

# **Laporan Praktikum Algoritma dan Struktur Data**

## **Jobsheet 12 - Double Linked Lists**

**Dosen Pengampu : Triana Fatmawati, S.T., M.T.**



**Nama : Annisa**  
**Nim : 2341760032**  
**Kelas : SIB 1E**  
**Prodi : D-IV Sistem Informasi Bisnis**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2023/2024**

## 12.1 Tujuan Praktikum

### 12.2.1 Kegiatan Praktikum 1

1. Perhatikan diagram class Node dan class DoubleLinkedLists di bawah ini! Diagram class ini yang selanjutnya akan dibuat sebagai acuan dalam membuat kode program DoubleLinkedLists.
2. Buat paket baru dengan nama doublelinkedlists
3. Buat class di dalam paket tersebut dengan nama Node



```
1 package doublelinkedlists;
2
3 /**
4  * Node04
5  */
6 public class Node04 {
7
```

4. Di dalam class tersebut, deklarasikan atribut sesuai dengan diagram class di atas.



```
1 int data;
2 Node04 prev, next;
```

5. Selanjutnya tambahkan konstruktor default pada class Node sesuai diagram di atas.



```
1 Node04(Node04 prev, int data, Node04 next) {
2     this.prev = prev;
3     this.data = data;
4     this.next = next;
5 }
6 }
```

6. Buatlah sebuah class baru bernama DoubleLinkedLists pada package yang sama dengan node seperti gambar berikut:

```
1 package doublelinkedlists;
2
3 /**
4  * DoubleLinkedLists04
5  */
6 public class DoubleLinkedLists04 {
```

7. Pada class DoubleLinkedLists tersebut, deklarasikan atribut sesuai dengan diagram class di atas.

```
1 Node04 head;
2     int size;
```

8. Selajutnya, buat konstruktor pada class DoubleLinkedLists sesuai gambar berikut.

```
1 public DoubleLinkedLists04() {
2     head = null;
3     size = 0;
4 }
```

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

```
1 public boolean isEmpty() {
2     return head == null;
3 }
```

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

```
1 public void addFirst (int item) {
2     if (isEmpty()) {
3         head = new Node04(null, item, null);
4     } else {
5         Node04 newNode04 = new Node04(null, item, head);
6         head.prev = newNode04;
7         head = newNode04;
8     }
9     size++;
10 }
```

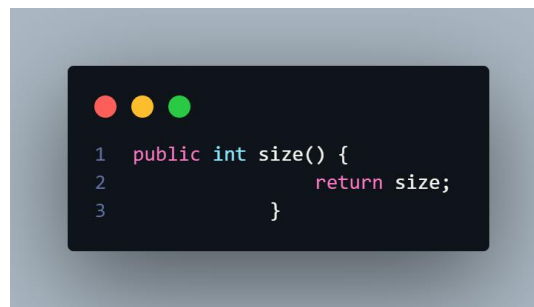
11. Selain itu pembuatan method `addLast()` akan menambahkan data pada bagian belakang linked list.

```
1 public void addLast (int item) {
2     if (isEmpty()) {
3         addFirst(item);
4     } else {
5         Node04 current = head;
6         while (current.next != null) {
7             current = current.next;
8         }
9         Node04 newNode04 = new Node04(current, item, null);
10        current.next = newNode04;
11        size++;
12    }
13 }
```

12. Untuk menambahkan data pada posisi yang telah ditentukan dengan indeks, dapat dibuat dengan method `add(int item, int index)`.

```
1 public void add (int item, int index) throws Exception {
2     if (isEmpty()) {
3         addFirst(item);
4     } else if (index < 0 || index > size) {
5         throw new Exception ("Nilai indeks di luar batas");
6     } else {
7         Node04 current = head;
8         int i = 0;
9         while (i < index) {
10            current = current.next;
11            i++;
12        }
13        if (current.prev == null) {
14            Node04 newNode04 = new Node04(null, item, current);
15            current.prev = newNode04;
16            head = newNode04;
17        } else {
18            Node04 newNode04 = new Node04(current.prev, item, current);
19            newNode04.prev = current.prev;
20            newNode04.next = current;
21            current.prev.next = newNode04;
22            current.prev = newNode04;
23        }
24    }
25    size++;
26 }
```

