LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN STRUKTUR DATA

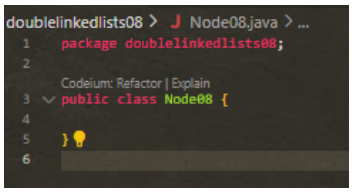
Oleh:
DZULFIKAR MUHAMMAD AL GHIFARI
NIM. 2341760071
SIB-1F / 08
D-IV SISTEM INFORMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

PRAKTIKUM 12

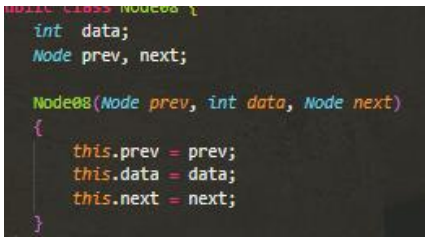
12.2 Percobaan 1

12.2.1 Langkah Langkah percobaan

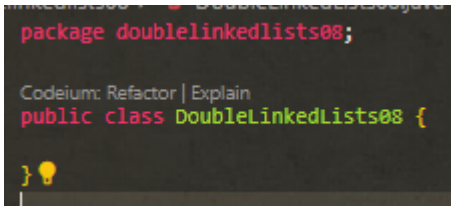
1. Membuat package dan class



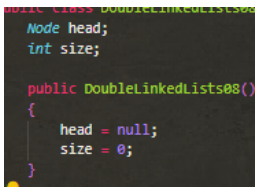
2. Deklarasi variable dan constructor



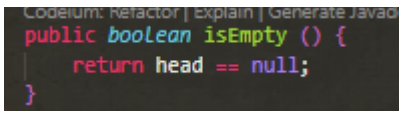
3. Membuat package dan class double linked lists



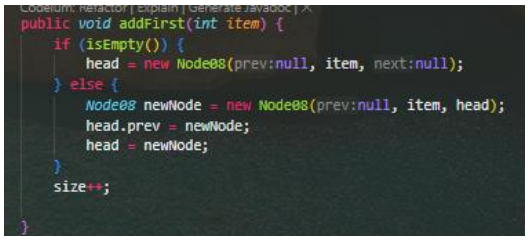
4. Deklarasi variable dan konnstruktor



5. Menambahkan method isEmpty



6. Menambahkan method addFirst



7. Menambahkan method addLast

```
public void addlast(int item) {
    if (isEmpty()) {
        addFirst(item);
    } else {
        Node08 current = head;
        while (current.next != null) {
            current = current.next;
        }

        Node08 newNode = new Node08(current, item, next:null);
        current.next = newNode;
        size++;
    }
}
```

8. Menambahkan method add

```
public void add(int item, int index) throws Exception {
    if (isEmpty()) {
        addFirst(item);
    } else if (index < 0 || index > size) {
        throw new Exception(message:"Nilai indeks di luar batas");
    } else {
        Node08 current = head;
        int i = 0;
        while (i < index) {
            current = current.next;
            i++;
        }

        if (current.prev == null) {
            Node08 newNode = new Node08(prev:null, item, current);
            current.prev = newNode;
            head = newNode;
        } else {
            Node08 newNode = new Node08(current.prev, item, current);
            newNode.prev = current.prev;
            newNode.next = current;
            current.prev.next = newNode;
            current.prev = newNode;
        }
    }
    size++;
}
```

9. Membuat method size

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

10. Membuat method clear

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

11. Membuat method print

```
public void print() {
    if (!isEmpty()) {
        Node08 tmp = head;
        while (tmp != null) {
            System.out.print(tmp.data + "\t");
            tmp = tmp.next;
        }
        System.out.println(x:"\nberhasil diisi");
    } else {
        System.out.println(x:"Linked Lists Kosong");
    }
}
```

12. Membuat class main

```
package doublelinkedlists08;

Codeium: Refactor | Explain
public class DoubleLinkedListsMain08 {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) {

    }
}
```

