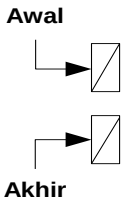
3. <u>Penghapusan</u>

a. Penghapusan di awal/depan

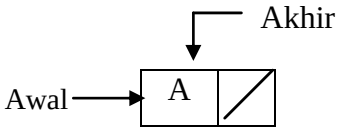
Penghapusan data di awal adalah proses menghapus simpul pertama (yang ditunjuk oleh variabel pointer Awal), sehingga variabel pointer Awal akan berpindah ke simpul berikutnya. Ada 3 kondisi yang perlu diperhatikan yaitu kondisi linked list masih kosong, kondisi linked list hanya memiliki 1 data, dan kondisi linked list yang memiliki data lebih dari 1 elemen

- **Kondisi linked list kosong**

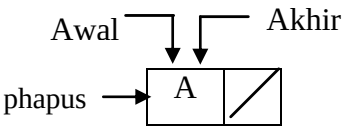


Pada kondisi ini proses penghapusan tidak bisa dilakukan

- **Kondisi linked list memiliki hanya 1 data{Satu simpul}**

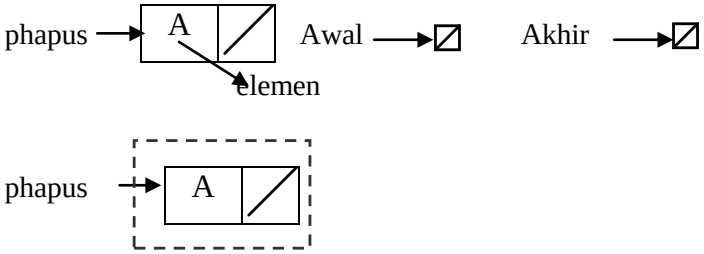


Berikan satu variabel pointer **phapus** yang menunjuk ke simpul yang ditunjuk oleh Awal/Akhir.



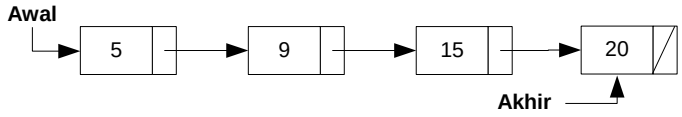
Kemudian berikan nilai nil untuk variabel pointer Awal dan variabel pointer Akhir.

Simpan terlebih dahulu nilai dari simpul yang akan dihapus ke dalam sebuah variabel e**lemen**.

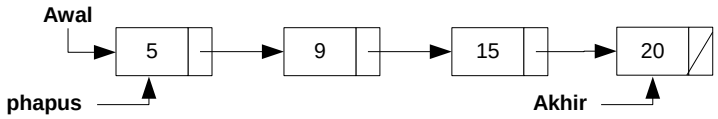


Setelah itu simpul yang ditunjuk oleh variable pointer phapus dapat dihapus.

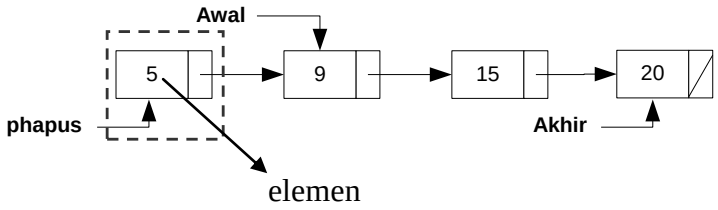
- **Kondisi linked list memiliki data lebih dari 1 data**



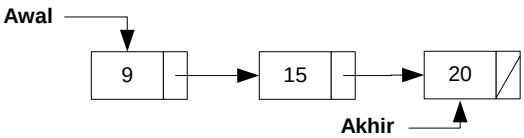
Tempatkan variabel pointer bantuan yang bernama **phapus** ke simpul yang sudah ditunjuk oleh variable pointer Awal (simpul pertama).



Setelah itu pindahkan variabel pointer Awal ke simpul berikutnya.

