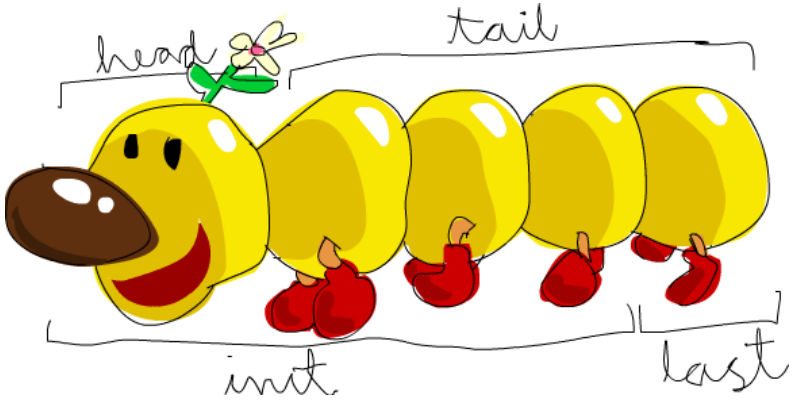
Double linked list

==Senarai berantai ganda==

Tenia Wahyuningrum, S.Kom., M.T

Double Link List adalah link list yang memiliki dua buah pointer yang menunjuk ke simpul sebelah kiri atau sebelumnya **(Prev)** dan yang menunjuk ke simpul sebelah kanan atau sesudahnya **(Next).**



Representasi data

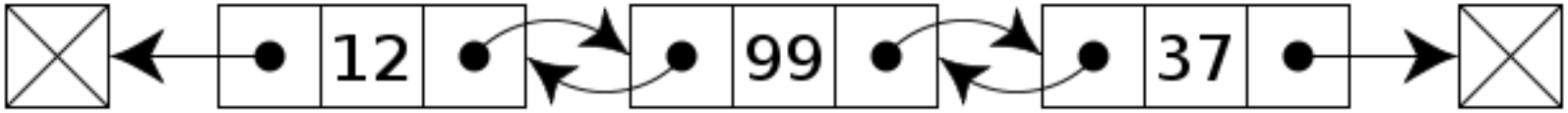
Representasi data

JENIS DOUBLE LINKED LIST

a. Double Linked List Non Circular

b. Double Linked List Circular

Double Linked List
Non Circular



Senarai ganda dengan tiap-tiap node yang terdiri atas tiga elemen, data integer, dan dua elemen rujukan ke node sebelum serta berikutnya

Setiap node/field pada linked list mempunyai field yang berisi data dan pointer. Node-node saling berkait melalui pointer.

- Untuk pembentukan node baru, mulanya pointer next dan prev akan menunjuk ke nilai NULL.

- Pointer prev akan menunjuk ke node sebelumnya, dan pointer next akan menunjuk ke node selanjutnya

ada beberapa fungsi yang diperlukan untuk menambahkan (menyisipkan) simpul baru, yaitu :

