Double linked list

Struktur Data

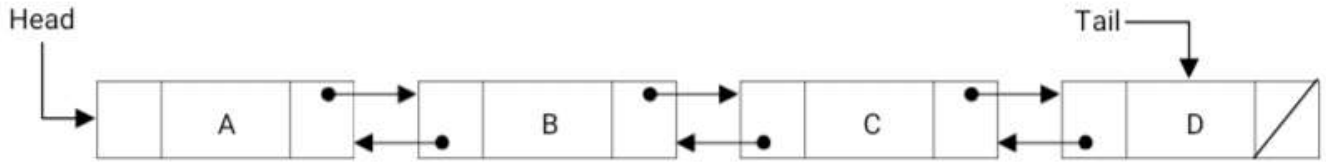
Resume

- Linked list yang kita pelajari sebelumnya hanya mempunyai sebuah pointer pada setiap simpulnya. Hal ini merupakan kelemahan bahwa linked list tersebut hanya bisa dibaca dalam satu arah saja, yaitu dari kiri ke kanan. Hal yang seperti ini kurang cepat jika kita ingin mencari data di dalam linked list.
- Untuk itulah kita memerlukan linked list yang setiap simpulnya mempunyai 2 buah pointer, dengan pointer pertama menunjuk ke simpul sebelumnya (sebelah kiri) dan pointer kedua menunjuk ke simpul sesudahnya (disebelah kanan). Hal ini akan mempermudah pembacaan (bisa dilakukan dari kiri ke kanan dan dari kanan ke kiri), begitu juga untuk penambahan simpul baru dan penghapusan simpul.
- Linked list seperti ini disebut dengan Seranai Beranai Ganda (Double linkeded list).

Double Linked list

- Secara garis besar Double linked list adalah linked list yang memiliki dua buah pointer yang menunjuk ke simpul sebelumnya (Prev) dan yang menunjuk ke simpul sesudahnya (Next).

Gambar berikut menunjukkan gambaran Double Linked list:



Gambar 1 Double Linked list dengan Empat Simpul

Deklarasi Dalam Pascal

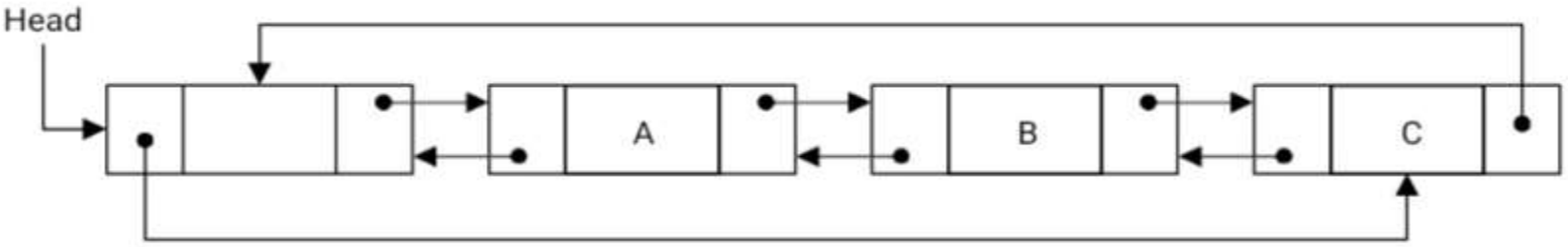
- Type
  - PSimpul = ^Simpul
  - Simpul = Record
    - Info : Tipe Data;
    - Prev : PSimpul;
    - Next : PSimpul;
- End;
- Var
  - Head, Tail : PSimpul;

Beberapa hal yang harus diketahui mengenai Double linked list

- Double Linked list selalu memiliki pointer petunjuk yang selalu menunjuk pada awal dari list yang disebut Head.
- Double Linked list juga selalu memiliki pointer petunjuk menunjuk pada akhir dari list yang disebut Tail.
- Setiap simpul yang terbentuk selalu memiliki nilai NIL, kecuali jika simpul tersebut sudah ditunjuk oleh simpul yang lainnya (Double Linked list belum terhubung).
- Posisi simpul terakhir pada Doube linked list selalu bernilai NIL karena ia tidak menunjuk pada simpul yang lainnya, kecuali bentuk circular.
- Operasi yang dapat dilakukan pada Double Linked list diantaranya adalah :
  - Inisialisasi.
  - Menambah Simpul (di Depan, Belakang dan Tengah).
  - Menghapus Simpul (di Depan, Belakang dan Tengah).
  - Membaca isi linked list (Membaca maju dan mundur).

