Nama : Rhainy Aulia

Kelas : TI-23-PA2

NPM : 232310034

**Algoritma Pemrograman dan Struktur Data**

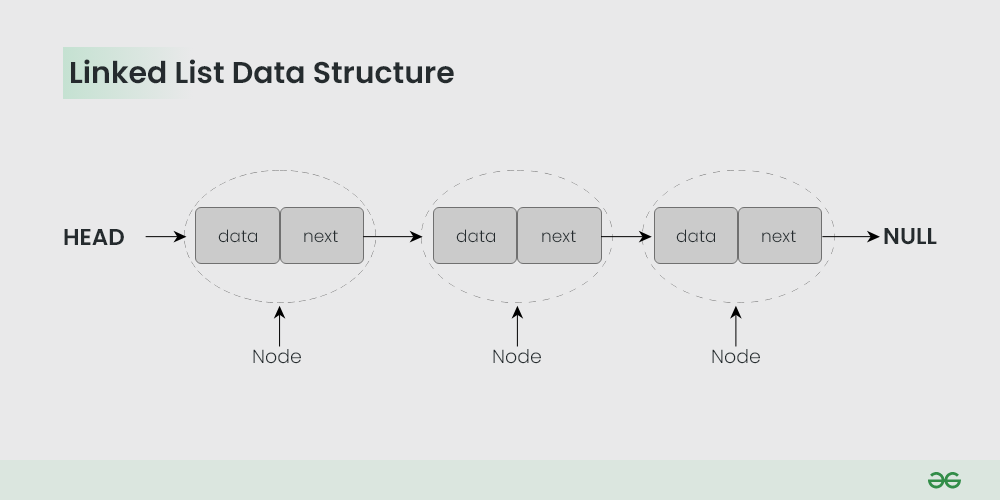
**Rangkuman**

**Linked List**

1. **Pengertian**

Linked list adalah struktur data linier berbentuk rantai simpul di mana setiap simpul menyimpan 2 item, yaitu nilai data dan pointer ke simpul elemen berikutnya.

Tidak seperti array, elemen-elemen dari daftar tertaut tidak ditempatkan pada alamat memori yang berdekatan; sebaliknya, elemen-elemen tersebut dihubungkan menggunakan pointer.

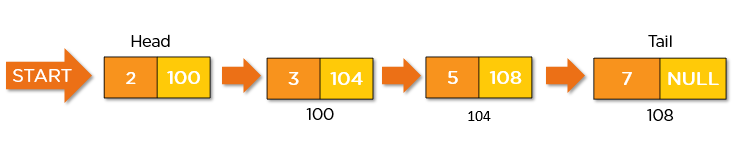


1. **Jenis**

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.

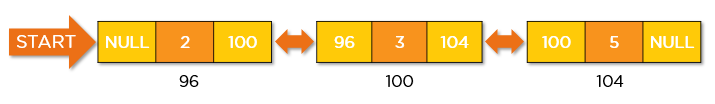
1. Singly linked list

Singly linked list adalah linked list unidirectional. Jadi, kita hanya dapat melintasinya dalam satu arah, yaitu dari simpul kepala ke simpul ekor. Suatu linked list dikatakan single linked list apabila hanya ada satu pointer yang menghubungkan setiap node (satu arah “next”).



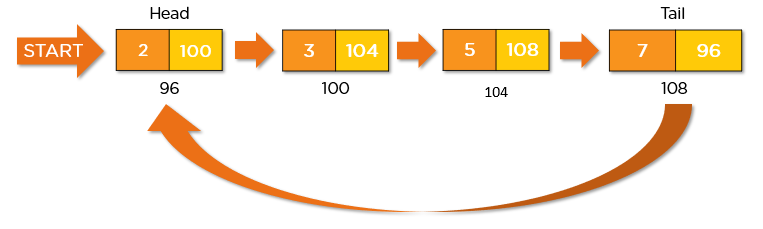
1. Doubly linked list

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.



