LAPORAN PRAKTIKUM
Modul 04
“SINGLE LINKED LIST (BAGIAN PERTAMA)”

Disusun Oleh:
M.Faris (2311104017)
SE-07-01

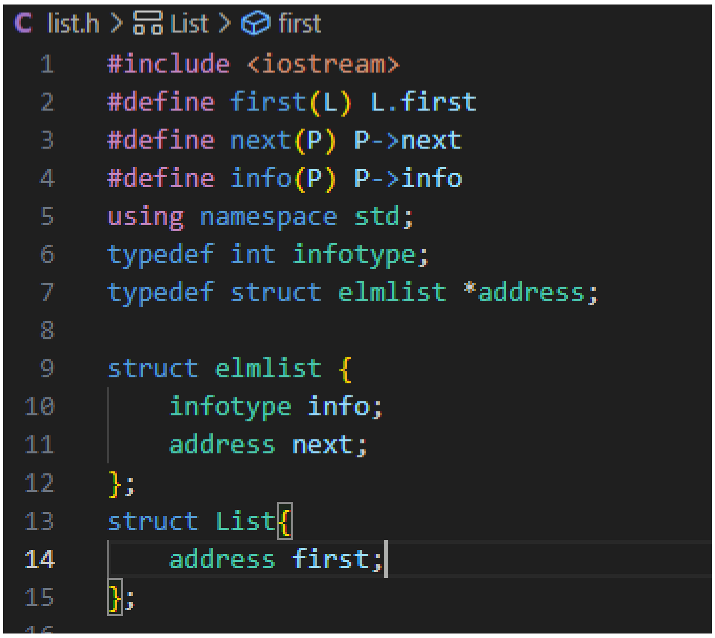
Dosen :
Yudha Islami Sulistiya

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM
PURWOKERTO
2024

A. Soal Tugas Pendahuluan

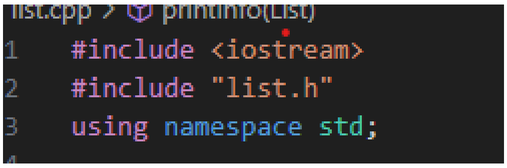
1. Membuat deklarasi tipe list

   List.h



```cpp
#include <iostream>
#define first(L) L.first
#define next(P) P->next
#define info(P) P->info
using namespace std;
typedef int infotype;
typedef struct elmlist *address;

struct elmlist {
    infotype info;
    address next;
};
struct List{
    address first;
};
```

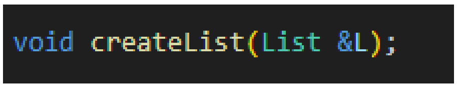
   List.cpp



```cpp
#include <iostream>
#include "list.h"
using namespace std;
```

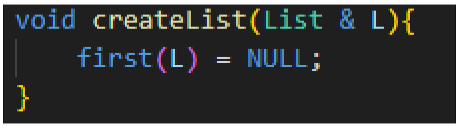
2. Membuat list kosong, yaitu procedure createList

   List.h



```cpp
void createList(List &L);
```

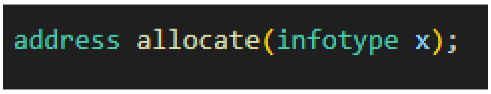
   List.cpp



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

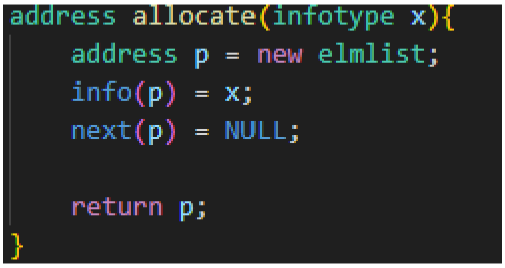
3. Setelah list sudah ada , selanjutnya buat elemen dengan menggunakan fungsi allocate.

