|  |  |  |
| --- | --- | --- |
| Nama | : | Imron |
| NPM | : | 232310021 |
| Kelas | : | TI-23-PA1 |
| Mata Kuliah | : | Lab Algoritma Pemrograman dan Struktur Data |
| Tugas | : | Tugas 14 Rangkuman Linked List |

1. Pengertian Linked List

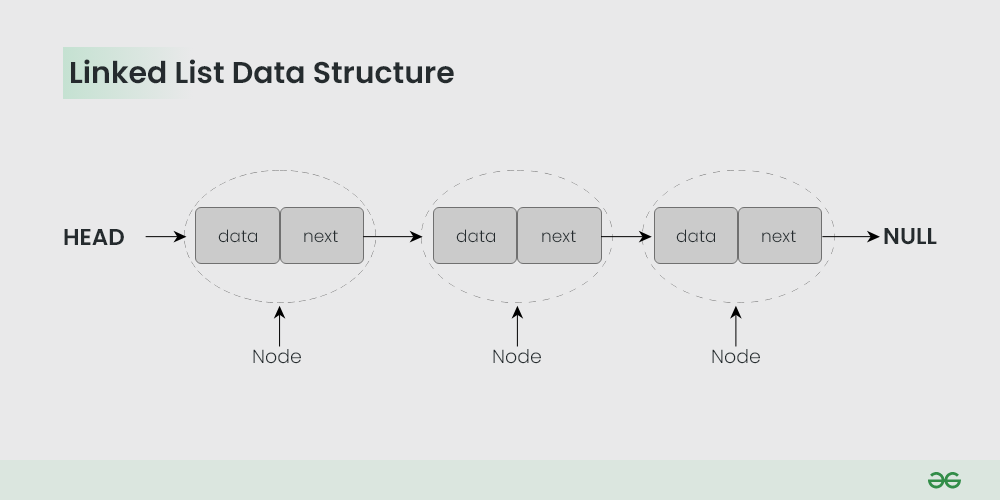
Linked list adalah struktur data yang menyimpan kumpulan data secara linear. Data dalam linked list disusun dalam urutan linier, artinya data pertama terletak di awal, data kedua terletak di belakang data pertama, dan seterusnya.

Linked list terdiri dari kumpulan simpul (node). Setiap simpul terdiri dari dua bagian, yaitu data dan pointer. Data adalah nilai yang akan disimpan dalam linked list. Pointer adalah variabel yang menyimpan alamat simpul berikutnya.

Pointer digunakan untuk menghubungkan satu simpul dengan simpul lain. Pointer pada simpul pertama akan menyimpan alamat simpul kedua. Pointer pada simpul kedua akan menyimpan alamat simpul ketiga, dan seterusnya.

1. Struktur Dasar Linked List

Struktur dasar dari linked list biasa disebut sebagai single linked list



Gambar 1 Single Linked List

(Sumber : https://media.geeksforgeeks.org/wp-content/cdn-uploads/20230726162542/Linked-List-Data-Structure.png)

Setiap simpul (node) dari suatu linked list terdiri atas dua bagian, yaitu :

1. Data, merupakan nilai yang tersimpan dalam node
2. Next (pointer), berisi alamat dari simpul (node) selanjutnya yang ditunjuk.

Pada gambar di atas, Node pertama (paling kiri) adalah simpul pertama dalam linked list disebut sebagai head. Node pertama menyimpan data dan pointer yang mengarah ke Node kedua (tengah). Node kedua menyimpan data dan pointer yang mengarah ke Node ketiga (kanan). Node ketiga menyimpan data dan pointer yang mengarah ke NULL, yang berarti linked list berakhir di Node ketiga disebut sebagai tail (ekor).

1. Jenis-Jenis Linked List
2. Single Linked List



Gambar 2 Single Linked List

(Sumber : <https://media.geeksforgeeks.org/wp-content/uploads/20220816144425/LLdrawio.png>)

Single linked list adalah jenis linked list yang paling sederhana. Singly linked list terdiri dari serangkaian node yang saling terhubung satu sama lain melalui pointer. Setiap node dalam singly linked list memiliki data dan pointer yang mengarah ke node berikutnya.

1. Double Linked List



Gambar 3Double Linked List

(Sumber : <https://media.geeksforgeeks.org/wp-content/cdn-uploads/gq/2014/03/DLL1.png>)

Double linked list adalah jenis linked list yang lebih kompleks daripada singly linked list. Doubly linked list memiliki dua pointer, yaitu pointer yang mengarah ke node berikutnya dan pointer yang mengarah ke node sebelumnya.

