JOBSHEET
Double Linked Lists

12.1 Tujuan Praktikum

Setelah melakukan praktikum ini, mahasiswa mampu:

1. memahami algoritma double linked lists;
2. membuat dan mendeklarasikan struktur algoritma double linked lists;
3. menerapkan algoritma double linked lists dalam beberapa *study case*.

12.2 Kegiatan Praktikum 1

12.2.1 Percobaan 1

Pada percobaan 1 ini akan dibuat class data, class Node dan class DoubleLinkedLists yang didalamnya terdapat operasi-operasi untuk menambahkan data dengan beberapa cara (dari bagian depan dan belakang linked list)

1. Perhatikan diagram class Mahasiswa01, Node01 dan class DoublelinkedLists di bawah ini! Diagram class ini yang selanjutnya akan dibuat sebagai acuan dalam membuat kode program DoubleLinkedLists.

Mahasiswa01
nim: String nama: String kelas: String ipk: Double
Mahasiswa01(String nim, String nama, String kelas, Double IPK) tampil()

Node01
data: Mahasiswa01 prev: Node01 next: Node01
Node01(prev:null, data: Mahasiswa01 data, next:null)

DoubleLinkedLists
head: Node01 tail : Node01
DoubleLinkedLists() isEmpty(): boolean addFirst (): void addLast(): void add(item: int, index:int): void print(): void removeFirst(): void removeLast(): void search(): null insertAfter: void

2. Buat paket baru dengan nama **dll** (opsional)
3. Buat class di dalam paket tersebut dengan nama **Mahasiswa01.** Di dalam class tersebut, deklarasikan atribut sesuai dengan diagram class di atas. Tambahkan juga konstruktor dan method sesuai diagram di atas

```java
package dll;

public class Mahasiswa01 {
    public String nim;
    public String nama;
    public String kelas;
    public double ipk;

    public Mahasiswa01(String nim, String nama, String kelas, double ipk) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.ipk = ipk;
    }

    public void tampil() {
        System.out.println("NIM: " + nim + ", Nama: " + nama + ", Kelas: " + kelas + ", IPK: " + ipk);
    }
}
```

4. Buat class di dalam paket tersebut dengan nama **Node01.** Di dalam class tersebut, deklarasikan atribut sesuai dengan diagram class di atas. Selanjutnya tambahkan konstruktor sesuai diagram di atas

```java
package dll;

public class Node01 {
    Mahasiswa01 data;
    Node01 prev;
    Node01 next;

    public Node01(Mahasiswa01 data) {
        this.data = data;
        this.prev = null;
        this.next = null;
    }
}
```

5. Buatlah sebuah class baru bernama **DoubleLinkedLists** pada package yang sama dengan **Node01**. Pada class **DoubleLinkedLists** tersebut, deklarasikan atribut sesuai dengan diagram class di atas.

```java
package dll;

public class DoubleLinkedList01 {
    Node01 head;
    Node01 tail;
```

6. Selajuntnya, buat konstruktor pada class DoubleLinkedLists sesuai gambar berikut.

```java
public DoubleLinkedList01() {
    head = null;
    tail = null;
}
```

7. Buat method **isEmpty()**. Method ini digunakan untuk memastikan kondisi linked list kosong.

```java
public boolean isEmpty() {
    return head == null;
}
```

8. Kemudian, buat method **addFirst().** Method ini akan menjalankan penambahan data di bagian depan linked list.

```java
public void addFirst(Mahasiswa01 data) {
    Node01 newNode = new Node01(data);
    if (isEmpty()) {
        head = tail = newNode;
    } else {
        newNode.next = head;
        head.prev = newNode;
        head = newNode;
    }
}
```

9. Selain itu pembuatan method **addLast()** akan menambahkan data pada bagian belakang linked list.

```
public void addLast(Mahasiswa01 data) {
    Node01 newNode = new Node01(data);
    if (isEmpty()) {
        head = tail = newNode;
    } else {
        tail.next = newNode;
        newNode.prev = tail;
        tail = newNode;
    }
}
```

10. Untuk menambahkan data pada posisi setelah node yang menyimpan data *key,* dapat dibuat dengan cara sebagai berikut

```
public void insertAfter(String keyNim, Mahasiswa01 data) {
Node01 current = head;

// Cari node dengan nim = keyNim
while (current != null && !current.data.nim.equals(keyNim)) {
    current = current.next;
}

if (current == null) {
    System.out.println("Node dengan NIM " + keyNim + " tidak ditemukan."
    return;
}

Node01 newNode = new Node01(data);

// Jika current adalah tail, cukup tambahkan di akhir
if (current == tail) {
    current.next = newNode;
    newNode.prev = current;
    tail = newNode;
} else {
    // Sisipkan di tengah
    newNode.next = current.next;
    newNode.prev = current;
    current.next.prev = newNode;
    current.next = newNode;
}

System.out.println("Node berhasil disisipkan setelah NIM " + keyNim);
```

11. Untuk mencetak isi dari linked lists dibuat method **print().** Method ini akan mencetak isi linked lists berapapun size-nya.

```
public void print() {
    Node01 current = head;
    while (current != null) {
        current.data.tampil();
        current = current.next;
    }
}
```

