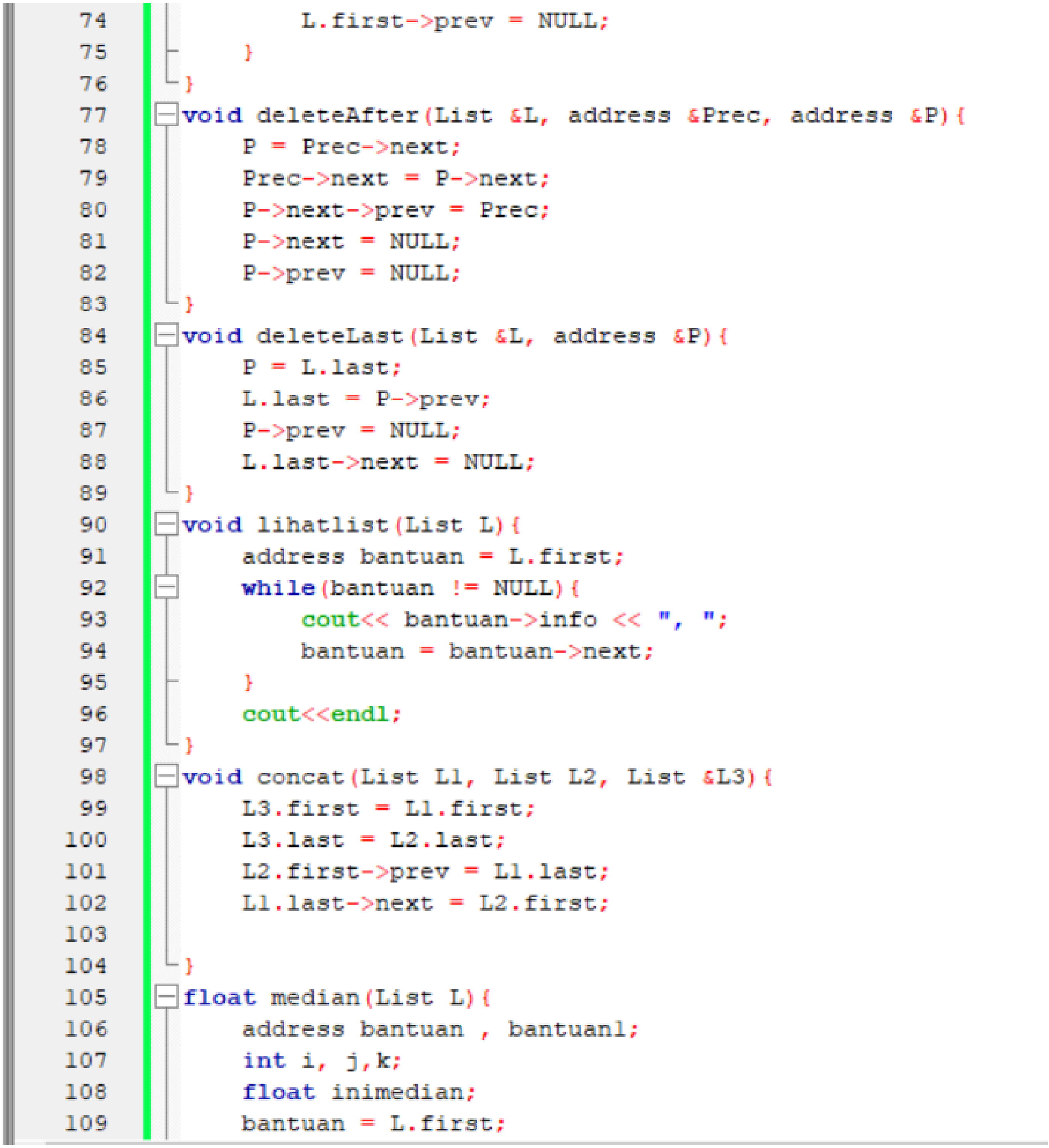
```
#ifndef DOUBLELINKEDLIST_H_INCLUDED
#define DOUBLELINKEDLIST_H_INCLUDED
/*Nama: Kadek Rizky Fransisca Putra
  Nim : 1301194035
*/
using namespace std;
typedef int infotype;
typedef struct elmtList *address;
struct elmtList{
    infotype info;
    address prev;
    address next;
};
struct List{
    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 &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 lihatlist(List L);
float median(List L);

#endif // DOUBLELINKEDLIST_H_INCLUDED

```



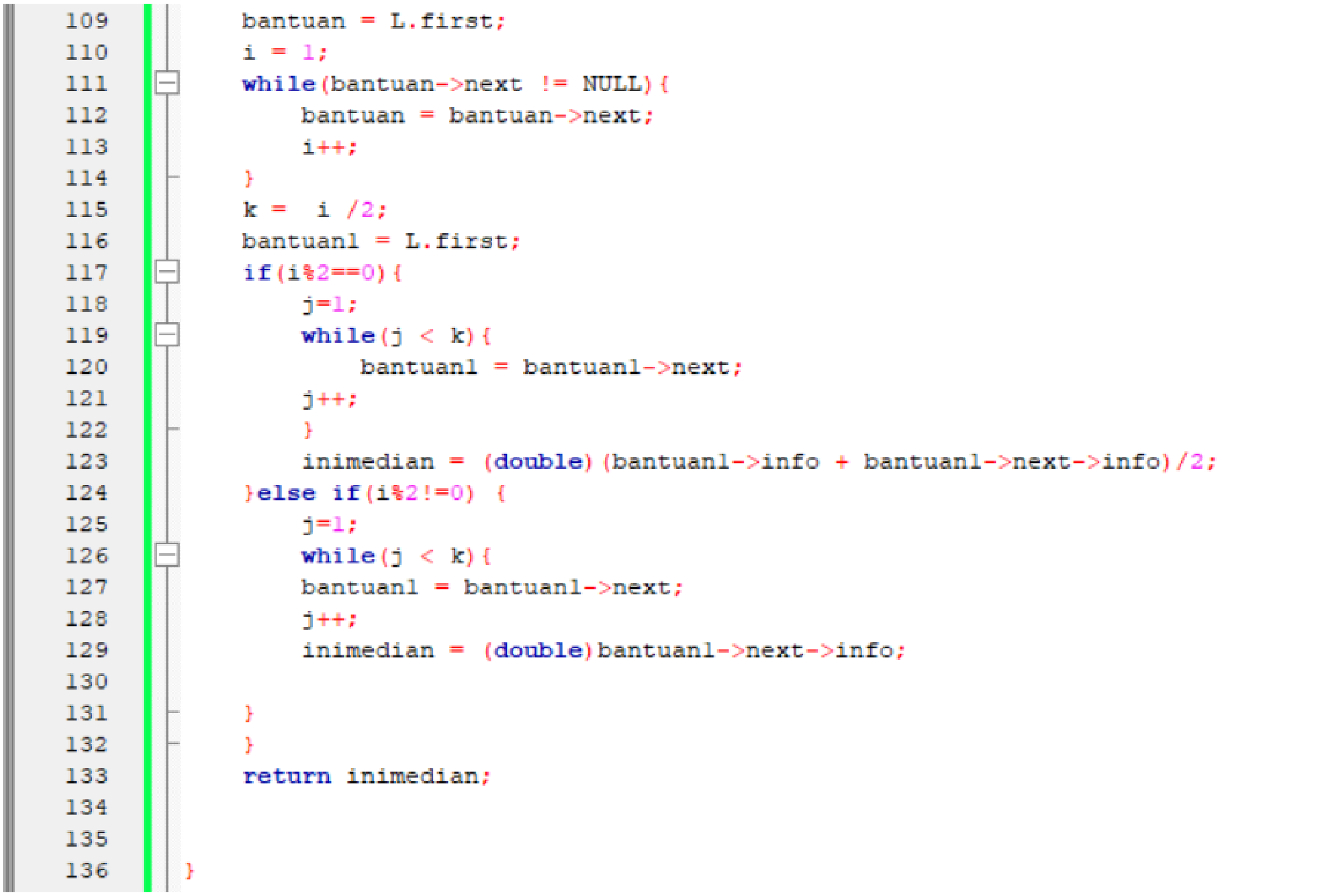
```cpp
#include <iostream>
#include "doublelinkedlist.h"
using namespace std;
/*Nama: Kadek Rizky Fransisca Putra
  Nim : 1301194035
*/
bool isEmpty(List L){
    bool cek;
    if(L.first == NULL){
        cek = true;
    }else{
        cek = false;
    }
    return cek;
}
void createList(List &L){
    L.first = NULL;
    L.last = NULL;
}
void createNewElmt(infotype X, address &P){
    P = new elmtList;
    P->info = X;
    P->next = NULL;
    P->prev = NULL;
}
void insertFirst(List &L, address P){
    if(isEmpty(L)==true){
        L.first = P;
        L.last = P;
    }
    else{
        P->next = L.first;
        L.first->prev = P;
        L.first = P;
    }
}
