POLITEKNIK NEGERI MALANG

TEKNOLOGI INFORMASI

TEKNIK INFORMATIKA

Nama	: Muhammad Nuril Huda
Kelas	: TI-1A
No	: 19
Mata Kuliah	: Algoritma dan Struktur Data

12.2 Kegiatan Praktikum 1

12.2.1 Kode Program

- Kode Program Node

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

- Kode Program Doublelinkedlist

```
public class Doublelinkedlist {
    Node head;
    int size;

    Doublelinkedlist(){
        head = null;
        size = 0;
    }
    public boolean isEmpty(){
        return head == null;
    }
    public void addFirst(int item) {
        if (isEmpty()) {
            head = new Node(null, item, null);
        } else {
            Node newNode = new Node(null, item, head);
            head.prev = newNode;
            head = newNode;
        }
        size++;
    }
```

```
    public void addLast(int item) {
        if (isEmpty()) {
            addFirst(item);
        } else {
            Node current = head;
            while (current.next != null) {
                current = current.next;
            }
            Node newNode = new Node(current, item, null);
            current.next = newNode;
        }
        size++;
    }
    public void add(int item, int index) throws Exception {
        if (isEmpty()) {
            addFirst(item);
        } else if (index < 0 || index > size) {
            throw new Exception("Nilai indeks di luar batas");
        } else {
            Node current = head;
            int i = 0;
            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;
            }
        }
        size++;
    }
```

```
    public int size(){
        return size;
    }
    public void clear(){
        head = null;
        size = 0;
    }
    public void print() {
        if (!isEmpty()) {
            Node 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");
        }
    }
}
```

- Kode Program Doublelinkedlistmain

```
ublic class Doublelinkedlistmain {
    public static void main(String[] args) throws Exception {
        Doublelinkedlist dll = new Doublelinkedlist();
        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());
    }
}
```

12.2.2 Hasil Kode Program

```
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
```

12.2.3 Pertanyaan

1. Jelaskan perbedaan antara single linked list dengan double linked lists!
   - Single linked list hanya bisa bergerak ke depan yaitu dari head ke tail dan tidak bisa mundur ke node sebelumnya secara langsung. Sedangkan double linked list bisa bergerak ke depan dan ke belakang karena ada prev dan next.
2. Perhatikan class Node, didalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?
   - Next merupakan referensi ke node berikutnya dalam list, digunakan saat ingin maju ke node setelahnya. Sedangkan prev merupakan referensi ke node sebelumnya dalam list, digunakan saat ingin mundur ke node sebelumnya.
3. Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan inisialisasi atribut head dan size seperti pada gambar berikut ini?
   - Atribut head diinisialisasi dengan nilai null dan size dengan nilai 0 bertujuan untuk menandai bahwa linked list masih dalam keadaan kosong saat pertama kali dibuat.
4. Pada method addFirst(), kenapa dalam pembuatan object dari konstruktor class Node prev dianggap sama dengan null?
   `Node newNode = new Node(null, item, head);`
   - Prev diisi dengan null karena node tersebut akan menjadi node pertama (head) dalam double linked list. Pada struktur double linked list, node

pertama tidak memiliki node sebelumnya, sehingga referensi ke node sebelumnya (prev) harus diatur menjadi null.

5. Perhatikan pada method addFirst(). Apakah arti statement head.prev = newNode?
   - Statement head.prev = newNode; digunakan untuk menghubungkan node lama (head sebelumnya) dengan node baru yang ditambahkan di depan, agar pointer prev dari node lama mengarah ke node baru.
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);`
   - Pembuatan objek digunakan untuk menghubungkan node baru ke node terakhir sebelumnya (current) lewat prev dan Menandai bahwa node baru adalah **node terakhir** dengan next = null.

