MODUL IX-A

TEMA

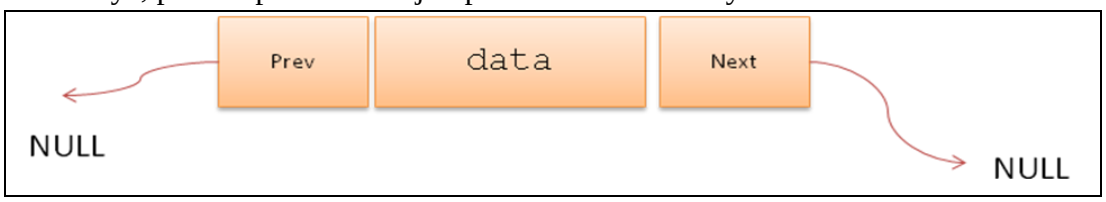
Double Link List Non Circular dengan HEAD

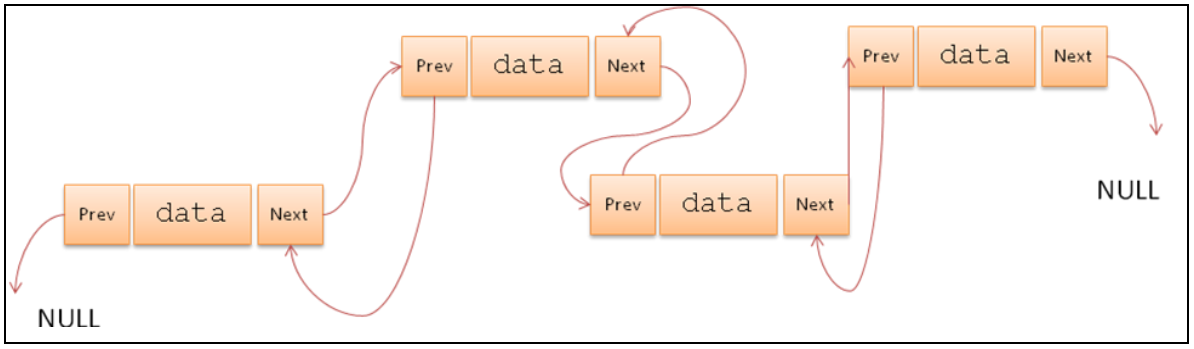
TUJUAN

Agar mahasiswa dapat mengetahui, memahami dan menggunakan konsep double link list non circular untuk menyelesaikan permasalahan dalam kehidupan sehari-hari.

MATERI

- Double Linked List Non Circular adalah sebuah linked list yang tidak hanya memiliki satu pointer tetapi dua pointer , yaitu next dan prev. Pointer Next menujuk pada node setelahnya, pointer prev menunjuk pada node sebelumnya



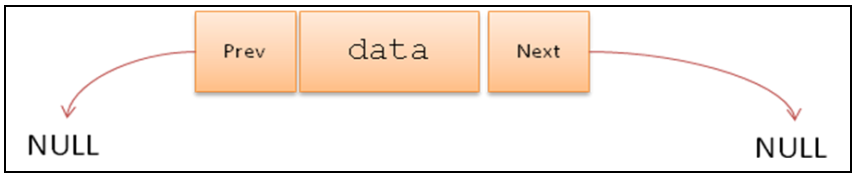


- Deklarasi DLNC :

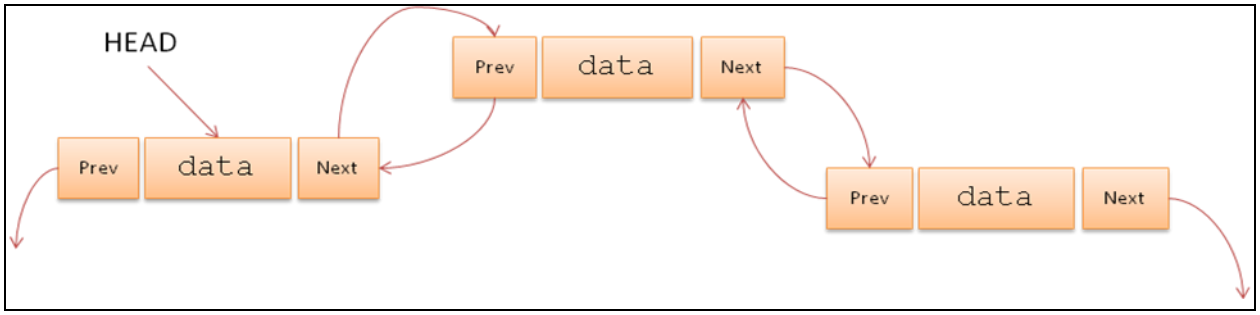
```
class Node {
    public $data;
    public $next;
    public $prev;
```