13. Jumlah data yang ada di dalam linked lists akan diperbarui secara otomatis, sehingga dapat dibuat method `size()` untuk mendapatkan nilai dari `size`.



```
1 public int size() {
2     return size;
3 }
```

14. Selanjutnya dibuat method `clear()` untuk menghapus semua isi linked lists, sehingga linked lists dalam kondisi kosong.



```
1 public void clear() {
2     head = null;
3     size = 0;
4 }
```

15. Untuk mencetak isi dari linked lists dibuat method `print()`. Method ini akan mencetak isi linked lists berapapun size-nya. Jika kosong akan dimunculkan suatu pemberitahuan bahwa linked lists dalam kondisi kosong



```
1 public void print() {
2     if (!isEmpty()) {
3         Node04 tmp = head;
4         while (tmp != null) {
5             System.out.print(tmp.data + "\t");
6             tmp = tmp.next;
7         }
8         System.out.println("\nBerhasil diisi");
9     } else {
10        System.out.println("Linked Lists Kosong");
11    }
12 }
13 }
```

16. Selanjutnya dibuat class Main DoubleLinkedListsMain untuk mengeksekusi semua method yang ada pada class DoubleLinkedLists.

```
1 package doublelinkedlists;
2
3 /**
4  * DoubleLinkedLists04Main
5  */
6 public class DoubleLinkedLists04Main {
7
8     public static void main(String[] args) {
```

17. Pada main class pada langkah 16 di atas buatlah object dari class DoubleLinkedLists kemudian eksekusi potongan program berikut ini

```
1 DoubleLinkedLists04 dll = new DoubleLinkedLists04();
2 dll.print();
3 System.out.println("Size: " + dll.size());
4 System.out.println("=====");
5 dll.addFirst(3);
6 dll.addLast(4);
7 dll.addFirst(7);
8 dll.print();
9 System.out.println("Size: " + dll.size());
10 System.out.println("=====");
11 try {
12     dll.add(40, 1);
13     dll.print();
14     System.out.println("Size: " + dll.size());
15     System.out.println("=====");
16     dll.clear();
17     dll.print();
18     System.out.println("Size: " + dll.size());
19 } catch (Exception e) {
20     e.printStackTrace();
21 }
22 System.out.println("=====");
23 System.out.println("-----");
24 System.out.println("BUILD SUCCESS");
25 System.out.println("-----");
26 }
27 }
```

### 12.2.2 Verifikasi Hasil Percobaan

```
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
=====
BUILD SUCCESS
=====
C:\Users\user\Documents\Jobsheet 12>
```

### 12.2.3 Pertanyaan Percobaan

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

Single linked list hanya memiliki satu pointer (next) per node yang menunjuk ke node berikutnya, sehingga hanya dapat diakses secara berurutan dari awal. Sementara double linked list memiliki dua pointer (next dan prev) per node, yang memungkinkan akses data dari dua arah (maju dan mundur), namun membutuhkan lebih banyak memori.

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

Dalam class Node04 atribut prev digunakan untuk menyimpan referensi atau pointer ke node sebelumnya, sementara atribut next digunakan untuk menyimpan referensi atau pointer ke node berikutnya. Dengan adanya dua atribut ini, setiap node dalam double linked list terhubung dengan node sebelum dan sesudahnya, memungkinkan traversal atau akses data dari dua arah

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

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

Dalam konstruktor DoubleLinkedLists04(), inisialisasi atribut head dengan null dan size dengan 0 dilakukan untuk menyiapkan kondisi awal double linked list yang kosong. Head diinisialisasi null berarti belum ada node atau elemen di dalamnya, sedangkan size 0 mengonfirmasi bahwa jumlah node-nya adalah kosong.

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

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

Pada pembuatan objek node baru newNode04 dalam method addFirst() prev diinisialisasi dengan null karena node baru tersebut akan menjadi node pertama (head) dalam double linked list, sehingga tidak memiliki node sebelumnya. Sedangkan next diinisialisasi dengan 'head' untuk menghubungkan node baru dengan node pertama sebelumnya (jika ada) sehingga node baru menjadi head baru dalam struktur double linked list.

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

Statement head.prev = newNode04; dalam method addFirst() digunakan untuk memperbarui pointer prev dari node yang sebelumnya menjadi head (node pertama) dalam double linked list agar menunjuk ke node baru (newNode04) yang menjadi head baru.