Circular Header Double Linked list

- Jenis linked list ini merupakan jenis double linked list yang memiliki simpul kepala dan tidak mempunyai tail (Head = Tail). Fungsi simpul kepala dapat digunakan untuk menyimpan informasi tambahan pada list. Berikut ini merupakan ilustrasi dari circular header double linked list.

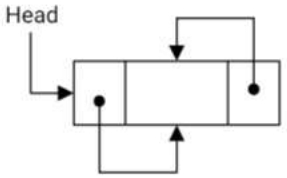


Gambar 2 Circular Double Linked list dengan tiga simpul

Operasi Pada Circular Header Double Linked list

- Ada empat jenis operasi pada linked list, yaitu :
- 1. Inisialisasi

  Proses ini dilakukan untuk mendefinisikan Circular double linked list untuk pertama kalinya atau dengan kata lain ingin membuat Head. Pada proses ini kita menginginkan agar pointer kiri (Prev) dan kanan (Next) dari simpul kepala tidak bernilai NIL, sehingga simpul kepala pada saat inisialisasi bisa kita gambarkan sebagai berikut :

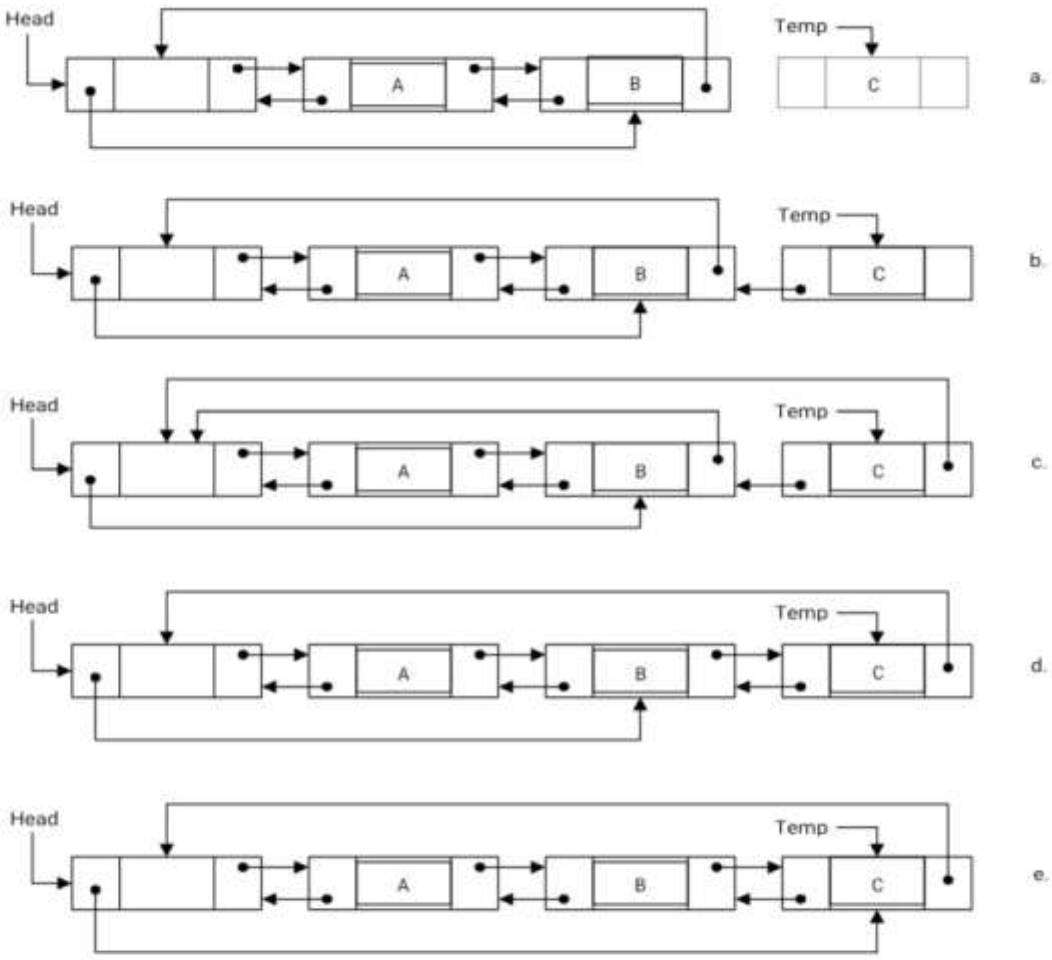


Gambar 3 Inisialisasi Circular Double Linked list

- Procedure inisialisasi dari gambar diatas dapat kita susun sebagai berikut :