- Pembentukan node baru

```
public function __construct($d)
{
    $this->data = $d;
    $this->next = null;
    $this->prev = null;
}
```



Penggunaan HEAD pada SLLC




```
private $head;
```

Inisialiasi dan LEmpty

```
public function __construct()
{
    $this->head = null;
    $this->prev = null;
}

public function LEmpty()
{
    if ($this->head == null)
        return 1;
    else
        return 0;
}
```

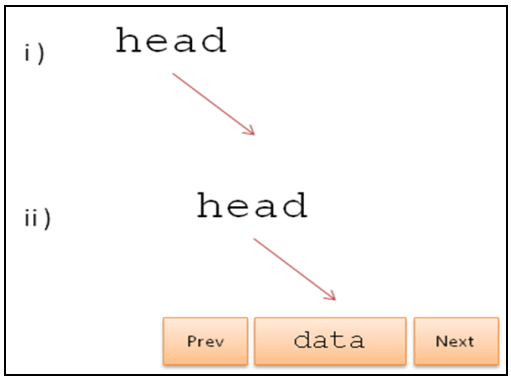
Menambah data di depan

- Ada dua jenis penambahan data, apabila linkedlist kosong maka head langsung dikaitkan ke data baru
- Apabila linked list tidak kosong, maka pointer next dari node baru akan diarahkan ke node pertama, selanjutnya pointer prev dari node pertama akan diarahkan ke node baru, kemudian head diarahkan ke node baru tersebut.

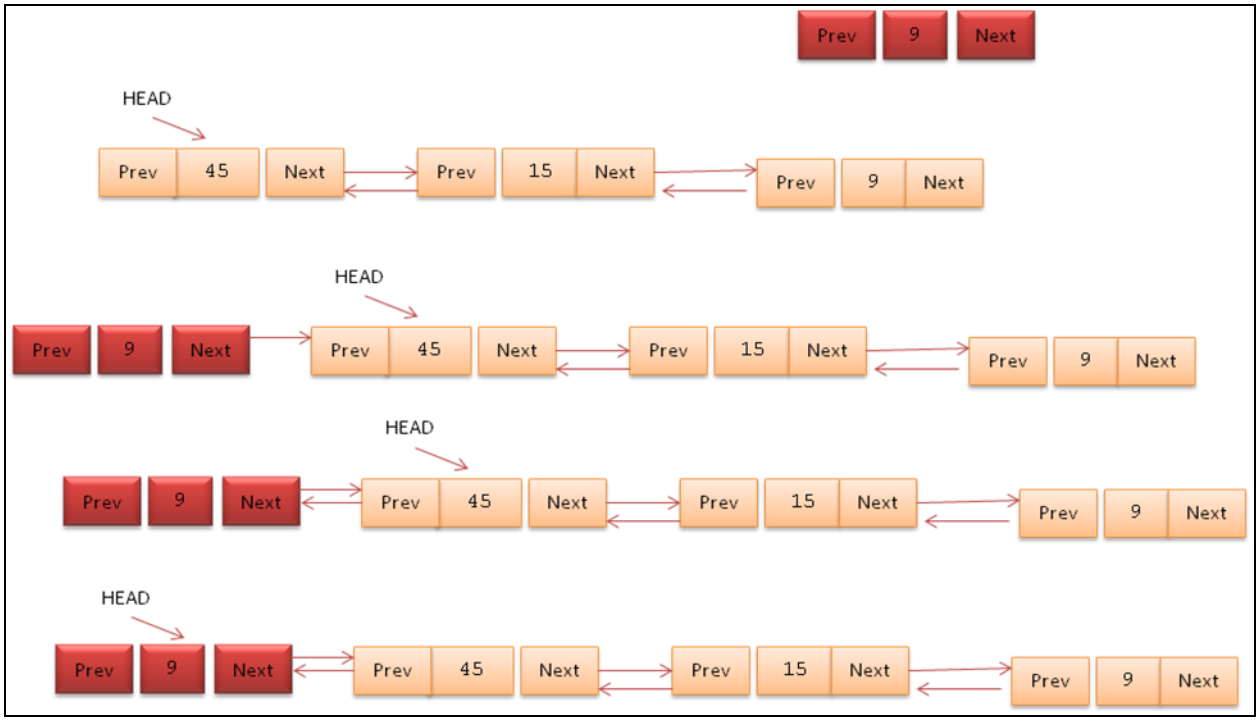
```php
public function insertD($d)
{
    $newNode = new Node($d);

    if ($this->LEmpty()) {
        $this->head = $newNode;
    } else
        {
            $newNode->next = $this->head;
            $this->head->prev = $newNode;
            $this->head = $newNode;
        }
}
```