12.3 Kegiatan Praktikum 2

12.3.1 Kode Program

- Perubahan Kode Program Doublelinkedlist

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

- Perubahan Kode Program Doublelinkedlistmain

```
        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());
```

12.3.2 Hasil Kode Program

```
50      40      10      20
berhasil diisi
Size : 5
========================
40      10      20
berhasil diisi
Size : 4
========================
40      10
berhasil diisi
Size : 3
========================
40
berhasil diisi
Size : 2
```

12.3.3 Pertanyaan

1. Apakah maksud statement berikut pada method removeFirst()?
   ```
   head = head.next;
   head.prev = null;
   ```
   - `head = head,next`
     Baris ini menggeser pointer head ke node setelah node pertama. Artinya, node yang sebelumnya adalah node kedua kini menjadi node pertama atau head baru.
   - `head.prev = null;`
     Setelah head menunjuk ke node baru yang dulunya adalah node kedua, maka node ini perlu menghapus referensi ke node sebelumnya, yaitu node yang sudah dihapus tadi.
2. Bagaimana cara mendeteksi posisi data ada pada bagian akhir pada method removeLast()?
   - Pada method removeLast(), deteksi bahwa sebuah node berada di posisi akhir dilakukan dengan cara memeriksa apakah atribut next dari node tersebut bernilai null.

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;
```

   - Potongan kode tersebut tidak bisa menjadi perintah remove yang lengkap karena hanya spesifik menghapus node kedua dalam daftar. Kode ini tidak bisa menghapus node pertama, node terakhir, atau node di posisi lain. Selain itu, kode ini tidak memperbarui jumlah total node (size) dan tidak aman untuk kasus-kasus khusus seperti list kosong atau list yang hanya punya satu node, yang bisa menyebabkan error.

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

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

   - Dua baris kode tersebut berfungsi untuk menghapus node current dari tengah Double Linked List. Mereka bekerja dengan cara membuat node sebelum current langsung menunjuk ke node setelah current untuk maju, dan node setelah current langsung menunjuk ke node sebelum current untuk mundur, sehingga current terputus dari rantai.

12.4 Kegiatan Praktikum 3

12.4.1 Kode Program

- Perubahan Kode Program Doublelinkedlist

```
    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");
        }
        Node 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.");
        }
        Node tmp = head;
        for (int i = 0; i < index; i++) {
            tmp = tmp.next;
        }
        return tmp.data;
    }
```

- Perubahan Kode Program Doublelinkedlistmain

```
        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("=========================");
        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));
```

12.4.2 Hasil Kode Program

```
40
berhasil diisi
Size : 2
========================
7       3       40      4
berhasil diisi
Size : 5
========================
7       40      3       40      4
berhasil diisi
Size : 6
========================
Data awal pada Linked Lists adalah: 7
Data akhir pada Linked Lists adalah: 4
Data indeks ke-1 pada Linked Lists adalah: 40
```

12.4.3 Pertanyaan

1. Jelaskan method size() pada class DoubleLinkedLists!
    - Method size() pada DoubleLinkedLists berfungsi untuk mengembalikan jumlah total elemen atau node yang saat ini tersimpan di dalam linked list tersebut. Nilai ini dikelola oleh variabel size yang bertambah setiap kali ada penambahan node dan berkurang setiap kali ada penghapusan node.
2. Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke- 1!
    - Memodifikasi semua method yang menggunakan indeks dengan indeks awal 1 bukn 0
3. Jelaskan perbedaan karakteristik fungsi Add pada Double Linked Lists dan Single Linked Lists!
    - Perbedaan utama fungsi Add pada Double Linked Lists dan Single Linked Lists terletak pada penanganan pointer prev. Pada Single Linked Lists, setiap node hanya memiliki pointer next yang menunjuk ke node berikutnya, sehingga penambahan node hanya melibatkan pembaruan pointer next. Sebaliknya, pada Double Linked Lists, setiap node memiliki pointer next dan prev menunjuk ke node berikutnya dan sebelumnya, sehingga fungsi Add harus memperbarui kedua pointer next dan prev baik pada node yang baru ditambahkan maupun pada node-node di sekitarnya untuk menjaga koneksi dua arah yang benar.
4. Jelaskan perbedaan logika dari kedua kode program di bawah ini!

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

(a)

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

(b)

    - Logika a memeriksa apakah list kosong berdasarkan nilai dari variabel size. Jika size adalah 0, berarti list kosong.
    - Logika b memeriksa apakah list kosong berdasarkan status head. Jika head (pointer ke node pertama) adalah null, berarti tidak ada node di list dan list dianggap kosong.

12.5 Tugas

1. Kode Program
   - Kode Program VaksinasiNode19