**6. Perhatikan isi method addLast(), apa arti dari pembuatan object Node dengan mengisi parameter prev dengan current, dan next dengan null? Node newNode = new Node(current, item, null);**

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

Pada pembuatan objek node baru newNode04 dalam method addLast(), parameter prev diisi dengan current yang merujuk pada node terakhir yang sudah ada dalam double linked list. Ini untuk menghubungkan node baru tersebut sebagai node berikutnya dari node terakhir sebelumnya melalui pointer prev. Sedangkan parameter next diisi dengan null karena node baru tersebut akan menjadi node terakhir dalam double linked list, sehingga tidak ada node berikutnya lagi.

**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.**

Pertama, kode memeriksa apakah current.prev bernilai null, yang mengindikasikan linked list masih kosong. Jika kondisi tersebut terpenuhi, kode akan membuat instance baru dari objek Node dengan nilai item yang akan ditambahkan, dan menghubungkannya dengan current. Kemudian, current.prev dan head (node pertama dalam linked list) diatur menunjuk ke node baru yang baru saja dibuat. Dengan demikian, node baru tersebut menjadi node pertama dan satu-satunya dalam linked list yang sebelumnya masih kosong.



## 12.3 Kegiatan Praktikum 2

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

```
1 public void removeFirst() throws Exception {
2     if (isEmpty()) {
3         throw new Exception("Linked List masih kosong, tidak dapat dihapus!");
4     } else if (size == 1) {
5         removeLast();
6     } else {
7         head = head.next;
8         head.prev = null;
9         size--;
10    }
11 }
```

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

```
1 public void removeLast() throws Exception {
2     if (isEmpty()) {
3         throw new Exception("Linked List masih kosong, tidak dapat dihapus!");
4     } else if (head.next == null) {
5         head = null;
6         size--;
7         return;
8     }
9     Node04 current = head;
10    while (current.next.next != null) {
11        current = current.next;
12    }
13    current.next = null;
14    size--;
15 }
```

3. Tambahkan pula method `remove(int index)` pada class `DoubleLinkedLists` dan amati hasilnya.

```
1 public void remove(int index) throws Exception {
2     if (isEmpty() || index >= size) {
3         throw new Exception("Nilai indeks di luar batas");
4     } else if (index == 0) {
5         removeFirst();
6     } else {
7         Node04 current = head;
8         int i = 0;
9         while (i < index) {
10            current = current.next;
11            i++;
12        }
13        if (current.next == null) {
14            current.prev.next = null;
15        } else if (current.prev == null) {
16            current = current.next;
17            current.prev = null;
18            head = current;
19        } else {
20            current.prev.next = current.next;
21            current.next.prev = current.prev;
22        }
23        size--;
24    }
25 }
```

4. Untuk mengeksekusi method yang baru saja dibuat, tambahkan potongan kode program berikut pada main class.

```
1 dll.addLast(50);
2 dll.addLast(40);
3 dll.addLast(10);
4 dll.addLast(20);
5 dll.print();
6 System.out.println("Size : "+dll.size());
7 System.out.println("=====");
8 dll.removeFirst();
9 dll.print();
10 System.out.println("Size : "+dll.size());
11 System.out.println("=====");
12 dll.removeLast();
13 dll.print();
14 System.out.println("Size : "+dll.size());
15 System.out.println("=====");
16 dll.remove(1);
17 dll.print();
18 System.out.println("Size : "+dll.size());
```

### 12.3.2 Verifikasi Hasil Percobaan

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

### 12.3.3 Pertanyaan Percobaan

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

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

Maksud dari potongan kode tersebut adalah:

head = head.next; => artinya adalah mengganti nilai head ke head.next

head.prev = null; => artinya adalah merubah nilai head.prev yang tadinya masih ada nilai Node nya menjadi null untuk memutus pointer ke Node yang paling awal sebelumnya.

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



```
1 Node04 current = head;  
2 while (current.next.next != null) {  
3     current = current.next;  
4 }  
5 current.next = null;  
6 size--;  
7 }
```

Pada potongan kode diatas menjelaskan akan dilakukan traversal dimulai dari head hingga Node current.next.next bernilai null. Sehingga bisa dilakukan remove atau penghapusan pada node terakhir ditandai dengan 'current.next = 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 program tersebut tidak cocok karena potongan program tersebut hanya menghapus node kedua dalam linked list, bukan node yang sebenarnya ingin dihapus. berikut penjelasannya:

1) Node tmp = head.next; => baris ini menginisialisasi tmp dengan node kedua dalam linked list, ditandai dengan adanya 'head.next'

2) head.next = tmp.next => baris ini memperbarui nilai head.next untuk melompati tmp. ini menghapus koneksi dari head ke tmp dan langsung menghubungkan head ke node setelah tmp.

