LAPORAN PRAKTIKUM 05

Single Linked List 2

Semester 2 Tahun Akademik 2022/2023

DOSEN PENGAMPU:

Entin Martiana Kusumaningtyas, S.kom, M.kom

PENYUSUN:

Arsyita Devanaya Arianto (3122500008)

PROGRAM STUDI VOKASI

D-III TEKNIK INFORMATIKA

DEPARTEMEN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

PERCOBAAN DAN LATIHAN 1

Soal

1. Implementasikan operasi dasar Single linked list : Menghapus simpul tertentu. Tambahkan kondisi jika yang dihapus adalah data yang paling depan atau data yang paling terakhir.
2. Implementasikan operasi dasar Single linked list : Menyisipkan setelah simpul tertentu. Tambahkan kondisi jika data yang disisipkan setelahnya adalah data terakhir.
3. Implementasikan operasi dasar Single linked list : Menyisipkan sebelum simpul tertentu. Tambahkan kondisi jika data yang disisipkan setelahnya adalah data terakhir.
4. Gabungkan semua operasi di atas dalam sebuah Menu Pilihan.

Listing Program

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

//Membuat struktur simpul
struct simpul{
    char nama[25];
    int nrp;
    struct simpul *next;
};
//Mendeklarasikan variabel struktur
struct simpul *head=NULL, *tail=NULL;
struct simpul *baru, *tampil;

//Untuk mengalokasikan simpul
struct simpul *alokasi_simpul(){
    struct simpul *x;
    x=(struct simpul *)malloc(sizeof(struct simpul));
    if(x==NULL){
        return NULL;
    }else{
        x->next=NULL;
        return x;
    }
};

//Untuk menambahkan simpul awal
void sisip_awal(){

    baru=alokasi_simpul();
    if(baru==NULL){
        printf("Alokasi Gagal");
    }else{
        printf("Nama\t: ");scanf("%s", &baru->nama);
        printf("NRP\t: ");scanf("%d", &baru->nrp);
        if(head==NULL){
            head=baru;
            tail=baru;
        }else{
            baru->next=head;
            head=baru;
        }
    }
}
```

```c
//Untuk menambahkan simpul di akhir
void sisip_akhir(){
    baru=alokasi_simpul();

    if(baru==NULL){
        printf("Alokasi Gagal");
    }else{
        printf("Nama\t: ");scanf("%s", &baru->nama);
        printf("NRP\t: ");scanf("%d", &baru->nrp);
        if(head==NULL){
            head=baru;
            tail=baru;
        }else{
            baru->next=NULL;
            tail->next=baru;
            tail=baru;
        }
    }
}

//Untuk menampilkan simpul
void Tampil(){
    tampil=head;
    while(tampil!=NULL){
        printf("Nama\t: %s\n",tampil->nama);
        printf("NRP\t: %d", tampil->nrp);
        tampil=tampil->next;
        printf("\n\n");
    }
}

//Untuk menghapus simpul
void Hapus_simpul(){
    struct simpul *hapus;
    struct simpul *sbl;
    int nrp;
    hapus=head;

    printf("Simpul yang ingin dihapus (NRP): ");
    scanf("%d", &nrp);

    if(head->nrp==nrp){
        hapus=head;
        head=hapus->next;
        hapus->next=NULL;
        free(hapus);
    }else if(tail->nrp==nrp){
        sbl=tail;
        while(hapus->nrp!=nrp){
            sbl=hapus;
            hapus=hapus->next;
        }
            sbl->next=NULL;
            free(hapus);

            tree(hapus);
    }else{
        while(hapus->nrp!=nrp){
            sbl=hapus;
            hapus=hapus->next;
        }
        sbl->next=hapus->next;
        free(hapus);
    }
}
```

```c
//Untuk menyisipkan simpul setelah simpul
void menyisipkan_setelah(){
    int NRP;
    struct simpul*cari;
    printf("Menyisipkan Simpul Setelah (NRP): ");
    scanf("%d", &NRP);

    cari=head;

    while(cari!=NULL && cari->nrp != NRP){
        cari=cari->next;
    }
    if(cari==NULL){
        printf("Simpul tidak ditemukan\n");
    }else{
        if(cari==tail){
            sisip_akhir();
        }else{
            baru=alokasi_simpul();
            printf("Nama\t: ");scanf("%s", &baru->nama);
            printf("NRP\t: ");scanf("%d", &baru->nrp);