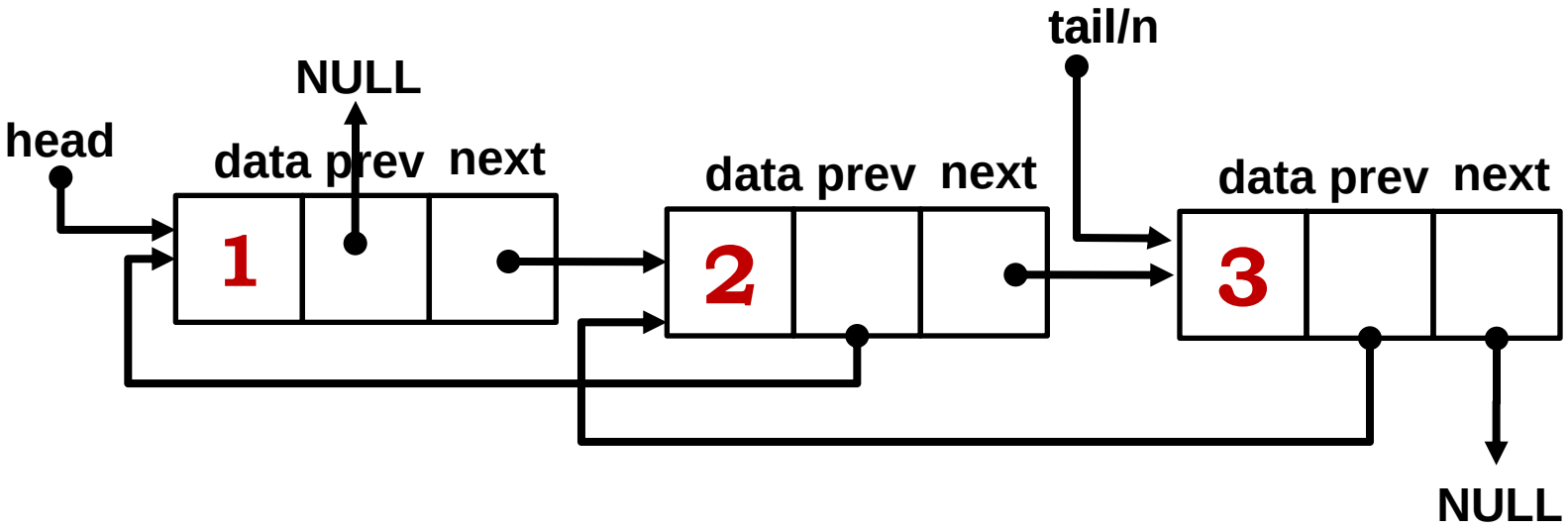
- sisip awal
- sisip akhir
- sisip sebelum
- sisip sesudah

dan untuk menghapus simpul, diperlukan beberapa fungsi, yaitu :

- hapus awal
- hapus akhir
- hapus simpul

Perhatikan script code double linked list non Circular berikut ini dan tuliskan urutan langkah Output node nya!

```
#include<iostream>
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

typedef struct node
{
        int data;
        node* prev;
        node* next;
    };

int main()
{
    node *head;
    node *tail;
    node *n;


    n= new node;
    n->data = 1;
    n->prev=NULL;
    head = n;
    tail = n;
    n= new node;
    n->data = 2;
    n->prev = tail;
    tail->next = n;
    tail=n;
    n=new node;
    n->data = 3;
    n->prev = tail;
    tail->next= n;
    tail=n;

    tail->next=NULL;

    tail = head ;


    while( tail!= NULL ){
        cout << "Data : " << tail->data << endl;
        tail = tail->next;

    }
    system("PAUSE");
    return 0;
}
```

