Teknik Informatika Kampus 3 Nganjuk
Jurusan Teknologi Informasi

Politeknik Negeri Jember

Circular Linked List

Struktur Data

Ulfa Emi Rahmawati, S.Kom., M.Kom.

Definisi

Circular linked list

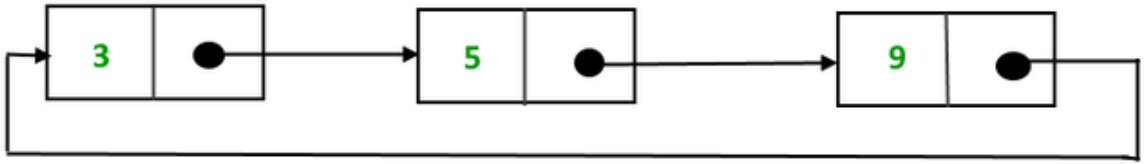
Circular linked list

Circular Linked List => linked list dimana semua node terhubung untuk membentuk lingkaran. Dalam circular linked list, node pertama dan node terakhir terhubung satu sama lain yang membentuk lingkaran.

Note: Tidak ada NULL di akhir.

Circular singly linked list

- Circular Singly Linked List => Dalam circular singly linked list, node terakhir dari list berisi penunjuk ke node pertama dari list.
- Melintasi circular singly linked list sampai mencapai simpul yang sama di mana dimulainya penelusuran.
- Circular singly linked list tidak memiliki awal atau akhir. Tidak ada nilai null yang ada di bagian *next* dari node manapun.



Representasi

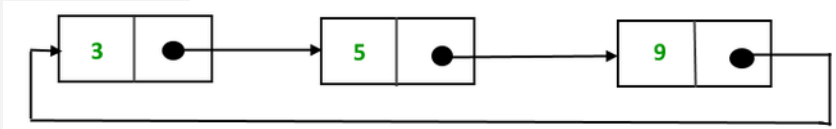
Circular linked list

Representasi circular linked list

- Circular linked list mirip dengan singly linked list dengan pengecualian menghubungkan simpul terakhir ke simpul pertama.
- Representasi node dari circular linked list menggunakan C++

```
// Class Node, similar to the linked list
class Node{
    int value;

  // Points to the next node.
    Node next;
}
```

Contoh circular singly linked list




```
// Initialize the Nodes.
Node one = new Node(3);
Node two = new Node(5);
Node three = new Node(9);

// Connect nodes
one.next = two;
two.next = three;
three.next = one;
```

Penjelasan:

Pada program di samping => satu, dua, dan tiga adalah node dengan nilai masing-masing 3, 5, dan 9 yang dihubungkan secara melingkar sebagai:

- Untuk Node Satu: Next pointer menunjuk alamat node dua.
- Untuk Node Dua: Next menunjuk alamat node tiga
- Untuk Node Tiga: Next menunjuk alamat node satu.

Insertion

Penyisipan Node baru

Insertion

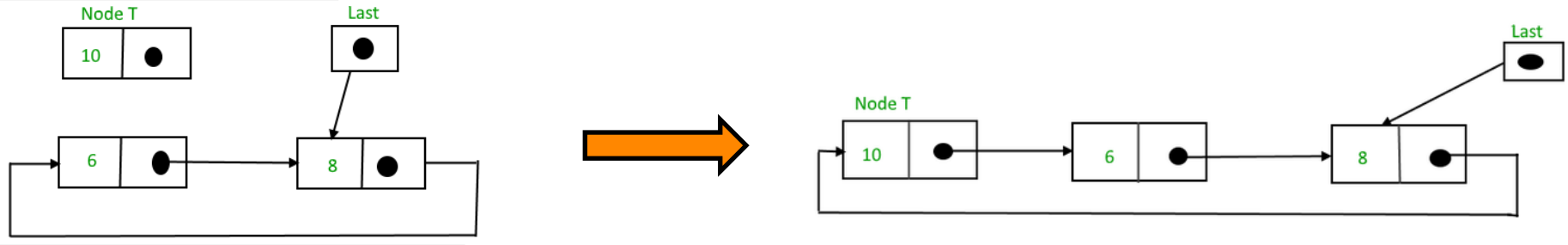
Merupakan penyisipan node baru pada suatu linked list

- Insertion di awal
- Insertion di akhir
- Insertion setelah node tertentu atau diantara node

Insertion di Awal

Untuk menyisipkan simpul di awal daftar, ikuti langkah-langkah berikut:

- Buat node, misalnya T.
- Buat T -> next = last -> next.
- last -> next = T.