Kondisi 1 : Linked List Kosong



Kondisi 2 : Linked List telah terisi



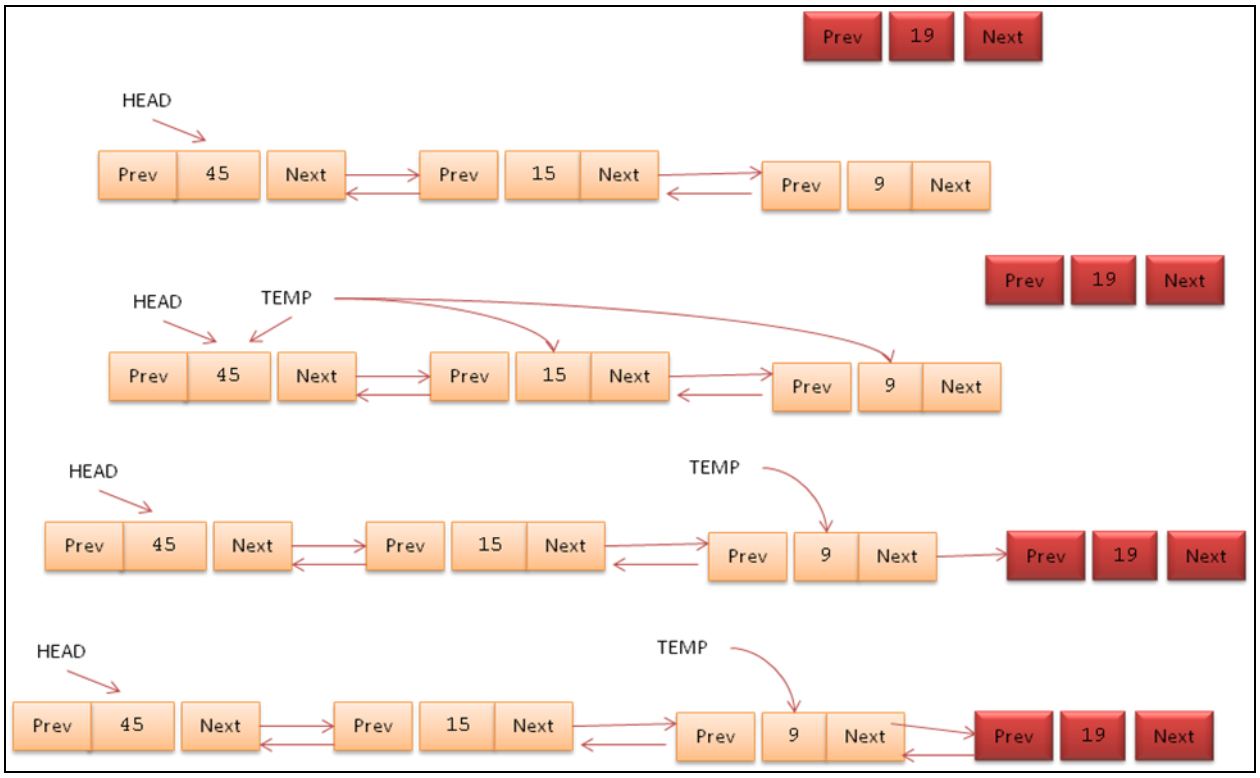
Menambah data di belakang

- Untuk menambah data dibelakang perlu menggunakan pointer bantuan untuk mencari node terakhir yang akan disambung dengan node baru

- Seteleh pointer next dari node terakhir di arahkan kenode baru melalui pointer bantuan, pointer prev dari node baru diarahkan ke node terakhir dari linked list.

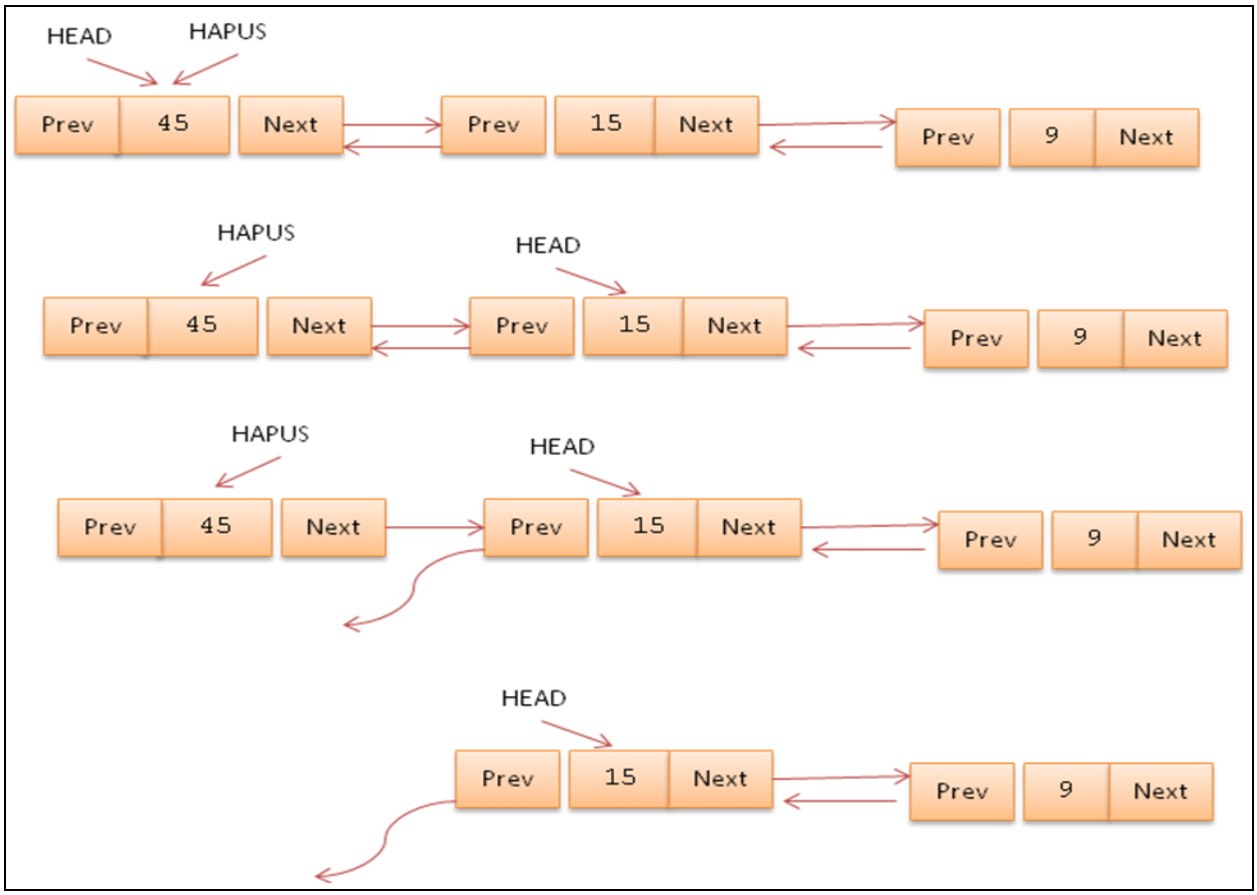
```
public function insertB($d)
{
    $newNode = new Node($d);

    if ($this->LEmpty())
    {
        $this->head = $newNode;
        $this->tail = $newNode;
    } else
    {
        $temp = $this->head;
        while ($temp->next != null)
        {
            $temp = $temp->next;
        }
        $temp->next = $newNode;
        $newNode->prev = $temp;
        $newNode->next = null;
    }
}
```



```php
public function HapusD()
{
    if (!$this->LEmpty())
    {
        if ($this->head->next == null) {
            $this->head = $this->tail = null;
        } else {
            $hapus = $this->head;
            $this->head = $this->head->next;
            $this->head->prev = null;
            $hapus->next = null;
            unset ($hapus);
        }
    }
    else
    {echo "<br>List kosong";}
}
```

