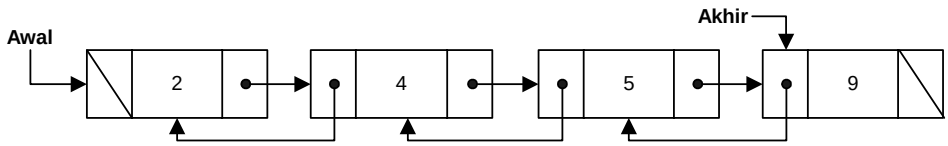
BAB IX
LINKED LIST
(SENARAI BERANTAI)

Double Linked List

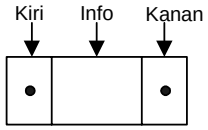
Double Linked List adalah suatu linked list yang mempunyai 2 penunjuk yaitu penunjuk ke data sebelumnya dan berikutnya. Perhatikan gambar di bawah ini :



Beberapa operasi yang dapat dilakukan dalam double linked list adalah :

1. **Pendeklarasian Struktur dan Variabel Double Linked List**

   Jika dilihat 1 elemen listnya, maka secara umum struktur dari elemen listnya adalah sebagai berikut :

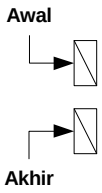


   Dari gambar di atas, untuk setiap elemen terdiri dari 3 buah field yaitu kiri (prev), info (data), dan kanan (next). Field kiri dan kanan merupakan sebuah pointer ke data struktur elemen (data).

   Pendeklarasian struktur double linked list dalam bahasa Pascal adalah :

```
Type elemen = integer;
     simpul = ^data;
     Data = record
              Info : elemen;
              Kiri, kanan : simpul;
            End;
Var awal, akhir : simpul;
```

   Kondisi awal ketika awal dan akhir telah dideklarasikan.



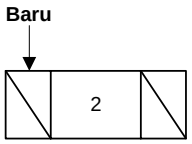
2. **View data**

   Untuk view data, langkah yang dilakukan adalah dengan menelusuri semua elemen list sampai elemen terakhir. Setelah melakukan penelurusan posisi awal dan akhir elemen tidak boleh berubah sehingga untuk melakukan penelusuran dibutuhkan sebuah variabel bantu.

Kelebihan dari view data pada linked list adalah kita dapat menampilkan data dari data terakhir dengan lebih mudah.

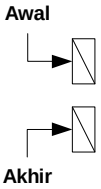
3. Tambah di awal

Operasi ini berguna untuk menambahkan elemen baru di posisi pertama. Langkah pertama untuk penambahan data adalah pembuatan elemen baru dan pengisian nilai infonya. Pointer yang menunjuk ke data tersebut dipanggil dengan nama **baru**. Kondisi di setelah ada pembuatan elemen baru tersebut adalah :



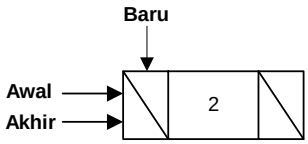
Perhatikan gambar di bawah ini :

- Kondisi sebelum disisipkan



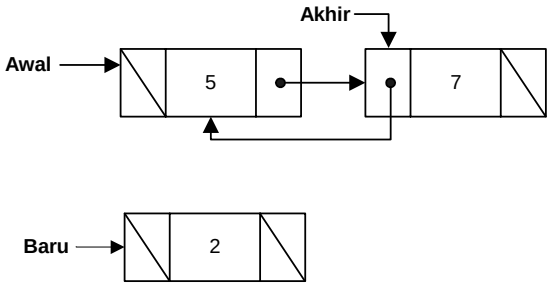
- Kondisi setelah operasi penambahan

  Operasi penambahan awal ketika linked list masih kosong adalah dengan mengisikan alamat pointer baru ke pointer awal dan pointer akhir. Lihat gambar di bawah ini.



