Laporan Resmi
Praktikum Algoritma dan Struktur Data
Single Linked List Delete (Data Struct)

Dr. Tita Karlita S.Kom, M.Kom

Nama : Marits Ikmal Yasin

Kelas : 1D4 IT B

NRP : 3121600047

1. Delete Awal

Kode :

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

typedef struct{
    int id;
    char nama[25];
    float nilai;
}mahasiswa;

typedef struct simpul node;
struct simpul{
    mahasiswa student;
    node *next;
};
 node *head=NULL, *p, *tail;

void input(mahasiswa *);
void tampil();
void akhir();
void alokasi();
void menghapus_awal();

int main(){
    char jwb;

    puts("Linked List untuk aplikasi INSERT DI AKHIR");
    puts("Membentuk linked list dengan Insert di akhir");
    do{
        fflush(stdin);
        alokasi();
        akhir();
        fflush(stdin);
        printf("Ada data lagi (y/t)? ");
        jwb=getchar();
        puts("");
    }while(jwb=='y');
    tampil();
    fflush(stdin);
    printf("Mau menghapus node pertama? ");
    jwb = getchar();
    if(jwb == 'y'){
        puts("\nMenghapus simpul pertama dari node...\n");
        menghapus_awal();
    }
    tampil();
    return 0;
}

void menghapus_awal(){
    node *hapus;

    hapus = head;
    head = hapus->next;
    free(hapus);
    hapus == NULL;
}

//Step 1) Memesan Alamat
void alokasi(){
    mahasiswa murid;
```

```
    input(&murid);
    p=(node *)malloc(sizeof(node));
    if(p == NULL)
        exit(0);
    else{
        p->student.id = murid.id;
        strcpy(p->student.nama, murid.nama);
        p->student.nilai = murid.nilai;
        p->next=NULL;
    }
}

void akhir(){
    node *tail;

    if(head == NULL)
        head = p;
    //Step 2) Mencari Posisi
    else{
        tail = head;
        while(tail->next != NULL)
            tail = tail->next;
        //Step 3) Menyambungkan Node
        tail->next = p;
        tail = tail->next;
    }
}

void tampil(){
    node *baca;

    baca = head;
    puts("Data yang telah diinputkan :");
    puts("No\tNama\tNilai");
    while(baca != NULL){
        printf("%d\t",baca->student.id);
        printf("%s\t", baca->student.nama);
        printf("%g\t", baca->student.nilai);
        puts("");
        baca=baca->next;
    }

}

void input(mahasiswa *student){
    int x, y;

    printf("No\t : ");
    scanf("%d",&student->id);
    fflush(stdin);
    printf("Nama\t : ");
    gets(student->nama);
    fflush(stdin);
    printf("Nilai\t : ");
    scanf("%f",&student->nilai);
}
```

Output :

```
Linked List untuk aplikasi INSERT DI AKHIR
Membentuk linked list dengan Insert di akhir
No       : 1
Nama     : Ikmal
Nilai    : 98
Ada data lagi (y/t)? y

No       : 2
Nama     : Ling
Nilai    : 87
Ada data lagi (y/t)? t

Data yang telah diinputkan :
No      Nama    Nilai
1       Ikmal   98
2       Ling    87
Mau menghapus node pertama? y

Menghapus simpul pertama dari node...