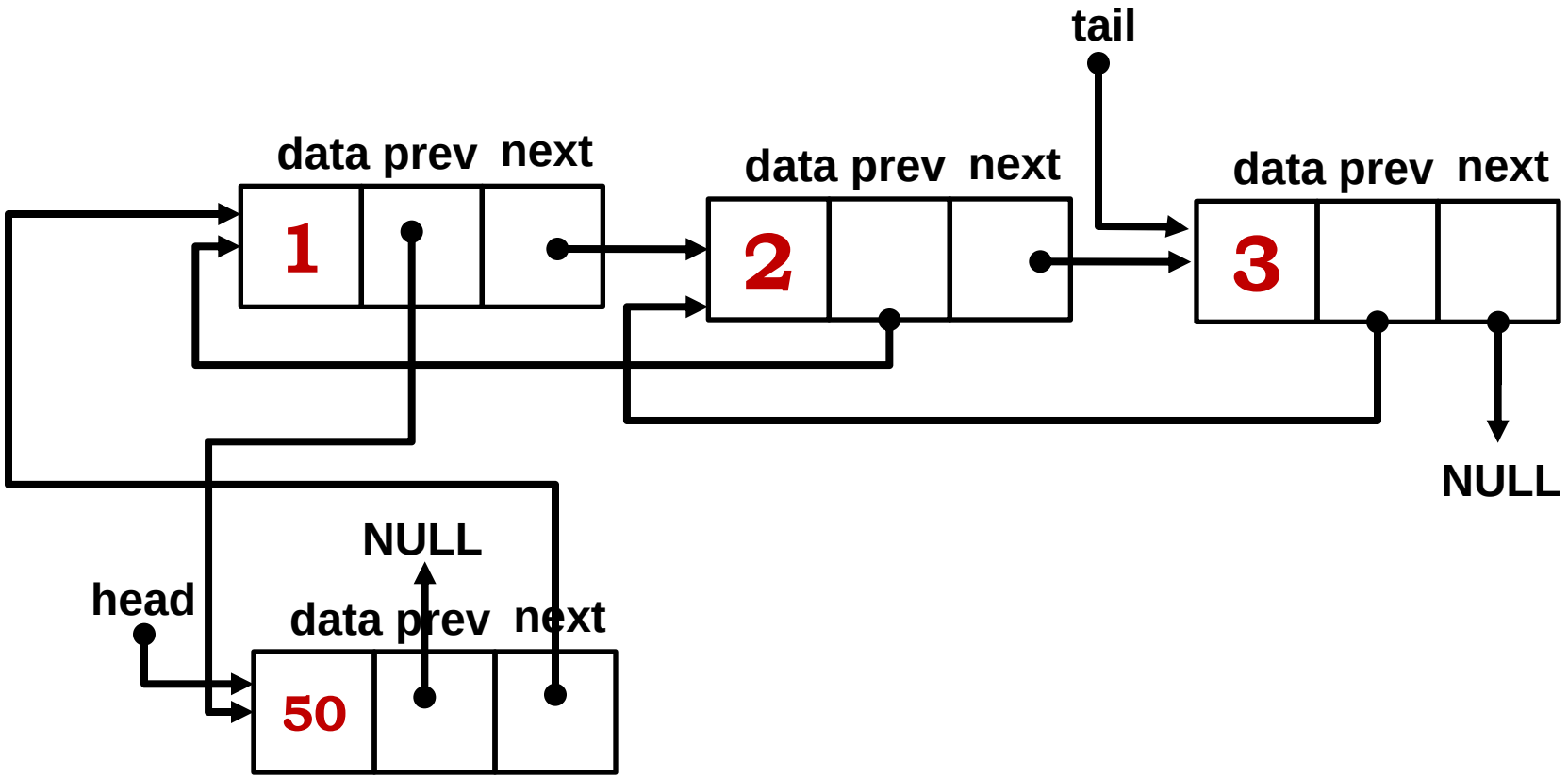

tail/n
NULL
head
data prev next
data prev next
data prev next
1
2
3
NULL


latihan

Tuliskan keluarannya, jika ditambahkan statement berikut !

```
    n=new node;
    n->data=50;
    n->prev=NULL;
    n->next = head;
    head->prev = n;
    head = n;

    tail->next=NULL;

    tail = head ;

    while( tail!= NULL ){
        cout << "Data : " << tail->data << endl;
        tail = tail->next;

    }
    system("PAUSE");
    return 0;
}
```