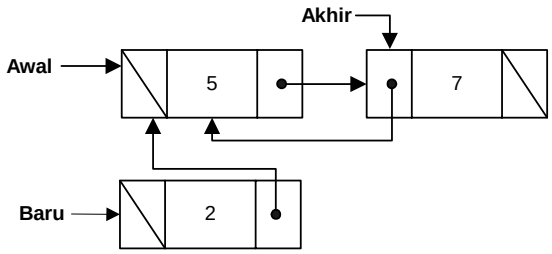
a. Ketika linked list sudah mempunyai data

Kondisi linked list ketika sudah mempunyai data elemen dan elemen yang baru telah dibuat, dapat dilihat di gambar di bawah ini.

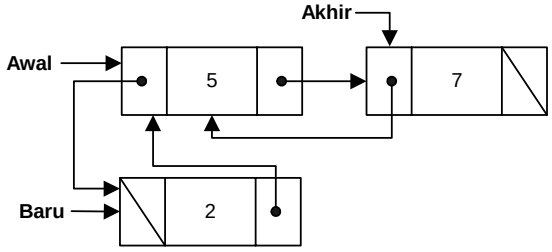


Proses penambahan data di awal linked list adalah :

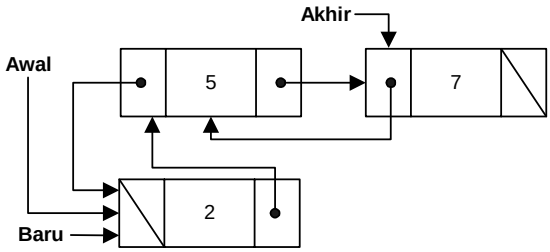
- Hubungkan baru→kanan agar menunjuk ke awal



- Hubungkan awal→kiri agar menunjuk ke posisi pointer baru

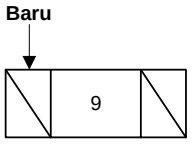


- Pindahkan pointer awal ke pointer baru



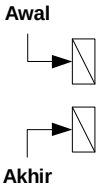
4. Tambah di akhir

Operasi ini berguna untuk menambahkan elemen baru di posisi akhir. Langkah pertama untuk penambahan data adalah pembuatan elemen baru dan pengisian nilai infonya. Pointer yang menunjuk ke data tersebut dipanggil dengan nama **baru**. Kondisi di setelah ada pembuatan elemen baru tersebut adalah :



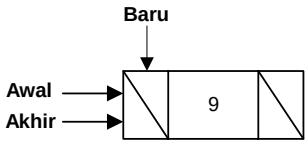
Perhatikan gambar di bawah ini :

- Kondisi sebelum penambahan

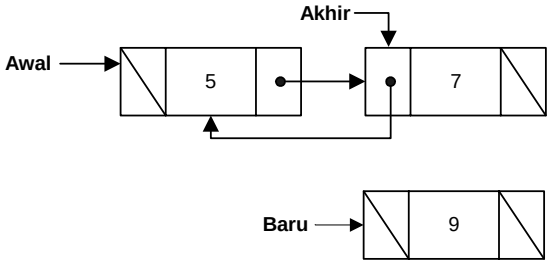


- Kondisi setelah operasi penambahan

Operasi penambahan awal ketika linked list masih kosong adalah dengan mengisikan alamat pointer baru ke pointer awal dan pointer akhir. Lihat gambar di bawah ini.

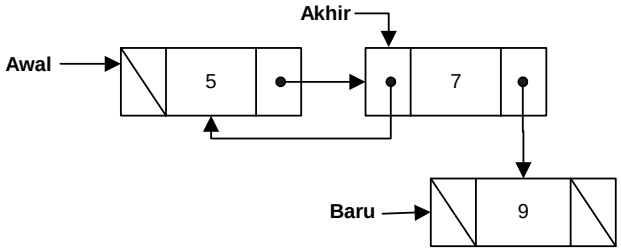


Kondisi linked list ketika sudah mempunyai data elemen dan elemen yang baru telah dibuat, dapat dilihat di gambar di bawah ini.

