# Modul 4 : ADT Single Linked List

## 4.1 Waktu Pelaksanaan Praktikum

Durasi kegiatan praktikum = **170 menit,** dengan rincian sebagai berikut :

a. 15 menit untuk pengerjaan Tes Awal atau wawancara Tugas Pendahuluan
b. 60 menit untuk penyampaian materi
c. 45 menit untuk pengerjaan tugas / Studi Kasus
d. 50 menit **Pengayaan**

## 4.2 Tujuan

Setelah mengikuti praktikum ini mahasiswa diharapkan dapat:

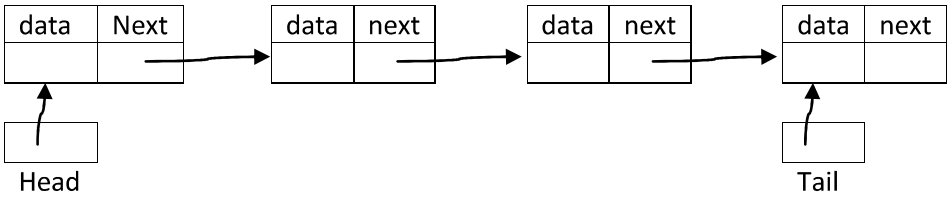
1. Mampu memahami konsep ADT linked list
2. Mampu mengaplikasikan ADT single linked list

## 4.3 Alat & Bahan

1. Komputer
2. Java IDE

## 4.4 Dasar Teori

Linked List merupakan struktur data yang dibangun dari satu atau lebih node yang menempati alokasi memori secara dinamis. Dalam setiap node menyimpan dua informasi utama yakni data dan pointer yang menunjuk ke node yang lain sehingga membentuk rangkaian node sebagaimana digambarkan berikut :




Jika linked list hanya berisi satu node maka pointernya akan menunjuk ke NULL. Jika linked list memiliki lebih dari satu node maka pointer menyimpan alamat dari node berikutnya. Sehingga antara node satu dengan node yang lain akan terhubung. Kecuali node paling ujung akan menunjuk ke NULL.

**Single Linked List (SLL)**

Single artinya pointer-nya hanya satu buah dan satu arah, yaitu menunjuk ke node sesudahnya. Node terakhir akan menunjuk ke NULL yang akan digunakan sebagai kondisi berhenti pada saat pembacaan isi linked list.

Dari ilustrasi gambar di atas ADT single Linked-list dapat direpresentasikan sebagai berikut :

| **Node** |
|---|
| **Data : Object**<br>**Next : Node** |

| **SLL** |
|---|
| **head,tail: Node**<br>**size=0 : int** |
| **Inisialisasi():void**<br>**isEmpty(): boolean**<br>**size (): int**<br>**addFirst(Node input): void**<br>**addLast (Node input): void** |

## 4.5 Prosedur Praktikum

**A. Percobaan Pertama: Memahami *Node* yang digunakan pada *Single Linked List***

1. Buatlah projek di IDE Java dengan nama SingleLinkedListProject
2. Buatlah Class *Node* di projek *SingleLinkedListProject* dan Tuliskan kode dibawah :

```
1   public class Node {
2       Object data;
3       Node next;
4
5       public static void main(String[] args)
6       {
7
8
9
10
11
12      }
13  }
```

3. Lakukan beberapa langkah berikut untuk memahami konsep Node pada Single Linked List:
   a. Tambahkan kode `Node head = new Node();` pada baris ke 7 class Node.
   b. Tambahkan kode `System.out.println("data : " + head.data);` pada baris ke 9 class Node kemudian jalankan program.
   c. Tambahkan kode `System.out.println("pointer: " + head.next);` pada baris ke 10 class Node kemudian jalankan program.
   d. Tambahkan kode `head.data = "A";` pada baris ke 8 class Node kemudian jalankan program.
4. Tuliskan hasil percobaan, analisis hasil dan kesimpulannya.