13. Mengisi method main

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

VERIFIKASI HASIL PERCOBAAN 12.2.2

```
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 12.2.3

1. **Jelaskan perbedaan antara single linked list dengan double linked lists!**
   Jika single linked hanya ada satu arah, yaitu dari node awal – akhir, dan hanya memiliki satu node yaitu next.
   Double linked memiliki dua arah, yaitu bolak balik antar node, dan memiliki dua node yaitu next dan prev.

2. **Perhatikan class Node, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?**
   Attribute next berfungsi untuk menunjukan node berikutnya, prev berfungsi untuk menunjukan node sebelumnya.

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

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

Head menunjukan pointer node pertama dalam daftar, size menyimpan jumlah node dalam daftar

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

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

AddFirst befungsi untuk menambahkan data pada saat pertama kali. Karena isi daftar node pada awal berisikan 0, tidak ada data. Jadi tidak ada yang bisa untuk dilakukan prev

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

Untuk mengatur node yang sebelumnya menjadi head untuk menunjuk ke node baru yang kita tambahkan 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);

AddLast befungsi untuk menambahkan data pada urutan terakhir. Karena akan di tempatkan pada posisi terakhir. Jadi tidak ada yang bisa untuk dilakukan next

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

untuk mendeteksi apakah posisi sebelumnya null, jika ya maka posisi sebelumnya akan di isi oleh data baru, dan head akan ada pada posisi data yang baru saja ditambahkan

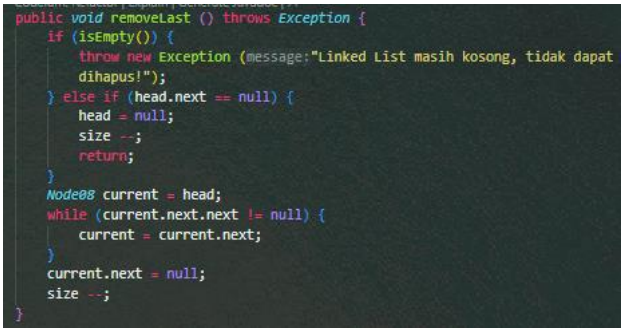
12.3 Percobaan 2

12.3.1 Langkah-langkah Percobaan

1. Menambahkan method removeFirst

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

2. Menambahkan method removeLast



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

3. Menambahkan method remove

```
public void remove(int index) throws Exception {
    if (isEmpty() || index >= size) {
        throw new Exception (message:"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 = current.next;
            current.prev = null;
            head = current;
        } else {
            current.prev.next = current.next;
            current.next.prev = current.prev;
        }
        size --;
    }
}
```

4. Menambahkan kode pada main

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

VERIFIKASI HASIL PERCOBAAN 12.3.2

```
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 12.3.3

1. **Apakah maksud statement berikut pada method *removeFirst()*?**
   head = head.next;
   head.prev = null;

   head=head.next berfungsi untuk memperbarui pointer head yang menunjukan ke node berikutnya setelah node pertama yang akan dihapus.
   head.prev=null berfungsi untuk menghapus ponter prev karena head berasa pada daftar pertama

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

```
Node08 current = head;
while (current.next.next != null) {
    current = current.next;
}
current.next = null;
```

   Dengan menggunakan iterasi loop while, mencari current.next hingga ke daftar yang terakhir. Karena daftar terakhir tidak bisa melakukan .next, jadi data tersebut menunjukan data terakhir, dan ponter akan di simpan di variable current

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;
```

Karena kode program di atas tidak memeriksa kondisi spesifik dari daftar

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

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

Kode program pertama berfungsi memperbarui pointer next dari node sebelumnya untuk menunjuk ke node berikutnya
Kode program kedua berfungsi memperbarui ponter prev dari node berikutnya untuk menunjuk ke node sebelumnya

12.4 Percobaan 2

12.4.1 Langkah-langkah Percobaan

1. Menambahkan method getFIrst

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

2. Menambahkan method getLast

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

3. Menambahkan method get

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

4. Menambahkan kode pada main

