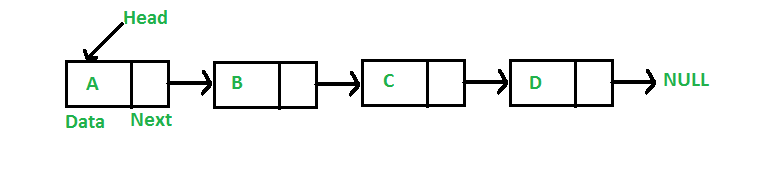
**LINKED LIST**

**Pengertian Linked List**

Linked list adalah [strukur data](https://www.trivusi.web.id/2022/06/mengenal-struktur-data.html" \t "_blank) linier berbentuk rantai simpul di mana setiap simpul menyimpan 2 item, yaitu **nilai data** dan **pointer ke simpul elemen berikutnya**. Berbeda dengan [array](https://www.trivusi.web.id/2022/07/struktur-data-array.html), 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.

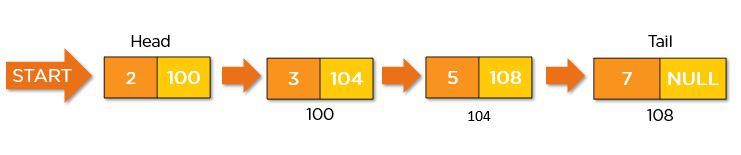
Linked list biasanya digunakan untuk membuat file system, adjacency list, dan hash table.

## 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**.

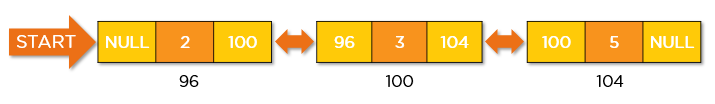
### 1. Singly linked list

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



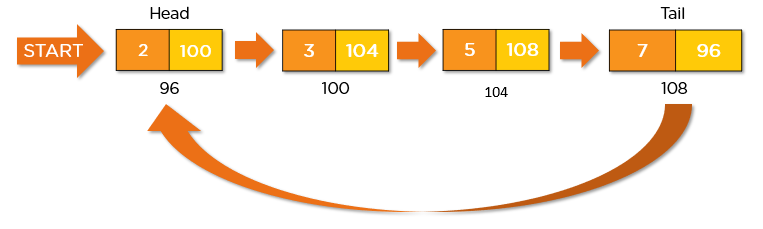
### 2. Doubly linked list

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.



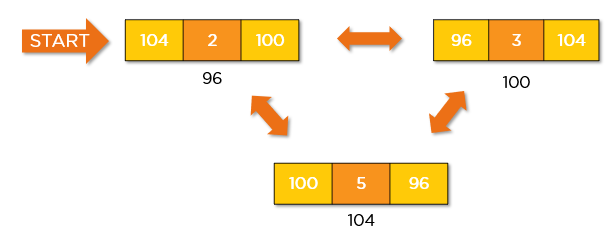
### 3. Circular linked list

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
* #include <stdio.h>  
  #include <iostream>  
  #include <conio.h>  
  #include <stdlib.h>  
    
  **using** **namespace** std;  
    
  **struct** TNode{  
   **int** data;  
   TNode \*next;  
  };  
    
  TNode \*head, \*tail;  
    
  **void** init (){  
   head = NULL;  
   tail = NULL;  
  }  
    
  **int** isEmpty(){  
   **if**(tail==NULL)**return** 1;  
   **else** **return** 0;  
  }  
    
  **void** insertDepan(**int** databaru)  
  {  
   TNode \*baru;  
   baru = **new** TNode;  
   baru->data = databaru;  
   baru->next = NULL;  
   **if**(isEmpty()==1)  
   {  
   head=tail=baru;  
   tail->next=NULL;  
   }**else**  
   {  
   baru->next = head;  
   head=baru;  
   }  
   cout<<"Data masuk\n";  
  }  
    
  **void** tampil(){  
   TNode \*bantu;  
   bantu=head;  
   **if**(isEmpty()==0)  
   {  
   **while**(bantu!=NULL)  
   {  
   cout<<bantu->data<<" ";  
   bantu=bantu->next;  
   }  
   }**else** cout <<"Masih kosong\n";  
  }  
    
  **void** hapusDepan()  
  {  
    
   TNode \*hapus;  
   **int** d;  
   **if** (isEmpty()==0)  
   {  
   **if**(head!=tail)  
   {  
   hapus=head;  
   d=hapus->data;  
   head=head->next;  
   **delete** hapus;  
   }**else**  
   {  
   d=tail->data;  
   head=tail=NULL;  
   }  
   cout<<d<<"terhapus";  
   }**else** cout<<"Masih kosong\n";  
  }  
    
  **void** clear()  
  {  
   TNode \*bantu, \*hapus;  
   bantu =head;  
   **while**(bantu!=NULL)  
   {  
   hapus=bantu;  
   bantu=bantu->next;  
   **delete** hapus;  
   }  
   head = NULL;  
   printf("clear");  
  }  
    
  main()  
  {  
   **int** pil, databaru;  
   **do**  
   {  
   system("cls");  
   cout<<endl;  
   cout<<" ============================"<<endl;  
   cout<<" = PROGRAM LINKED LIST ="<<endl;  
   cout<<" ============================"<<endl;  
   cout<<" = 1. Insert Depan ="<<endl;  
   cout<<" = 2. Delete Depan ="<<endl;  
   cout<<" = 3. Tampil Data ="<<endl;  
   cout<<" = 4. Clear ="<<endl;  
   cout<<" = 5. Exit ="<<endl;  
   cout<<" ============================"<<endl;  
   cout<<" Masukan Pilihan : ";cin>>pil;  
   **switch** (pil)  
   {  
   **case** 1: system("cls");{  
   cout<<"Masukan Data = ";cin>>databaru;  
   insertDepan(databaru);  
   **break**;  
   }  
   **case** 2: system("cls");{  
   hapusDepan();  
   **break**;  
   }  
   **case** 3: system("cls");{  
   tampil();  
   **break**;  
   }  
   **case** 4: system("cls");{  
   clear();  
   **break**;  
   }  
   **case** 5: system("cls");{  
   **return** 0;  
   **break**;  
   }  
   **default** : system("cls");  
   {  
   cout<<"\n Maaf, Pilihan yang anda pilih tidak tersedia!";  
   }  
   }  
   getch();  
   }  
   **while** (pil!=7);  
  }