   List.h



```cpp
address allocate(infotype x);
```

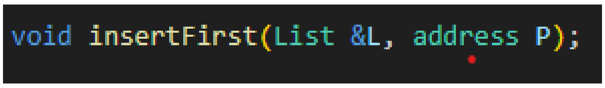
List.cpp



```
address allocate(infotype x){
    address p = new elmlist;
    info(p) = x;
    next(p) = NULL;

    return p;
}
```

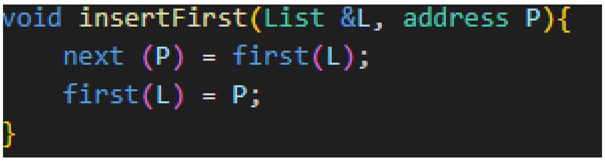
4. Setelah List dan elemen sudah ada, maka selanjutnya elemen tersebut harus diinsert ke List agar bisa menjadi elemen list.

   List.h

   ```
   void insertFirst(List &L, address P);
   ```

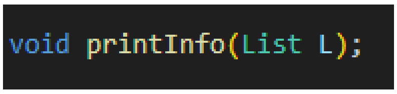
   List.cpp

   ```
   void insertFirst(List &L, address P){
       next (P) = first(L);
       first(L) = P;
   }
   ```

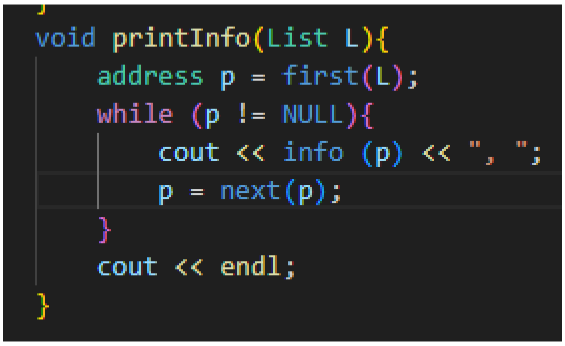
5. Setelah proses insert elemen, maka agar bisa mengetahui apakah elemen berhasil diinsertkan, maka kita perlu menampilkan isi list.

