Pemahaman Modul

Praktikum Struktur Data.

Kode_Asisten_PJ_:_MTN Kelompok = A11.4301U

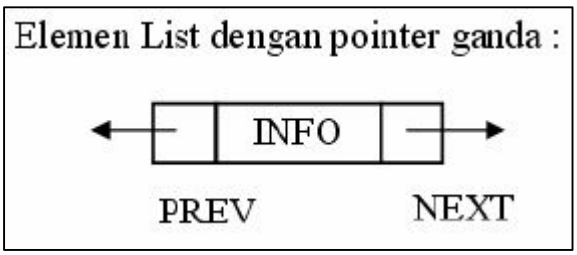
NIM :A11.2012.06758 NIM : A11.2012.06748

Nama :Muhammad Adhi D Nama : M. Azwar Adli

PSDA – 09 (List Linier Dengan Pointer Ganda).

Penjelasan singkat List Linier Pointer Ganda.

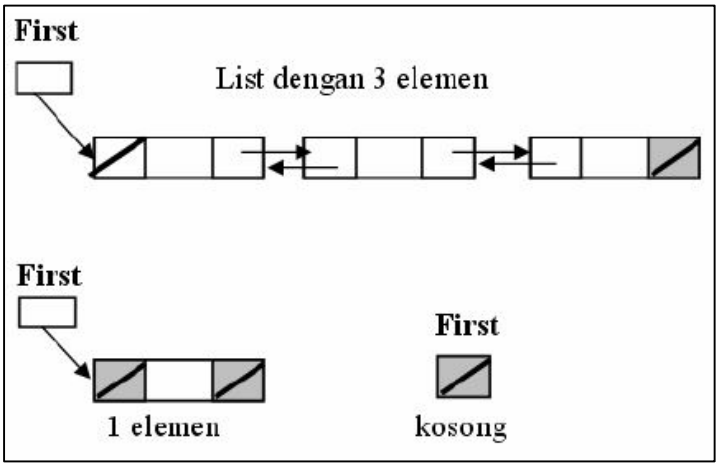
List dengan representasi ini mempunyai 1 identitas tambahan daripada elemen yang biasanya yaitu adalah selekto "Prev".Dimana jika selector NEXT digunakan untuk mengacu pada addres selnajutya pada suatu elemen, maka selector PREV di gunakan untuk mengacu pada address sebelumnya pada suatu elemen. Sehingga benuk fisik sebuah elemen dengan representasi ini adalah seperti pada gambar berikut.



Ciri dari representasi ini adalah :

Elemen pertama : First(L)= P
Elemen terakhir :Last(L)= P,
Next(P) = NIL,
Prev(P) = NIL,
List kosong : First(L) = Nil

Ilustrasi List Pointer Ganda dengan 3 Elemen.



Pada gambar disamping menggambarkan List Pointer ganda dengan 3 buah elemen.Dapat dilihat bahwa bentuk representasi fisik pada variasi ini adalah pada selector Prev dan Next di mana keduanya saling menunjuk pada elemen yang bersangkutan.

Next(L) = akan selalu menunjuk pada address elemen **selanjutnya**, sedangkan LAST(L) = akan selalu menunjuk pada address elemen **sebelumnya,** kelebihna dari repsenentasi ini adalah dapat melakukan traversal dari depan (FIRST) , traversa menggunakan NEXT, maupun dari belakang (LAST) , traversal menggunkana PREV.

