

LAPORAN PRAKTIKUM
Modul 5
Single Linked List Bagian 2



Disusun Oleh :
Arzario Irsyad Al Fatih/2211104032
SE 06 2

Asisten Praktikum :
Aldi Putra
Andini Nur Hidayah

Dosen Pengampu :
Wahyu Andi Saputra

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2024

1. Tujuan

- Memahami penggunaan linked list dengan pointer operator dalam program.
- Memahami operasi dasar dalam linked list.
- Membuat program dengan menggunakan linked list dengan prorotype yang ada.

2. Landasan Teori

- Searching

Searching merupakan operasi dasar list dengan melakukan aktivitas pencarian terhadap node tertentu. Proses ini berjalan dengan mengunjungi setiap node dan berhenti setelah node yang dicari ketemu. Dengan melakukan operasi searching, operasi-operasi seperti insert after, delete after, dan update akan lebih mudah.

3. Guided

- Searching

Output

```
PS C:\Users\toshiba\Documents\Tugas Kuliah\Semester 5\Praktikum\STD_Arzario_Irsyad_Al_Fatih_2211104032\05_Single
ster 5\Praktikum\STD_Arzario_Irsyad_Al_Fatih_2211104032\05_Single_Linked_List_Bagian_2\GUIDED\> if ($?) { g++
nnerFile }
Elemen dalam linked list: 7 5 3 11 14 18
Masukan elemen yang ingin dicari: 11
Elemen ditemukan dalam index 3
Masukan elemen yang ingin dihapus: 7
Elemen dalam linked list setelah penghapusan: 5 3 11 14 18
```

Source Code

```

#include <iostream>
using namespace std;

// Struktur untuk node dalam linked list
struct Node {
    int data;
    Node* next;
};

// Fungsi untuk menambahkan elemen baru ke awal linked list
void insertFirst(Node*& head, Node*& tail, int new_data) {
    Node* new_node = new Node();
    new_node->data = new_data;
    new_node->next = head;
    head = new_node;

    if (tail == nullptr) {
        tail = new_node;
    }
}

// Fungsi untuk menambahkan elemen baru ke akhir linked
void insertLast(Node*& head, Node*& tail, int new_data) {
    Node* new_node = new Node();
    new_node->data = new_data;
    new_node->next = nullptr;

    if (head == nullptr) {
        head = new_node;
        tail = new_node;
    } else
        tail->next = new_node;
    tail = new_node;
}

// Fungsi untuk mencari elemen dalam linked list
int findElement(Node* head, int x){
    Node* current = head;
    int index = 0;

    while (current != nullptr){
        if(current->data == x) {
            return index;
        }
        current = current->next;
        index++;
    }
    return -1;
}
```

4. Unguided

a. Soal 1

Source Code

singlelist.h

```



#ifndef SINGLELIST_H
#define SINGLELIST_H

#include <iostream>

using namespace std;

typedef int infotype;
typedef struct elmlist *address;

struct elmlist {
    infotype info;
    address next;
};

struct List {
    address first;
};

// Prosedur pembuatan list kosong
void createList(List &L);

// Fungsi alokasi elemen baru
address alokasi(infotype x);

// Prosedur dealokasi elemen
void dealokasi(address P);

// Prosedur untuk menampilkan info semua elemen dalam
void printInfo(List L);

// Prosedur untuk menambahkan elemen di awal list
void insertFirst(List &L, address P);

// Fungsi untuk mencari elemen dengan info tertentu
address findElm(List L, infotype x);

// Fungsi untuk menghitung total info semua elemen
int totalInfo(List L);

#endif
```

singlelist.cpp

```
#include "singlelist.h"

// Prosedur pembuatan list kosong
void createList(List &L) {
    L.first = NULL;
}

// Fungsi alokasi elemen baru
address alokasi(infotype x) {
    address P = new elm1ist;
    P->info = x;
    P->next = NULL;
    return P;
}

// Prosedur dealokasi elemen
void dealokasi(address P) {
    delete P;
}

// Prosedur untuk menampilkan info semua elemen dalam
void printInfo(List L) {
    address P = L.first;
    while (P != NULL) {
        cout << P->info << " ";
        P = P->next;
    }
    cout << endl;
}

// Prosedur untuk menambahkan elemen di awal list
void insertFirst(List &L, address P) {
    P->next = L.first;
    L.first = P;
}

// Fungsi untuk mencari elemen dengan info tertentu
address findElm(List L, infotype x) {
    address P = L.first;
    while (P != NULL) {
        if (P->info == x)
            return P;
        P = P->next;
    }
    return NULL; // Jika tidak ditemukan
}

// Fungsi untuk menghitung total info semua elemen
int totalInfo(List L) {
    address P = L.first;
    int total = 0;
    while (P != NULL) {
        total += P->info;
        P = P->next;
    }
    return total;
}
```