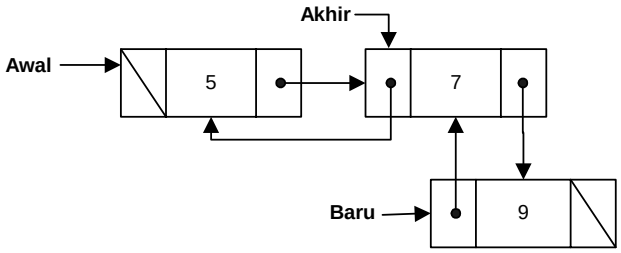


Proses penambahan data di akhir linked list adalah :

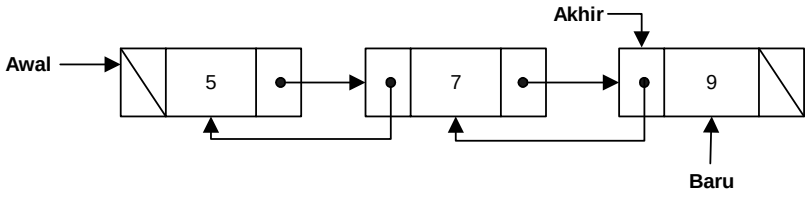
- Hubungkan akhir→kanan agar menunjuk ke pointer baru



- Hubungkan baru→kiri agar menunjuk ke posisi pointer akhir



- Pindahkan pointer akhir ke pointer baru



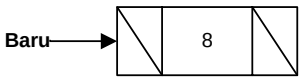
5. Sisip di tengah

Operasi penyisipan data di tengah linked list adalah suatu operasi menambah data di posisi tertentu di dalam linked list. Contohnya adalah jika ingin menyisipkan data di posisi ke-3 atau ke-4.

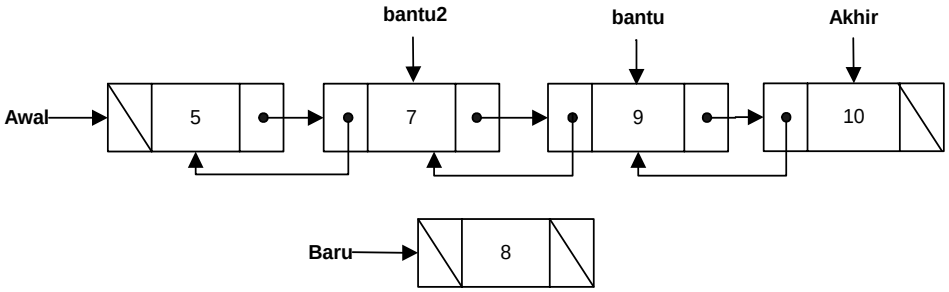
Untuk proses tersebut ada 2 hal yang harus diperhatikan yaitu :

a. Kondisi linked list masih kosong atau posisi penyisipan kurang dari atau sama dengan 1
   Jika kondisi ini terjadi, maka langkah yang dilakukan adalah sangat mudah yaitu dengan memanggil proses penambahan data awal atau akhir. (untuk lebih jelas lihat penambahan data awal atau akhir ketika kondisi linked list masih kosong).