void insertAfter(List &L, address &Prec, address &P){
```



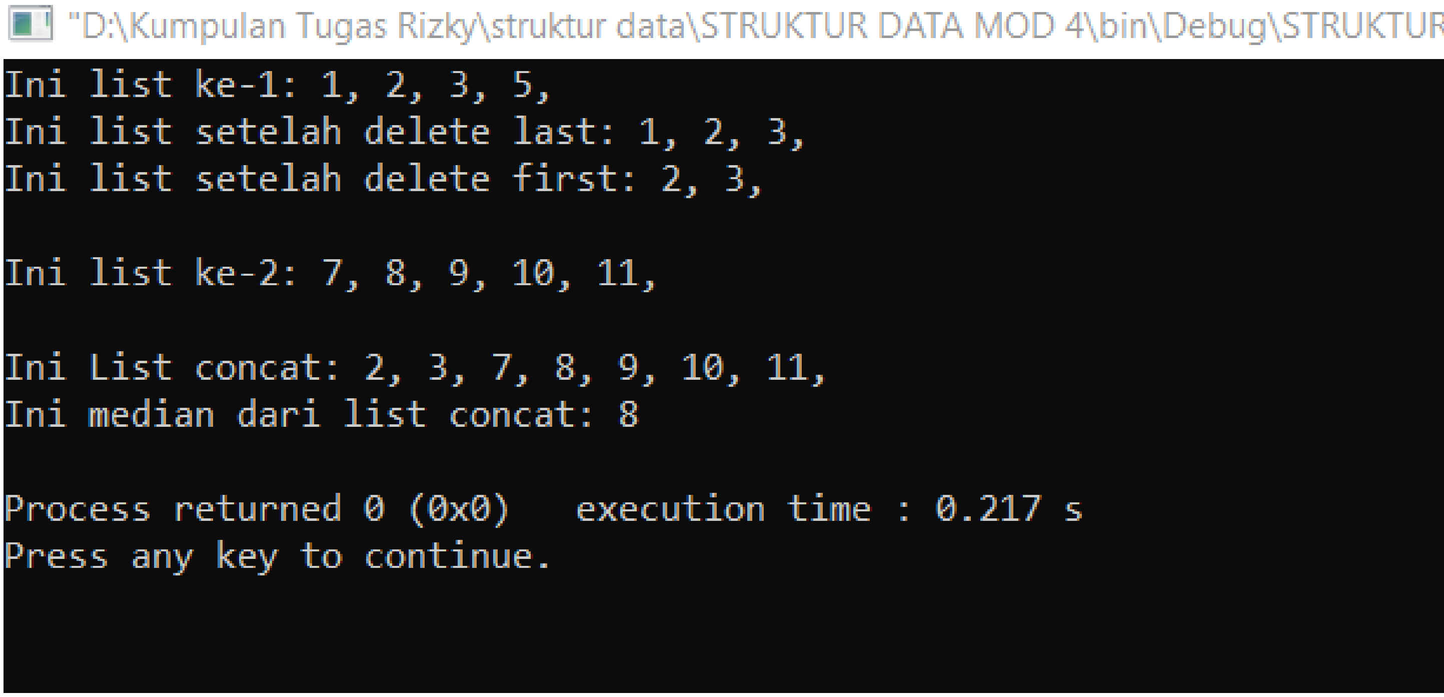
```
void insertAfter(List &L, address &Prec, address &P){
    if(isEmpty(L)==true){
        cout<<"List masih kosong !";
    }
    if(Prec->next == NULL){
        P->prev = Prec;
        Prec->next = P;
        L.last = P;
    }
    else{
        P->next = Prec->next;
        Prec->next = P;
        P->next->prev = P;
        P->prev = Prec;
    }

}
void insertLast(List &L, address P){
    if(L.first == NULL){
        L.first = P;
        L.last = P;
    }
    else{
        L.last->next =P;
        P->prev = L.last;
        L.last = P;
    }
}
void deleteFirst(List &L, address &P){
    if(L.first == NULL){
        L.first = NULL;
        L.last = NULL;
    }
    else{
        P = L.first;
        L.first = P->next;
        P->next = NULL;
```



```
        L.first->prev = NULL;
    }
}
void deleteAfter(List &L, address &Prec, address &P){
    P = Prec->next;
    Prec->next = P->next;
    P->next->prev = Prec;
    P->next = NULL;
    P->prev = NULL;
}
void deleteLast(List &L, address &P){
    P = L.last;
    L.last = P->prev;
    P->prev = NULL;
    L.last->next = NULL;
}
void lihatlist(List L){
    address bantuan = L.first;
    while(bantuan != NULL){
