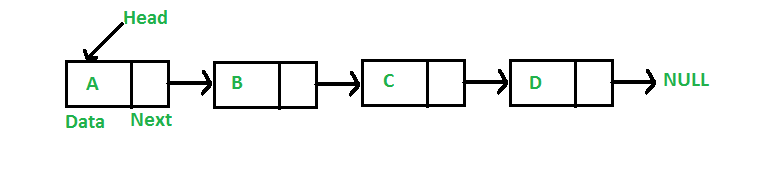
**RANGKUMAN LINKED LIST**

Pengertian Linked List

Linked list adalah strukur data linier berbentuk rantai simpul di mana setiap simpul menyimpan 2 item, yaitu nilai data dan pointer ke simpul elemen berikutnya. Berbeda dengan array, elemen linked list tidak ditempatkan dalam alamat memori yang berdekatan melainkan elemen ditautkan menggunakan pointer.



Simpul pertama dari linked list disebut sebagai **head** atau simpul kepala. Apabila linked list berisi elemen kosong, maka nilai pointer dari head menunjuk ke NULL. Begitu juga untuk pointer berikutnya dari simpul terakhir atau simpul ekor akan menunjuk ke NULL.

Ukuran elemen dari linked list dapat bertambah secara dinamis dan mudah untuk menyisipkan dan menghapus elemen karena tidak seperti array, kita hanya perlu mengubah pointer elemen sebelumnya dan elemen berikutnya untuk menyisipkan atau menghapus elemen.

Linked list biasanya digunakan untuk membuat file system, adjacency list, dan hash table.

**Jenis Jenis Linked List**

Secara umum, linked list dapat dibagi ke dalam 4 jenis, yakni: **Singly linked list**, **Doubly linked list**, **Circular linked list**, dan **Circular doubly linked list**.

**1. Singly linked list**

Singly linked list adalah linked list unidirectional. Jadi, kita hanya dapat melintasinya dalam satu arah, yaitu dari simpul kepala ke simpul ekor.

**2. Doubly linked list**

Doubly linked list adalah linked list bidirectional. Jadi, kita bisa melintasinya secara dua arah. Tidak seperti singly linked list, simpul doubly linked list berisi satu pointer tambahan yang disebut previous pointer. Pointer ini menunjuk ke simpul sebelumnya.

**3. Circular linked list**

Circular linked list adalah linked list unidirectional. Kita hanya dapat melintasinya dalam satu arah. Tetapi jenis linked list ini memiliki simpul terakhir yang menunjuk ke simpul kepala. Jadi saat melintas, kita harus berhati-hati dan berhenti saat mengunjungi kembali simpul kepala.

**4. Circular doubly linked list**

Circular doubly linked list adalah gabungan dari Doubly linked list dan Circular linked list. Seperti Doubly linked list, linked list ini memiliki pointer tambahan yang disebut previous pointer, dan mirip dengan Circular linked list, simpul terakhirnya menunjuk pada simpul kepala. Jenis linked list ini adalah bidirectional. Jadi, kita bisa melintasinya dua arah.

**Karakteristik Linked List**

Sebuah linked list memiliki beberapa karakteristik sebagai berikut:

* Linked list menggunakan memori tambahan untuk menyimpan link (tautan)
* Untuk inisialiasi awal linked list, kita tidak perlu tahu ukuran dari elemen.
* Linked list umumnya dapat digunakan untuk mengimplementasikan struktur data lain seperti stack, queue, ataupun graf
* Simpul pertama dari linked list disebut sebagai Head.
* Pointer setelah simpul terakhir selalu bernilai NULL
* Dalam struktur data linked list, operasi penyisipan dan penghapusan dapat dilakukan dengan mudah
* Tiap-tiap simpul dari linked list berisi pointer atau tautan yang menjadi alamat dari simpul berikutnya
* Linked list bisa menyusut atau bertambah kapan saja dengan mudah.