Insertion di Awal

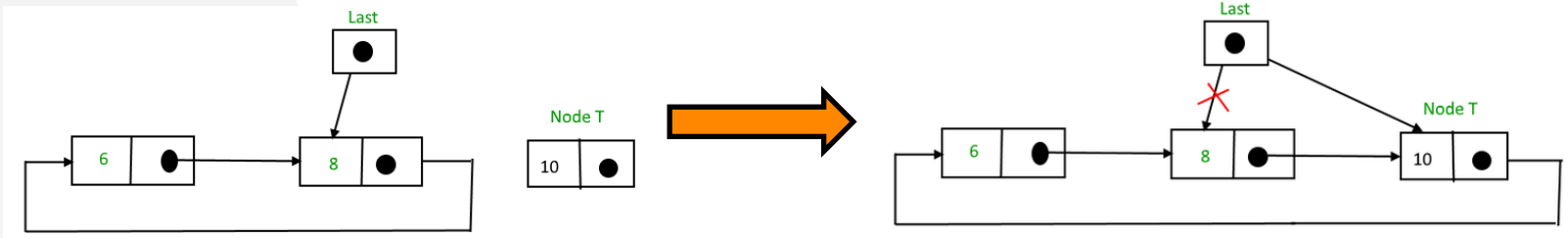
Di bawah ini adalah implementasi kode untuk menyisipkan node di awal list menggunakan C++

```
struct Node *addBegin(struct Node *last, int data)
{
if (last == NULL)
    return addToEmpty(last, data);

// Creating a node dynamically.
struct Node *temp
        = (struct Node *)malloc(sizeof(struct Node));

// Assigning the data.
temp -> data = data;

// Adjusting the links.
temp -> next = last -> next;
last -> next = temp;

return last;
}
```

Insertion di Akhir

Untuk menyisipkan node di akhir list, ikuti langkah-langkah berikut:

- Buat node, misalnya T.
- Buat T -> next = last -> next;
- last -> next = T.
- last = T.



Insertion di Akhir

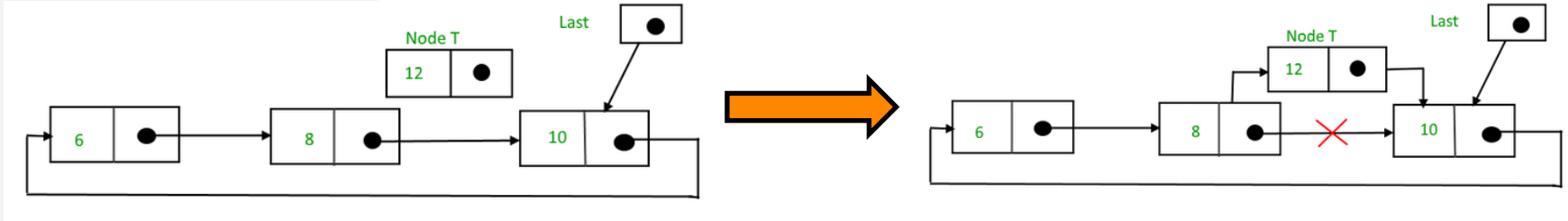
Di bawah ini adalah implementasi kode untuk menyisipkan node di akhir list menggunakan C++

```
struct Node *addEnd(struct Node *last, int data)
{
if (last == NULL)
    return addToEmpty(last, data);

// Creating a node dynamically.
struct Node *temp =
        (struct Node *)malloc(sizeof(struct Node));

// Assigning the data.
temp -> data = data;

// Adjusting the links.
temp -> next = last -> next;
last -> next = temp;
last = temp;

return last;
}
```

Insertion diantara node

Untuk menyisipkan node di antara dua node, ikuti langkah-langkah berikut:

- Buat node, misalnya T.
- Cari node sebelum T dimasukkan, misalnya node itu adalah P.
- Buat T -> next = P -> next;
- P -> next = T.



Insertion diantara node

Di samping ini adalah implementasi kode untuk menyisipkan node di akhir list menggunakan C++

```
struct Node *addAfter(struct Node *last, int data, int item)
{
    if (last == NULL)
    return NULL;

    struct Node *temp, *p;
    p = last -> next;

    // Searching the item.
    do
    {
        if (p ->data == item)
        {
            // Creating a node dynamically.
            temp = (struct Node *)malloc(sizeof(struct Node));

            // Assigning the data.
            temp -> data = data;

            // Adjusting the links.
            temp -> next = p -> next;

            // Adding newly allocated node after p.
            p -> next = temp;

            // Checking for the last node.
            if (p == last)
                last = temp;

            return last;
        }
        p = p -> next;
    } while (p != last -> next);

    cout << item << " not present in the list." << endl;
    return last;
}
```

THANKS!

Any questions?