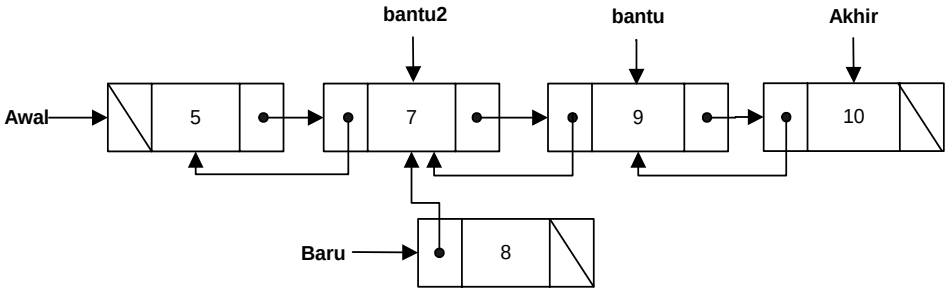
b. Kondisi linked list sudah mempunyai data
   Langkah untuk penyisipan data ketika linked list sudah mempunyai data adalah :
   - Cari posisi pointer pada data ke-***posisi sisip***.
   - Setelah posisi penyisipan (bantu) ditemukan, maka periksa apakah posisi penyisipan (bantu) bernilai NIL atau tidak. Jika posisi penyisipan (bantu) bernilai NIL berarti posisi sisip berada di luar atau melebihi linked list. Oleh karena itu berarti penambahan datanya dilakukan dengan melakukan operasi penambahan akhir.
   - Jika posisi penyisipan (bantu) tidak sama dengan NIL, maka itu berarti posisi penyisipan ada di dalam jangkauan linked list. Proses yang dilakukan untuk penyisipannya adalah :
     - Buat elemen baru di memori dan isi infonya (contoh data info = 8).



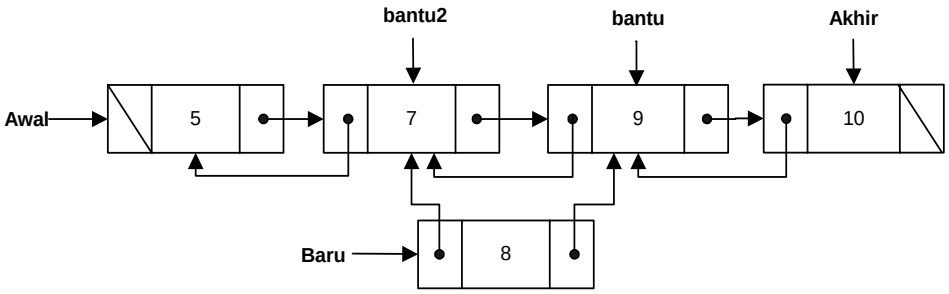
     - Untuk mempermudah proses penyambungan/penyisipan data baru, maka buat variabel pointer baru dengan nama bantu2 untuk memegang data di sebelah kiri dari posisi sisip (bantu→kiri). (Contoh posisi penyisipan adalah 3)



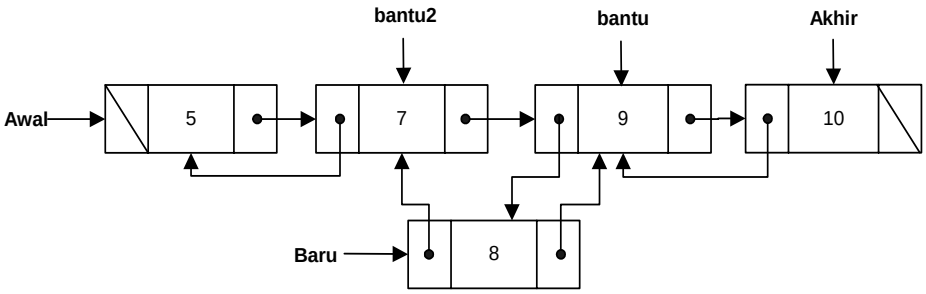
     - Isi baru→kiri dengan pointer bantu2



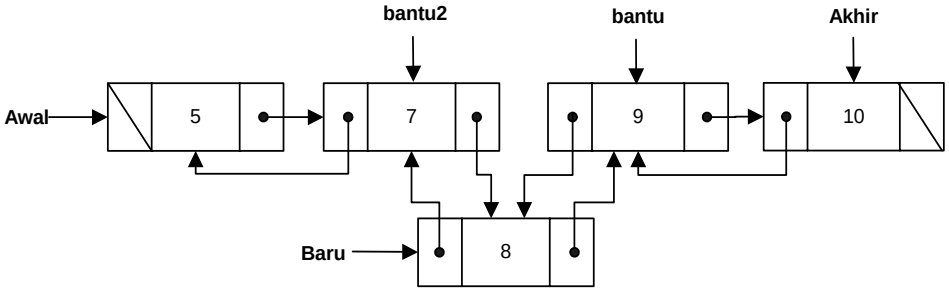
     - Isi/sambungkan baru→kanan ke pointer bantu