```
dll.print ();
System.out.println("Size: " + dll.size());
System.out.println (x:"================================");
dll.addFirst(item:3);
dll.addlast(item:4);
dll.addFirst (item:7);
dll.print();
System. out.println("Size: " + dll.size());
System.out.println (x:"================================");
dll.add(item:40, index:1);
dll.print();
System. out.println("Size: " + dll.size());
System.out.println (x:"================================");
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
(index:1));
```

VERIFIKASI HASIL PERCOBAAN 12.4.2

```
Linked Lists Kosong
Size: 0
================================
7       3       4
berhasil diisi
Size: 3
================================
7       40      3       4
berhasil diisi
Size: 4
================================
Data awal pada Linked Lists adalah: 7
Data akhir pada Linked Lists adalah: 4
Data indeks ke-1 pada Linked Lists adalah: 40
```

PERTANYAAN 12.4.3

1. **Jelaskan method size() pada class DoubleLinkedLists!**
   Size() mengembalikan nilai size yang telah di atur pada awal menjalankan program
2. **Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke-1**
   Secara default index umumnya dimulai dari 0. Namun dengan dengan sedikit mengubah pada method add dengan memberikan validasi untuk index seperti dibawah ini

```
(index < 1 || index > size) {
```

3. **Jelaskan perbedaan karakteristik fungsi Add pada Double Linked Lists dan Single Linked Lists!**
   Penambahan pada single linked hanya satu arah dari depan ke belakang, tidak bisa menambahkan elemen di tengah daftar
   Penambahan pada double linked memiliki 2 arah, bisa dari depan dan belakang, bisa menyisipkan kode di mana saja
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)

   pada kode A akan memeriksa jumlah elemen, jika jumlah elemen tersebut bernilai 0 maka akan mengembalikan true, namun jika tidak sama dengan 0 maka akan mengembalikan false
   Pada kode B akan memeriksa ponter head apakah berisikan null, jika iya maka akan mengembalikan true, jika tidak null maka kan mengembalikan false

8.4 TUGAS

1. Buat program antrian vaksinasi menggunakan queue berbasis double linked list sesuai ilustrasi dan menu di bawah ini! (counter jumlah antrian tersisa di menu cetak(3) dan data orang yang telah divaksinasi di menu Hapus Data(2) harus ada

Class faksinasi

```java
package tugas1;

Codeium: Refactor | Explain
public class Faksinasi08 {
    String nama;
    int noAntri;

    Faksinasi08(){}

    Faksinasi08 (int noAntri, String nama) {
        this.noAntri = noAntri;
        this.nama = nama;
    }

}
```

Class Node

```java
package tugas1;

// import org.w3c.dom.Node;

Codeium: Refactor | Explain
public class Node08 {
    Faksinasi08  data;
    Node08 prev, next;

    Node08(Node08 prev, Faksinasi08 data, Node08 next)
    {
        this.prev = prev;
        this.data = data;
        this.next = next;
    }
}
```

Class double berisikan hasil copy dari percobaan 1-3

```java
package tugas1;

Codeium: Refactor | Explain
public class DoubleLinkedLists08 {
    Node08 head;
    int size;

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

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public boolean isEmpty() {
        return head == null;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public void addFirst(Faksinasi08 item) {
        if (isEmpty()) {
            head = new Node08(prev:null, item, next:null);
        } else {
            Node08 newNode = new Node08(prev:null, item, head);
            head.prev = newNode;
            head = newNode;
        }
        size++;

    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public void addlast(Faksinasi08 item) {
        if (isEmpty()) {
            addFirst(item);
        } else {
            Node08 current = head;
            while (current.next != null) {
                current = current.next;
            }
```

Class Main

