PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

JOBSHEET 10

NAMA : DIMAS ADI BAYU SAMUDRA

KELAS : 1A

NO. ABSEN : 08

NIM : 2341720169

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

JOBSHEET 10

Double Linked Lists

Praktikum

Percobaan 1

Class Node08

```
package doublelinkedlists;
public class Node08 {
    int data;
    Node08 prev, next;

    public Node08 (Node08 prev, int data, Node08 next){
        this.prev = prev;
        this.data = data;
        this.next = next;
    }
}
```

Class Double Linked List 08

```
package doublelinkedlists;

public class DoubleLinkedLists08 {
    Node08 head;
    int size;

    public DoubleLinkedLists08(){
        head = null;
        size = 0;
    }

    public boolean isEpmty(){
        return head == null;
    }


    public void addFirst(int item){
        if (isEpmty()) {
            head = new Node08(null, item, null);

        } else {
            Node08 newNode08 = new Node08(null, item, head);
            head.prev = newNode08;
            head = newNode08;
        }
        size++;
    }
    public void addLast (int item){
        if (isEpmty()) {
            addLast(item);
            size++;
```

```
        } else {
            Node08 current = head;
            while (current.next != null) {
                current = current.next;
            }
            Node08 newNode08 = new Node08(current, item, null);
            current.next = newNode08;
            size++;
        }
    }

    public void add(int item, int index)throws Exception{
        if (isEpmty()) {
            addFirst(item);

        } else if (index < 0 || index > size) {
            throw new Exception ("Nilai indeks di luar batas");
        } else {
            Node08 current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }
            if (current.prev == null) {
                Node08 newNode08 = new Node08(null, item, current);
                current.prev = newNode08;
                head =  newNode08;
            } else {
                Node08 newNode08 = new Node08(current.prev, item, current);
                newNode08.prev = current.prev;
                newNode08.next = current;
                current.prev.next = newNode08;
                current.prev = newNode08;
            }
        }
        size++;
    }

    public int size(){
        return size;
    }

    public void clear(){
        head = null;
        size = 0;
    }

    public void print(){
        if (!isEpmty()) {
            Node08 tmp = head;
            while (tmp != null) {
                System.out.print(tmp.data + "\t");
```

```
                tmp = tmp.next;
            }
            System.out.println("\nberhasil diisi");
        } else {
            System.out.println("Linked lists kosong");
        }
    }
}
```

Class Main

```
package doublelinkedlists;

public class DoubleLinkedListMain08 {
    public static void main(String[] args) throws Exception {
        DoubleLinkedLists08 dll =  new DoubleLinkedLists08();
        dll.print();
        System.out.println("Size : " + dll.size() );
        System.out.println("==========================================");
        dll.addFirst(3);
        dll.addLast(4);
        dll.addFirst(7);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println("==========================================");
        dll.add(40, 1);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println("==========================================");
        dll.clear();
        dll.print();
        System.out.println("Size : " + dll.size());
    }
}
```

Hasil run percobaan 1

```
Linked lists kosong
Size : 0
==========================================
7       3       4
berhasil diisi
Size : 3
==========================================
7       40      3       4
berhasil diisi
Size : 4
==========================================
Linked lists kosong
Size : 0
```

Pertanyaan Percobaan

1. Jelaskan perbedaan antara single linked list dengan double linked lists!

2. Perhatikan class Node, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?

3. Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan inisialisasi atribut head dan size seperti pada gambar berikut ini?

```
public DoubleLinkedLists() {
    head = null;
    size = 0;
}
```

4. Pada method addFirst(), kenapa dalam pembuatan object dari konstruktor class Node prev dianggap sama dengan null?

Node newNode = new Node(null, item, head);

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

6. Perhatikan isi method addLast(), apa arti dari pembuatan object Node dengan mengisikan parameter prev dengan current, dan next dengan null?

Node newNode = new Node(current, item, null);

7. Pada method add(), terdapat potongan kode program sebagai berikut:

```
while (i < index) {
    current = current.next;
    i++;
}
if (current.prev == null) {
    Node newNode = new Node(null, item, current);
    current.prev = newNode;
    head = newNode;
} else {
    Node newNode = new Node(current.prev, item, current);
    newNode.prev = current.prev;
    newNode.next = current;
    current.prev.next = newNode;
    current.prev = newNode;
}
```

jelaskan maksud dari bagian yang ditandai dengan kotak kuning.

