PRAKTIKUM 16
SINGLE LINKED LIST 2

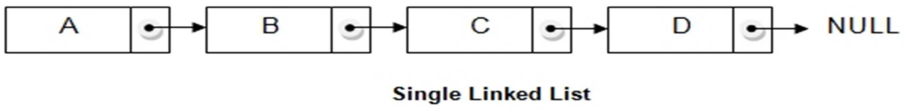
A. TUJUAN PEMBELAJARAN

Mahasiswa diharapkan mampu :

1. Memahami konsep SingleLinkedList untuk menghapus sebuah node
2. Memahami konsep SingleLinkedList untuk mencari index untuk node tertentu

B. DASAR TEORI

Linked list adalah sekumpulan elemen bertipe sama, yang mempunyai keterurutan tertentu, yang setiap elemennya terdiri dari dua bagian. Struktur berupa rangkaian elemen saling berkait dimana setiap elemen dihubungkan elemen lain melalui pointer. Pointer adalah alamat elemen.Penggunaan pointer untuk mengacu elemen berakibat elemen-elemen bersebelahan secara logik walau tidak bersebelahan secara fisik di memori.



Single Linked List

Gambar 1. Single Linked List

Terdapat tempat yang disediakan pada satu area memori tertentu untuk menyimpan data dikenal dengan sebutan node atau simpul.Setiap node memiliki pointer yang menunjuk ke simpul berikutnya sehingga terbentuk satu untaian, dengan demikian hanya diperlukan sebuah variabel pointer.Susunan berupa untaian semacam ini disebut Single Linked List (NULL memilik nilai khusus yang artinya tidak menunjuk ke mana-mana. Biasanya Linked List pada titik akhirnya akan menunjuk ke NULL). Salah satu kelemahan single linked list adalah pointer (penunjuk) hanya dapat bergerak satu arah saja, maju/mundur, atau kanan/kiri sehingga pencarian data pada single linked list hanya dapat bergerak dalam satu arah saja. Untuk mengatasi kelemahan tersebut, dapat menggunakan metode double linked list. Linked list ini dikenal dengan nama Linked list berpointer Ganda atau Double Linked List.

C. TUGAS PENDAHULUAN

1. Dengan menggunakan gambar, jelaskan langkah-langkah menghapus sebuah node di awal SingleLinkedList.

2. Dengan menggunakan gambar, jelaskan langkah-langkah menghapus sebuah node di akhir SingleLinkedList.

D. PERCOBAAN

Pada percobaan SingleLinkedList (SLL), membuat class SingleLinkedList yang mengimplementasikan interface List<T>. Implementasikan method-method pada interface List<T> seperti tabel 1. Method yang bercetak tebal adalah method yang dibuat sendiri.

Pada praktikum ini yang dikerjakan adalah praktikum SLL 2.

Tabel 1. Method-method pada class SingleLinkedList<T>

`class SingleLinkedList<T> implements List<T>`		`Praktikum`
`private Node<T> front = null;`	Variable reference untuk menandai node awal dari SSL	**SLL 1**
`private int size;`	Variablel untuk mengetahui jumlah node pada SSL	
`private void addFirst(T item) {}`	Method addFirst() untuk menambahkan node baru di awal SSL	
`private void addLast(T item) {}`	Method addLast()untuk menambahkan node baru di akhir SSL	
`public boolean add(T e){}`	Method add()untuk menambahkan node baru di akhir SSL	
`public int size() {}`	Method size() untuk mengetahui jumlah node pada SSL	
`public void add(int index, T element) {}`	Method add(index,elemen) untuk menambahkan node baru dengan value elemen dengan pada index tertentu	
`public boolean isEmpty(){}`	Method isEmpty() untuk mengetahui apakah SSL masih null/kosong, jika ya maka mengembalikan nilai true, jika tidak null maka mengembalikan nilai false.	
`public T get(int index) {}`	Method get(index) untuk mendapatkan value pada node pada index tertentu dari SSL	
`public T set(int index, T element){}`	Method set(index, element) untuk merubah value pada node pada index tertentu dengan	