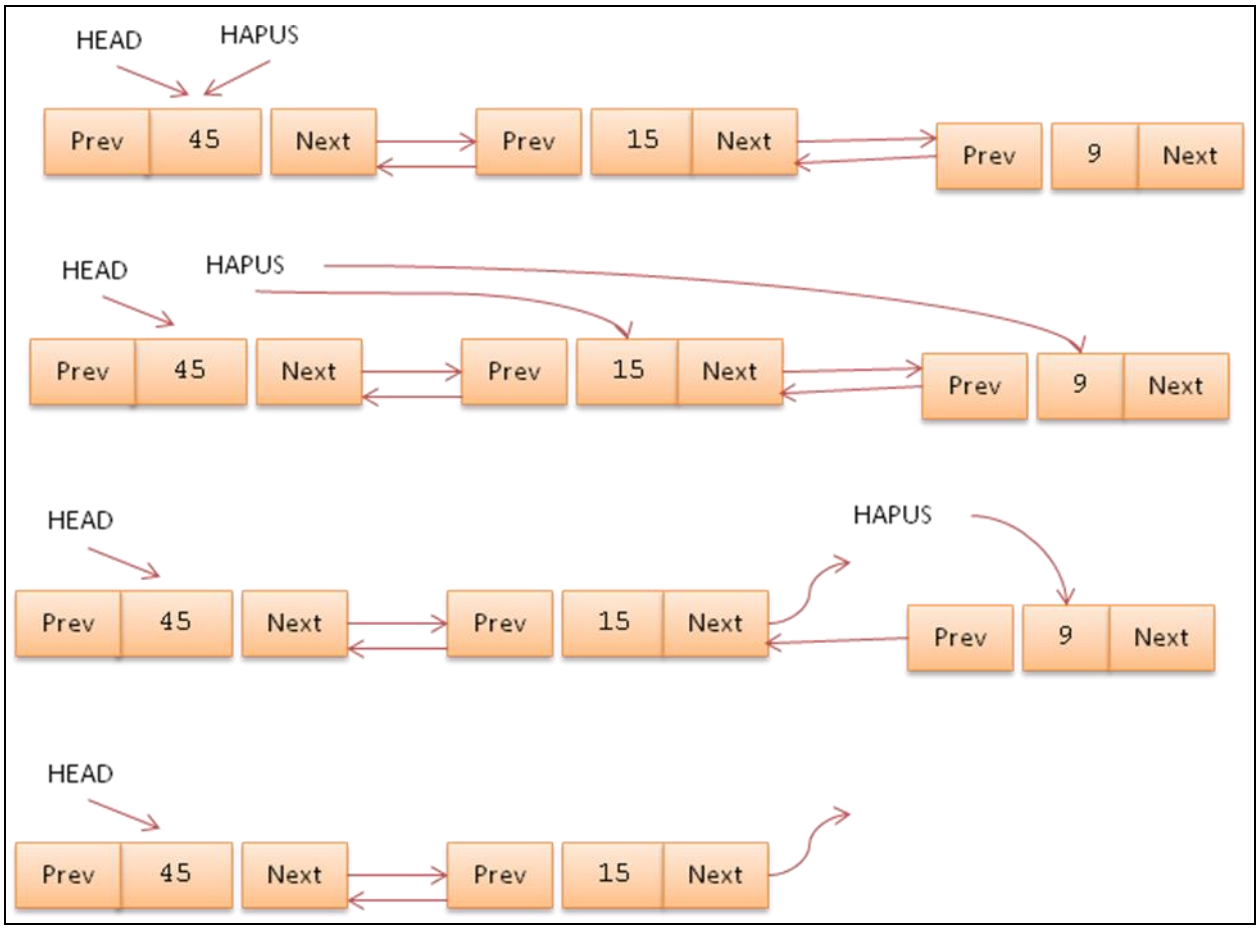
Menghapus data di depan



Menghapus datadi belakang

- Prinsip dari menghapus data di belakang adalah mencari node terakhir
- Kemudian menset pointer yang mengarah kepada node terakhir ke NULL

```php
public function HapusB()
{
    if ($this->head == null) {
        echo "Linked list kosong\n";
        return;
    }
    if ($this->head->next == null)
    {
        $this->head = null;
        return;
    } else
    {   $temp = $this->head;
        while ($temp->next->next != null)
        {
            $temp = $temp->next;
        }
        $hapus = $temp->next;
        $temp->next = null;
        $hapus->prev = null;
        unset ($hapus);
    }
}
```



Menampilkan data

```php
public function printList()
{
    if ($this->head == null) {
        echo "Linked list kosong\n";
        return;
    }
    $current = $this->head;
    while ($current != null) {
        echo $current->data . " ";
        $current = $current->next;
    }
    echo "\n";
}
```

Menghapus semua data

```php
public function clear()
{
    if ($this->LEmpty())
    {
        echo "Link list kosong\n";
        return;
    }
    $temp = $this->head;
    $hapus = null;

    do {
        $hapus = $temp;
        $temp = $temp->next;
        unset ($hapus);
    } while ($temp != null);

    $this->head = null;
    echo "Link List berhasil dihapus\n";
}
```

1. PRAKTIKUM 1