Jawaban

1. Single linked list

- hanya memiliki pointer node untuk next
- Hanya dapat dilintasi dalam satu arah, dari awal daftar ke akhir.

Double linked list

- memiliki pointer next dan prev
- Dapat dilintasi dalam dua arah, dari awal ke akhir dan dari akhir ke awal.

2. atribut next untuk menunjukan pointer selanjutnya sedangkan atribut prev utnuk menunjukan pointer sebelumnya

3. kegunaan atribut head untuk menunjukan sebagai node pertama , dan size sebagai jumlah total node.

4. di saat kondisi isEmpty maka double linked list kosong maka prev dianggap null

5. Ketika menambahkan elemen baru ke awal linked list, kita harus membuat node baru dan menjadikannya sebagai kepala (head) dari linked list.

head.prev = newNode menghubungkan node yang sebelumnya menjadi kepala (sekarang menjadi head.prev) ke node baru yang akan menjadi kepala (sekarang menjadi head).

6. mengisikan prev dengan current, kita menghubungkan node baru tersebut dengan node terakhir saat ini dalam linked list, sehingga node baru tersebut akan menjadi node sebelumnya dari node yang akan kita tambahkan.

Mengisikan next dengan null menandakan bahwa node baru tersebut akan menjadi node terakhir dalam linked list, sehingga tidak ada node lain yang akan terhubung ke node tersebut setelahnya.

7.

Percobaan 2

Class DoubleLinkedList08

```
    public void removeFirst() throws Exception{
        if (isEmpty()) {
            throw new Exception("Linked List masih kosong, tidak dapat di
hapus");
        } else if (size == 1) {
            removeLast();
            size--;
        } else {
            head =  head.next;
            head.prev = null;
            size--;
        }
    }

    public void removeLast() throws Exception {
        if (isEmpty()) {
            throw new Exception("Linked list masih kosong, tidak dapat
dihapus");
        } else if (head.next == null) {
            head = null;
        } else {
            Node08 current = head;
            while (current.next.next != null) {
                current = current.next;
            }
            current.next = null;
        }
        size--;
    }


    public void remove(int index) throws Exception{
        if (isEmpty() || index >= size) {
            throw new Exception("Nilai indeks di luar batas");
        } else if (index == 0) {
            removeFirst();
        } else {
            Node08 current = head;
            int i =0;
            while (i < index) {
                current = current.next;
                i++;
            }
            if (current.next == null) {
                current.prev.next = null;
            } else if(current.prev == null){
                current.prev.next = null;
            } else {
                current.prev.next = current.next;
                current.next.prev = current.prev;
```

```
            }
            size--;
        }
    }
}
```

Tambahan Class Main

```
        dll.print();
        System.out.println("Size : " + dll.size());
        dll.addLast(50);
        dll.addLast(40);
        dll.addLast(10);
        dll.addLast(20);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println("==========================================");
        dll.removeFirst();
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println("==========================================");
        dll.removeLast();
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println("==========================================");
        dll.remove(1);
        dll.print();
        System.out.println("Size : " + dll.size());
```

Hasil run

```
Linked lists kosong
Size : 0
==========================================
7      3      4
berhasil diisi
Size : 3
==========================================
7      40     3      4
berhasil diisi
Size : 4
==========================================
Linked lists kosong
Size : 0
50     40     10     20
berhasil diisi
Size : 4
==========================================
40     10     20
berhasil diisi
Size : 3
==========================================
40     10
berhasil diisi
Size : 2
==========================================
40
berhasil diisi
Size : 1
```

Pertanyaan Percobaan 2

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

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

2. Bagaimana cara mendeteksi posisi data ada pada bagian akhir pada method removeLast()?

3. Jelaskan alasan potongan kode program di bawah ini tidak cocok untuk perintah remove!

```
Node tmp = head.next;

head.next=tmp.next;
tmp.next.prev=head;
```

4. Jelaskan fungsi kode program berikut ini pada fungsi remove!

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

Jawaban

1. head = head.next adalah menjadikan node setelah next menjadi head dan pointer head sebelumnya menjadi null.