	element. Method ini mengembalikan value yang lama.	
`public String toString() {}`	Method toString() untuk mengubah objek SSL menjadi String	
`public Object[] toArray() {}`	Method toArray() untuk mengubah objek SSL menjadi array dengan tipe Object	**SLL 2**
`public boolean contains(Object o) {}`	Method contains() untuk mengecek apakah terdapat node o pada SSL	
`public boolean remove(Object o){}`	Method remove(Object o) untuk menghapus node o pada SSL	
`private T removeFirst() {}`	Method removeFirst() untuk menghapus node yang paling depan	
`private T removeLast() {}`	Method removeLast() untuk menghapus node yang paling akhir	
`public T remove(int index){}`	Method remove(index) untuk menghapus node pada index tertentu pada SSL	
`public int indexOf(Object o){}`	Method indexOf(Object o) untuk mendapatkan index pertama kali node o pada SSL	
`public int lastIndexOf(Object o){}`	Method lastIndexOf(Object o) untuk mendapatkan index terakhir node o pada SSL	
`public Iterator<T> iterator() {}`	Method iterator() untuk melakukan iterasi pada SSL	**SLL 3**

Percobaan 1. Membuat method removeFirst() untuk menghapus node di awal SLL

```
public class SingleLinkedList<T> implements List<T>{
…
     private T removeFirst() {
        Node<T> curr = front;
        front = curr.next;
        curr.next = null;
        T value = curr.nodeValue;
        curr = null;
        size--;
        return value;
    }
}
```

Percobaan 2. Membuat method removeLast() untuk menghapus node di akhir SLL

```
public class SingleLinkedList<T> implements List<T>{
…
    private T removeLast() {
        Node<T> curr = front, prev = null;
        while (curr.next != null) {
            prev = curr ;
            curr = curr.next;
        }
        T value = curr.nodeValue ;
        prev.next = null ;
        curr = null ;
        size--;
        return value;

    }

}
```

Percobaan 3. Membuat method indexOf() untuk mencari index pertama kali sebuah Node yang dicari pada SLL

```
public class SingleLinkedList<T> implements List<T>{
…
    public int indexOf(Object o) {
        Node<T> curr = front;
        Node<T> o2 = (Node<T>) o ;
        int i = 0;
        while (curr != null) {
            if (o2.nodeValue.equals(curr.nodeValue)) {
                return i;
            } else {
                curr = curr.next;
                i++;
            }
        }
        return -1;
    }
}
```

E. LATIHAN

Selesaikan method-method yang belum diimplementasikan pada Class SingleLinkedList

`class SingleLinkedList<T> implements List<T>`	
`public Object[] toArray() {}`	Method toArray() untuk mengubah objek SSL menjadi array dengan tipe Object
`public boolean contains(Object o) {}`	Method contains() untuk mengecek apakah terdapat

	node o pada SSL
`public boolean remove(Object o){}`	Method remove(Object o) untuk menghapus node o pada SSL
`public T remove(int index){}`	Method remove(index) untuk menghapus node pada index tertentu pada SSL
`public int lastIndexOf(Object o){}`	Method lastIndexOf(Object o) untuk mendapatkan index terakhir node o pada SSL

Selanjutnya ujilah method-method pada Class SingleLinkedList dengan membuat class Main, sebagai contoh berikut.

```
public class Main {
    public static void main(String[] args) {
        SingleLinkedList<String> list = new
SingleLinkedList<String>();
        System.out.println(list.toString());
        list.add("Kuning");
        list.add("Coklat");
        list.add("Kuning");
        list.add("Merah");
        System.out.println(list.toString());
        list.remove(1);
        System.out.println(Arrays.toString(list.toArray()));
        System.out.println("Warna Kuning di index : " +
list.indexOf(new Node<String>("Kuning")));
    }
}
```

Output program adalah

```
SLL masih kosong
[Kuning, Coklat, Kuning, Merah]
[Kuning, Kuning, Merah]
Warna Kuning di index : 0
```

F. LAPORAN RESMI

Kerjakan hasil percobaan(D) dan latihan(E) di atas dan tambahkan analisa.