**B. Percobaan Kedua: Memahami Operasi sederhana pada Single Linked List**

1. Tambahkan *Class SLL* di projek *SingleLinkedListProject* dan Tuliskan kode dibawah :

```
1   public class SLL {
2     Node head,tail;
3     int size=0;
4
5     void inisialisasi(){
6       head=null;
7     }
8
9     boolean isEmpty(){
10      return (size==0);
11    }
12
13    int size()
14    {
15      return size;
16    }
17
18    void addFirst(Node input){
19      if (isEmpty()){
20        head=input;
21        tail=input;
22      }
23      else
```

```
24          {
25          input.next = head;
26          head = input;
27          } size++;
28      }
29
30      void addLast(Node input){
31          if (isEmpty()){
32              head = input;
33              tail = input;
34          }
35          else
36          {
37          tail.next = input;
38          tail = input;
39          } size++;
40      }
41  }
```

2. Tambahkan method main pada class SLL dengan dengan kode sebagai berikut kemudian jalankan.

```
1   public static void main(String[] args)
2       {
3           SLL list = new SLL();
4           System.out.println("head : " + list.head);
5           System.out.println("tail : " + list.tail);
6           list.addFirst(new Node());
7           System.out.println("head : " + list.head);
8           System.out.println("tail : " + list.tail);
9           list.addFirst(new Node());
10          System.out.println("head : " + list.head);
11          System.out.println("tail : " + list.tail);
12          list.addLast(new Node());
13          System.out.println("head : " + list.head);
14          System.out.println("tail : " + list.tail);
15      }
```

3. Ganti isi dari method main pada class SLL dengan kode sebagai berikut kemudian jalankan.

```
1   public static void main(String[] args)
2       {
3           SLL list = new SLL();
4           System.out.println("head : " + list.head);
5           System.out.println("tail : " + list.tail);
6           list.addLast(new Node());
7           System.out.println("head : " + list.head);
8           System.out.println("tail : " + list.tail);
9           list.addLast(new Node());
10          System.out.println("head : " + list.head);
11          System.out.println("tail : " + list.tail);
12          list.addLast(new Node());
```

```
13          System.out.println("head : " + list.head);
14          System.out.println("tail : " + list.tail);
15      }
```

4. Tuliskan hasil percobaan, analisis hasil dan kesimpulannya.

### 4.6 Hasil Percobaan

1. Tuliskan hasil dari percobaan pertama diatas.
2. Tuliskan hasil dari percobaan kedua diatas.

### 4.7 Analisis Hasil

1. Tuliskan Analisis hasil dari percobaan pertama diatas.
2. Tuliskan Analisis hasil dari percobaan kedua diatas.

### 4.8 Kesimpulan

1. Tuliskan kesimpulan dari percobaan pertama diatas.
2. Tuliskan kesimpulan dari percobaan kedua diatas.

### 4.9 Latihan

Operasi pada Single Linked List secara lengkap adalah sebagai berikut:

1. Inisialisasi
2. isEmpty
3. size
4. Penambahan
5. Penghapusan
6. Penyisipan
7. Pencarian
8. Pengaksesan

Lengkapi kode class SLL diatas untuk operasi-operasi (method) yang belum ada pada Single Linked List.

### 4.10 Tugas

1. Modifikasilah program Latihan di atas sehingga SLL dapat menampung sembarang object. Untuk itu anda perlu membuat class baru bernama Mahasiswa dengan data dan method sebagai berikut :

| **Mahasiswa** |
|---|
| **String nim**<br>**String nama**<br>**double ipk** |
| **Constructor Mahasiswa**<br>**double getIpk**<br>**String getNim**<br>**String getNama** |

2. Tambahkan method penyisipan data yang bisa membentuk linked list urut sejak awal dengan pengurutan berdasarkan ipk.

### 4.11 DAFTAR PUSTAKA


- Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, “Data Structures and Algorithms Using Java 6 edition”, Wiley, USA, 2014.

- John R. Hubbard, “Scaum’s Outline of Data Structures With Java second Edition”, McGraw-Hill, New york, 2007.
- Robert Lafore, “Data Structures and Algorithm in Java second Edition”, Sams Publishing, Indiana, 2003