        cout<< bantuan->info << ", ";
        bantuan = bantuan->next;
    }
    cout<<endl;
}
void concat(List L1, List L2, List &L3){
    L3.first = L1.first;
    L3.last = L2.last;
    L2.first->prev = L1.last;
    L1.last->next = L2.first;

}
float median(List L){
    address bantuan , bantuan1;
    int i, j,k;
    float inimedian;
    bantuan = L.first;
```



```
    bantuan = L.first;
    i = 1;
    while(bantuan->next != NULL){
        bantuan = bantuan->next;
        i++;
    }
    k =  i /2;
    bantuanl = L.first;
    if(i%2==0){
        j=1;
        while(j < k){
            bantuanl = bantuanl->next;
        j++;
        }
        inimedian = (double)(bantuanl->info + bantuanl->next->info)/2;
    }else if(i%2!=0) {
        j=1;
        while(j < k){
        bantuanl = bantuanl->next;
        j++;
        inimedian = (double)bantuanl->next->info;

    }
    }
    return inimedian;


}
```



"D:\Kumpulan Tugas Rizky\struktur data\STRUKTUR DATA MOD 4\bin\Debug\STRUKTUR

```
Ini list ke-1: 1, 2, 3, 5,
Ini list setelah delete last: 1, 2, 3,
Ini list setelah delete first: 2, 3,

Ini list ke-2: 7, 8, 9, 10, 11,

Ini List concat: 2, 3, 7, 8, 9, 10, 11,
Ini median dari list concat: 8

Process returned 0 (0x0)   execution time : 0.217 s
Press any key to continue.
```