```
public static void menu() {
    System.out.println(x:"+++++++++++++++++++++++++++++++++++ ");
    System.out.println(x:"PENGANTRI VAKSIN EXTRAVAGANZA 08 ");
    System.out.println(x:"+++++++++++++++++++++++++++++++++++ ");
    System.out.println();
    System.out.println(x:"1. Tammbah Data Penerima Vaksin");
    System.out.println(x:"2. Hapus Data Pengantri Vaksin");
    System.out.println(x:"3. Daftar Penerima Vaksin");
    System.out.println(x:"4. Keluar");
    System.out.println(x:"+++++++++++++++++++++++++++++++++++ ");
}
```

```
do {
    menu();
    pilih = sc.nextInt();
    switch (pilih) {
        case 1:
            System.out.println(x:"----------------------------");
            System.out.println(x:"Masukkan Data Penerima Vaksin");
            System.out.println(x:"----------------------------");
            System.out.print(s:"Nomor Antrian: ");
            int noAntrian = sc.nextInt();
            System.out.print(s:"Nama Penerima: ");
            String nama = sc.next();
            Faksinasi08 nb = new Faksinasi08(noAntrian, nama);
            dll.addlast(nb);
            System.out.println();
            break;
        case 2:
            Faksinasi08 penerima = dll.getFirst();
            System.out.println(penerima.nama + " telah selesain divaksinasi.");
            dll.removeFirst();
            break;
        case 3:
            System.out.println(x:"----------------------------");
            System.out.println(x:"Daftar Pengantri Vaksin");
            System.out.println(x:"----------------------------");
            dll.print();
            System.out.println("Sisa Antrian: " + dll.size());
            System.out.println();
            break;
        case 4:
            return;
    }
} while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
```

Hasil Compile

```
+++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA 08
+++++++++++++++++++++++++++++++++++

1. Tammbah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
+++++++++++++++++++++++++++++++++++
3
----------------------------
Daftar Pengantri Vaksin
----------------------------
Nomor   |        Nama
123     |       JOko
124     |       Mely
135     |       Johan
146     |       Rosi

berhasil diisi
Sisa Antrian: 4
```

```
+++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA 08
+++++++++++++++++++++++++++++++++++

1. Tammbah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
+++++++++++++++++++++++++++++++++++
1
----------------------------
Masukkan Data Penerima Vaksin
----------------------------
Nomor Antrian: 146
Nama Penerima: Rosi
```

```
+++++++++++++++++++++++++++++++++++
PENGANTRI VAKSIN EXTRAVAGANZA 08
+++++++++++++++++++++++++++++++++++

1. Tammbah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
+++++++++++++++++++++++++++++++++++
2
JOko telah selesain divaksinasi.
```

2. Buatlah program daftar film yang terdiri dari id, judul dan rating menggunakan double linked lists, bentuk program memiliki fitur pencarian melalui ID Film dan pengurutan Rating secara descending. Class Film wajib diimplementasikan dalam soal ini.

Class film

```java
package tugas2;

Codeium: Refactor | Explain
public class Film08 {
    int id;
    String judul;
    double rating;

    Film08(){}

    Film08 (int id, String judul, double rating) {
        this.id = id;
        this.judul = judul;
        this.rating = rating;
    }
}
```

Class node

```java
Codeium: Refactor | Explain
public class Node08 {
    Film08  data;
    Node08 prev, next;

    Node08(Node08 prev, Film08 data, Node08 next)
    {
        this.prev = prev;
        this.data = data;
        this.next = next;
    }
}
```

Perubahan pada class doublelinkedlist