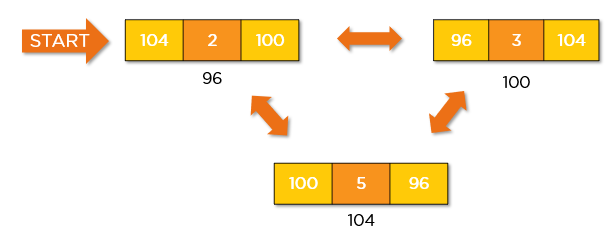
1. Circular linked list

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.

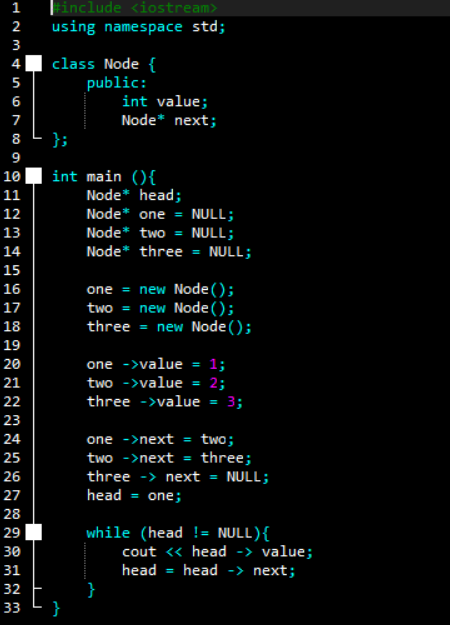


1. Circular doubly linked list

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.



1. **Karakteristik**
   1. Linked list menggunakan memori tambahan untuk menyimpan link (tautan)
   2. Untuk inisialisasi awal linked list, kita tidak perlu tahu ukuran dari elemen.
   3. Linked list umumnya dapat digunakan untuk mengimplementasikan struktur data lain seperti stack, queue, ataupun graf
   4. Simpul pertama dari linked list disebut sebagai Head.
   5. Pointer setelah simpul terakhir selalu bernilai NULL
   6. Dalam struktur data linked list, operasi penyisipan dan penghapusan dapat dilakukan dengan mudah
   7. Tiap-tiap simpul dari linked list berisi pointer atau tautan yang menjadi alamat dari simpul berikutnya
   8. Linked list bisa menyusut atau bertambah kapan saja dengan mudah.
2. **Operasi-operasi**
3. Traversal - mengakses setiap elemen dari linked list
4. Insertion - menambahkan elemen baru ke linked list
5. Deletion - menghapus elemen yang ada
6. Searching - menemukan simpul pada linked list
7. Sorting - mengurutkan simpul dari struktur linked list
8. **Fungsi**
9. Linked list dapat digunakan untuk mengimplementasikan struktur data lain seperti stack, queue, graf, dll.
10. Digunakan untuk melakukan operasi aritmatika pada bilangan long integer
11. Dipakai untuk representasi matriks rongga.
12. Digunakan dalam alokasi file yang ditautkan.
13. Membantu dalam manajemen memori.
14. **Kegunaan**
15. Linked list digunakan dalam penjadwalan Round-Robin untuk melacak giliran dalam permainan multi-pemain.
16. Digunakan dalam aplikasi penampil gambar. Gambar sebelumnya dan berikutnya ditautkan, sehingga dapat diakses oleh tombol prev dan next.
17. Dalam playlist musik, lagu yang sedang diputar ditautkan ke lagu sebelumnya dan berikutnya.
18. **Kelebihan**
19. 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.
20. 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.
21. Implementasi: Struktur data linier seperti stack dan queue seringkali mudah diimplementasikan menggunakan linked list.
22. 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.
23. **Kelemahan**
24. 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.
25. 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.
26. 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.
27. Akses Acak: Akses acak tidak bisa dilakukan dalam linked list karena alokasi memorinya yang dinamis.
28. **Contoh Kode**

 Output : 123

Sumber :

1. <https://www.trivusi.web.id/2022/07/struktur-data-linked-list.html>
2. <https://www.geeksforgeeks.org/data-structures/linked-list/>
3. <https://youtu.be/Zgzoe8jjidk?feature=shared>