Data yang telah diinputkan :
No      Nama    Nilai
2       Ling    87
```

2. Delete Akhir

Kode :

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

typedef struct{
   int id;
   char nama[25];
   float nilai;
}mahasiswa;

typedef struct simpul node;
struct simpul{
   mahasiswa student;
   node *next;
};
 node *head=NULL, *p, *tail;

void input(mahasiswa *);
void tampil();
void akhir();
void alokasi();
void menghapus_akhir();

int main(){
   char jwb;

   puts("Linked List untuk aplikasi INSERT DI AKHIR");
   puts("Membentuk linked list dengan Insert di akhir");
   do{
      fflush(stdin);
      alokasi();
      akhir();
      fflush(stdin);
      printf("Ada data lagi (y/t)? ");
      jwb=getchar();
      puts("");
   }while(jwb=='y');
   tampil();
   fflush(stdin);
   printf("Mau menghapus node terakhir? ");
   jwb = getchar();
   if(jwb == 'y'){
```

```
        puts("\nMenghapus simpul terakhir dari node...\n");
        menghapus_akhir();
    }
    tampil();
    return 0;
}

void menghapus_akhir(){
    node *hapus, *phapus;

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


//Step 1) Memesan Alamat
void alokasi(){
    mahasiswa murid;

    input(&murid);
    p=(node *)malloc(sizeof(node));
    if(p == NULL)
        exit(0);
    else{
        p->student.id = murid.id;
        strcpy(p->student.nama, murid.nama);
        p->student.nilai = murid.nilai;
        p->next=NULL;
    }
}

void akhir(){
    node *tail;

    if(head == NULL)
        head = p;
    //Step 2) Mencari Posisi
    else{
        tail = head;
        while(tail->next != NULL)
            tail = tail->next;
        //Step 3) Menyambungkan Node
        tail->next = p;
        tail = tail->next;
    }
}

void tampil(){
    node *baca;

    baca = head;
    puts("Data yang telah diinputkan :");
    puts("No\tNama\tNilai");
    while(baca != NULL){
```

```
        printf("%d\t",baca->student.id);
        printf("%s\t", baca->student.nama);
        printf("%g\t", baca->student.nilai);
        puts("");
        baca=baca->next;
    }

}

void input(mahasiswa *student){
    int x, y;

    printf("No\t : ");
    scanf("%d",&student->id);
    fflush(stdin);
    printf("Nama\t : ");
    gets(student->nama);
    fflush(stdin);
    printf("Nilai\t : ");
    scanf("%f",&student->nilai);
}
```

Output :

```
Linked List untuk aplikasi INSERT DI AKHIR
Membentuk linked list dengan Insert di akhir
No       : 1
Nama     : Ikmal
Nilai    : 90
Ada data lagi (y/t)? y

No       : 2
Nama     : Ling
Nilai    : 99
Ada data lagi (y/t)? t

Data yang telah diinputkan :
No      Nama    Nilai
1       Ikmal   90
2       Ling    99
Mau menghapus node terakhir? y

Menghapus simpul terakhir dari node...

Data yang telah diinputkan :
No      Nama    Nilai
1       Ikmal   90
```

3. Delete After

Kode :

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

typedef struct{
    int id;
    char nama[25];
    float nilai;
}mahasiswa;

typedef struct simpul node;
struct simpul{
    mahasiswa student;
    node *next;
};
 node *head=NULL, *p, *tail;

void input(mahasiswa *);
void tampil();
void akhir();
void alokasi();
void menghapus_setelah();

int main(){
    char jwb;

    puts("Linked List untuk aplikasi INSERT DI AKHIR");
    puts("Membentuk linked list dengan Insert di akhir");
    do{
        fflush(stdin);
        alokasi();
        akhir();
        fflush(stdin);
        printf("Ada data lagi (y/t)? ");
        jwb=getchar();
        puts("");
    }while(jwb=='y');
    tampil();
    fflush(stdin);
    printf("Mau menghapus node setelah (y/t)? ");
    jwb = getchar();
    if(jwb == 'y')
        menghapus_setelah();
    tampil();
    return 0;
}

void menghapus_setelah(){
    int key;
    node *hapus, *phapus;

    printf("Node yang akan dihapus setelah node berapa? ");
    scanf("%d", &key);
    hapus = head;
    while(hapus->student.id != key){
        if(hapus->next != NULL)
            hapus = hapus->next;
    }
    if(hapus->next == NULL){
        printf("Setelah node %d tidak ada yang bisa dihapus\n", key);
        exit(0);
```

```
    }
    else{
        phapus = hapus;
        hapus = hapus->next;
        phapus->next = hapus->next;
        free(hapus);
        hapus == NULL;
    }
}