```
public class VaksinasiNode19 {

    String noAntrian;

    String namaPenerima;

    VaksinasiNode19 prev;

    VaksinasiNode19 next;

    VaksinasiNode19 (VaksinasiNode19 prev, String noAntrian, String
namaPenerima, VaksinasiNode19 next){

        this.prev = prev;

        this.noAntrian = noAntrian;

        this.namaPenerima = namaPenerima;

        this.next = next;

    }

}
```

   - Kode Program VaksinasiDLL19

```
public class VaksinasiDLL19 {

    VaksinasiNode19 head;

    VaksinasiNode19 tail;

    int size;

    VaksinasiDLL19 (){

        head = null;

        tail = null;

        size = 0;

    }

    boolean isEmpty (){

        return head == null;

    }

    void Enqueue (String noAntrian, String namaPenerima){

        VaksinasiNode19 newNode = new VaksinasiNode19 (tail, noAntrian,
namaPenerima, null);

        if (isEmpty()) {

            head = newNode;

            tail = newNode;

        } else {

            tail.next = newNode;

            tail = newNode;

        }

        size++;

    }
```

```
void Dequeue () {
        if (isEmpty()) {
        System.out.println("Antrian masih kosong, tidak dapat dihapus!");
        }
        String namaDihapus = head.namaPenerima;
        if (size == 1) {
            head = null;
            tail = null;
        } else {
            head = head.next;
            head.prev = null;
        }
        size--;
        System.out.println(namaDihapus + " telah selesai divaksinasi.");
    }
    int size(){
        return size;
    }
    void print() {
        if (isEmpty()) {
            System.out.println("Linked Lists Kosong");
        } else {
            System.out.println("+++++++++++++++++++++++++++++");
            System.out.println("Daftar Pengantri Vaksin");
            System.out.println("+++++++++++++++++++++++++++++");
            System.out.println("|No.    |Nama   |");
            System.out.println("-----------------------------");
            VaksinasiNode19 tmp = head;
            while (tmp != null) {
                System.out.printf("|%-7s|%-7s|\n", tmp.noAntrian,
tmp.namaPenerima);
                tmp = tmp.next;
            }
            System.out.println("Sisa Antrian: " + size);
        }
    }
}
```

- Kode Program VaksinasiMain19

```
import java.util.Scanner;
public class VaksinasiMain19 {
    public static void main(String[] args) {
        Scanner sc = new Scanner (System.in);
        VaksinasiDLL19 antrianVaksin = new VaksinasiDLL19();
        int pilihan;
        do {
            System.out.println("\n++++++++++++++++++++++++++++");
            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("Pilih menu: ");
            pilihan = sc.nextInt();
            sc.nextLine();

        switch (pilihan) {
            case 1:
                System.out.println("----------------------------");
                System.out.println("Masukkan Data Penerima Vaksin");
                System.out.print("Nomor Antrian: ");
                String noAntrian = sc.nextLine();
                System.out.print("Nama Penerima: ");
                String nama = sc.nextLine();
                antrianVaksin.Enqueue(noAntrian, nama);
                System.out.println("Data " + nama + " dengan Nomor
Antrian " + noAntrian + " berhasil ditambahkan.");
                break;
```

```
            case 2:
                antrianVaksin.Dequeue();
                break;
            case 3:
                System.out.println("-----------------------------");
                antrianVaksin.print();
                break;
            case 4:
                System.out.println("Terima kasih telah menggunakan
layanan antrian vaksinasi!");
                break;
            default:
                System.out.println("Pilihan tidak valid. Silakan coba
lagi.");
        }
    } while (pilihan != 4);


    }


}
```

Hasil Kode Program

```
++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
----------------------------
Pilih menu: 1
----------------------------
Masukkan Data Penerima Vaksin
Nomor Antrian: 123
Nama Penerima: Joko
Data Joko dengan Nomor Antrian 123 berhasil ditambahkan.