3) tmp.next.prev = head => baris ini memperbarui nilai prev dari node setelah tmp (node ke-3) untuk mentarget head. namun, jika tmp adalah node kedua, maka tmp.next akan menjadi null dan ketika mengakses prev dari null akan terjadi 'NullPointerException' atau data tidak ada.

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

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

Fungsi kode program tersebut adalah untuk menghapus node yang akan kita target. menggambarkan fungsi dari potongan program tersebut adalah untuk menghubungkan node sebelum dan sesudah dari Node yang telah kita target untuk dihapus. Sehingga setelah node tersebut dihapus maka Node sebelum dan sesudah nya akan tetap masih terhubung. Seperti ilustrasi berikut.

### 12.4 Kegiatan Praktikum 3

1. Buatlah method `getFirst()` di dalam class `DoubleLinkedLists` untuk mendapatkan data pada awal linked lists.

```
1 public int getFirst() throws Exception {  
2     if (isEmpty()) {  
3         throw new Exception("Linked List Kosong");  
4     }  
5     return head.data;  
6 }
```

2. Selanjutnya, buatlah method `getLast()` untuk mendapat data pada akhir linked lists.

```
1 public int getLast() throws Exception {  
2     if (isEmpty()) {  
3         throw new Exception("Linked List kosong");  
4     }  
5     Node04 tmp = head;  
6     while (tmp.next != null) {  
7         tmp = tmp.next;  
8     }  
9     return tmp.data;  
10 }
```

4. Method `get(int index)` dibuat untuk mendapatkan data pada indeks tertentu

```
1 public int get(int index) throws Exception {
2     if (isEmpty() || index >= size) {
3         throw new Exception("Nilai indeks di luar batas.");
4     }
5     Node04 tmp = head;
6     for (int i = 0; i < index; i++) {
7         tmp = tmp.next;
8     }
9     return tmp.data;
10 }
11 }
```

5. Pada main class tambahkan potongan program berikut dan amati hasilnya!

```
1 dll.print();
2 System.out.println("Size: " + dll.size());
3 System.out.println("=====");
4 dll.addFirst(3);
5 dll.addLast(4);
6 dll.addFirst(7);
7 dll.print();
8 System.out.println("Size: " + dll.size());
9 System.out.println("=====");
10 dll.add(40, 1);
11 dll.print();
12 System.out.println("Size: " + dll.size());
13 System.out.println("=====");
14 System.out.println("Data awal pada Linked Lists adalah: " + dll.getFirst());
15 System.out.println("Data akhir pada Linked Lists adalah: " + dll.getLast());
16 System.out.println("Data indeks ke-1 pada Linked Lists adalah: " + dll.get(1));
17 } catch (Exception e) {
18     e.printStackTrace();
19 }
20 System.out.println("=====");
21 System.out.println("-----");
22 }
23 }
24 }
```

#### 12.4.2 Verifikasi Hasil Percobaan

```
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
40
Berhasil diisi
Size : 1
=====
7      3      40      4
Berhasil diisi
Size : 4
=====
7      40      3      40      4
Berhasil diisi
Size : 5
=====
Data awal pada Linked Lists adalah: 7
Data akhir pada Linked Lists adalah: 4
Data indeks ke-1 pada Linked Lists adalah: 40
=====
C:\Users\user\Documents\Jobsheet 12>

```

### 12.4.3 Pertanyaan Percobaan

#### 1. Jelaskan method size() pada class DoubleLinkedLists!

Method size digunakan mengembalikan nilai dari atribut size untuk memberitahukan berapa jumlah data pada linked lists.

#### 2. Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke1!

Pertama adalah mengatur pada konstruktor double linked lists itu sendiri di set menjadi bernilai size = 1; Kedua, setiap memulai proses traversal langsung mengarahkan nilai indeks nya adalah satu. Sehingga proses perhitungan indeks akan dimulai dari satu.