2. Dalam loop ini, kita memeriksa apakah current.next.next tidak null. Jika tidak null, artinya ada setidaknya dua node lagi di depan current, yang berarti current belum berada di node kedua terakhir. Loop akan terus berlanjut hingga kita mencapai node kedua terakhir.

Setelah keluar dari loop, current akan menunjuk pada node kedua terakhir, dan kita menghapus koneksi dari node kedua terakhir tersebut ke node terakhir dengan current.next = null, sehingga menjadikan node kedua terakhir sebagai node terakhir dalam linked list.

3.

Percobaan 3

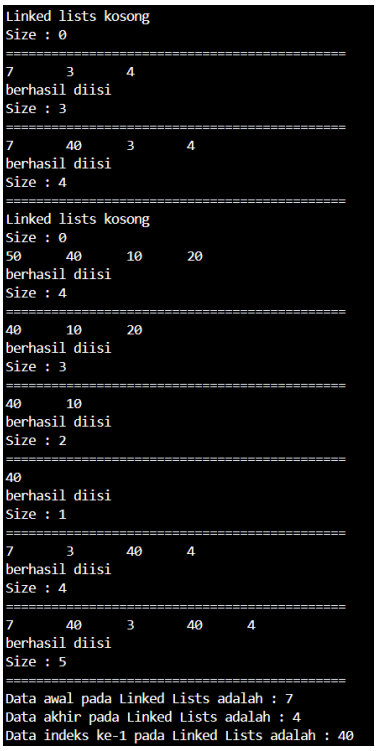
Tambahan pada class DoubleLinkedLists08

```
public int getFirst()throws Exception{
        if (isEmpty()) {
            throw new Exception("linked List kosong");
        }
        return head.data;
    }

    public int getLast()throws Exception{
        if (isEmpty()) {
            throw new Exception("Linked List kosong");
        }
        Node08 tmp = head;
        while (tmp.next != null) {
            tmp = tmp.next;
        }
        return tmp.data;
    }

    public int get(int index)throws Exception{
        if (isEmpty() || index >= size) {
            throw new Exception("Nilai indeks di luar batas.");
        }
        Node08 tmp = head;
        for (int i = 0; i < index; i++) {
            tmp = tmp.next;
        }
        return tmp.data;
    }
```

Tambahan pada class Main08

```
        dll.addFirst(3);
        dll.addLast(4);
        dll.addFirst(7);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println("=========================================");
        dll.add(40, 1);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println("=========================================");
        System.out.println("Data awal pada Linked Lists adalah : " +
dll.getFirst());
        System.out.println("Data akhir pada Linked Lists adalah : " +
dll.getLast());
        System.out.println("Data indeks ke-1 pada Linked Lists adalah : " +
dll.get(1));
```

Hasil run



Pertanyaan Percobaan

1. Jelaskan method size() pada class DoubleLinkedLists!

2. Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke- 1!

3. Jelaskan perbedaan karakteristik fungsi Add pada Double Linked Lists dan Single Linked Lists!

4. Jelaskan perbedaan logika dari kedua kode program di bawah ini!

```
public boolean isEmpty(){
    if(size ==0){
        return true;
    } else{
        return false;
    }
}
```

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

Tugas 1

Class Node08

```
package Tugas1;

import java.util.Scanner;

public class MainTugas108 {
    public static void main(String[] args)throws Exception {
        Scanner sc = new Scanner(System.in);
```

```
        DoubleLinkedListTugas108 dll = new DoubleLinkedListTugas108();
        int noAntrian;
        String nama;
        boolean run = true;
        do {
        System.out.println("++++++++++++++++++++++++++++++++++");
        System.out.println("PENGANTRI VAKSIN EXTRAVAGANZA");
        System.out.println("++++++++++++++++++++++++++++++++++");
        System.out.println("1. Tambah Data Penerima Vaksin");
        System.out.println("2. Hapus Data Pengantri Vaksin");
        System.out.println("3. Daftar Penerima Vaksin");
        System.out.println("4. Keluar");
        System.out.println("++++++++++++++++++++++++++++++++++");
        System.out.print("Masukan Pilihan Anda : ");
        int pilihan = sc.nextInt();

        switch (pilihan) {
            case 1:
            System.out.println("----------------------------------");
            System.out.println("Masukan Data Penerima Vaksin");
            System.out.println("----------------------------------");
            System.out.print("Nomor Antrian : ");
            noAntrian = sc.nextInt();
            System.out.print("Nama Penerima : ");
            nama = sc.next();
            dll.addLast(noAntrian, nama);
                break;
            case 2:
            System.out.println("----------------------------------");
            System.out.println("Masukan Data Penerima Vaksin");
            System.out.println("----------------------------------");
            dll.removeFirst();
                break;
            case 3:
            dll.print();
                break;
            case 4:
                run = false;
                break;
            default:
                break;
        }
        } while (run);
    }

}
```