++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
----------------------------
Pilih menu: 1
----------------------------
Masukkan Data Penerima Vaksin
Nomor Antrian: 456
Nama Penerima: Huda
Data Huda dengan Nomor Antrian 456 berhasil ditambahkan.
```

```
++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
----------------------------
Pilih menu: 1
----------------------------
Masukkan Data Penerima Vaksin
Nomor Antrian: 789
Nama Penerima: Nuril
Data Nuril dengan Nomor Antrian 789 berhasil ditambahkan.

++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
----------------------------
Pilih menu: 3
----------------------------
++++++++++++++++++++++++++++++
Daftar Pengantri Vaksin
++++++++++++++++++++++++++++++
|No.    |Nama   |
----------------------------
|123    |Joko   |
|456    |Huda   |
|789    |Nuril  |
Sisa Antrian: 3

++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
----------------------------
Pilih menu: 2
Joko telah selesai divaksinasi.
```

```
++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
----------------------------
Pilih menu: 3
----------------------------
++++++++++++++++++++++++++++++
Daftar Pengantri Vaksin
++++++++++++++++++++++++++++++
|No.    |Nama   |
----------------------------
|456    |Huda   |
|789    |Nuril  |
Sisa Antrian: 2

++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA
++++++++++++++++++++++++++++++
1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
----------------------------
Pilih menu: 4
Terima kasih telah menggunakan layanan antrian vaksinasi!
```

2. Kode Program
   - Kode Program Film19

```
public class Film19 {
    int id;
    String judul;
    double rating;
    public Film19(int id, String judul, double rating) {
        this.id = id;
        this.judul = judul;
        this.rating = rating;
    }
    public String toString() {
        return "ID: " + id + ", Judul: " + judul + ", Rating: " +
rating;
    }
}
```

   - Kode Program FilmNode19

```
public class FilmNode19 {
    Film19 data;
    FilmNode19 prev;
    FilmNode19 next;
    public FilmNode19(FilmNode19 prev, Film19 data, FilmNode19 next) {
        this.prev = prev;
        this.data = data;
        this.next = next;
    }
}
```

   - Kode Program FilmDLL19

```
public class FilmDLL19 {
    FilmNode19 head;
    FilmNode19 tail;
    int size;
    public FilmDLL19() {
        head = null;
        tail = null;
        size = 0;
    }
```

```
    public boolean isEmpty() {
        return head == null;
    }
    public void addLast(Film19 film) {
        FilmNode19 newNode = new FilmNode19(tail, film, null);
        if (isEmpty()) {
            head = newNode;
            tail = newNode;
        } else {
            tail.next = newNode;
            tail = newNode;
        }
        size++;
    }
    public void addFirst(Film19 film) {
        FilmNode19 newNode = new FilmNode19(null, film, head);
        if (isEmpty()) {
            head = newNode;
            tail = newNode;
        } else {
            head.prev = newNode;
            head = newNode;
        }
        size++;
    }
    public void add(Film19 film, int index) throws Exception {
        if (index < 0 || index > size) {
            throw new Exception("Nilai indeks di luar batas");
        }
        if (index == 0) {
            addFirst(film);
        } else if (index == size) {
            addLast(film);
        } else {
            FilmNode19 current = head;
            for (int i = 0; i < index; i++) {
                current = current.next;
            }
            FilmNode19 newNode = new FilmNode19(current.prev, film, current);
            current.prev.next = newNode;
            current.prev = newNode;
            size++;
        }
    }
```