            baru->next=cari->next;
            cari->next=baru;
        }
    }
}

//Untuk menyisipkan simpul sebelum simpul
void menyisipkan_sebelum(){
    int NRP;
    struct simpul*cari, *stl;
    printf("Menyisipkan Simpul Sebelum (NRP): ");
    scanf("%d", &NRP);

    cari=head;

    while(cari!=NULL && cari->nrp != NRP){
        stl=cari;
        cari=cari->next;
    }
    if(cari==NULL){
        printf("Simpul tidak ditemukan");
    }else{
        if(cari==head){
            sisip_awal();
        }else{
            baru=alokasi_simpul();
            printf("Nama\t: ");scanf("%s", &baru->nama);
            printf("NRP\t: ");scanf("%d", &baru->nrp);

            baru->next=cari;
            stl->next=baru;
        }
    }
}
```

```c
void main(){

    int pilihan;
    char jawab;
    char simpul;
    printf("Pilih Simpul (L/F): ");
    scanf("%s",&simpul);
    printf("\n");
    switch(simpul){
    case 'L':
        printf("====== 3 Simpul Pertama LIFO ======\n\n");
        sisip_awal();printf("\n");
        sisip_awal();printf("\n");
        sisip_awal();printf("\n");
        printf("(1)Sisip Awal\n");
        printf("(2)Sisip Akhir\n");
        printf("(3)Menghapus Simpul Tertentu\n");
        printf("(4)Menyisipkan Setelah Simpul\n");
        printf("(5)Menyisipkan Simpul Sebelum\n");
        printf("(6)Menampilkan \n\n");

        do{
            printf("Masukkan Pilihan Anda\t\t: ");
            scanf("%d", &pilihan);
            switch(pilihan){
                case 1:
                    printf("\n====== Sisip Awal ======\n");
                    sisip_awal(); break;
                case 2:
                    printf("\n====== Sisip Akhir ======\n");
                    sisip_akhir(); break;
                case 3:
                    printf("\n====== Hapus Simpul Tertentu ======\n");
                    Hapus_simpul();break;
                case 4:
                    printf("\n====== Menyisipkan Setelah ======\n");
                    menyisipkan_setelah(); break;
                case 5:
                    printf("\n====== Menyisipkan Sebelum ======\n");
                    menyisipkan_sebelum(); break;
                case 6:
                    printf("\n====== Menampilkan (LIFO) ======\n");
                    Tampil();break;
            }
            printf("\n(y/n)==> ");
            scanf("%s", &jawab);
            printf("\n");
        }while(jawab=='y'||jawab=='Y');
        break;

    case 'F':
        printf("====== 3 Simpul Pertama FIFO ======\n\n");
        sisip_akhir();printf("\n");
        sisip_akhir();printf("\n");
        sisip_akhir();printf("\n");
        printf("(1)Sisip Awal\n");
        printf("(2)Sisip Akhir\n");
        printf("(3)Menghapus Simpul Tertentu\n");
        printf("(4)Menyisipkan Setelah Simpul\n");
        printf("(5)Menyisipkan Simpul Sebelum\n");
        printf("(6)Menampilkan \n\n");
```

```
        do{
            printf("Masukkan Pilihan Anda\t\t: ");
            scanf("%d", &pilihan);
            switch(pilihan){
                case 1:
                    printf("\n====== Sisip Awal ======\n");
                    sisip_awal(); break;
                case 2:
                    printf("\n====== Sisip Akhir ======\n");
                    sisip_akhir(); break;
                case 3:
                    printf("\n====== Hapus Simpul Tertentu ======\n");
                    Hapus_simpul();break;
                case 4:
                    printf("\n====== Menyisipkan Setelah ======\n");
                    menyisipkan_setelah(); break;
                case 5:
                    printf("\n====== Menyisipkan Sebelum ======\n");
                    menyisipkan_sebelum(); break;
                case 6:
                    printf("\n====== Menampilkan (FIFO) ======\n");
                    Tampil();break;
            }
            printf("\n(y/n)==> ");
            scanf("%s", &jawab);
            printf("\n");
        }while(jawab=='y'||jawab=='Y');
        break;
    }
}
```

Output

FIFO :

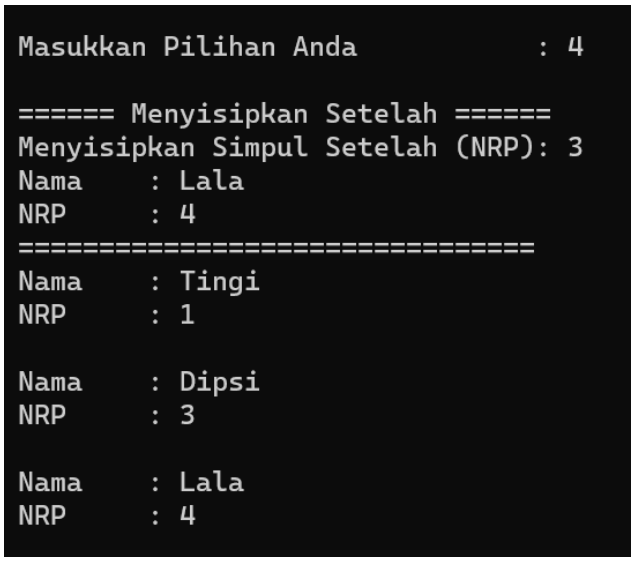
```
Pilih Simpul (L/F): L