A. Penjelasan Mengenai beberapa fungsi.

```
void createList (List *L)
{
    FIRST(*L)=NIL;
    LAST(*L)=NIL;
}
```

Pada fungsi CreateList yang ada pada poko bahasan List Linier bSirkuler, fungsi ini akan hanya berisi

*FIRST(*L) = NULL.*

*LAST(*L) = NULL.*

Dimana pada tahap fungsi ini berfungsi sebagai pembentuk nilai default dari identitas FIRST dan LAST, yaitu menunjuk pada alamat kosong (NULL).

```
void InsVFirst(List *L, infotype X)
{
    P = Alokasi(X);
    if(P!=Nil)
    {
        InsertFirst(&(*L),P);
    }
}
```

```
void InsertFirst(List *L, address P)
{
    if (ListEmpty(*L))
    {
        First(*L) = P;
        LAST(*L) = P;
        Next(P) = Nil;
        Prev(P) = Nil;
    }
    else
    {
        Next(P) = First(*L);
        Prev(First(*L)) = P;
        Prev(P) = Nil;
        First(*L) = P;
    }
}
```

Pada fungsi InsertFirst ini perbedaanya adalah pada tahap awal jika FIRST(*L) = NILL,atau jika List tersebut masih kosong, maka dilakukan pencatatan sebagai elemen terakhir pada variabel LAST(*L). Pernytaan ini dilakukan sekali hanya jika kondisi tersebut terpenuhi.Hal tersebut dilakukan karena yang diinsert adalah hanya pada posisi index pertama, sehingga nilai dari addres LAST akan tetap seperti kondisi 1 Elemen, dan tidak akan pernah berubah walaupun dilakukan InsertFirst secara berkelanjutan.

Berbeda dengan InsertVLast yang dimana elemen terakhirnya selalu berubah, maka dari itu selalu dilakukan pemindahan address pada elemen terakhir pada variabel LAST(*L).

Fungsi InsertFirst di atas berguna untuk mengkonfigurasi pointer-pointer pada suatu elemen, agar terjadi penambahan pada elemen pertama.

Statemen setelah “else” menyatakan penunjuk dari elemen yang baru, agar menunjuk pada elemen pertama.

NEXT(P) = FIRST(*L);

yang artinya adalah alamat selanjutnya dari elemen dengan addres P (yang baru) adalah mengacu pada addres elemen pertama FIRST(*L).Misalkan

terdapat List dengan elemen "5 -> 6 -> 7". Akan diinsert elemen **"8"**.Maka posisinya akan menjadi "8 -> 5 -> 6 -> 7".

Statemen kedua menytakan bahwa address sebelum FIRST menunjuk pada addres yang baru yaitu P, di sinilah proses penyatuan antara NEXT dan PREV dilakukan.

```
PREV(FIRST(*L)) = P;
```

Statemen ke tiga adalah sebagai penanda bawah sebelum elemen yang baru P menunjuk pada address kosong atau NIL.

```
PREV(P) = NIL;
```

Statemen yang ke empat adalah sebagai penunjuk bahwa elemen yang telah dimasukan tadi yaitu elemen "8" ditunjuk sebagai FIRST, dengan address elemen selanjutnya telah tersusun sedemikian rupa sesuai dengan urutan semula.

```
FIRST(*L) = P;
```

```
void InsertLast(List *L, address P)
{
    address Prec;
    if (ListEmpty(*L))
    {
        InsertFirst(L,P);
    }
    else
    {
        Next(LAST(*L)) = P;
        Prev(P) = LAST(*L);
        LAST(*L) = P;
    }
}
```

Pada Fungsi InsertLast , Hampir sama dengan Fungsi FIRST, hanya saja perbedaanya pada setiap dilakukan pemanggilan fungsi Last, akan dilakukan juga perubahan addres elemen terakhir (LAST). Karena penambahan elemen dilakukan pada address terakhir, maka nilai adderss dari LAST akan terus berubah. Hal tersebut ditunjukan pada statement.

LAST(*L) = P;

Dimana "P" di sini adalah addrss yang baru

Pada bagian ini tidak perlu dilakukan traversal,karena pencatatan address last telah dilakukan pada setiap penambahan elemen, sehingga jika akan melakukan pemasukan elemen terakhir tinggal menggunkana variabel LAST. Hal tersebut ditunjukan pada statement.

Next(LAST(*L)) = P;

Pada statement selanjutnya yang menyatakan.

Prev(P) = LAST(*L);

Yaitu alamat sebelumnya dari elemen yang baru ditambahkan P, harus menunjuk pada elemen LAST sebelumnya.