```
    public Film19 remove(int index) throws Exception {
        if (isEmpty() || index < 0 || index >= size) {
            throw new Exception("Nilai indeks di luar batas");
        }
        if (index == 0) {
            return removeFirst();
        } else if (index == size - 1) {
            return removeLast();
        } else {
            FilmNode19 current = head;
            for (int i = 0; i < index; i++) {
                current = current.next;
            }
            Film19 removedFilm = current.data;
            current.prev.next = current.next;
            current.next.prev = current.prev;
            size--;
            return removedFilm;
        }
    }
    public Film19 removeById(int idFilm) throws Exception {
        if (isEmpty()) {
            throw new Exception("Linked List kosong, tidak ada film untuk
dihapus!");
        }
        FilmNode19 current = head;
        int index = 0;
        while (current != null) {
            if (current.data.id == idFilm) {
                return remove(index);
            }
            current = current.next;
            index++;
        }
        throw new Exception("Film dengan ID " + idFilm + " tidak
ditemukan!");
    }
```

```
    public Film19 findById(int idFilm) {
        FilmNode19 current = head;
        while (current != null) {
            if (current.data.id == idFilm) {
                return current.data;
            }
            current = current.next;
        }
        return null;
    }
    public void sortByRatingDescending() {
        if (size <= 1) {
            return;
        }
        boolean swapped;
        do {
            swapped = false;
            FilmNode19 current = head;
            while (current != null && current.next != null) {
                // Bandingkan rating dari current dan current.next
                if (current.data.rating < current.next.data.rating) {
                    // Tukar data filmnya saja (bukan node-nya)
                    Film19 tempFilm = current.data;
                    current.data = current.next.data;
                    current.next.data = tempFilm;
                    swapped = true;
                }
                current = current.next;
            }
        } while (swapped);
    }
```