12. Selanjutya dibuat class Main DoubleLinkedListsMain untuk mengeksekusi semua method yang ada pada class DoubleLinkedLists.

```
public class DLLMain {
    Run | Debug
    public static void main(String[] args) {
    DoubleLinkedList01 list = new DoubleLinkedList01();
    Scanner scan = new Scanner(System.in);
    int pilihan;
```

13. Buatlah menu pilihan pada class main

```
do {
        System.out.println(x:"\nMenu Double Linked List Mahasiswa");
        System.out.println(x:"1. Tambah di awal");
        System.out.println(x:"2. Tambah di akhir");
        System.out.println(x:"3. Hapus di awal");
        System.out.println(x:"4. Hapus di akhir");
        System.out.println(x:"5. Tampilkan data");
        System.out.println(x:"7. Cari Mahasiswa berdasarkan NIM");
        System.out.println(x:"0. Keluar");
        System.out.print(s:"Pilih menu: ");
        pilihan = scan.nextInt();
        scan.nextLine();
```

14. Tambahkan switch case untuk menjalankan menu pilihan di atas

```
switch (pilihan) {
    case 1 -> {
        Mahasiswa01 mhs = inputMahasiswa(scan);
        list.addFirst(mhs);
    }
    case 2 -> {
        Mahasiswa01 mhs = inputMahasiswa(scan);
        list.addLast(mhs);
    }
    case 3 -> list.removeFirst();
    case 4 -> list.removeLast();
    case 5 -> list.print();
    case 6 -> {
        System.out.print(s:"Masukkan NIM yang dicari: ");
        String nim = scan.nextLine();
        Node01 found = list.search(nim);
        if (found != null) {
            System.out.println(x:"Data ditemukan:");
            found.data.tampil();
        } else {
            System.out.println(x:"Data tidak ditemukan.");
        }
    }
    case 0 -> System.out.println(x:"Keluar dari program.");
    default -> System.out.println(x:"Pilihan tidak valid!");
}
```

15. Jangan lupa tambahkan while di bawah switch case dan **close** untuk menutup object **scanner**

```
} while (pilihan != 0);
scan.close();
```

16. Ada satu karakter yang perlu ditambahkan agar code bisa berjalan. Silakan dianalisis kekurangannya dan ditambahkan sendiri.

12.2.2 Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 1
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 5
NIM: 20304050, Nama: Hermione, Kelas: Gryffindor, IPK: 4.0
```

12.2.3 Pertanyaan Percobaan

1. Jelaskan perbedaan antara single linked list dengan double linked lists!
2. Perhatikan class Node01, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?
3. Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan dari konstruktor tersebut?

```
public DoubleLinkedList01() {
    head = null;
    tail = null;
}
```

4. Pada method **addFirst()**, apa maksud dari kode berikut?

```
if (isEmpty()) {
    head = tail = newNode;
```

5. Perhatikan pada method **addFirst().** Apakah arti statement head.prev = newNode ?

6. Modifikasi code pada fungsi **print()** agar dapat menampilkan warning/ pesan bahwa linked lists masih dalam kondisi.

7. Pada insertAfter(), apa maksud dari kode berikut ?

```
current.next.prev = newNode;
```

8. Modifikasi menu pilihan dan switch-case agar fungsi **insertAfter()** masuk ke dalam menu pilihan dan dapat berjalan dengan baik.

12.3 Kegiatan Praktikum 2

12.3.1 Tahapan Percobaan

Pada praktikum 2 ini akan dibuat beberapa method untuk menghapus isi LinkedLists pada class DoubleLinkedLists. Penghapusan dilakukan dalam tiga cara di bagian paling depan, paling belakang, dan sesuai indeks yang ditentukan pada linkedLists.

1. Buatlah method **removeFirst()** di dalam class **DoubleLinkedLists.**

```
public void removeFirst() {
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }
    if (head == tail) {
        head = tail = null;
    } else {
        head = head.next;
        head.prev = null;
    }
}
```

2. Tambahkan method **removeLast()** di dalam class **DoubleLinkedLists.**

```
public void removeLast() {
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }
    if (head == tail) {
        head = tail = null;
    } else {
        tail = tail.prev;
        tail.next = null;
    }
}
```

12.3.2 Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 2
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 3

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
```

12.3.3 Pertanyaan Percobaan

1. Apakah maksud statement berikut pada method **removeFirst()**?

```
head = head.next;
head.prev = null;
```

2. Modifikasi kode program untuk menampilkan pesan “Data sudah berhasil dihapus. Data yang terhapus adalah … “

12.5 Tugas Praktikum

1. Tambahkan fungsi add() pada kelas DoubleLinkedList untuk menambahkan node pada indeks tertentu
2. Tambahkan removeAfter() pada kelas DoubleLinkedList untuk menghapus node setelah data *key*.
3. Tambahkan fungsi remove() pada kelas DoubleLinkedList untuk menghapus node pada indeks tertentu.
4. Tambahkan fungsi getFirst(), getLast() dan getIndex() untuk menampilkan data pada node head, node tail dan node pada indeks tertentu.
5. tambahkan kode program dan fungsi agar dapat membaca size/ jumlah data pada Double Linked List

--- *** ---