====== 3 Simpul Pertama LIFO ======

Nama    : Tingki
NRP     : 1

Nama    : Wingki
NRP     : 2

Nama    : Dipsi
NRP     : 3

(1)Sisip Awal
(2)Sisip Akhir
(3)Menghapus Simpul Tertentu
(4)Menyisipkan Setelah Simpul
(5)Menyisipkan Simpul Sebelum
(6)Menampilkan

Masukkan Pilihan Anda           : 3

====== Hapus Simpul Tertentu ======
Simpul yang ingin dihapus (NRP): 2
Nama    : Dipsi
NRP     : 3

Nama    : Tingki
NRP     : 1
```

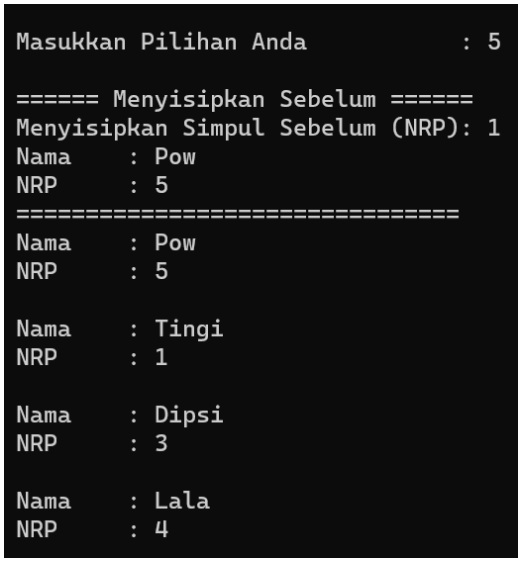
```
Masukkan Pilihan Anda           : 4

====== Menyisipkan Setelah ======
Menyisipkan Simpul Setelah (NRP): 3
Nama    : Lala
NRP     : 4
================================
Nama    : Dipsi
NRP     : 3

Nama    : Lala
NRP     : 4

Nama    : Tingki
NRP     : 1
```

```
Masukkan Pilihan Anda           : 5

====== Menyisipkan Sebelum ======
Menyisipkan Simpul Sebelum (NRP): 1
Nama    : Pow
NRP     : 5
================================
Nama    : Dipsi
NRP     : 3

Nama    : Lala
NRP     : 4

Nama    : Pow
NRP     : 5

Nama    : Tingki
NRP     : 1
```

FIFO:

```
Pilih Simpul (L/F): F

====== 3 Simpul Pertama FIFO ======

Nama    : Tingi
NRP     : 1

Nama    : Wingki
NRP     : 2

Nama    : Dipsi
NRP     : 3

(1)Sisip Awal
(2)Sisip Akhir
(3)Menghapus Simpul Tertentu
(4)Menyisipkan Setelah Simpul
(5)Menyisipkan Simpul Sebelum

(6)Menampilkan

Masukkan Pilihan Anda           : 3

====== Hapus Simpul Tertentu ======
Simpul yang ingin dihapus (NRP): 2
Nama    : Tingi
NRP     : 1