   List.h

   ```
   void printInfo(List L);
   ```

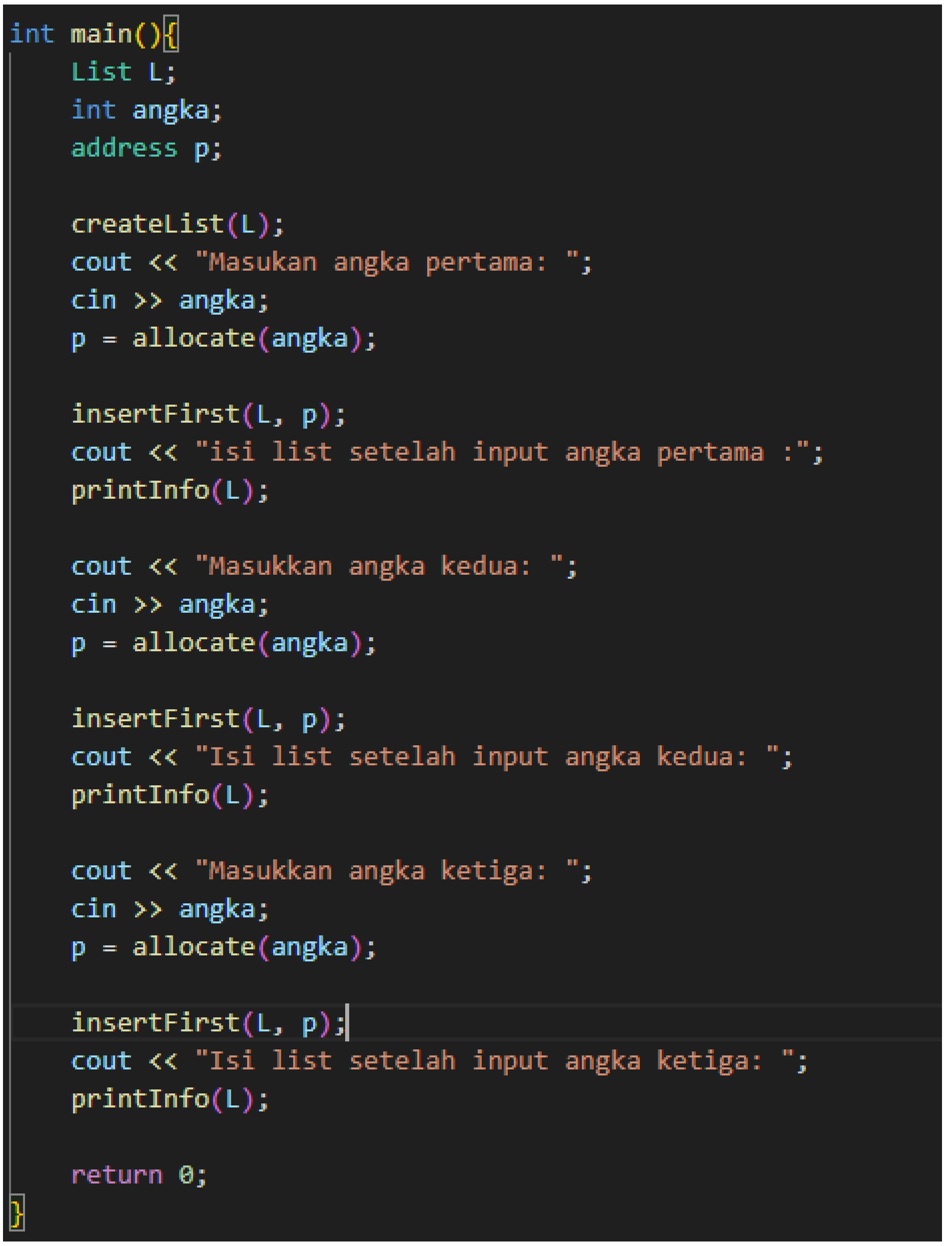
   List.cpp

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

6. Sekarang, setelah ADT List sudah terisi dengan beberapa fungsi Procedur di atas, maka mari buat sebuah List berisi 3 elemen yang berisi 3 digit nim terakhir Anda di main.cpp

   Main.cpp



```
int main(){
    List L;
    int angka;
    address p;

    createList(L);
    cout << "Masukan angka pertama: ";
    cin >> angka;
    p = allocate(angka);

    insertFirst(L, p);
    cout << "isi list setelah input angka pertama :";
    printInfo(L);

    cout << "Masukkan angka kedua: ";
    cin >> angka;
    p = allocate(angka);

    insertFirst(L, p);
    cout << "Isi list setelah input angka kedua: ";
    printInfo(L);

    cout << "Masukkan angka ketiga: ";
    cin >> angka;
    p = allocate(angka);

    insertFirst(L, p);
    cout << "Isi list setelah input angka ketiga: ";
    printInfo(L);

    return 0;
}
```