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

Perbedaan karakteristik nya adalah pada double linked lists diperlukan parameter Node Next dan Prev. Sedangkan pada single linked list hanya memerlukan parameter Next saja.

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

<pre> public boolean isEmpty(){     if(size == 0){         return true;     } else{         return false;     } } </pre> <p>(a)</p>	<pre> public boolean isEmpty(){     return head == null; } </pre> <p>(b)</p>
---	--

Perbedaan kedua logika tersebut berdasarkan apa yang menjadi perhitungannya.

Pada point (a) kondisinya adalah menghitung apakah size = 0. Jika iya maka akan menghasilkan nilai true, jika tidak maka akan menghasilkan nilai false. Sedangkan pada point (b) kondisinya akan melihat apakah nilai head == null. Jika iya maka akan menghasilkan nilai true. Pada point (b) tidak terlihat secara tertulis seperti pada point (a) untuk memberitahukan kondisi apa yang menghasilkan nilai true ataupun false.

## 12.5 Tugas Praktikum

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) Ilustrasi Program

```
*****
PENGANTRI VAKSIN EXTRAVAGANZA
*****

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar

*****

*****
PENGANTRI VAKSIN EXTRAVAGANZA
*****

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
*****
3
*****
Masukkan Data Penerima Vaksin

Nomor Antrian:
123
Nama Penerima:
Joko

Cetak Data (Komponen di area merah harus ada)
*****
PENGANTRI VAKSIN EXTRAVAGANZA
*****

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
*****
3
*****
Daftar Pengantri Vaksin
*****
[No. | Nama |
|123 | Joko |
|124 | Mely |
|135 | Johan |
|146 | Resi |
Sisa Antrian: 4]

Hapus Data (Komponen di area merah harus ada)
*****
PENGANTRI VAKSIN EXTRAVAGANZA
*****

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar
*****
2
*****
Joko telah selesai divaksinasi.
*****
Daftar Pengantri Vaksin
*****
[No. | Nama |
|124 | Mely |
|135 | Johan |
|146 | Resi |
Sisa Antrian: 3]
```

Berikut merupakan kode program dari studi kasus diatas

```
1 package tugasdoublelinkedlists;
2 /**
3  * Vaksin04
4  */
5 public class Vaksin04 {
6
7     String no_antrian, nama;
8     public Vaksin04(String no_antrian, String nama) {
9         this.no_antrian = no_antrian;
10        this.nama = nama;
11    }
12 }
```

```
1 package tugasdoublelinkedlists;
2
3 /**
4  * NodeVaksin04
5  */
6 public class NodeVaksin04 {
7
8     Vaksin04 data;
9     NodeVaksin04 prev, next;
10
11     public NodeVaksin04(NodeVaksin04 prev, Vaksin04 data, NodeVaksin04 next) {
12         this.data = data;
13         this.prev = prev;
14         this.next = next;
15     }
16     public void printData() {
17         System.out.println("|"+data.no_antrian+"\t|"+data.nama+"\t|");
18     }
19 }
```



```

1 package tugasdoublelinkedlists;
2 public class DoubleLinkedListsVks04 {
3
4     NodeVaksin04 head;
5     int size;
6
7     public DoubleLinkedListsVks04() {
8         head = null;
9         size = 0;
10    }
11
12    public boolean isEmpty() {
13        return head == null;
14    }
15
16    public void addFirst(Vaksin04 item) {
17        if (isEmpty()) {
18            head = new NodeVaksin04(null, item, null);
19        } else {
20            NodeVaksin04 newNode = new NodeVaksin04(null, item, head);
21            head.prev = newNode;
22            head = newNode;
23        }
24        size++;
25    }
26
27    public void addLast(Vaksin04 item) {
28        if (isEmpty()) {
29            addFirst(item);
30        } else {
31            NodeVaksin04 current = head;
32            while (current.next != null) {
33                current = current.next;
34            }
35            NodeVaksin04 newNode = new NodeVaksin04(current, item, null);
36            current.next = newNode;
37            size++;
38        }
39    }
40
41    public int size() {
42        return size;
43    }
44
45    public void clear() {
46        head = null;
47        size = 0;
48    }
49
50    public void print() {
51        if (isEmpty()) {
52            System.out.println("Linked Lists Kosong");
53        } else {
54            NodeVaksin04 tmp = head;
55            while (tmp != null) {
56                tmp.printData();
57                tmp = tmp.next;
58            }
59            System.out.println("Sisa Antrian: " + size());
60        }
61    }
62
63    public void removeLast() throws Exception {
64        if (isEmpty()) {
65            throw new Exception("Linked List masih kosong, tidak dapat menghapus elemen terakhir");
66        }
67        if (size == 1) {
68            head = null;
69        } else {
70            NodeVaksin04 current = head;
71            while (current.next != null) {
72                current = current.next;
73            }
74            current.prev.next = null;
75        }
76        size--;
77    }
78
79    public void removeFirst() throws Exception {
80        NodeVaksin04 tmp = head;
81        if (isEmpty()) {
82            throw new Exception("Linked List masih kosong, tidak dapat dihapus");
83        } else if (size == 1) {
84            removeLast();
85        } else {
86            head = head.next;
87            head.prev = null;
88            size--;
89            System.out.println(tmp.data.nama + " telah dihapus");
90        }
91    }
92
93    // Menambahkan metode enqueue untuk menambahkan elemen ke akhir linked list
94    public void enqueue(Vaksin04 item) {
95        addLast(item);
96    }
97
98    // Menambahkan metode dequeue untuk menghapus elemen dari awal linked list
99    public void dequeue() throws Exception {
100        removeFirst();
101    }
102 }
103