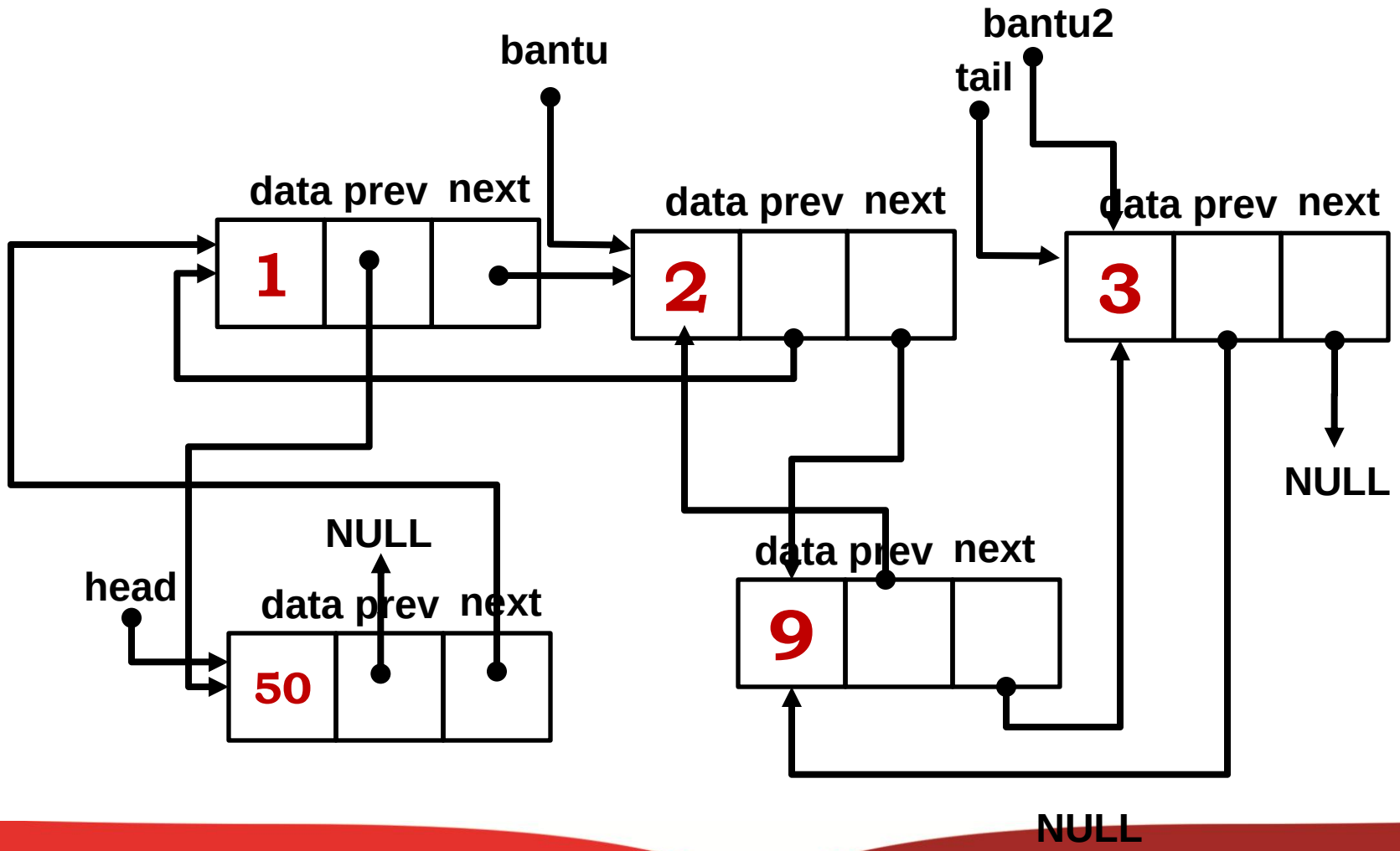
Penambahan di depan



Tuliskan keluarannya, jika ditambahkan statement berikut !

```
node *bantu, *bantu2;

n=new node;
n->data=9;
n->prev=NULL;
n->next=NULL;
bantu = head;

while(bantu->data != 2)
{
bantu = bantu->next;}

bantu2 = bantu->next;
n->next = bantu2;
bantu2->prev = n;
bantu->next = n;
n->prev = bantu;

tail->next=NULL;

tail = head ;

    while( tail!= NULL ){
        cout << "Data : " << tail->data << endl;
        tail = tail->next;

    }

    system("PAUSE");
    return 0;
}
```