Nama    : Dipsi
NRP     : 3
```





Analisa

Program ini adalah program dengan menggunakan struktur data linked list yang menyimpan variabel nama dengan penyimpanan maksimal 25 karakter, variabel nrp, dan variabel pointer struct simpul *next yang digunakan untuk menunjuk simpul selanjutnya.

Fungsi-fungsi dalam program di atas:

1. alokasi_simpul()
   Fungsi ini digunakan untuk mengalokasikan simpul linked list. Fungsi ini mengembalikan pointer ke simpul baru yang dialokasikan jika alokasi berhasik, atau mengembalikan nilai NULL jika alokasi gagal.
2. sisip_awal()
   Fungsi ini digunakan untuk menambahkan sisip simpul baru di awal linked list. Simpul baru dapat diinputkan oleh user.
3. sisip_akhir()
   fungsi ini digunakan untuk menambahkan sisip simpul baru di akhir linked list. Simpul baru dapat diinputkan oleh user.
4. Tampil()
   Fungsi ini digunakan untuk menampilkan semua simpul linked list. Fungsi ini mengeksekusi perintah looping while setiap simpul dengan isi nama dan nrp.
5. menyisipkan_setelah()
   Fungsi ini untuk menyisipkan simpul baru di tengah linked list, awal, atau akhir. Yang diutamakan adalah untuk menyisipkan di tengah setelah simpul akan disispkan simpul baru dan jika sisip baru berada di awal atau di akhir akan ada pengondisian.
6. menyisipkan_sebelum()
   Fungsi ini untuk menyisipkan simpul baru di tengah linked list, awal, atau akhir. Yang diutamakan adalah untuk menyisipkan di tengah sebelum simpul akan disispkan simpul baru dan jika sisip baru berada di awal atau di akhir akan ada pengondisian.

LATIHAN 2

Soal

Merepresentasikan sebuah bilangan polinomial dengan single linked list Masalah aritmatika polinom adalah membuat sekumpulan subrutin manipulasi terhadap polinom simbolis (symbolic Polynomial). Misalnya:

P1 = 6x8 + 8x7 + 5x5 + x3 + 15

P2 = 3x9 + 4x7 + 3x4 + 2x3 + 2x2 + 10

Representasikan bilangan polinom dengan menggunakan linked list dan buatlah prosedur-prosedur untuk :

• Menyisipkan simpul di awal jika pangkat yang dimasukkan lebih dari pangkat tertinggi dari bilangan polinomial.

• Menyisipkan simpul di tengah jika pangkat dari bilangan yang kita sisipkan berada di tengah.

• Menyisipkan simpul di akhir jika pangkat dari bilangan yang disisipkan adalah 0.

• Menghapus simpul, baik di awal, di tengah, ataupun di akhir.

Listing Program

```
#include <stdio.h>
#include <stdlib.h>

//Membuat struktur sipul
struct simpul{
    float koef;
    int pangkat;
    struct simpul *next;
};
//Deklarasi variabel struktur simpul
struct simpul *head, *tail;

//Mengalokasikan simpul baru
struct simpul *alokasi(){
    struct simpul *x;
    x=(struct simpul*)malloc(sizeof(struct simpul));
    if(x==NULL){
        return NULL;
    }else{
        x->next=NULL;
        return x;
    }
};
```

```
//Menyisipkan sipmpul baru di awal
void Sisipan_Awal(int x, int y){
    struct simpul *baru;
    baru=alokasi();
    baru->koef=x;
    baru->pangkat=y;
    if(head==NULL){
        head=baru;
        tail=baru;
    }else{
        baru->next=head;
        head=baru;
    }

}
```

```
//Menampilkan simpul
void Tampil(){
    struct simpul *tampil;
    tampil=head;
    while(tampil!=NULL){
        printf("%0.fx^%d", tampil->koef,tampil->pangkat);
        if(tampil->next!=NULL){
            printf(" + ");
        }
        tampil=tampil->next;
    }
     printf("\n");
}

//Menghapus simpul
void Hapus(int h){
    struct simpul *hapus, *sbl;
    hapus=head;
    if(head->koef==h){
        head=hapus->next;
        hapus->next=NULL;
        free(hapus);
    }else if(tail->koef==h){
        while(hapus->koef!=h){
        sbl=hapus;
        hapus=hapus->next;
        }
        sbl->next=NULL;
        free(hapus);
    }else{
        while(hapus->koef!=h){
            sbl=hapus;
            hapus=hapus->next;
        }
        sbl->next=hapus->next;
        free(hapus);
    }
}

//Untuk mengurutkan bilangan polinomial
void Sisip_Urut(int x, int y){

    struct simpul *baru,  *setelah, *sebelum;
    sebelum=head;
    baru=alokasi();
    baru->koef=x;
    baru->pangkat=y;


    if(head==NULL|| baru->pangkat > head->pangkat){
        Sisipan_Awal(baru->koef,baru->pangkat);
    }else{

        while(sebelum!=NULL && sebelum->pangkat >= baru->pangkat){
            if(sebelum->pangkat==baru->pangkat){
                sebelum->koef += baru->koef;
                return;
            }
            setelah=sebelum;
            sebelum=sebelum->next;
        }
        baru->next=sebelum;
        setelah->next=baru;
    }
}
```