//Step 1) Memesan Alamat
void alokasi(){
    mahasiswa murid;

    input(&murid);
    p=(node *)malloc(sizeof(node));
    if(p == NULL)
        exit(0);
    else{
        p->student.id = murid.id;
        strcpy(p->student.nama, murid.nama);
        p->student.nilai = murid.nilai;
        p->next=NULL;
    }
}

void akhir(){
    node *tail;

    if(head == NULL)
        head = p;
    //Step 2) Mencari Posisi
    else{
        tail = head;
        while(tail->next != NULL)
            tail = tail->next;
        //Step 3) Menyambungkan Node
        tail->next = p;
        tail = tail->next;
    }
}

void tampil(){
    node *baca;

    baca = head;
    puts("Data yang telah diinputkan :");
    puts("No\tNama\tNilai");
    while(baca != NULL){
        printf("%d\t",baca->student.id);
        printf("%s\t", baca->student.nama);
        printf("%g\t", baca->student.nilai);
        puts("");
        baca=baca->next;
    }

}

void input(mahasiswa *student){
    int x, y;

    printf("No\t : ");
    scanf("%d",&student->id);
```

```
    fflush(stdin);
    printf("Nama\t : ");
    gets(student->nama);
    fflush(stdin);
    printf("Nilai\t : ");
    scanf("%f",&student->nilai);
}
```

Output :

```
Linked List untuk aplikasi INSERT DI AKHIR
Membentuk linked list dengan Insert di akhir
No       : 1
Nama     : Ling
Nilai    : 98
Ada data lagi (y/t)? y

No       : 2
Nama     : Ikmal
Nilai    : 89
Ada data lagi (y/t)? y

No       : 3
Nama     : Ney
Nilai    : 90
Ada data lagi (y/t)? t

Data yang telah diinputkan :
No      Nama    Nilai
1       Ling    98
2       Ikmal   89
3       Ney     90
Mau menghapus node setelah (y/t)? y
Node yang akan dihapus setelah node berapa? 1
Data yang telah diinputkan :
No      Nama    Nilai
1       Ling    98
3       Ney     90
```

4. Delete Tertentu

Kode :

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

typedef struct{
    int id;
    char nama[25];
    float nilai;
}mahasiswa;

typedef struct simpul node;
struct simpul{
    mahasiswa student;
    node *next;
};
 node *head=NULL, *p, *tail;

void input(mahasiswa *);
void tampil();
void akhir();
void alokasi();
void menghapus_tertentu();

int main(){
    char jwb;

    puts("Linked List untuk aplikasi INSERT DI AKHIR");
    puts("Membentuk linked list dengan Insert di akhir");
    do{
        fflush(stdin);
        alokasi();
        akhir();
```

```
        fflush(stdin);
        printf("Ada data lagi (y/t)? ");
        jwb=getchar();
        puts("");
    }while(jwb=='y');
    tampil();
    fflush(stdin);
    printf("Mau menghapus node ? ");
    jwb = getchar();
    if(jwb == 'y')
        menghapus_tertentu();
    tampil();
    return 0;
}

void menghapus_tertentu(){
    int key;
    node *hapus, *phapus;

    printf("Node yang akan dihapus? ");
    scanf("%d", &key);
    hapus = head;
    if(hapus->student.id == key){
        head = hapus->next;
    }
    else{
        while(hapus->student.id != key){
            if(hapus->next != NULL){
                phapus = hapus;
                hapus = hapus->next;
            }
            else{
                printf("%d tidak ada di dalam SLL\n", key);
                exit(0);
            }
        }
        phapus->next = hapus->next;
    }
    free(hapus);
    hapus == NULL;
}

//Step 1) Memesan Alamat
void alokasi(){
    mahasiswa murid;

    input(&murid);
    p=(node *)malloc(sizeof(node));
    if(p == NULL)
        exit(0);
    else{
        p->student.id = murid.id;
        strcpy(p->student.nama, murid.nama);
        p->student.nilai = murid.nilai;
        p->next=NULL;
    }
}

void akhir(){
    node *tail;

    if(head == NULL)
        head = p;
```

```
    //Step 2) Mencari Posisi
    else{
        tail = head;
        while(tail->next != NULL)
            tail = tail->next;
        //Step 3) Menyambungkan Node
        tail->next = p;
        tail = tail->next;
    }
}

void tampil(){
    node *baca;

    baca = head;
    puts("Data yang telah diinputkan :");
    puts("No\tNama\tNilai");
    while(baca != NULL){
        printf("%d\t",baca->student.id);
        printf("%s\t", baca->student.nama);
        printf("%g\t", baca->student.nilai);
        puts("");
        baca=baca->next;
    }

}

void input(mahasiswa *student){
    int x, y;

    printf("No\t : ");
    scanf("%d",&student->id);
    fflush(stdin);
    printf("Nama\t : ");
    gets(student->nama);
    fflush(stdin);
    printf("Nilai\t : ");
    scanf("%f",&student->nilai);
}
```