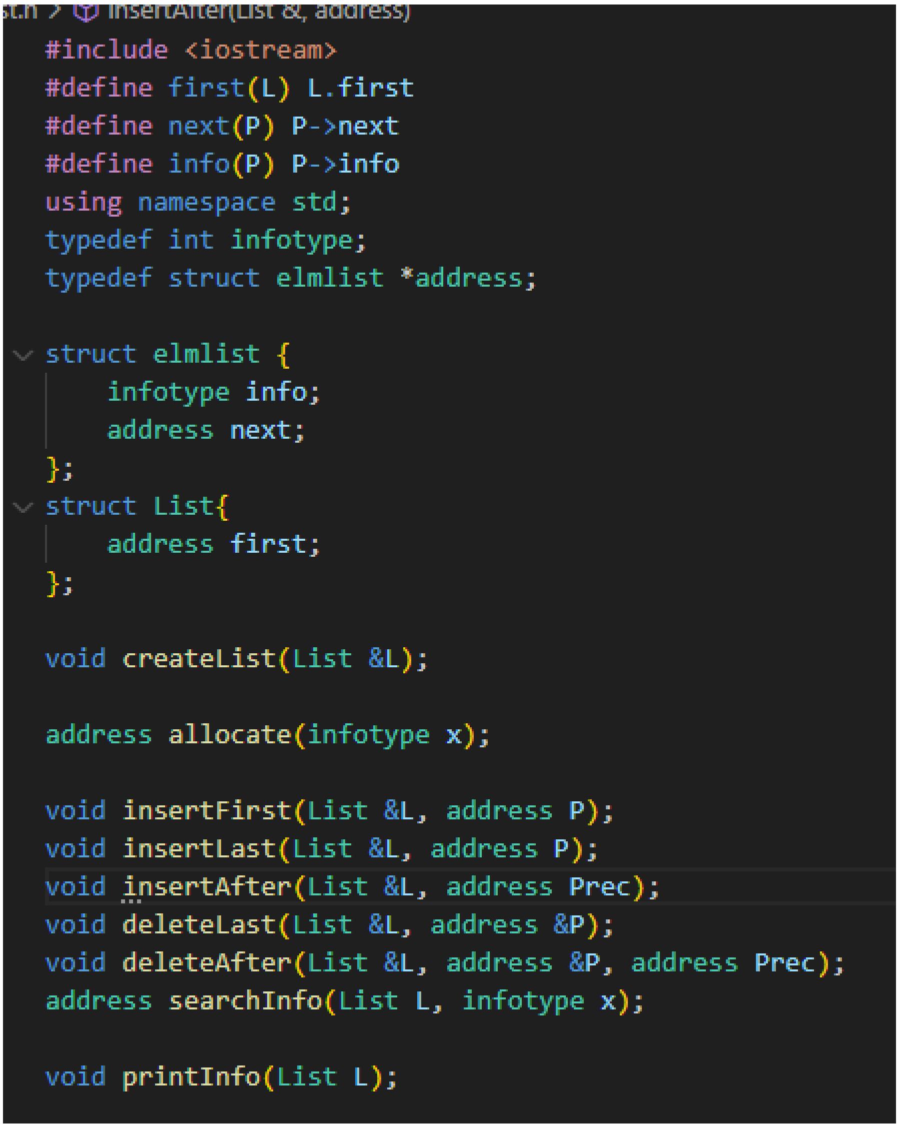
7. SESI HAVE FUN

   a. Tambahkan procedure insertLast, insertAfter, deleteLast, deleteAfter pada list.h dan list.cpp

   b. Tambahkan Function searchInfo pada list.h dan list.cpp

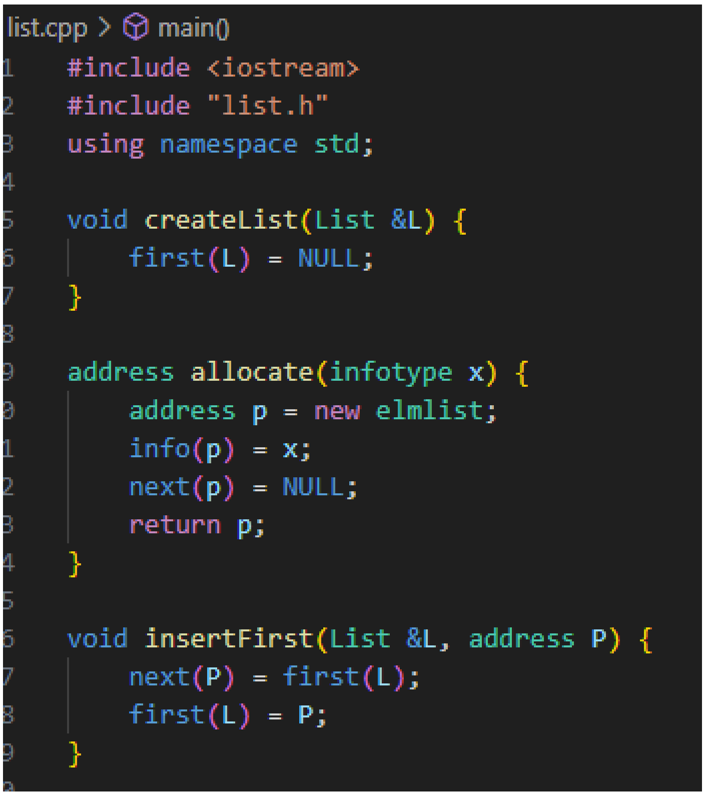
   c. Ubah main.cpp agar proses insert N data tidak satu persatu, tapi sesuai dengan jumlah digit NIM yaitu 10 data (clue : gunakan looping). Dan NIM yang diinput, saat di show tidak boleh terurut terbalik (clue : gunakan insert Last) Tampilan (underscore adalah inputan user):