- Isi/sambungkan bantu→kiri ke posisi pointer baru



- Isi/sambungkan bantu2→kanan ke posisi pointer baru



6. Hapus data awal

Operasi ini berguna untuk menghapus data pada posisi pertama. Ada 3 keadaan yang mungkin terjadi ketika akan melakukan proses hapus yaitu :

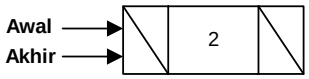
a. Kondisi linked list masih kosong

Jika kondisi ini terjadi, maka proses penghapusan data tidak bisa dilakukan karena linked list masih kosong.

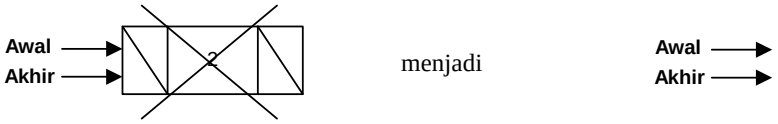
b. Kondisi linked list hanya memiliki 1 data

Langkah yang perlu dilakukan ketika ingin melakukan proses penghapusan linked list yang memiliki hanya 1 data adalah dengan langsung menghapus data dari memori dan kemudian pointer awal dan akhir di-NIL-kan. Untuk lebih jelas perhatikan urutan penghapusannya di bawah ini :

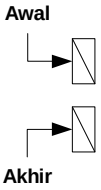
- Kondisi data sebelum dihapus



- Proses penghapusan yaitu dengan menghilangkan data dari memori.



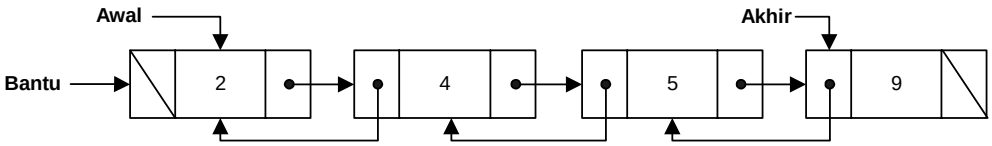
Kemudian pointer awal dan akhir diisi dengan NIL.



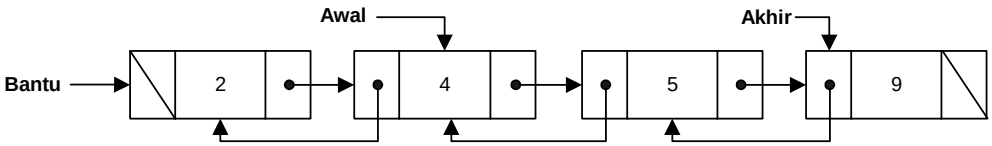
c. Kondisi linked list memiliki data lebih dari 1 buah

Untuk operasi penghapusan data di posisi pertama pada double linked list yang mempunyai data lebih dari 1 buah adalah :

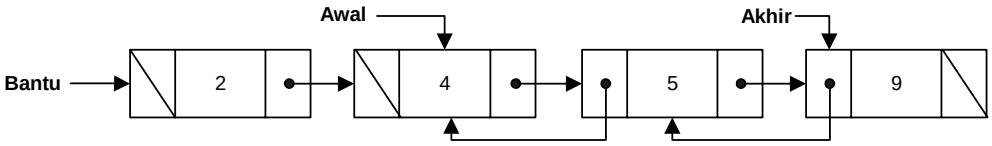
- Simpan pointer yang akan dihapus (awal) ke suatu pointer lain yang diberi nama pointer **bantu**.