Output :

```
Linked List untuk aplikasi INSERT DI AKHIR
Membentuk linked list dengan Insert di akhir
No       : 1
Nama     : Ikmal
Nilai    : 98
Ada data lagi (y/t)? y

No       : 2
Nama     : Ney
Nilai    : 90
Ada data lagi (y/t)? y

No       : 3
Nama     : Ling
Nilai    : 89
Ada data lagi (y/t)? t

Data yang telah diinputkan :
No      Nama    Nilai
1       Ikmal   98
2       Ney     90
3       Ling    89
Mau menghapus node ? y
Node yang akan dihapus? 2
Data yang telah diinputkan :
No      Nama    Nilai
1       Ikmal   98
3       Ling    89
```

5. Menu Delete

Kode :

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

typedef struct{
   int id;
   char nama[25];
   float nilai;
}mahasiswa;

typedef struct simpul node;
struct simpul{
   mahasiswa student;
   node *next;
};

node *head = NULL, *p;

void menu();
void masukkan();
void menghapus();
void alokasi();
void input(mahasiswa *);
void awal();
void akhir();
void setelah();
void sebelum();
void menghapus_awal();
void menghapus_akhir();
void menghapus_tertentu();
void tampil();

int main(){
   menu();

   return 0;

}

void menu(){
   int pilihan;

   do{
      puts("\nMenu SLL");
      puts("1. Insert");
      puts("2. Delete");
      puts("3. Keluar");
      printf("Masukkan Pilihan Anda : ");
      scanf("%d", &pilihan);
      if(head == NULL && pilihan == 2)
         puts("Tidak Bisa Melakukan Delete karena Data Kosong");
      else{
          switch(pilihan){
            case 1 :
               masukkan();
               break;
            case 2 :
               menghapus();
               break;
            case 3 :
               exit(0);
```

```
                break;
            default :
                puts("Pilih yang benar");
        }
    }
    tampil();
  }while(pilihan != 3);
}

void masukkan(){
  int pilih;

  puts("Menu Insert");
  puts("1. Insert di Awal");
  puts("2. Insert di Akhir");
  puts("3. Insert Sebelum Node Tertentu");
  puts("4. Insert Sesudah Node Tertentu");
insert:
  printf("Masukkan Pilihan Anda : ");
  scanf("%d",&pilih);
  if(head == NULL && pilih == 3 || head == NULL && pilih == 4){
    puts("Data masih kosong tidak bisa melakukan insert sebelum atau sesudah");
    goto insert;
  }
  else{
     switch(pilih){
        case 1 :
            alokasi();
            awal();
            break;
        case 2 :
            alokasi();
            akhir();
            break;
        case 3 :
            alokasi();
            sebelum();
            break;
        case 4 :
            alokasi();
            setelah();
            break;
        default :
            puts("Pilih yang benar");
            goto insert;
    }
  }
}

void menghapus(){
  int jawab;

  puts("Menu Delete");
  puts("1. Delete di Awal");
  puts("2. Delete di Akhir");
  puts("3. Delete Node Tertentu");
hapus:
  printf("Masukkan Pilihan Anda : ");
  scanf("%d",&jawab);
  switch(jawab){
    case 1 :
        menghapus_awal();
        break;
```

```
        case 2 :
            menghapus_akhir();
            break;
        case 3 :
            menghapus_tertentu();
            break;
        default :
            puts("Pilih yang benar");
            goto hapus;
    }