List.h

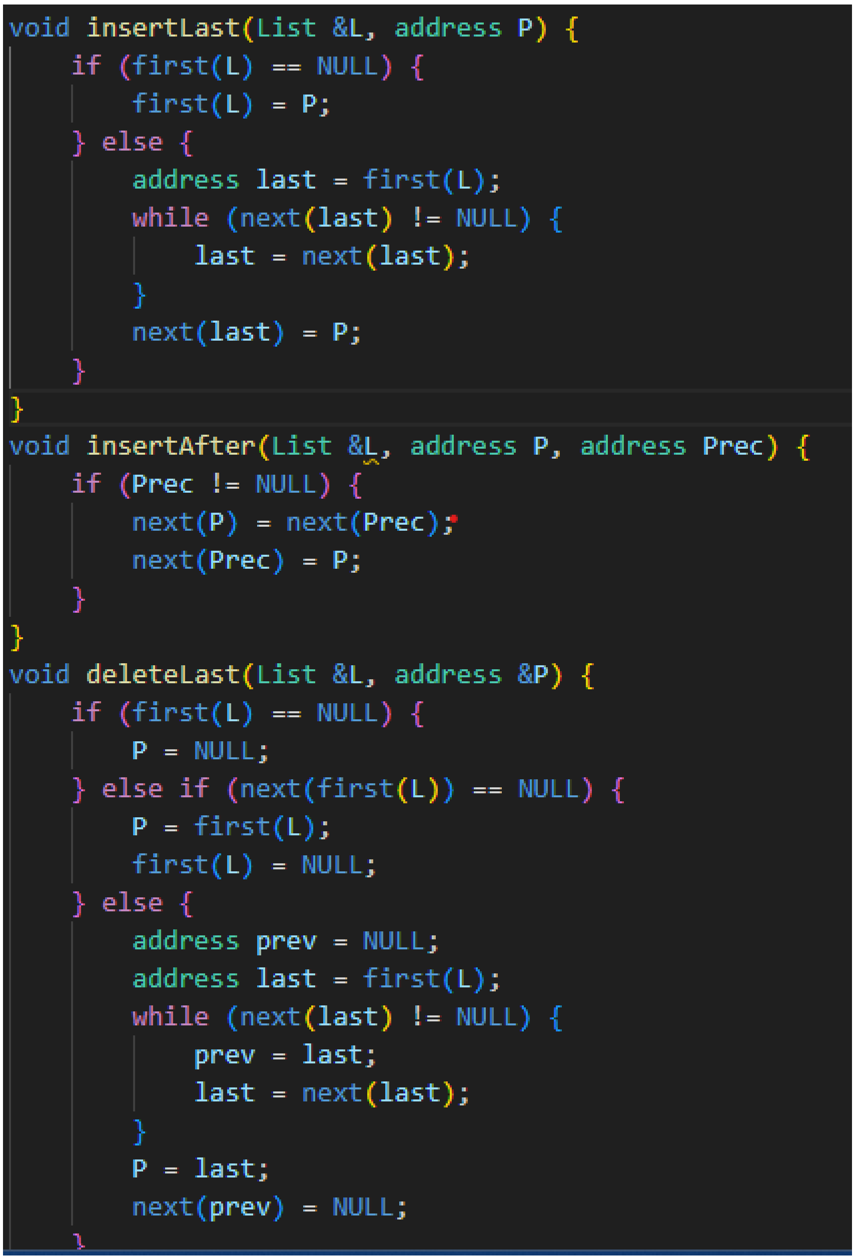


```cpp
#include <iostream>
#define first(L) L.first
#define next(P) P->next
#define info(P) P->info
using namespace std;
typedef int infotype;
typedef struct elmlist *address;

struct elmlist {
    infotype info;
    address next;
};
struct List{
    address first;
};

void createList(List &L);

address allocate(infotype x);

void insertFirst(List &L, address P);
void insertLast(List &L, address P);
void insertAfter(List &L, address Prec);
void deleteLast(List &L, address &P);
void deleteAfter(List &L, address &P, address Prec);
address searchInfo(List L, infotype x);

void printInfo(List L);
```