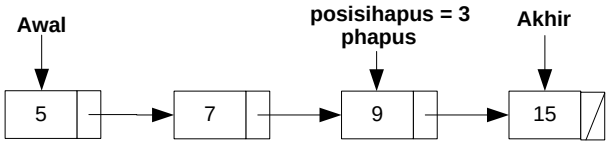


Setelah itu simpan data yang ada pada simpul yang akan dihapus ke dalam sebuah variabel. Lalu hapus/hancurkan simpul yang ditunjuk oleh variabel pointer phapus. Sehingga linked list menjadi seperti di bawah ini:

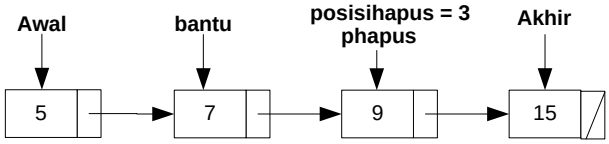


b. Penghapusan di tengah

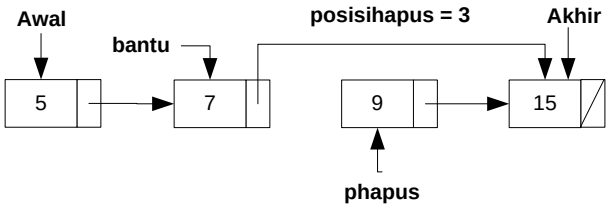
Misalnya akan menghapus simpul ke-3.



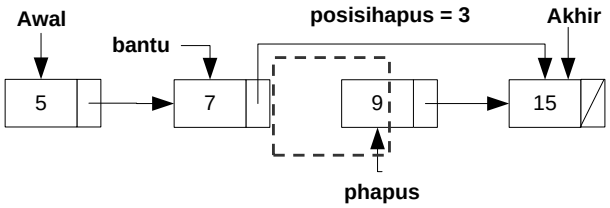
Kemudian cari posisi simpul sebelum posisi simpul yang akan dihapus, yang dilakukan oleh variabel pointer bernama **bantu**.



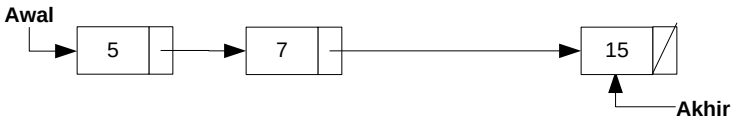
Kemudian field next dari simpul yang ditunjuk oleh pointer bantu harus menunjuk ke alamat yang ditunjuk oleh field next dari simpul yang ditunjuk oleh pointer phapus.



Simpan data yang ada pada simpul yang akan dihapus ke dalam sebuah variabel **elemen**, lalu hapus simpul yang ditunjuk oleh variabel pointer phapus.

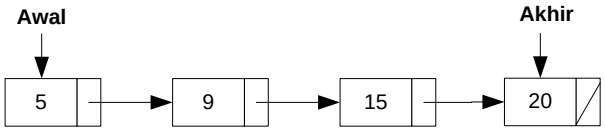


Setelah langkah tersebut, maka linked list seperti gambar di bawah ini:

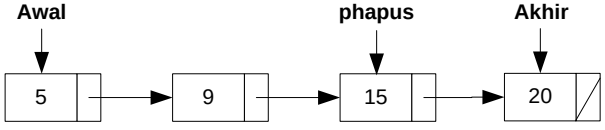


c. Penghapusan di belakang/akhir

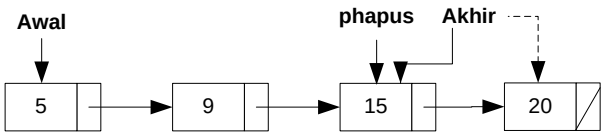
Kondisi linked list memiliki lebih dari 1 data.



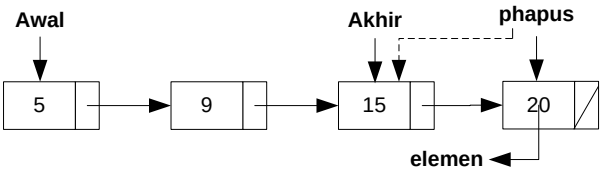
Karena posisi simpul yang akan dihapus adalah simpul terakhir, maka nanti posisi pointer akhir harus pindah ke posisi simpul sebelumnya. Oleh karena itu harus dicari posisi simpul sebelum simpul terakhir, sebut dengan variabel pointer phapus.



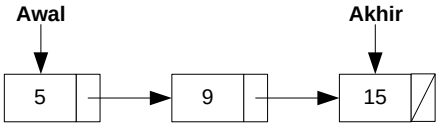
Kemudian pindahkan variabel pointer akhir ke simpul yang ditunjuk oleh varibel pointer phapus.