Class DoubleLinkedLists108

```
package Tugas1;

public class DoubleLinkedListTugas108 {
    Node08 head,tail;
    int size;
```

```
    public DoubleLinkedListTugas108(){
        head = null;
        size = 0;
    }

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

    public void addFirst(int item, String nama){
        if (isEmpty()) {
            head = new Node08(head, tail, item, nama);

        } else {
            Node08 newNode08 = new Node08(head, tail, item, nama);
            head.prev = newNode08;
            head = newNode08;
        }
        size++;
    }

    public void addLast (int item, String nama){
        if (isEmpty()) {
            addFirst(item, nama);
        } else {
            Node08 current = head;
            while (current.next != null) {
                current = current.next;
            }
            Node08 newNode08 = new Node08(current, tail, item, nama);
            current.next = newNode08;
            size++;
        }
    }

        void print() {
            if (!isEmpty()) {
                Node08 tmp = head;
                System.out.println("Isi Linked List:");
                while (tmp != null) {
                    System.out.println("\nnoAntrian : " + tmp.noAntrian +
"\tNama: " + tmp.nama);
                    tmp = tmp.next;
                }
            } else {
                System.out.println("Linked List kosong");
            }
        }

        public void removeFirst() throws Exception{
            if (isEmpty()) {
                throw new Exception("Linked List masih kosong, tidak dapat
di hapus");
```

```
            } else if (size == 1) {
                removeLast();
                size--;
            } else {
                head =  head.next;
                head.prev = null;
                size--;
            }
        }

        public void removeLast() throws Exception {
            if (isEmpty()) {
                throw new Exception("Linked list masih kosong, tidak dapat
dihapus");
            } else if (head.next == null) {
                head = null;
            } else {
                Node08 current = head;
                while (current.next.next != null) {
                    current = current.next;
                }
                current.next = null;
            }
            size--;
        }

        public int size(){
            return size;
        }
}
```

Class Main

```
package Tugas1;

import java.util.Scanner;

public class MainTugas108 {
    public static void main(String[] args)throws Exception {
        Scanner sc = new Scanner(System.in);
        DoubleLinkedListTugas108 dll = new DoubleLinkedListTugas108();
        int noAntrian;
        String nama;
        boolean run = true;
        do {
        System.out.println("++++++++++++++++++++++++++++++++++");
        System.out.println("PENGANTRI VAKSIN EXTRAVAGANZA");
        System.out.println("++++++++++++++++++++++++++++++++++");
        System.out.println("1. Tambah Data Penerima Vaksin");
        System.out.println("2. Hapus Data Pengantri Vaksin");
        System.out.println("3. Daftar Penerima Vaksin");
        System.out.println("4. Keluar");
        System.out.println("++++++++++++++++++++++++++++++++++");
        System.out.print("Masukan Pilihan Anda : ");
        int pilihan = sc.nextInt();
```

```
        switch (pilihan) {
            case 1:
            System.out.println("-----------------------------------");
            System.out.println("Masukan Data Penerima Vaksin");
            System.out.println("-----------------------------------");
            System.out.print("Nomor Antrian : ");
            noAntrian = sc.nextInt();
            System.out.print("Nama Penerima : ");
            nama = sc.next();
            dll.addLast(noAntrian, nama);
                break;
            case 2:
            System.out.println("-----------------------------------");
            System.out.println("Masukan Data Penerima Vaksin");
            System.out.println("-----------------------------------");
            dll.removeFirst();
                break;
            case 3:
            dll.print();
            System.out.println("Sisa Antrian : " + dll.size());
        System.out.println("=============================================");
                break;
            case 4:
                run = false;
                break;
            default:
                break;
        }
        } while (run);
    }

}
```