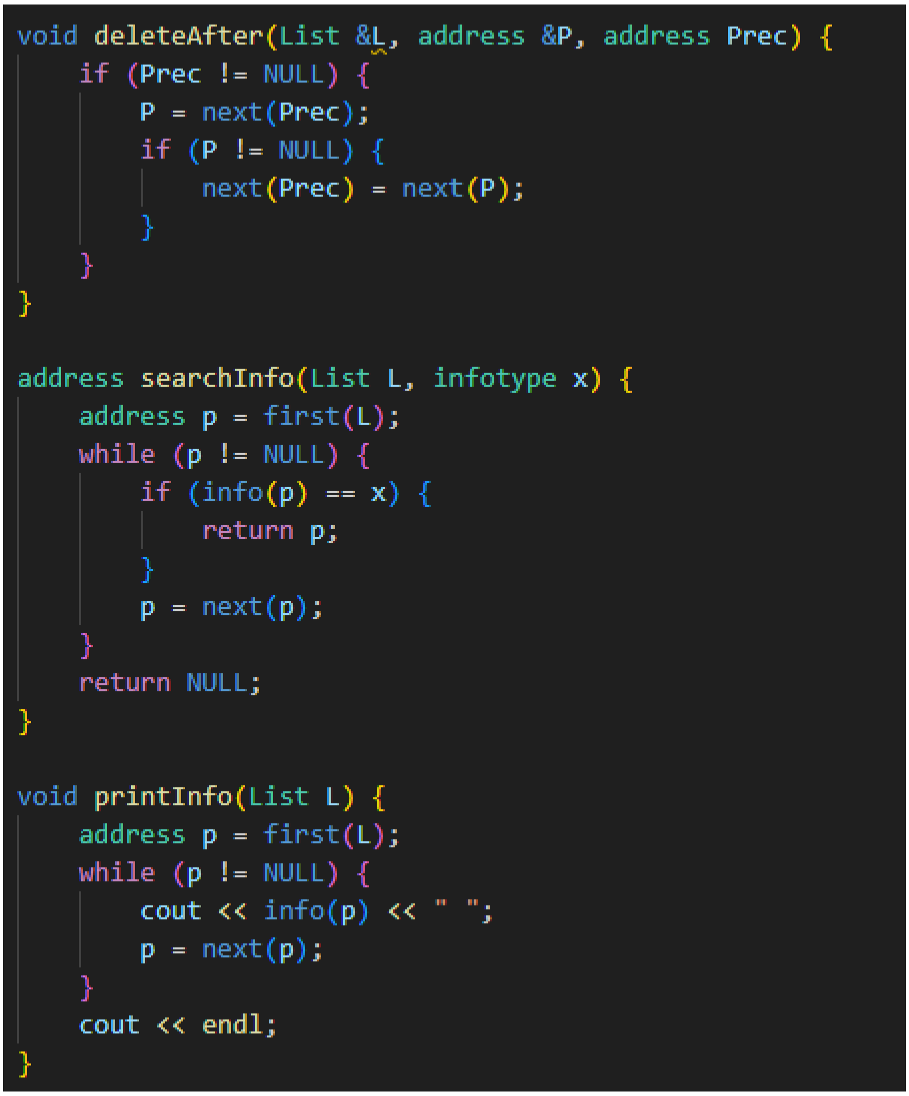
List.cpp



```cpp
#include <iostream>
#include "list.h"
using namespace std;

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

address allocate(infotype x) {
    address p = new elmlist;
    info(p) = x;
    next(p) = NULL;
    return p;
}

void insertFirst(List &L, address P) {
    next(P) = first(L);
    first(L) = P;
}
```