1. Circular Linked List

Circular Linked List

Gambar 4 Circular Linked List

(Sumber : https://media.geeksforgeeks.org/wp-content/uploads/CircularLinkeList.png)

Circular linked list adalah jenis linked list yang ujungnya terhubung kembali ke dirinya sendiri. Hal ini berarti bahwa node terakhir dalam circular linked list akan memiliki pointer yang mengarah ke node pertama.

1. Operasi pada Linked List

Berikut adalah daftar operasi dasar pada linked list:

1. Traversal adalah operasi untuk mengakses setiap elemen dari linked list. Traversal dapat dilakukan dengan dua cara, yaitu, iterasi maju dan iterasi mundur.
2. Insertion adalah operasi untuk menambahkan elemen baru ke linked list. Insertion dapat dilakukan di awal, di akhir, atau di tengah linked list.
3. Deletion adalah operasi untuk menghapus elemen dari linked list. Deletion dapat dilakukan di awal, di akhir, atau di tengah linked list.
4. Searching adalah operasi untuk menemukan simpul pada linked list.
5. Sorting adalah operasi untuk mengurutkan simpul dari struktur linked list. Sorting dapat dilakukan dengan berbagai algoritma, seperti bubble sort, selection sort, dan insertion sort.
6. Implementasi Linked List
7. Linked list dapat digunakan untuk mengimplementasikan struktur data lain seperti stack, queue, graf, dll.
8. Digunakan dalam aplikasi penampil gambar. Gambar sebelumnya dan berikutnya ditautkan, sehingga dapat diakses oleh tombol prev dan next. Undo/Redo Functionality
9. Dalam playlist musik, lagu yang sedang diputar ditautkan ke lagu sebelumnya dan berikutnya. Undo/Redo Functionality
10. Kelebihan dan Kekurangan

Kelebihan :

1. 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.
2. Efisiensi 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.
3. Mudah diimplementasikan : Struktur data linier seperti stack dan queue seringkali mudah diimplementasikan menggunakan linked list.

Kekurangan :

1. Memerlukan banyak memori : Linked list memerlukan lebih banyak memori dibandingkan dengan array. Hal ini disebabkan karena pointer juga perlu menyimpan alamat elemen berikutnya dan membutuhkan memori tambahan untuk dirinya sendiri.
2. Kurang efisien untuk 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.
3. Reverse traversing hanya dimungkinkan pada double linked list : 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.
4. Akses acak tidak dimungkinkan : Akses acak tidak bisa dilakukan dalam linked list karena alokasi memorinya yang dinamis.
5. Pebedaan Linked List dan Array

Perbedaan Antara Array dan Linked List

Array dan Linked List adalah dua cara berbeda untuk menyimpan dan mengatur data dalam pemrograman.

Pertama, Array memerlukan alokasi memori kontiguitas, yang berarti elemen data harus disusun secara berurutan dalam memori. Ini membuat penambahan atau penghapusan elemen menjadi lebih sulit, karena kita perlu menggeser elemen lain untuk memberi ruang pada posisi yang diinginkan. Selain itu, ukuran Array tetap dan ditentukan saat deklarasi, sehingga sulit untuk mengubah ukuran Array secara dinamis. Meskipun Array memberikan akses elemen yang cepat melalui indeks, namun ada keterbatasan dalam fleksibilitas dan efisiensi ketika datanya sering berubah.

Kedua, Linked List menggunakan pendekatan yang berbeda. Linked List tidak memerlukan alokasi memori kontiguitas, sehingga setiap elemen data dapat ditempatkan di lokasi memori yang berbeda dan dihubungkan melalui referensi. Ini membuat penambahan dan penghapusan elemen menjadi lebih efisien, karena kita hanya perlu mengatur ulang referensi antar Node.

Ukuran Linked List dapat berubah selama runtime, sehingga memungkinkan penambahan atau penghapusan elemen dengan mudah. Namun, Linked List memerlukan pencarian melalui elemen sebelumnya untuk mengakses data, sehingga akses elemen membutuhkan waktu lebih lama. Meskipun Linked List memiliki keterbatasan, namun ini merupakan pilihan yang fleksibel untuk situasi di mana data sering berubah atau ketika ukuran data tidak diketahui sebelumnya.