```java
Codeium: Refactor | Explain | Generate Javadoc | ×
public void print() {
    if (!isEmpty()) {
        Node08 tmp = head;
        System.out.println(x:"Cetak Data");
        while (tmp != null) {
            System.out.println("ID \t: " +tmp.data.id);
            System.out.println("  Judul \r: " +tmp.data.judul);
            System.out.println("  Rating \r: " +tmp.data.rating);
            tmp = tmp.next;
        }
        // System.out.println("\nberhasil diisi");
    } else {
        System.out.println(x:"Linked Lists Kosong");
    }
}
```

Penambahan pada class double linkedlist

```java
Codeium: Refactor | Explain | Generate Javadoc | ×
public Film08 searchById(int id) throws Exception {
    if (isEmpty()) {
        throw new Exception(message:"Linked List masih kosong");
    }

    Node08 current = head;
    while (current != null) {
        if (current.data.id == id) {
            return current.data;
        }
        current = current.next;
    }

    throw new Exception("Film dengan ID " + id + " tidak ditemukan");
}
```

```java
public void sortByRatingDesc() {
    if (isEmpty() || size == 1) {
        return;
    }

    for (int i = 0; i < size - 1; i++) {
        Node08 current = head;
        Node08 maxNode = current;

        for (int j = 0; j < size - i - 1; j++) {
            if (current.next != null && current.next.data.rating > maxNode.data.
            rating) {
                maxNode = current.next;
            }
            current = current.next;
        }

        if (maxNode != current) {
            Film08 temp = current.data;
            current.data = maxNode.data;
            maxNode.data = temp;
        }
    }
}
```

Class main

```
Codeium: Refactor | Explain | Generate Javadoc | ×
public static void menu() {
    System.out.println(x:"================================== ");
    System.out.println(x:"DATA FILM LAYAR LEBAR 08 ");
    System.out.println(x:"================================== ");
    System.out.println();
    System.out.println(x:"1. Tammbah Data Awal");
    System.out.println(x:"2. Tammbah Data Akhir");
    System.out.println(x:"3. Tammbah Data Index Tertentu");
    System.out.println(x:"4. Hapus Data Pertama");
    System.out.println(x:"5. Hapus Data Terkahir");
    System.out.println(x:"6. Hapus Data Tertentu");
    System.out.println(x:"7. Cetak");
    System.out.println(x:"8. Cari ID Film");
    System.out.println(x:"9. Urut Data Rating Film-DESC");
    System.out.println(x:"10. Keluar");
    System.out.println(x:"================================== ");
}
```

```
switch (pilih) {
    case 1:
        System.out.println(x:"----------------------------");
        System.out.println(x:"Masukkan Data Posisi Awal");
        System.out.println(x:"----------------------------");
        System.out.print(s:"ID : ");
        int id = sc.nextInt();
        System.out.print(s:"Judul Film : ");
        String judul = sc.next();
        System.out.print(s:"Rating (ex. 4.5) : ");
        double rating = sc.nextDouble();
        Film08 nb = new Film08(id, judul, rating);
        dll.addFirst(nb);
        System.out.println();
        break;
    case 2:
        System.out.println(x:"----------------------------");
        System.out.println(x:"Masukkan Data Posisi Akhir");
        System.out.println(x:"----------------------------");
        System.out.print(s:"ID : ");
        int id1 = sc.nextInt();
        System.out.print(s:"Judul Film : ");
        String judul1 = sc.next();
        System.out.print(s:"Rating (ex. 4.5) : ");
        double rating1 = sc.nextDouble();
        Film08 nb1 = new Film08(id1, judul1, rating1);
        dll.addFirst(nb1);
        System.out.println();
        break;
```