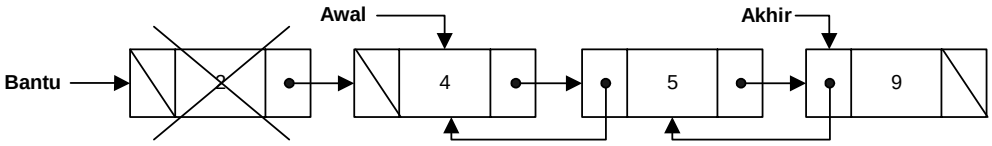
- Pindahkan pointer awal ke data berikutnya (bantu→kanan atau awal→kanan).



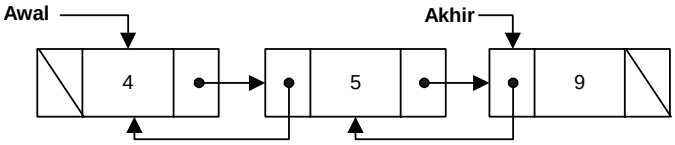
- Field kiri dari awal yang baru (awal→kiri) di-NIL-kan.



- Langkah terakhir adalah hapus elemen yang ditunjuk pointer bantu.



- Setelah data dihapus, maka kondisi linked list adalah seperti di gambar di bawah ini.



7. Hapus data terakhir

Operasi ini berguna untuk menghapus data pada posisi terakhir. Ada 3 keadaan yang mungkin terjadi ketika akan melakukan proses hapus yaitu :

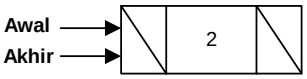
a. Kondisi linked list masih kosong

Jika kondisi ini terjadi, maka proses penghapusan data tidak bisa dilakukan karena linked list masih kosong.

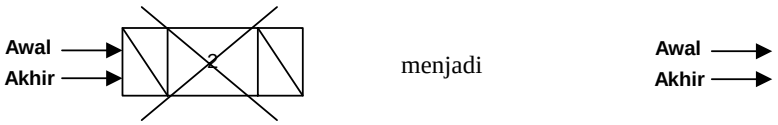
b. Kondisi linked list hanya memiliki 1 data

Langkah yang perlu dilakukan ketika ingin melakukan proses penghapusan linked list yang memiliki hanya 1 data adalah dengan langsung menghapus data dari memori dan kemudian pointer awal dan akhir di-NIL-kan. Untuk lebih jelas perhatikan urutan penghapusannya di bawah ini :

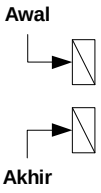
- Kondisi data sebelum dihapus



- Proses penghapusan yaitu dengan menghilangkan data dari memori dengan perintah free(awal) atau free(akhir).



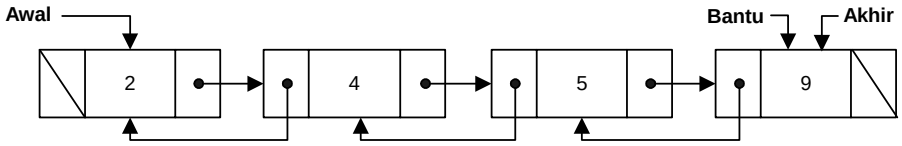
Kemudian pointer awal dan akhir diisi dengan NIL.