Hasil Run

```
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
++++++++++++++++++++++++++++++++++
Masukan Pilihan Anda : 1
---------------------------------
Masukan Data Penerima Vaksin
---------------------------------
Nomor Antrian : 1
Nama Penerima : Samid
++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
++++++++++++++++++++++++++++++++++
Masukan Pilihan Anda : 1
---------------------------------
Masukan Data Penerima Vaksin
---------------------------------
Nomor Antrian : 2
Nama Penerima : dimas
```

```
++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
++++++++++++++++++++++++++++++++++
Masukan Pilihan Anda : 1
----------------------------------
Masukan Data Penerima Vaksin
----------------------------------
Nomor Antrian : 3
Nama Penerima : adi
++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
++++++++++++++++++++++++++++++++++
Masukan Pilihan Anda : 3
Isi Linked List:

noAntrian : 1   Nama: Samid

noAntrian : 2   Nama: dimas

noAntrian : 3   Nama: adi
Sisa Antrian : 3
==========================================
++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
++++++++++++++++++++++++++++++++++
Masukan Pilihan Anda : 2
----------------------------------
```

```
++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
++++++++++++++++++++++++++++++++++
Masukan Pilihan Anda : 3
Isi Linked List:

noAntrian : 2   Nama: dimas

noAntrian : 3   Nama: adi
Sisa Antrian : 2
==========================================
++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
++++++++++++++++++++++++++++++++++
Masukan Pilihan Anda : 4
```

Tugas 2

1. tambah data pertama

```
==============================
DATA FILM LAYAR LEBAR
==============================
1. Tambah Data Awal
2. Tambah Data Akhir
3. Tambah Data Posisi Tertentu
4. Hapus Data Pertama
5. Hapus Data Terakhir
6. Hapus Data Tertentu
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
==============================
1
==============================
Masukkan Data Film Posisi Awal
==============================
ID Film:
100
Judul Film:
kakung
Rating Film:
4
==============================
```

2. Tambah data terakhir

```
==============================
DATA FILM LAYAR LEBAR
==============================
1. Tambah Data Awal
2. Tambah Data Akhir
3. Tambah Data Posisi Tertentu
4. Hapus Data Pertama
5. Hapus Data Terakhir
6. Hapus Data Tertentu
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
==============================
2
==============================
Masukkan Data Film Posisi Akhir
==============================
ID Film:
101
Judul Film:
kong
Rating Film:
5
==============================
```

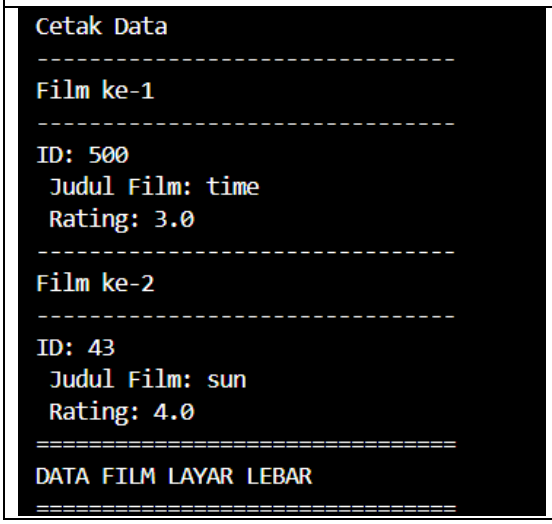
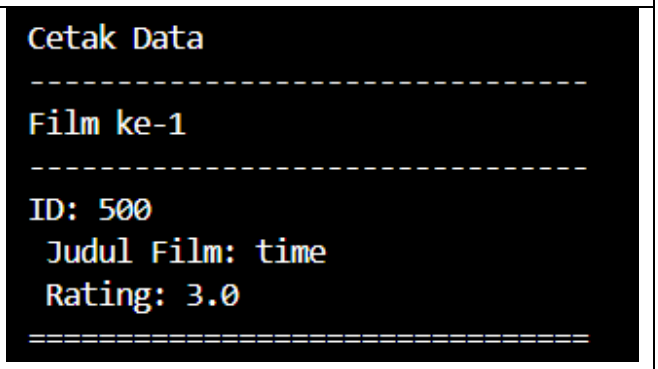
3. tambah data ke-

```
==============================
DATA FILM LAYAR LEBAR
==============================
1. Tambah Data Awal
2. Tambah Data Akhir
3. Tambah Data Posisi Tertentu
4. Hapus Data Pertama
5. Hapus Data Terakhir
6. Hapus Data Tertentu
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
==============================
3
==============================
Masukkan Data Film Urutan Ke-
==============================
ID Film:
500
Judul Film:
time
Rating Film:
3
Data Film ini akan masuk di urutan ke-
2
==============================
```