Buatlah program lengkap untuk double link list non circular dengan Head, dengan pemanggilan prosedur dan fungsi sebagai berikut :

```
CL = new DLLNCH();
CL->insertD(11);
CL->insertD(55);
CL->insertD(33);
CL->insertB(44);
cho "Isi linked list: ";
CL->printList();

cho "<hr><br>Hapus node pertama<br>";
CL->HapusD();
cho "Isi linked list setelah dihapus: ";
CL->printList();

cho "<hr><br>Hapus node terakhir<br>";
CL->HapusB();
cho "Isi linked list setelah dihapus: ";
CL->printList();

cho "<hr><br>Hapus semua node<br>";
CL->clear();
cho "<br>Isi linked list setelah dihapus: ";
CL->printList();
```

Output sebagai berkut :

```
Isi linked list: 33 55 11 44
---------------------------------------
Hapus node pertama
Isi linked list setelah dihapus: 55 11 44
---------------------------------------
Hapus node terakhir
Isi linked list setelah dihapus: 55 11
---------------------------------------
Hapus semua node
Link List berhasil dihapus
Isi linked list setelah dihapus: Linked list kosong
```

BUKU ACUAN

- Goodrich,Michael T.;Tamassia, Roberto; Mount, David. 2004. Data Structures and Algorithms in C++. WILEY.
- Hartono, Jogiyanto.1992.*Konsep Dasar Pemrograman Bahasa C.*Yogyakarta : Penerit Andi
- Wahyudi, Bambang.2004.”Pengantar Struktur Data & Algoritma”. Yogyakarta: Penerbit Andi
- Yatini B.,Indra; Nasution, Erliansya.2005.”Algoritma & Struktur Data dengan C++”.Graha Ilmu.