Variabel pointer phapus pindahkan ke simpul tetangganya.



Simpul yang ditunjuk oleh variabel pointer akhir field next-nya diberi harga nil. Sebelum simpul yang ditunjuk oleh variabel pointer phapus dihapus, maka simpan terlebih dahulu nilainya pada sebuah variabel elemen. Kemudian lakukan penghapusan pada simpul yang ditunjuk variabel pointer phapus, sehingga hasil akhirnya seperti dibawah ini:



4. **Penelusuran Single Linked List**

Penelusuran berarti menampilkan semua data yang ada di dalam linked list dari simpul pertamal sampai dengan simpul yang terakhir. Untuk itu diperlukan suatu variabel pointer pembantu (sebut saja variable pointer bantu) yang akan menelusuri simpul sampai simpul terakhir.

Langkah-langkah penelusuran adalah :

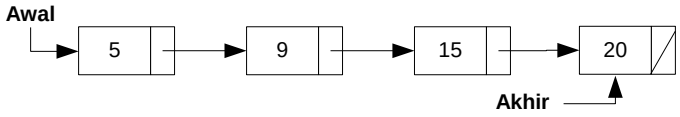
- Salin variabel pointer awal ke variabel pointer bantu.
- Selama variabel pointer bantu tidak NULL/nil, maka tampilkan field info yang ada di simpul yang ditunjuk variabel pointer bantu, kemudian variabel pointer bantu dipindahkan ke simpul berikutnya

5. **Pencarian**

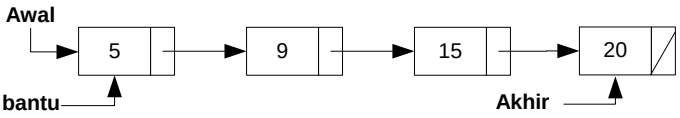
Langkah-langkah untuk melakuan pencarian data dalam linked list tidak begitu beda dengan langkah-langkah pencarian data pada array statis. Karena dengan linked list

tidak dapat diakses secara acak, maka pencarian yang dilakukan adalah pencarian secara sekuensial.

Contoh : Data yang akan dicari = 9



Berarti diperlukan variabel pointer bantuan (bantu) untuk menelusuri simpul pertama sampai dengan simpul terakhir untuk mendapatkan data yang dicari.

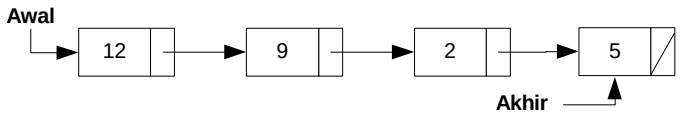


Setelah simpul ditelusuri data 9 akan ditemukan pada simpul yang ke 2.

6. **Pengurutan**

Langkah pengurutan data dalam linked list sama saja dengan pengurutan data dalam array. Berikut ini adalah implementasi pengurutan data dalam linked list dengan algoritma bubble sort.

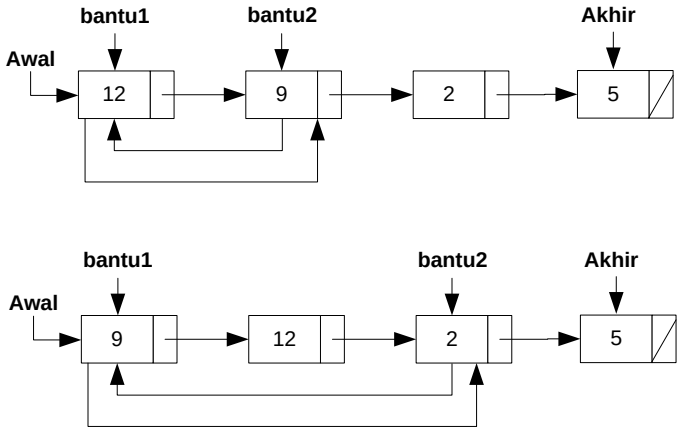
Contoh : Pengurutan menggunakan buble sort secara ascending