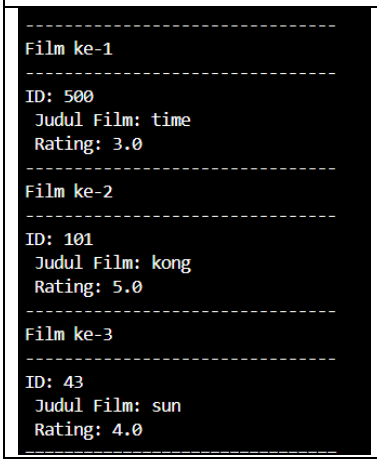
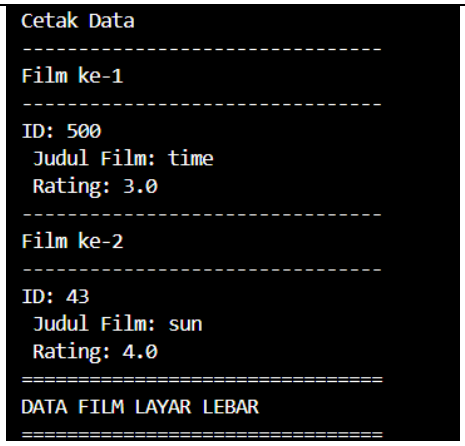
4. hapus data pertama

Sebelum	sesudah
========== Cetak Data ---------- Film ke-1 ---------- ID: 100 Judul Film: kakung Rating: 4.0 ---------- Film ke-2 ---------- ID: 500 Judul Film: time Rating: 3.0 ---------- Film ke-3 ---------- ID: 101 Judul Film: kong Rating: 5.0 ==========	Cetak Data ---------- Film ke-1 ---------- ID: 500 Judul Film: time Rating: 3.0 ---------- Film ke-2 ---------- ID: 101 Judul Film: kong Rating: 5.0 ----------

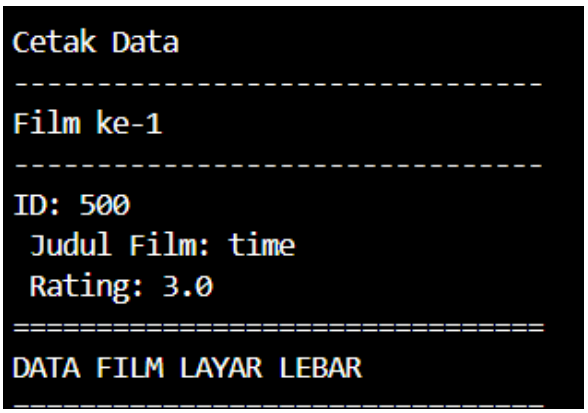
5. hapus data terakhir

Sebelum	sesudah
	

6. hapus data ke-

sebelum	sesudah
	

7. cetak data



8. cari id film

```
================================
Masukkan ID Film yang dicari:
================================
43
Data ID Film: 43 berada di node ke-1
IDENTITAS:
ID: 43
 Judul Film: sim
 Rating: 5.0
```

9. urutan rating film -decs

```
Urutan Film Terbaik Berdasarkan Rating:
--------------------------------
Film ke-1
--------------------------------
ID: 43
 Judul Film: sim
 Rating: 5.0
--------------------------------
Film ke-2
--------------------------------
ID: 500
 Judul Film: time
 Rating: 3.0
```