Pertemuan 5

Waktu : 135 menit

Tujuan Pembelajaran : Mahasiswa mampu menjelaskan teknik pemrograman menggunakan Linked List.

Substansi Materi : Single Linked List, LIFO, FIFO

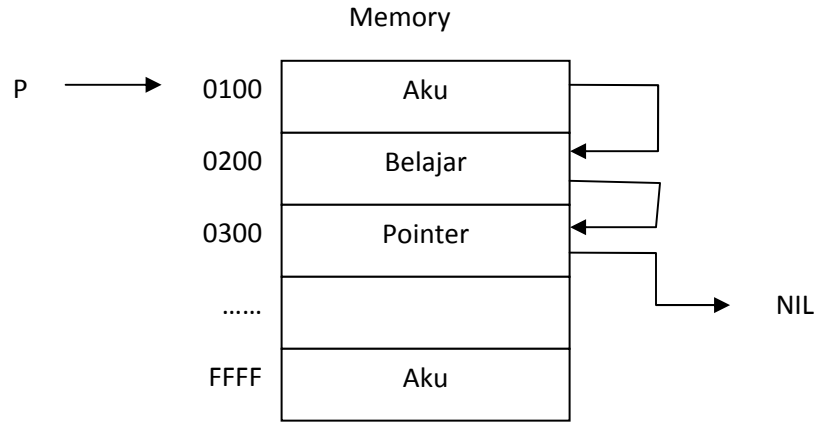
Tabulasi Kegiatan Perkuliahan

No	Tahap Kegiatan	Kegiatan Pengajar	Kegiatan Mahasiswa	Media & Alat	Waktu
1	Pendahuluan	1. Membuka pertemuan 2. Mengulang materi pertemuan sebelumnya	Menyimak Bertanya	Papan Tulis	20 Menit
2	Penyajian Materi	1. Singled Linked List 2. Last in first out 3. First in first out	Menyimak Bertanya Menjawab Pertanyaan	Papan Tulis	80 Menit
3	Penutup	1. Menyimpulkan materi pertemuan 2. Memberikan tugas kecil 3. Menutup pertemuan	Menyimak	Papan tulis	35 Menit

MATERI KULIAH

Single Linked List

Gambar berikut menunjukan sebuah data terletak pada sebuah lokasi memory. Tempat yang disediakan pada suatu area memory tertentu untuk menyimpan data dikenal dengan sebutan node / simpul. Pada setiap node memiliki pointer(penunjuk) yang menunjuk ke simpul berikutnya sehingga terbentuk suatu untaian dan dengan demikian hanya diperlukan sebuah variable pointer. Susunan berupa untaian ini disebut dengan Single Linked List. Nil tidak memiliki nilai apapun. Setiap linked list pada akhirnya akan menunjuk ke Nil.



Dalam pembuatan Single Linked List dapat menggunakan 2 Metoda :

- LIFO (Last In First Out), aplikasinya : Stack (Tumpukan)
- FIFO (First In First Out), aplikasinya : Queue (Antrian)

LIFO (Last In First Out)

LIFO adalah suatu metoda pembuatan Linked List dimana data yang masuk paling akhir adalah data yang keluar paling awal. Hal ini dapat dianalogikan dengan menumpukan barang pada kehidupan sehari-hari. Pembuatan simpul pada suatu linked list disebut dengan istilah INSERT. Jika linked list dibuat dengan Metoda LIFO maka penambahan/insert simpul dilakukan di BELAKANG.

Procedure Insert

Istilah INSERT berarti menambahkan sebuah simpul baru ke dalam suatu linked list. Berikut adalah deklarasi tipe data dan variabel yang dapat digunakan sebagai deklarasi awal dan penggalan procedure insert.

```
Type
   Point = ^RecPoint;
   RecPoint = Record
            Isi : TipeData;
            Next : Point;
            End;
Var
   Head, Tail, Now : Point;
```

```
Procedure INSERT(elemen:TipeData);
Var Now : Point;
   Begin
            New(Now);
            Now^.Isi := Elemen;
            If Head = Nil Then
               Now^.Next := Nil;
            Else
               Now^.Next := Head;
            Head := Now;
   End;
```

First In First Out

FIFO adalah suatu metoda pembuatan Linked List dimana data yang masuk paling awal adalah data yang keluar paling awal juga. Jika linked list dibuat dengan menggunakan FIFO, maka terjadi penambahan / Insert simpul di depan.

```
Procedure INSERT(elemen:TipeData);
Var Now : Point;
  Begin
        New(Now);
        If Head = Nil Then
          Head := Now;
        Else
           Tail^.Next := now;
        Tail := Now;
        Tail^.Next := Nil;
        Now^.Isi := Elemen;
End;
```

Procedure dan function Linked List lainnya

Selain procedure insert diatas, pada linked list juga masih terdapat procedure serta function lainnya. Dibawah ini diberikan procedure-procedure serta function umum dalam linked list.

- **Create**

  Membuat sebuah linked list yang baru dan masih kosong. Procedure ini wajib dipanggil untuk menggunakan linked list.

```
Procedure Create;
Begin
  Head := Nil;
  Tail := Nil;
End;
```

- **Empty**

Function untuk menentukan apakah linked list kosong atau tidak.

```
Function Empty : Boolean;
Begin
   If head = nil then
            Empty := true
   else
             Empty := false;
End;
```

- **Find First**

Mencari elemen pertama dari linked list

```
Procedure Find_First;
Begin
   Now := Head;
End;
```

- **Find Next**

Mencari elemen sesudah elemen yang ditunjuk Now

```
Procedure Find_Next;
Begin
   If Now^.Next <> Nil then
            Now := Now^.next;
End;
```

- **Retrieve**

Mengambil elemen yang ditunjuk oleh now. Elemen tersebut lalu ditampung pada suatu variabel, dalam potongan procedure ini ditampung dalam variabel r.

```
Procedure Retrieve(var r : TipeData);
Begin
   R := Now^.Isi;
End:
```

Update

Mengubah elemen yang ditunjuk oleh now dengan isi dari suatu variabel (dalam contoh ini digunakan variabel u).

```
Procedure UpDate(u :TipeData);
Begin
   Now^.Isi := U;
End:
```

Delete Now

Menghapus elemen yang ditunjuk oleh now. Jika yang dihapus adalah elemen yang pertama dari linked list (head), maka head akan berpindah ke elemen berikutnya.

```
Procedure DeleteNow;
Var x : point;
Begin
   If Now <> Head then
   Begin
         X := head;
         While x^.next <> now do
            X := x^.next;
         X^.next := now^.next;
   End
   Else head := head^.next;
         Dispose(now);
   Now := head;
End;
```

Delete Head

Menghapus elemen yang ditunjuk oleh head. Head akan berpindah ke elemen sesudahnya.

```
Procedure DeleteHead;
Begin
  If head <> nil then
        Begin
        Now := Head;
        Head := Head^.Next;
        Dispose(Now);
        Now := Head;
        End;
End;
```

Clear

Untuk menghapus linked list yang sudah ada. Wajib dilakukan bila ingin mengakhiri program yang menggunakan linked list. Jika tidak ada data-data yang dialokasikan ke memory pada program sebelumnya akan tetap tertinggal di dalam memory.

```
Procedure Clear;
Begin
  While head <> nil do
        Begin
        Now := head;
        Head := head^.next;
        Dispose(now);
        End;
End;
```