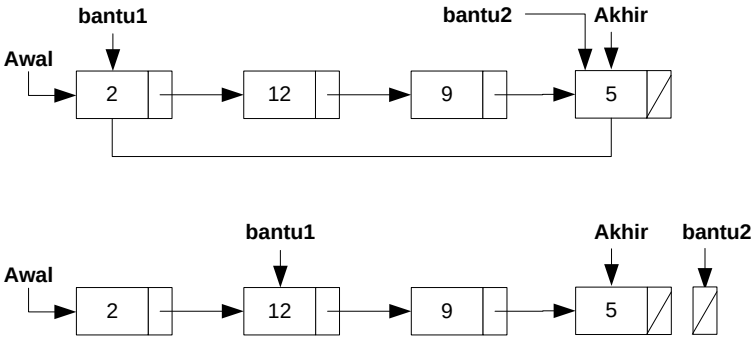


Berikan variabel pointer bantu1 yang menunjuk ke simpul awal dan variabel pointer bantu2 ditempatkan di tetangga dari simpul yang ditunjuk oleh variabel bantu1.

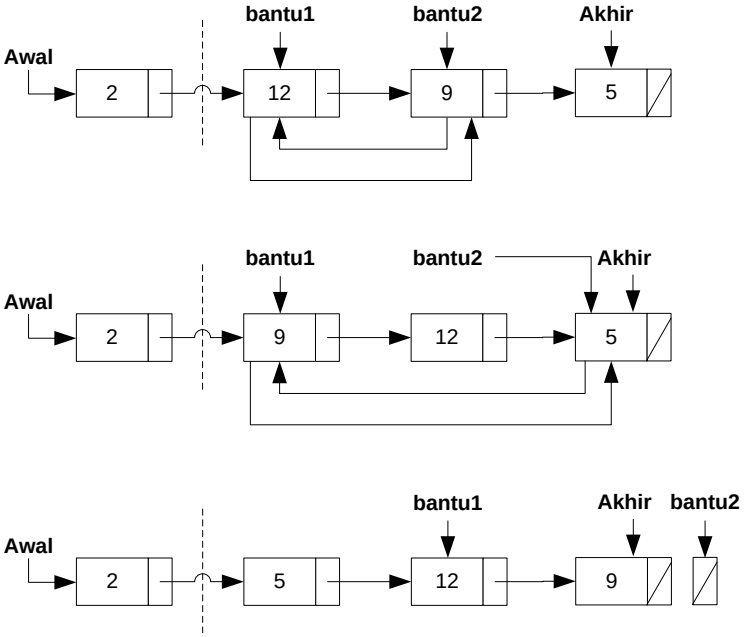
Penyusunan data acak menjadi data tersusun untuk data di atas akan mengalami 3 tahap, karena jumlah data berjumlah 4 data.

Tahap I :

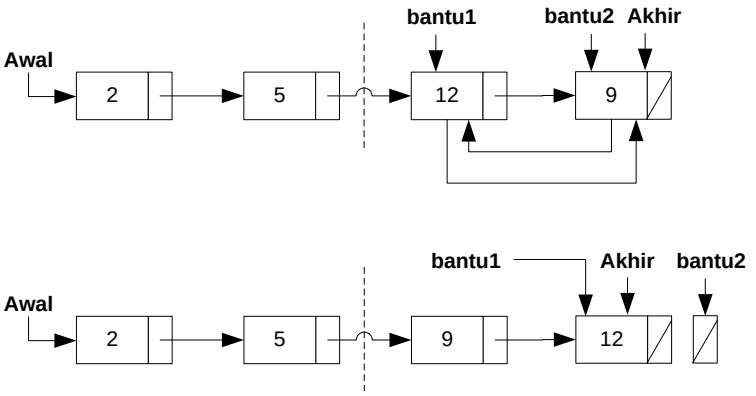




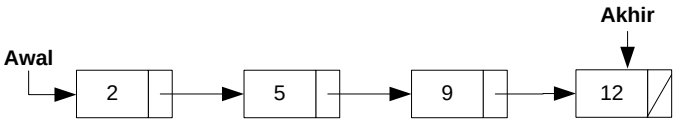
II :



III :



Jadi data terurut secara ascending sebagai berikut:



7. **Penghancuran**

Penghancuran berarti membebaskan memori dan mengembalikannya ke sistem operasi untuk dikelola.

Langkah-langkahnya:

1. Tempatkan variabel pointer bantuan (phapus) di simpul pertama
2. Pindahkan variabel pointer awal ke simpul tetangga kanannya.
3. Hapus simpul yang ditunjuk oleh variabel pointer phapus
4. Tempatkan kembali variabel pointer phapus ke simpul yang ditunjuk oleh variabel pointer awal
5. Lakukan langkah 2 – 4 berulang-ulang sampai list kosong.