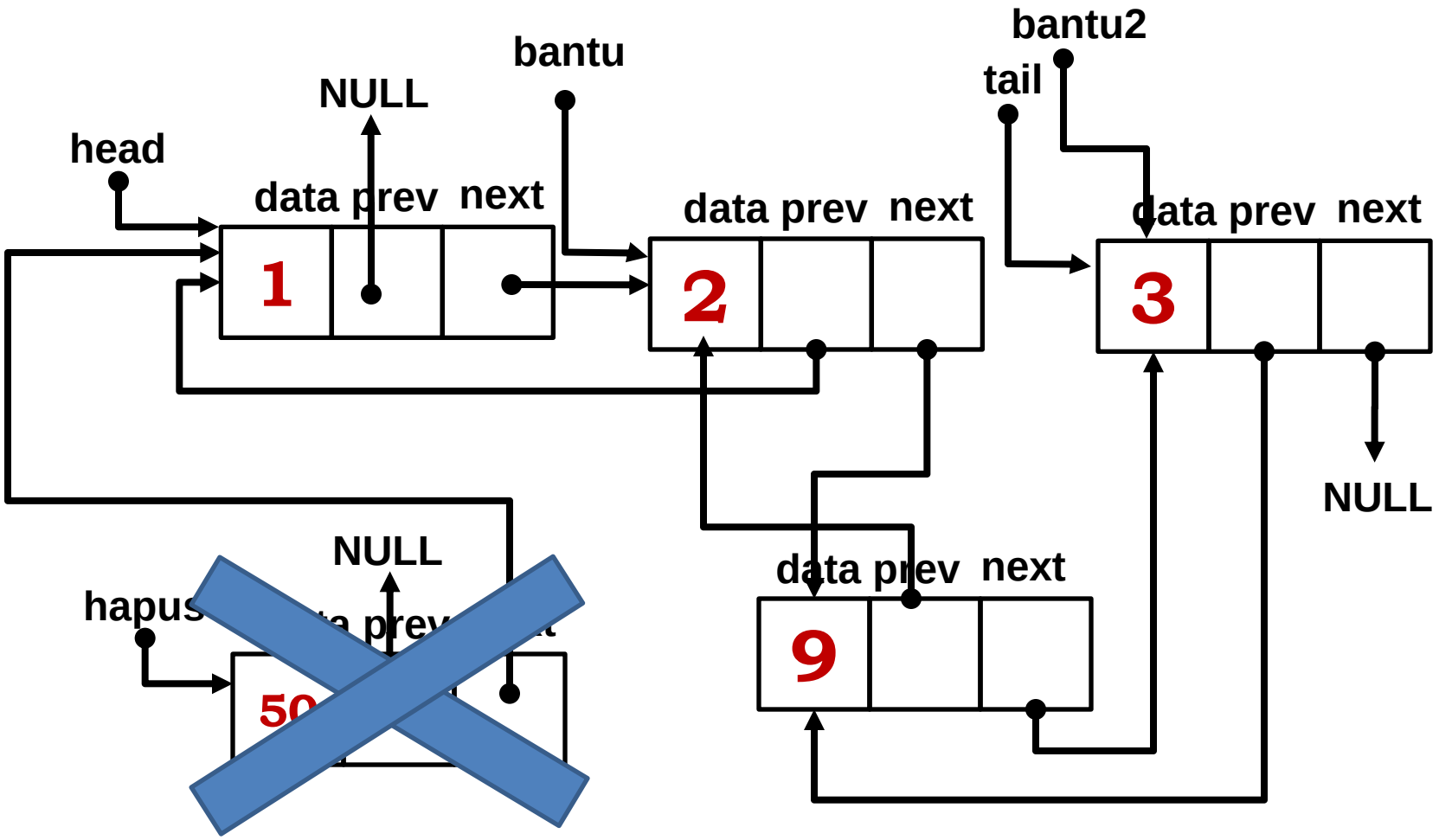
Penambahan di tengah



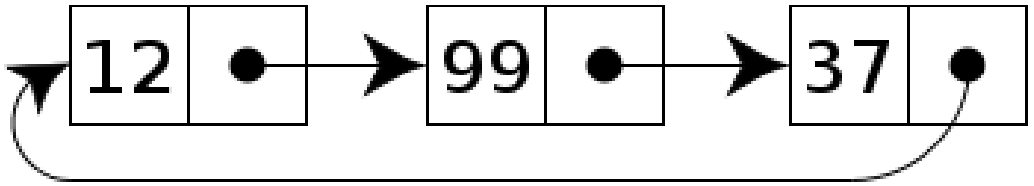
Tuliskan keluarannya, jika ditambahkan statement berikut !

```
while(bantu->data != 2)
{
bantu = bantu->next;}

bantu2 = bantu->next;
n->next = bantu2;
bantu2->prev = n;
bantu->next = n;
n->prev = bantu;


hapus = head;
head = head->next;
head->prev = NULL;
delete hapus;

tail->next=NULL;

tail = head ;

while( tail!= NULL ){
    cout << "Data : " << tail->data << endl;
    tail = tail->next;

}
```

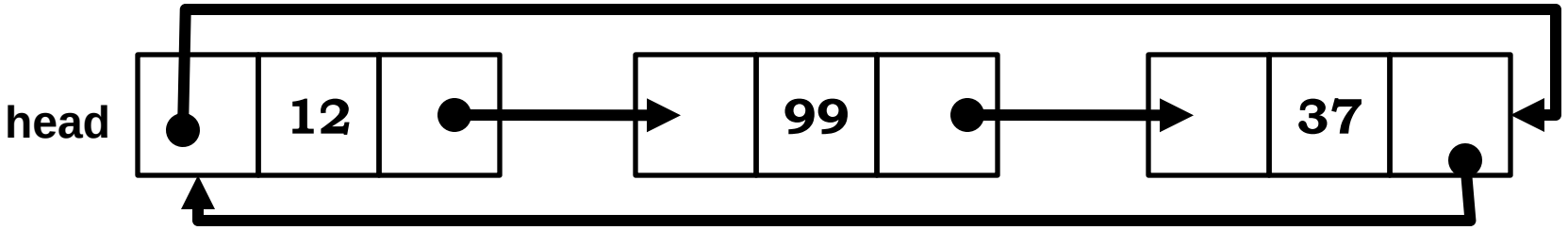
Hapus di depan



Double Linked List
Circular



Senarai sirkular dengan menggunakan model implementasi senarai tungal. Node terakhir menyimpan rujukan pada node pertama



Jenis linked list ini merupakan jenis double linked list yang memiliki simpul kepala dan tidak mempunyai tail (Head = Tail).

Perhatikan script code double linked list Circular sederhana berikut ini dan tuliskan urutan langkah Output node nya!

```
#include<iostream>
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>


//linked list circular
typedef struct node{
        int data;
        node* prev;
        node* next;
    };


int main()
{
    node* head;
    node* tail;
    node* n;
    node* bantu;
```

```
n = new node;
n->next = n;
n->prev = n;
n->data = 5;

head = tail = n;

n = new node;
n->next = n;
n->prev = n;
n->data = 8;

tail->next = n;
n->prev = tail;
tail = n;

tail->next = head;
head->prev = tail;
```

```
    n = new node;
    n->next = n;
    n->prev = n;
    n->data = 9;

    tail->next = n;
    n->prev = tail;
    tail = n;

    tail->next = head;
    head->prev = tail;

    bantu = head;
    do
    {
      cout<<bantu->data;
      bantu = bantu->next;
    } while(bantu!=head);


    system("PAUSE");
    return 0;
}
```