- Procedure Inisialisasi(Var Head : PSimpul); Begin
  - New(Head);
  - Head^.Prev := Head; Head^.Next := Head;
- End;

- 2. Menambah Simpul
  Menambah simpul pada double linked list ada tiga macam yaitu menambah di depan, belakang dan tengah, tapi hanya penambahan yang umum dipakai saja oleh double linked list yang akan dibahas yaitu penambahan simpul di belakang. Penambahan di belakang maksudnya menambahkan simpul-simpul baru pada posisi Tail. Gambar 3.4 ini merupakan ilustrasi penambahan simpul di belakang. Selanjutnya disertakan pula procedure untuk menambah simpul baru pada posisi belakang (Tail).



Gambar 4 Ilustrasi Penambahan Simpul pada Circular Header Double Linked List

- Procedure Menambah dari gambar di atas dapat kita susun sebagai berikut :

- Procedure Tambah(Var Head : PSimpul; Elemen : Char); Var Temp : PSimpul;
- Begin
  - New(Temp);
  - Temp^.Info := Elemen;
  - Temp^.Prev := Head^.Prev;
  - Temp^.Next := Head;
  - Head^.Prev^.Next := Temp;
  - Head^.Prev := Temp;
- End;

- 3. Menghapus Simpul

  Operasi menghapus simpul juga ada tiga macam yaitu menghapus simpul di depan, belakang dan tengah. Tapi dengan menggunakan double linked list proses pencarian simpul yang akan dihapus menjadi semakin cepat. Untuk menghapus sebuah simpul diperlukan satu buah tambahan variabel pointer yaitu variabel bantu yang berguna untuk menunjukkan simpul manakah yang akan dihapus.

- Procedure Hapus(Var Head : PSimpul; Elemen : Char); Var Temp : PSimpul;
- Begin
  - If Head^.Next = Head Then { Jika Linked list Masih Kosong } Writeln('Double Linked list Masih Kosong')
  - {Menghapus simpul tengah / }
- Else
  - Begin {Akhir }
    - Temp := Head
    - { Memulai proses pencarian elemen yang akan dihapus }
    - Repeat
      - Temp := Temp^.Next;
    - Until (Temp^.Info=Elemen) or (Bantu = Head);
    - If Temp^.Info = Elemen Then { Jika simpul ketemu }
      - Begin
        - Temp^.Prev^.Next := Temp^.Next;
        - Temp^.Next^.Prev := Temp^.Prev;
        - Dispose(Temp);
        - Temp := Head;
      - End

      Else
      - { Simpul yang akan dihapus tidak ketemu }
      - Writeln('Simpul Tidak di ketemukan !');
    - End;
- End;

- 4. Membaca Isi Double Linked list
  Dengan menggunakan double linked list proses pembacaan simpul dapat dilakukan dua arah tanpa harus dilakukan prosedur membalik simpul. Kita hanya menggunakan satu prosedur bantu yang pointer next-nya kita gunakan jika kita ingin membaca mundur atau pointer prev-nya jika kita ingin melakukan pembacaan secara maju. Procedure membaca isi double linked list selengkapnya adalah sebagai berikut :

- Procedure BacaMaju(Head : PSimpul); Var Temp : PSimpul;
- Begin
  - Temp := Head^.Prev;
  - Repeat
    - Write(Temp^.Info,' '); Temp := Temp^.Prev;
  - Until (Temp=Head);
- End;

- Procedure BacaMundur(Head : PSimpul); Var Temp : PSimpul;
- Begin
  - Temp := Head^.Next;
  - Repeat
    - Write(Temp^.Info,' '); Temp := Temp^.Next;
  - Until (Temp=Head);
- End;

- Latihan :
- Buat ilustrasi penambahan simpul didepan dan tengah
- Buat ilustrasi penghapusan simpul didepan dan tengah