```
    public void print() {
        if (isEmpty()) {
            System.out.println("Linked Lists Kosong");
        } else {
            System.out.println("+++++++++++++++++++++++++++++");
            System.out.println("Daftar Film");
            System.out.println("+++++++++++++++++++++++++++++");
            System.out.printf("| %-5s | %-25s | %-7s |\n", "ID", "Judul
Film", "Rating");
            System.out.println("-----------------------------------------
------");
            FilmNode19 tmp = head;
            while (tmp != null) {
                System.out.printf("| %-5d | %-25s | %-7.1f |\n",
tmp.data.id, tmp.data.judul, tmp.data.rating);
                tmp = tmp.next;
            }
            System.out.println("-----------------------------------------
------");
            System.out.println("Total Film: " + size);
        }
    }
}
```

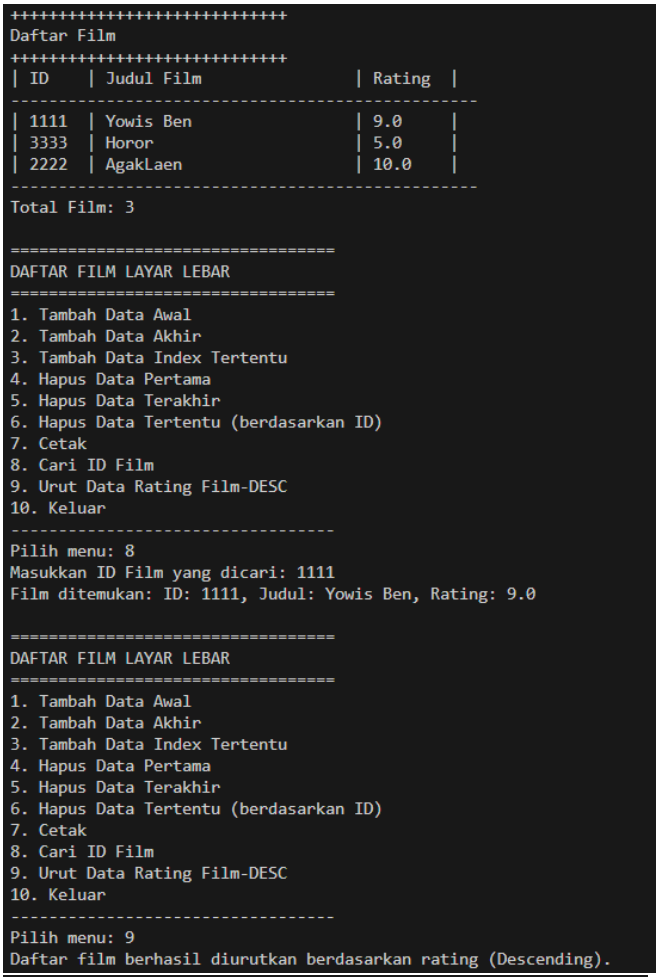
Hasil Kode Program

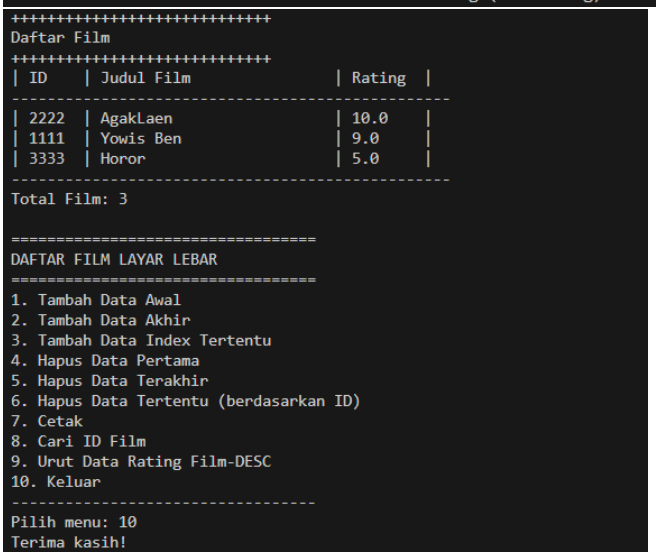
```
==================================
DAFTAR FILM LAYAR LEBAR
==================================
1. Tambah Data Awal
2. Tambah Data Akhir
3. Tambah Data Index Tertentu
4. Hapus Data Pertama
5. Hapus Data Terakhir
6. Hapus Data Tertentu (berdasarkan ID)
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
----------------------------------
Pilih menu: 1
--- Tambah Data Awal ---
ID Film: 1111
Judul Film: Yowis Ben
Rating Film: 9
Film berhasil ditambahkan di awal.
```

```
==================================
DAFTAR FILM LAYAR LEBAR
==================================
1. Tambah Data Awal
2. Tambah Data Akhir
3. Tambah Data Index Tertentu
4. Hapus Data Pertama
5. Hapus Data Terakhir
6. Hapus Data Tertentu (berdasarkan ID)
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
----------------------------------
Pilih menu: 2
--- Tambah Data Akhir ---
ID Film: 2222
Judul Film: AgakLaen
Rating Film: 10
Film berhasil ditambahkan di akhir.

==================================
DAFTAR FILM LAYAR LEBAR
==================================
1. Tambah Data Awal
2. Tambah Data Akhir
3. Tambah Data Index Tertentu
4. Hapus Data Pertama
5. Hapus Data Terakhir
6. Hapus Data Tertentu (berdasarkan ID)
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
----------------------------------
Pilih menu: 3
--- Tambah Data Index Tertentu ---
ID Film: 3333
Judul Film: Horor
Rating Film: 5
Masukkan Posisi Index: 1
Film berhasil ditambahkan pada indeks 1.
```

```
==================================
DAFTAR FILM LAYAR LEBAR
==================================
1. Tambah Data Awal
2. Tambah Data Akhir
3. Tambah Data Index Tertentu
4. Hapus Data Pertama
5. Hapus Data Terakhir
6. Hapus Data Tertentu (berdasarkan ID)
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
----------------------------------
Pilih menu: 7
```





Link Github: https://github.com/nurilhuda05/Algoritma-dan-Struktur-Data.git