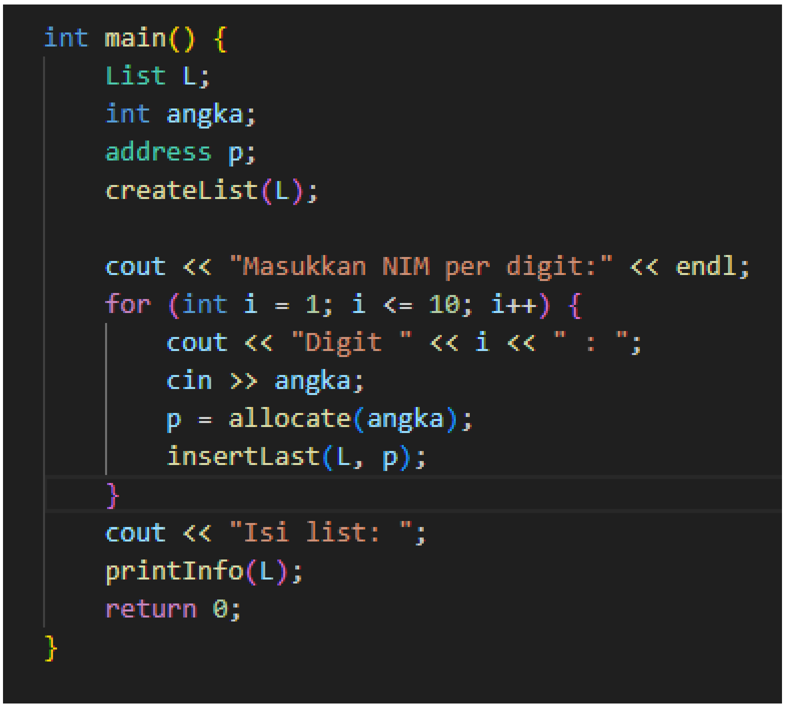


```cpp
void insertLast(List &L, address P) {
    if (first(L) == NULL) {
        first(L) = P;
    } else {
        address last = first(L);
        while (next(last) != NULL) {
            last = next(last);
        }
        next(last) = P;
    }
}
void insertAfter(List &L, address P, address Prec) {
    if (Prec != NULL) {
        next(P) = next(Prec);
        next(Prec) = P;
    }
}
void deleteLast(List &L, address &P) {
    if (first(L) == NULL) {
        P = NULL;
    } else if (next(first(L)) == NULL) {
        P = first(L);
        first(L) = NULL;
    } else {
        address prev = NULL;
        address last = first(L);
        while (next(last) != NULL) {
            prev = last;
            last = next(last);
        }
        P = last;
        next(prev) = NULL;
    }
```



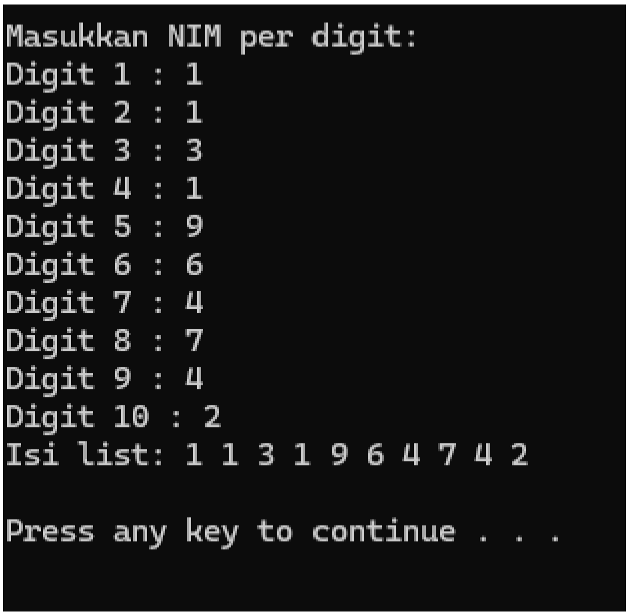
```cpp
void deleteAfter(List &L, address &P, address Prec) {
    if (Prec != NULL) {
        P = next(Prec);
        if (P != NULL) {
            next(Prec) = next(P);
        }
    }
}

address searchInfo(List L, infotype x) {
    address p = first(L);
    while (p != NULL) {
        if (info(p) == x) {
            return p;
        }
        p = next(p);
    }
    return NULL;
}

void printInfo(List L) {
    address p = first(L);
    while (p != NULL) {
        cout << info(p) << " ";
        p = next(p);
    }
    cout << endl;
}
```

Main.cpp



```cpp
int main() {
    List L;
    int angka;
    address p;
    createList(L);

    cout << "Masukkan NIM per digit:" << endl;
    for (int i = 1; i <= 10; i++) {
        cout << "Digit " << i << " : ";
        cin >> angka;
        p = allocate(angka);
        insertLast(L, p);
    }
    cout << "Isi list: ";
    printInfo(L);
    return 0;
}
```

Output:



```
Masukkan NIM per digit:
Digit 1 : 1
Digit 2 : 1
Digit 3 : 3
Digit 4 : 1
Digit 5 : 9
Digit 6 : 6
Digit 7 : 4
Digit 8 : 7
Digit 9 : 4
Digit 10 : 2
Isi list: 1 1 3 1 9 6 4 7 4 2

Press any key to continue . . .
```