```
case 3:
    System.out.println(x:"----------------------------");
    System.out.println(x:"Masukkan Data FILM");
    System.out.println(x:"----------------------------");
    System.out.print(s:"Urutan ke : ");
    int index = sc.nextInt();
    System.out.print(s:"ID : ");
    int id2 = sc.nextInt();
    System.out.print(s:"Judul Film : ");
    String judul2 = sc.next();
    System.out.print(s:"Rating (ex. 4.5) : ");
    double rating2 = sc.nextDouble();
    Film08 nb2 = new Film08(id2, judul2, rating2);
    dll.add(nb2, index);
    System.out.println();
    break;
case 4:
    Film08 film = dll.getFirst();
    System.out.println("Film " +film.judul + " telah dihapus.");
    dll.removeFirst();
    System.out.println();
    break;
case 5:
    Film08 film1 = dll.getLast();
    System.out.println("Film " +film1.judul + " telah dihapus.");
    dll.removeLast();
    System.out.println();
    break;
```

```
case 6:
    System.out.print(s:"Urutan ke : ");
    int index1 = sc.nextInt();
    Film08 film2 = dll.get(index1);
    System.out.println("Film " +film2.judul + " telah dihapus.");
    dll.remove(index1);
    break;
case 7:
    System.out.println(x:"----------------------------");
    System.out.println(x:"DATA FILM LAYAR LEBAR");
    System.out.println(x:"----------------------------");
    dll.print();
    System.out.println();
    break;
case 8:
    System.out.print(s:"Masukkan ID : ");
    int idSearch = sc.nextInt();
    Film08 seach = dll.searchById(idSearch);
    System.out.println("ID \t: " +seach.id);
    System.out.println("Judul \t: " +seach.judul);
    System.out.println("Rating \t: " +seach.rating);
    break;
case 9:
    dll.sortByRatingDesc();
    System.out.println(x:"----------------------------");
    System.out.println(x:"DATA FILM LAYAR LEBAR SORTING DESC");
    System.out.println(x:"----------------------------");
    dll.print();
    System.out.println();
    break;
case 10:
    return;
```

Hasil compile

```
==================================
DATA FILM LAYAR LEBAR 08
==================================

1. Tammbah Data Awal
2. Tammbah Data Akhir
3. Tammbah Data Index Tertentu
4. Hapus Data Pertama
5. Hapus Data Terkahir
6. Hapus Data Tertentu
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
==================================
1
----------------------------
Masukkan Data Posisi Awal
----------------------------
ID : 123
Judul Film : Simerah
Rating (ex. 4.5) : 3.4
```

```
DATA FILM LAYAR LEBAR 08
==================================

1. Tammbah Data Awal
2. Tammbah Data Akhir
3. Tammbah Data Index Tertentu
4. Hapus Data Pertama
5. Hapus Data Terkahir
6. Hapus Data Tertentu
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
==================================
2
----------------------------
Masukkan Data Posisi Akhir
----------------------------
ID : 124
Judul Film : Sibiru
Rating (ex. 4.5) : 2.8
```

```
==================================
DATA FILM LAYAR LEBAR 08
==================================

1. Tammbah Data Awal
2. Tammbah Data Akhir
3. Tammbah Data Index Tertentu
4. Hapus Data Pertama
5. Hapus Data Terkahir
6. Hapus Data Tertentu
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
==================================
3
----------------------------
Masukkan Data FILM
----------------------------
Urutan ke : 1
ID : 125
Judul Film : Sikuning
```

```
10. Keluar
==================================
7
----------------------------
DATA FILM LAYAR LEBAR
----------------------------
Cetak Data
ID      : 124
  Judul         : Sibiru
  Rating        : 2.8
ID      : 125
  Judul         : Sikuning
  Rating        : 5.0
ID      : 123
  Judul         : Simerah
  Rating        : 3.4

==================================
```

```
==================================
8
Masukkan ID : 123
ID      : 123
Judul   : Simerah
Rating  : 3.4
```

```
==================================
9
----------------------------
DATA FILM LAYAR LEBAR SORTING DESC
----------------------------
Cetak Data
ID      : 124
  Judul         : Sibiru
  Rating        : 2.8
ID      : 123
  Judul         : Simerah
  Rating        : 3.4
ID      : 125
  Judul         : Sikuning
  Rating        : 5.0
```