main.cpp

```
#include "singlelist.h"

// Prosedur pembuatan list kosong
void createList(List &L) {
    L.first = NULL;
}

// Fungsi alokasi elemen baru
address alokasi(infotype x) {
    address P = new elm1ist;
    P->info = x;
    P->next = NULL;
    return P;
}

// Prosedur dealokasi elemen
void dealokasi(address P) {
    delete P;
}

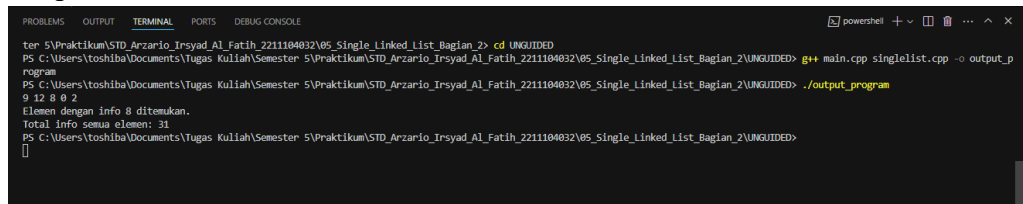
// Prosedur untuk menampilkan info semua elemen dalam
void printInfo(List L) {
    address P = L.first;
    while (P != NULL) {
        cout << P->info << " ";
        P = P->next;
    }
    cout << endl;
}

// Prosedur untuk menambahkan elemen di awal list
void insertFirst(List &L, address P) {
    P->next = L.first;
    L.first = P;
}

// Fungsi untuk mencari elemen dengan info tertentu
address findElm(List L, infotype x) {
    address P = L.first;
    while (P != NULL) {
        if (P->info == x)
            return P;
        P = P->next;
    }
    return NULL; // Jika tidak ditemukan
}

// Fungsi untuk menghitung total info semua elemen
int totalInfo(List L) {
    address P = L.first;
    int total = 0;
    while (P != NULL) {
        total += P->info;
        P = P->next;
    }
    return total;
}
```

Output



```
PS C:\Users\toshiba\Documents\Tugas Kuliah\Semester 5\Praktikum\STD_Arzario_Irsyad_Al_Fatih_2211104032\05_Single_Linked_List_Bagian_2\UNGUIDED> cd UNGUIDED
PS C:\Users\toshiba\Documents\Tugas Kuliah\Semester 5\Praktikum\STD_Arzario_Irsyad_Al_Fatih_2211104032\05_Single_Linked_List_Bagian_2\UNGUIDED> g++ main.cpp singlelist.cpp -o output_p
rogram
PS C:\Users\toshiba\Documents\Tugas Kuliah\Semester 5\Praktikum\STD_Arzario_Irsyad_Al_Fatih_2211104032\05_Single_Linked_List_Bagian_2\UNGUIDED> ./output_program
9 12 8 0 2
Elemen dengan info 8 ditemukan.
Total info semua elemen: 31
PS C:\Users\toshiba\Documents\Tugas Kuliah\Semester 5\Praktikum\STD_Arzario_Irsyad_Al_Fatih_2211104032\05_Single_Linked_List_Bagian_2\UNGUIDED>
```

Deskripsi

Kode diatas adalah implementasi dari struktur data Single Linked List dalam C++. Struktur data ini terdiri dari elemen-elemen yang saling terhubung secara berantai, di mana setiap elemen menyimpan sebuah nilai integer (`info`) dan pointer (`next`) ke elemen berikutnya.

- `createList`: Menginisialisasi list kosong dengan menetapkan pointer `first` ke `NULL`.
- `alokasi`: Membuat elemen baru dengan nilai `info` tertentu dan mengembalikan alamat dari elemen tersebut.
- `dealokasi`: Menghapus elemen yang telah dialokasikan dari memori.
- `printInfo`: Menampilkan semua elemen dalam list dengan mencetak nilai `info` tiap elemen.
`insertFirst`: Menambahkan elemen baru di awal list.
- `findElm`: Mencari elemen dengan nilai `info` tertentu dalam list dan mengembalikan alamatnya jika ditemukan.
- `totalInfo`: Menghitung total dari semua nilai `info` elemen dalam list.

Di bagian `main`:

- Dibuat list kosong dan diisi dengan beberapa elemen menggunakan fungsi `alokasi` dan `insertFirst`.
- List ditampilkan dengan `printInfo`.
- Dilakukan pencarian elemen dengan nilai `info` 8 menggunakan `findElm`.
- Total semua nilai `info` elemen dihitung dengan `totalInfo`.