```
void main(){
    int Pangkat, Pilihan, hps;
    float Koefisien;
    char jwb;
    printf("====== Pilih Perintah Berikut ======\n");
    printf("(1) Menambahkan Bilangan Polinom\n(2) Menghapus Bilangan Polinom\n");
    printf("(Tekan y untuk melanjutkan pemilihan)\n\n");
    do{
        printf("Masukkan Pilihan Anda\t: ");
        scanf("%d", &Pilihan);
        switch(Pilihan){
        case 1:
            printf("====== Menambahkan Bilangan ======\n");
            printf("Masukkan Bilangan Koefisien\t: ");
            scanf("%f", &Koefisien);
            printf("Masukkan Bilangan Pangkat\t: ");
            scanf("%d", &Pangkat);
            Sisip_Urut(Koefisien, Pangkat);
            Tampil();
            break;
        case 2:
            printf("====== Menghapus Bilangan ======\n");
            printf("Masukkan Koefisien yang Ingin Dihapus\t");
            scanf("%d", &hps);
            Hapus(hps);
            Tampil();
            break;
        default:
            printf("Anda Salah Menginputkan Pilihan\n");
            break;
        }
        printf("\n(y/n)==> ");
        scanf("%s", &jwb);
        printf("\n");
    }while(jwb=='y');

}
```

Output

```
Masukkan Pilihan Anda   : 1
====== Menambahkan Bilangan ======
Masukkan Bilangan Koefisien     : 1
Masukkan Bilangan Pangkat       : 3
6x^8 + 8x^7 + 5x^5 + 1x^3

(y/n)==> y

Masukkan Pilihan Anda   : 1
====== Menambahkan Bilangan ======
Masukkan Bilangan Koefisien     : 5
Masukkan Bilangan Pangkat       : 1
6x^8 + 8x^7 + 5x^5 + 1x^3 + 5x^1

(y/n)==> y
```

```
Masukkan Pilihan Anda   : 2
====== Menghapus Bilangan ======
Masukkan Koefisien yang Ingin Dihapus   8
6x^8 + 5x^5 + 1x^3 + 5x^1

(y/n)==> y

Masukkan Pilihan Anda   : 2
====== Menghapus Bilangan ======
Masukkan Koefisien yang Ingin Dihapus   1
6x^8 + 5x^5 + 5x^1

(y/n)==> n
```

Analisa

Program ini merupakan program untuk mengolah bilangan polynomial dengan menggunakan single linked list. Program ini menggunakan struktur yang menyimpan nilai koefisien bertipe float, pangkat dengan tipe integer, dan pointer next dengan tipe struktur simpul.

Fungsi-fungsi dari program ini:

1. alokasi()
   Fungsi ini digunakan untuk mengalokasikan simpul linked list. Fungsi ini mengembalikan pointer ke simpul baru yang dialokasikan jika alokasi berhasik, atau mengembalikan nilai NULL jika alokasi gagal.
2. Sisipan_Awal()
   Fungsi ini digunakan untuk menyisipkan simpul baru di awal linked list.
3. Tampil()
   Fungsi ini digunakan untuk menampilkan semua simpul linked list. Fungsi ini mengeksekusi perintah looping while setiap simpul.
4. Hapus()
   Fungsi ini digunakan untuk menghapus simul dalam linked list saat fungsinya dipanggil.
5. Sisip_Urut()
   Fungsi ini digunakan untuk mengurutkan bilangan polynomial. Dalam fungsi ini menggunakan pengondisian untuk pangkat tertinggi akan berada di depan dan urutannya berdasarkan dengan pangkat terbesar hingga kecil.

KESIMPULAN

Dalam praktikum ini kita mempelajari tentang single linked list untuk menghapus, menyisipkan setelah, dan menyisipkan sebelum. Single linked list menyerupai array tetapi sangat berbeda. Single linked list jika diibaratkan seperti sekumpulan kotak yang saling berhubungan. Agar memahami alur kerja dari single linked list harus mengetahui alokasi awal simpul baru dan mendefinisikannya diawal agar tidak kebingungan dalam proses selanjutnya.