Single Linked List

L HAFIDL ALKHAIR

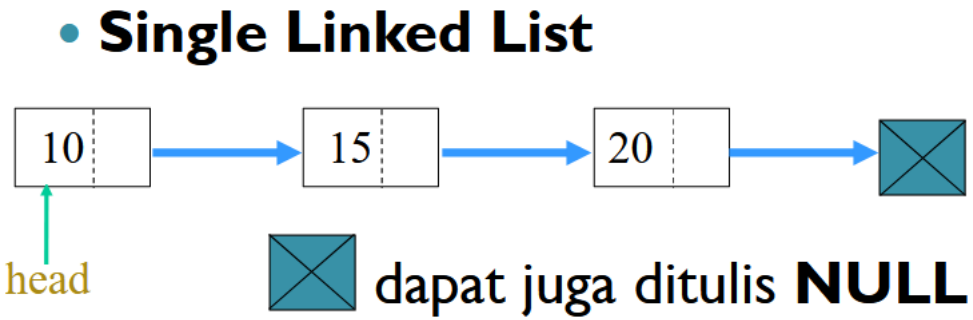
TRKJ 1C

2023903430060

Single Linked List

Single linked list adalah struktur data linear yang terdiri dari sejumlah simpul (node) di mana setiap simpul memiliki dua komponen utama:

- Data yang menyimpan nilai.
- Dan sebuah pointer yang menunjuk ke simpul berikutnya dalam daftar

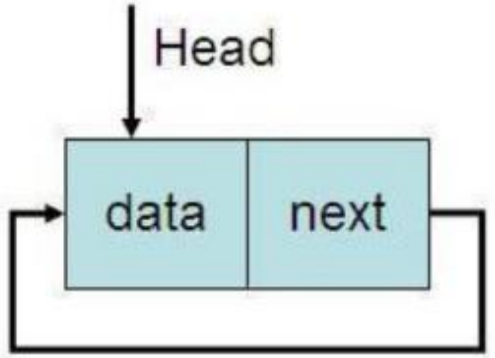


Ilustrasi Single Linked List Circular :
Setiap node pada linked list mempunyai field yang berisi pointer ke node berikutnya, dan juga memiliki field yang berisi data.

Pada akhir linked list, node terakhir akan menunjuk ke node terdepan sehingga linked list tersebut berputar. Node terakhir akan menunjuk lagi ke head.

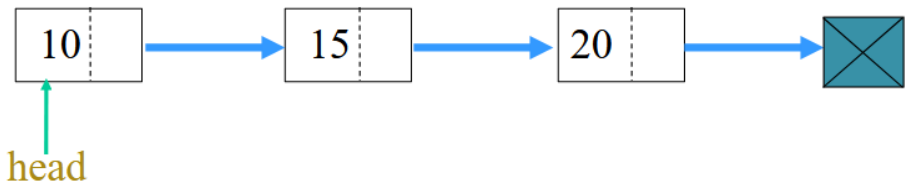
Deklarasi dan Node Baru SLLC

- #include “stdio.h”
  typedef struct Tnode
  { int data;
  Tnode *next;
  };
  Tnode *bantu; //mendeklarasikan pointer bantu
  Tnode *baru; //mendeklarasikan pointer baru
  Tnode *head; //mendeklarasikan pointer head
  Tnode *tail; //mendeklarasikan pointer tail
  void main()
  {
  head=new Tnode;
  head ->next=head;
  }

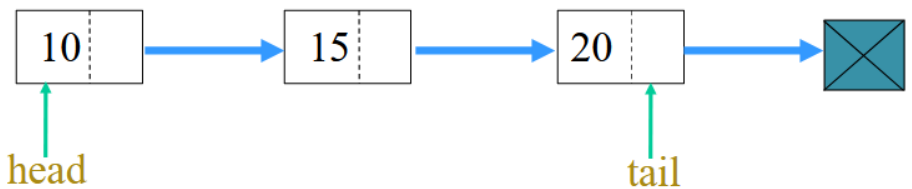


Jenis Single Linked List

- Single linked list dengan HEAD



- Single linked list dengan HEAD dan TAIL



Deklarasi Single Linked List

Deklarasi Node

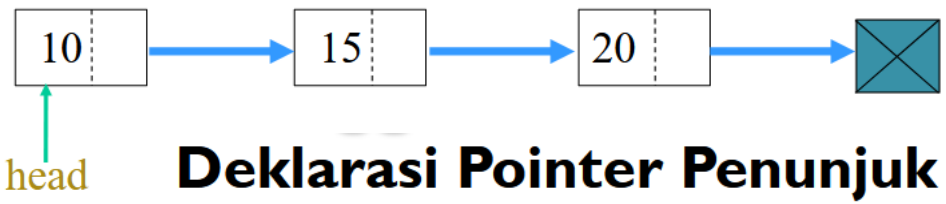
```
typedef struct TNode
{
        int data;
        TNode *next;
};
```