```

```

1 package tugasdoublelinkedlists;
2
3 import java.util.Scanner;
4
5 public class VaksinMain04 {
6
7     public static void menu() {
8         System.out.println("+++++++");
9         System.out.println("PENGANTRI VAKSIN EXTRAVAGANZA");
10        System.out.println("+++++++");
11        System.out.println("\n1. Tambah Data Penerima Vaksin");
12        System.out.println("2. Hapus Data Pengantri Vaksin");
13        System.out.println("3. Daftar Penerima Vaksin");
14        System.out.println("4. Keluar");
15        System.out.println("+++++++");
16    }
17
18    public static void main(String[] args) {
19        DoubleLinkedListsVks04 dll = new DoubleLinkedListsVks04();
20        Scanner sc04 = new Scanner(System.in);
21        int pilih = 0;
22
23        do {
24            menu();
25            pilih = sc04.nextInt();
26            sc04.nextLine(); // consume newline
27            switch (pilih) {
28                case 1:
29                    System.out.println("-----");
30                    System.out.println("Masukkan Data Penerima Vaksin");
31                    System.out.println("-----");
32                    System.out.print("Nomor Antrian: ");
33                    String no_antrian = sc04.nextLine();
34                    System.out.print("Nama Penerima: ");
35                    String nama = sc04.nextLine();
36                    Vaksin04 vn = new Vaksin04(no_antrian, nama);
37                    dll.enqueue(vn);
38                    break;
39                case 2:
40                    try {
41                        dll.dequeue();
42                        System.out.println("Data pengantri vaksin berhasil dihapus.");
43                    } catch (Exception e) {
44                        System.out.println(e.getMessage());
45                    }
46                    break;
47                case 3:
48                    System.out.println("+++++++");
49                    System.out.println("Daftar Pengantri Vaksin");
50                    System.out.println("+++++++");
51                    System.out.println("|No.\t|Nama\t|");
52                    dll.print();
53                    break;
54                case 4:
55                    System.out.println("Terima kasih!");
56                    break;
57                default:
58                    System.out.println("Pilihan tidak valid!");
59                    break;
60            }
61        } while (pilih != 4);
62
63        sc04.close();
64    }
65 }
66

```

Berikut merupakan hasil running dari kode program di atas

```
12. C:\szale23\0111 - tugas\double linked list\
+++++
PENGANTRI VAKSIN EXTRAVAGANZA
+++++

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar

+++++
1

-----
Masukkan Data Penerima Vaksin

-----
Nomor Antrian: 301130
Nama Penerima: Annisaa
+++++
PENGANTRI VAKSIN EXTRAVAGANZA
+++++

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar

+++++
1

-----
Masukkan Data Penerima Vaksin

-----
Nomor Antrian: 291130
Nama Penerima: Nnatalia
+++++
PENGANTRI VAKSIN EXTRAVAGANZA
+++++
```

```
-----
Nomor Antrian: 731130
Nama Penerima: Claudya
+++++
PENGANTRI VAKSIN EXTRAVAGANZA
+++++
```

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

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar

+++++
3

+++++
Daftar Pengantri Vaksin
+++++
|No. |Nama |
|301130 |Annisaa |
|291130 |Nnatalia |
|731130 |Claudya |
Sisa Antrian: 3
```

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

1. Tambah Data Penerima Vaksin
2. Hapus Data Pengantri Vaksin
3. Daftar Penerima Vaksin
4. Keluar

+++++
2
Annisaa telah dihapus
Data pengantri vaksin berhasil dihapus.
```

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.