**Operasi-operasi pada Linked List**

Ada beberapa operasi yang bisa kita lakukan pada struktur data linked list. Misalnya, operasi insertion yaitu tindakan menambahkan elemen baru ke linked list.

Berikut adalah daftar operasi dasar pada linked list:

* **Traversal** - mengakses setiap elemen dari linked list
* **Insertion** - menambahkan elemen baru ke linked list
* **Deletion** - menghapus elemen yang ada
* **Searching** - menemukan simpul pada linked list
* **Sorting** - mengurutkan simpul dari struktur linked list

**Fungsi dan Kegunaan Linked List**

Adapun fungsi dan kegunaan linked list adalah sebagai berikut:

* Linked list dapat digunakan untuk mengimplementasikan struktur data lain seperti [stack](https://www.trivusi.web.id/2022/07/struktur-data-stack.html), [queue](https://www.trivusi.web.id/2022/07/struktur-data-queue.html), graf, dll.
* Digunakan untuk melakukan operasi aritmatika pada bilangan long integer
* Dipakai untuk representasi matriks rongga.
* Digunakan dalam alokasi file yang ditautkan.
* Membantu dalam manajemen memori.

Penerapan linked list banyak ditemui dalam beberapa kasus berikut:

* Linked list digunakan dalam penjadwalan Round-Robin untuk melacak giliran dalam permainan multi-pemain.
* Digunakan dalam aplikasi penampil gambar. Gambar sebelumnya dan berikutnya ditautkan, sehingga dapat diakses oleh tombol prev dan next.
* Dalam playlist musik, lagu yang sedang diputar ditautkan ke lagu sebelumnya dan berikutnya.

**Kelebihan Linked List**

Berikut ini dalah keunggulan menggunakan linked list:

* **Struktur data dinamis**: Linked list adalah himpunan dinamis sehingga dapat bertambah dan menyusut saat runtime dengan mengalokasikan dan membatalkan alokasi memori. Jadi kita tidak perlu memberikan ukuran awal dari linked list.
* **Tidak boros memori**: Dalam linked list, pemanfaatan memori yang efisien dapat dicapai karena ukuran linked list bertambah atau berkurang pada runtime sehingga tidak ada pemborosan memori dan tidak perlu mengalokasikan memori sebelumnya.
* **Implementasi**: Struktur data linier seperti stack dan queue seringkali mudah diimplementasikan menggunakan linked list.
* **Operasi penyisipan dan penghapusan**: Operasi penyisipan dan penghapusan cukup mudah dalam linked list. Kita tidak perlu menggeser elemen setelah operasi penyisipan atau penghapusan elemen, hanya alamat yang ada di pointer berikutnya saja yang perlu diperbarui.

**Kelemahan Linked List**

Adapun kelemahan menggunakan linked list adalah sebagai berikut:

* **Penggunaan memori**: Linked list memerlukan lebih banyak memori dibandingkan dengan array. Karena dalam linked list, pointer juga perlu menyimpan alamat elemen berikutnya dan membutuhkan memori tambahan untuk dirinya sendiri.
* **Traversal**: Dalam traversal, linked list lebih banyak memakan waktu dibandingkan dengan array. Akses langsung ke elemen tidak bisa dilakukan pada linked list seperti array yang dapat akses elemen berdasarkan indeks. Untuk mengakses sebuah simpul pada posisi n dari linked list, kita harus melintasi semua simpul sebelumnya.
* **Reverse Traversing**: Dalam single linked list, reverse traversing tidak dimungkinkan, tetapi dalam kasus double-linked list, ini dapat dimungkinkan karena berisi pointer ke node yang terhubung sebelumnya dengan setiap node. Untuk melakukannya, diperlukan memori tambahan untuk pointer sebelumnya sehingga ada pemborosan memori.
* **Akses Acak**: Akses acak tidak bisa dilakukan dalam linked list karena alokasi memorinya yang dinamis.