}

void menghapus_tertentu(){
    int key;
    node *hapus, *phapus;

input:
    printf("Masukkan Node berapa yang ingin dihapus : ");
    scanf("%d", &key);
    hapus = head;
    if(hapus->student.id == key){
        head = hapus->next;
    }
    else{
        while(hapus->student.id != key){
            if(hapus->next != NULL){
                phapus = hapus;
                hapus = hapus->next;
            }
            else{
                printf("%d tidak ada di dalam SLL\n", key);
                goto input;
            }
        }
        phapus->next = hapus->next;
    }
    free(hapus);
    hapus == NULL;
}

void menghapus_akhir(){
    node *hapus, *phapus;

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

void menghapus_awal(){
    node *hapus;

    hapus = head;
    head = hapus->next;
    free(hapus);
```

```
    hapus == NULL;
}

void alokasi(){
    mahasiswa murid;

    input(&murid);
    p=(node *)malloc(sizeof(node));
    if(p == NULL)
        exit(0);
    else{
        p->student.id = murid.id;
        strcpy(p->student.nama, murid.nama);
        p->student.nilai = murid.nilai;
        p->next=NULL;
    }
}

void input(mahasiswa *student){
    int x, y;

    printf("No\t : ");
    scanf("%d",&student->id);
    fflush(stdin);
    printf("Nama\t : ");
    gets(student->nama);
    fflush(stdin);
    printf("Nilai\t : ");
    scanf("%f",&student->nilai);
}

void awal(){
    if(head != NULL)
        p->next = head;
    head = p;
}

void akhir(){
    node *tail;

    if(head == NULL)
        head = p;
    else{
        tail = head;
        while(tail->next != NULL)
            tail = tail->next;
        tail->next = p;
        tail = tail->next;
    }
}

void setelah(){
    int key;
    node *after;

sesudah:
    printf("Disisipkan setelah data berapa ? ");
    scanf("%d",&key);
    after = head;
    while(after->student.id != key){
        if(after->next == NULL){
            printf("%d tidak ditemukan di dalam SLL\n",key);
            goto sesudah;
```

```
        }
        else{
            after = after->next;
        }
    }
        p->next = after->next;
        after->next = p;
}

void sebelum(){
    int key;
    node *bef, *pbef;

before:
    printf("Disisipkan sebelum data berapa ? ");
    scanf("%d",&key);
    bef = head;
    while(bef->student.id != key){
        if(bef->next == NULL){
            printf("%d tidak ditemukan di dalam SLL\n",key);
            goto before;
        }
        else{
            pbef = bef;
            bef = bef->next;
        }
    }
    if(bef == head){
        p->next = bef;
        head = p;
    }
    else{
        p->next = bef;
        pbef->next = p;
    }
}

void tampil(){
    node *baca;

    baca = head;
    puts("Data yang telah diinputkan :");
    puts("No\tNama\tNilai");
    while(baca != NULL){
        printf("%d\t",baca->student.id);
        printf("%s\t", baca->student.nama);
        printf("%g\t", baca->student.nilai);
        puts("");
        baca=baca->next;
    }
    if(head == NULL)
        puts("Kosong");

}
```

Output :

```
Menu SLL
1. Insert
2. Delete
3. Keluar
Masukkan Pilihan Anda : 1

Menu Insert
1. Insert di Awal
2. Insert di Akhir
3. Insert Sebelum Node Tertentu
4. Insert Sesudah Node Tertentu
Masukkan Pilihan Anda : 1
No       : 1
Nama     : Ikmal
Nilai    : 98
Membentuk Linked List dg insert di awal
Data yang telah diinputkan :
No      Nama    Nilai
1       Ikmal   98

Menu SLL
1. Insert
2. Delete
3. Keluar
Masukkan Pilihan Anda : 1
```

```
Menu Insert
1. Insert di Awal
2. Insert di Akhir
3. Insert Sebelum Node Tertentu
4. Insert Sesudah Node Tertentu
Masukkan Pilihan Anda : 2
No       : 2
Nama     : Ikmal
Nilai    : 98
Membentuk Linked List dg insert di akhir
Data yang telah diinputkan :
No      Nama    Nilai
1       Ikmal   98
2       Ikmal   98