#### Contoh Ilustrasi Program

##### Menu Awal dan Penambahan Data

```
DATA 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
=====
```

```
DATA 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
=====
1
Masukkan Data Film Posisi Awal
ID Film:
1232
Judul Film: Spider-Man: No Way Home
Rating Film:
6.7
```

```
DATA 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
=====
2
Masukkan Data Posisi Akhir
ID Film:
1346
Judul Film: Uncharted
Rating Film:
6.7
```

```
DATA 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
=====
3
Masukkan Data Film
Urutan ke-:
1234
Judul Film: Death on the Nile
Rating Film:
8.6
Data Film ini akan masuk di urutan ke-:
3
```

##### Cetak Data

```
DATA 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
=====
7
Cetak Data
ID: 1232
Judul Film: Spider-Man: No Way Home
Rpt: 6.7
ID: 1346
Judul Film: Skyfall
Rpt: 7.8
ID: 1307
Judul Film: The Bank Knight Rises
Rpt: 6.4
ID: 1234
Judul Film: Death on the Nile
Rpt: 8.6
ID: 1348
Judul Film: Uncharted
Rpt: 6.7
```

##### Pencarian Data

```
DATA 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film-DESC
10. Keluar
=====
8
Cari Data
Masukkan ID Film yang dicari:
1307
Data ID Film 1307 berada di node ke- 3
=====
ID Film 1307
Judul Film: The Bank Knight Rises
Rpt Rating: 6.4
```