Penjelasan:

- Pembuatan struct bernama TNode yang berisi 2 field, yaitu field data bertipe integer dan field next yang bertipe pointer dari TNode
- Setelah pembuatan struct, buat variabel haed yang bertipe pointer dari TNode yang berguna sebagai kepala linked list.

Single Linked List menggunakan HEAD

- Dibutuhkan satu buah variabel pointer: **head**
- Head akan selalu menunjuk pada **node pertama**



Deklarasi Pointer Penunjuk Kepala Single Linked List

Manipulasi linked list tidak bisa dilakukan langsung ke node yang dituju, melainkan harus menggunakan suatu pointer penunjuk ke node pertama dalam linked list (dalam hal ini adalah head).

Deklarasinya sebagai berikut:

TNode *head;

Single Linked List dengan Head (Penambahan data dari depan)

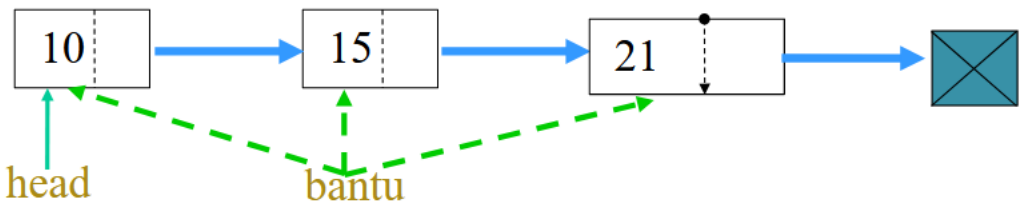
```
void  insertDepan(int databaru)
{
  TNode *baru;
  baru = new TNode;
  baru->data = databaru;
  baru->next = NULL;
  if(isEmpty()==1){
    head=baru;
    head->next = NULL;
  }
  else {
    baru->next = head;
    head = baru;
  }
  cout<<"Data masuk\n";
}
```

Single Linked List dengan Head (Penambahan data dari belakang)

```
void  insertBelakang (int databaru){
  TNode *baru,*bantu;
  baru = new TNode;
    baru->data = databaru;
  baru->next = NULL;
  if(isEmpty()==1){
    head=baru;
    head->next = NULL;
  }
  else {
    bantu=head;
    while(bantu->next!=NULL){
       bantu=bantu->next;
    }
    bantu->next = baru;
  }
  cout<<"Data masuk\n";
}
```

Single Linked List dengan Head (menampilkan data)

```
void  tampil(){
    TNode *bantu;
    bantu = head;
    if(isEmpty()==0){
          while(bantu!=NULL){
                cout<<bantu->data<<" ";
                bantu=bantu->next;
          }
          cout<<endl;
    } else cout<<"Masih kosong\n";
}
```



Function di atas digunakan untuk menampilkan semua isi list, di mana linked list ditelusuri satu-persatu dari awal node sampai akhir node. Penelusuran ini dilakukan dengan menggunakan suatu **pointer bantu**, karena pada prinsipnya pointer head yang menjadi tanda awal list tidak boleh berubah/berganti posisi.

- Function untuk mengetahui kosong tidaknya SLLC

```
int isEmpty(){
if(tail == NULL)
return 1;
else
return 0;
}
```

Kelebihan & Kekurangan

- Linked List memiliki kelebihan sebagai berikut :

  Penambahan elemen tidak terbatas Memungkinkan untuk dihapus

  Kekurangan :

  Hanya bisa diakses secara sekuensial Memerlukan memori dalam jumlah yang besar, untuk menyimpan data yang besar juga.

Function untuk menghapus semua elemen Linked List

```
void  clear(){
      TNode *bantu,*hapus;
      bantu = head;
      while(bantu!=NULL){
         hapus = bantu;
         bantu = bantu->next;
         delete hapus;
      }
      head = NULL;
}
```