Menu SLL
1. Insert
2. Delete
3. Keluar
Masukkan Pilihan Anda : 2
Menu Delete
1. Delete di Awal
2. Delete di Akhir
3. Delete Node Tertentu
Masukkan Pilihan Anda : 3
Masukkan Node berapa yang ingin dihapus : 1
Data yang telah diinputkan :
No      Nama    Nilai
2       Ikmal   98
```

```
Menu SLL
1. Insert
2. Delete
3. Keluar
Masukkan Pilihan Anda : 3
```

Algoritma umum pada operasi delete dalam single linked list yaitu ada 3. Pertama, pilih node yang ingin kita delete. Kedua, selamatkan node agar tidak terputus. Ketiga, bebaskan node yang ingin kita hapus dan jangan lupa me-NULL kan pointer yang digunakan untuk membebaskan node.

Dalam melakukan delete awal yang pertama kita lakukan yaitu kita harus menyiapkan sebuah pointer hapus dan kita assign head ke pointer hapus. Setelah itu kita selamatkan head dengan cara diassignkan oleh head->next. Setelah itu otomatis node masih tetap terhubung. Setelah node head dihubungkan maka langkah selanjutnya melakukan pembebasan pada node yang ditunjuk oleh hapus. Setelah alamat yang ditunjuk hapus dibebaskan jangan lupa untuk me-NULLkan hapus.

Dalam melakukan delete akhir kita memerlukan 2 buah pointer bantuan yaitu phapus dan hapus. Cara menggunakannya sama seperti algoritma di saat kita melakukan insert before. Apabila data hanya terdapat hanya satu node, maka langsung dilakukan pembebasan layaknya delete awal. Apabila data node lebih dari 1 maka hapus akan mencari data terakhir setelah itu phapus akan berada di belakang pointer hapus. Setelah node terakhir ditemukan maka kita harus menyambungkan node agar tidak terputus dengan cara, phapus->next disambungkan dengan hapus->next. Setelah aman maka dapat dilakukan pembebasan. Sama seperti sebelumnya apabila telah melakukan pembebasan maka harus me-NULLkan pointer hapus.

Dalam melakukan delete after kita memerlukan 2 pointer Bantuan yaitu phapus dan hapus. Pertama kita *assign head* ke hapus. Setelah itu lakukan pengecekan apakah hapus->data sesuai dengan *key*. Apabila tidak sesuai maka hapus akan bergerak menuju hapus->next. Apabila hapus sudah menemui data yang dicari langkah selanjutnya maka kita perlu melakukan *assign* yaitu hapus ke phapus yang berfungsi untuk memegang *node* itu. Setelah itu hapus pindahkan hapus menuju hapus->next. Setelah berpindah maka sambungkan node agar tidak terputus dan bebaskan hapus. Setelah hapus dibebaskan maka NULL kan pointer hapus.

Dalam melakukan delete node tertentu kita memerlukan sebuah key, untuk mencari data mana yang ingin dicari. Apabila key yang dicari ada di awal data maka seperti sebelumya akan dilakukan delete awal. Jika data yang dicari belum ketemu maka pointer hapus akan mencari sampai menemukannya. Di belakang pointer hapus ada pointer phapus yang berfungsi untuk memegang node di belakang pointer hapus. Apabila data yang dicari ketemu maka lakukan penyelamatan dengan cara melakukan assign antara phapus->next dengan hapus->next. Setelah dilakukan penyelamatan maka bisa dilakukan pembebasan. Setelah dilakukan pembebasan maka jangan lupa melakukan NULL pada pointer hapus. Akan tetapi apabila data yang dicari tidak ada di dalam SLL maka program akan melakukan perintah keluar.

Setelah itu terdapat sebuah menu untuk SLL, kita dapat memilih kita mau melakukan apa. Contohnya seperti kita dapat melakukan insert dan delete. Untuk memberhentikan program dengan cara memilih opsi 3 yaitu dengan cara keluar.