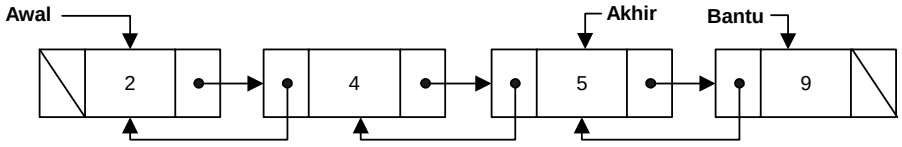
c. Kondisi linked list memiliki data lebih dari 1 buah

Untuk operasi penghapusan data di posisi terakhir pada double linked list yang mempunyai data lebih dari 1 buah adalah :

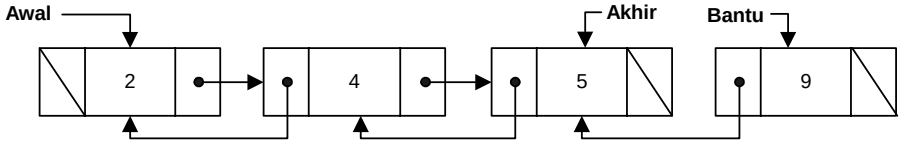
- Simpan pointer yang akan dihapus (akhir) ke suatu pointer lain yang diberi nama pointer **bantu**.



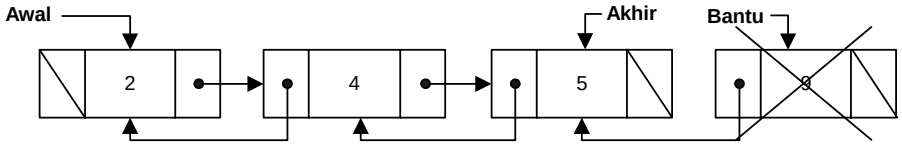
- Pindahkan pointer akhir ke data sebelumnya (bantu→kiri atau akhir→kiri).



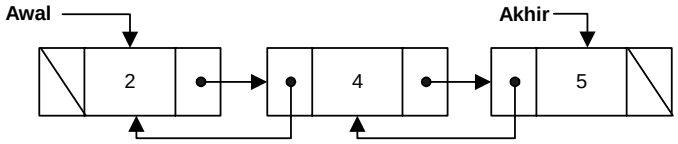
- Field kanan dari akhir baru (akhir→kanan) di-NIL-kan.



- Langkah terakhir adalah hapus elemen yang ditunjuk pointer bantu.



- Setelah data dihapus, maka kondisi linked list adalah seperti di gambar di bawah ini.



8. Hapus data di tengah

Untuk melakukan proses penghapusan di tengah linked list, ada 3 kondisi yang perlu diperhatikan yaitu :

a. Kondisi ketika linked list masih kosong atau ketika posisi hapus lebih kecil dari 1.
   Ketika kondisi ini terjadi, maka proses penghapusan tidak bisa dilakukan karena data masih kosong atau karena posisi hapus diluar jangkauan linked list (posisi kurang dari 1).

b. Kondisi ketika posisi hapus sama dengan 1 (hapus data pertama)
   Ketika kondisi ini terjadi, maka proses yang dilakukan adalah proses penghapusan di posisi awal (hapusawal).

c. Kondisi ketika posisi hapus lebih besar dari 1
   Langkah-langkah untuk penghapusan data di tengah linked list yang posisi hapusnya lebih besar dari 1 adalah :

- Cari pointer yang menunjuk ke data pada posisi ke-***posisi hapus***. Ketika kondisi ini, ada 2 kemungkinan yang bisa terjadi yaitu posisi hapus ada di dalam jangkauan linked list atau di luar linked list.
- Setelah pointer posisi penghapusan telah ditemukan, maka langkah selanjutnya adalah pemeriksaan apakah posisi penghapusan (bantu) tersebut bernilai NIL (diluar jangkauan linked list). Jika posisi penghapusan (bantu) bernilai NIL maka proses penghapusan tidak bisa dilakukan. Tetapi jika bantu tidak bernilai NIL, maka lanjutkan ke langkah berikutnya.
- Periksa juga apakah pointer posisi penghapusan (bantu) sama dengan posisi akhir, jika benar maka proses penghapusan yang dilakukan adalah proses penghapusan akhir.
- Tetapi jika posisi penghapusan tidak sama dengan posisi akhir itu menunjukan posisi penghapusan ada di tengah.