Berikut merupakan kode program dari studi kasus di atas

```
1 package tugasdoublelinkedlists;
2
3 /**
4  * NodeFilm04
5  */
6 public class NodeFilm04 {
7
8     int id;
9     String judulFilm;
10    double rating;
11
12    NodeFilm04 prev, next;
13
14    NodeFilm04(NodeFilm04 prev, int id, String judulFilm, double rating, NodeFilm04 next){
15        this.prev = prev;
16        this.id = id;
17        this.judulFilm = judulFilm;
18        this.rating = rating;
19        this.next = next;
20    }
21 }
```

```

1 package tugasdoubleLinkedList;
2
3 /*
4  * DoubleLinkedListFile.java
5  */
6 public class DoubleLinkedListFile {
7     NodeFile head;
8     int size;
9
10    public DoubleLinkedListFile(){
11        head = null;
12        size = 0;
13    }
14    public boolean isEmpty(){
15        return head == null;
16    }
17    public void addFirst(int id, String judulFile, double rating){
18        if (isEmpty()){
19            head = new NodeFile(null, id, judulFile, rating, null);
20        }
21        else{
22            NodeFile newNodeFile = new NodeFile(null, id, judulFile, rating, head);
23            head.prev = newNodeFile;
24            head = newNodeFile;
25        }
26        size++;
27    }
28    public void addLast(int id, String judulFile, double rating){
29        if (isEmpty()){
30            addFirst(id, judulFile, rating);
31        }
32        else{
33            NodeFile current = head;
34            while (current.next != null){
35                current = current.next;
36            }
37            NodeFile newNodeFile = new NodeFile(current, id, judulFile, rating, null);
38            current.next = newNodeFile;
39            size++;
40        }
41    }
42    public void add(int id, String judulFile, double rating, int index) throws Exception {
43        if (isEmpty()){
44            addFirst(id, judulFile, rating);
45        }
46        else if (index < 0 || index > size){
47            throw new Exception("Nilai Indeks di luar batas");
48        }
49        else{
50            NodeFile current = head;
51            int i = 0;
52            while (i < index){
53                current = current.next;
54                i++;
55            }
56            if (current.prev == null){
57                NodeFile newNodeFile = new NodeFile(null, id, judulFile, rating, current);
58                current.prev = newNodeFile;
59                head = newNodeFile;
60            }
61            else{
62                NodeFile newNodeFile = new NodeFile(current.prev, id, judulFile, rating, current);
63                newNodeFile.next = current;
64                current.prev = newNodeFile;
65                current.next = newNodeFile;
66            }
67        }
68        size++;
69    }
70    public void removeFirst() throws Exception{
71        if (isEmpty()){
72            throw new Exception("Linked list masih kosong, tidak dapat dihapus!");
73        }
74        else if (size == 1){
75            removeLast();
76        }
77        else{
78            head = head.next;
79            head.prev = null;
80            size--;
81        }
82    }
83    public void removeLast() throws Exception {
84        if (isEmpty()){
85            throw new Exception("Linked list masih kosong, tidak dapat dihapus!");
86        }
87        else if (head.next == null){
88            head = null;
89            size--;
90            return;
91        }
92        else{
93            NodeFile current = head;
94            while (current.next.next != null){
95                current = current.next;
96            }
97            current.next = null;
98            size--;
99        }
100    }
101    public void remove(int index) throws Exception {
102        if (isEmpty() || index > size){
103            throw new Exception("Nilai Indeks di luar batas");
104        }
105        else if (index == 0){
106            removeFirst();
107        }
108        else{
109            NodeFile current = head;
110            int i = 0;
111            while (i < index){
112                current = current.next;
113                i++;
114            }
115            if (current.next == null){
116                current.prev.next = null;
117            }
118            else if (current.prev == null){
119                current = current.next;
120                current.prev = null;
121                head = current;
122            }
123            else{
124                current.prev.next = current.next;
125                current.next.prev = current.prev;
126            }
127            size--;
128        }
129    }
130    public void print(){
131        System.out.println("-----");
132        System.out.println("Data Saat Ini Menjadi");
133        if (isEmpty()){
134            NodeFile tmp = head;
135            while (tmp != null){
136                System.out.println("ID\t\t: " + tmp.id);
137                System.out.println("Judul File\t\t: " + tmp.judulFile);
138                System.out.println("Rating\t\t\t: " + tmp.rating);
139                System.out.println();
140                tmp = tmp.next;
141            }
142        }
143        else{
144            System.out.println("Linked list Kosong");
145        }
146    }
147    public int findIndexSearch(int cari){
148        NodeFile tmp = head;
149        int posisi = -1;
150        int index = 0;
151        while (tmp != null){
152            if (tmp.id == cari){
153                posisi = index;
154                break;
155            }
156            index++;
157            tmp = tmp.next;
158        }
159        return posisi;
160    }
161    public void tampilPosisi(int x, int pos){
162        if (pos <= 1){
163            System.out.println("ID\t\t: " + x + " ditemukan pada indeks " + pos);
164        }
165        else {
166            System.out.println("ID\t\t: " + x + " tidak ditemukan");
167        }
168    }
169    public void sort(){
170        NodeFile current = null, index = null;
171        int temp0;
172        String temp00;
173        double temp01;
174        if (head == null) {
175            return;
176        }
177        else {
178            for (current = head; current.next != null; current = current.next) {
179                for (index = current.next; index != null; index = index.next) {
180                    if (current.rating > index.rating) {
181                        temp01 = current.rating;
182                        current.rating = index.rating;
183                        index.rating = temp01;
184                        temp00 = current.id;
185                        current.id = index.id;
186                        index.id = temp00;
187                        temp000 = current.judulFile;
188                        current.judulFile = index.judulFile;
189                        index.judulFile = temp000;
190                    }
191                }
192            }
193        }
194        print();
195    }
196 }

```

```

1 package tugasdoublelinkedlists;
2 import java.util.Scanner;
3
4 public class DoubleLinkedListsFilm04Main {
5
6     public static void menu() {
7         System.out.println("=====");
8         System.out.println(" Data Film Layer Lebar ");
9         System.out.println("=====");
10        System.out.println(" 1. Tambah Data Awal");
11        System.out.println(" 2. Tambah Data Akhir");
12        System.out.println(" 3. Tambah Data Index Tertentu");
13        System.out.println(" 4. Hapus Data Pertama");
14        System.out.println(" 5. Hapus Data Terakhir");
15        System.out.println(" 6. Hapus Data Tertentu");
16        System.out.println(" 7. Cetak");
17        System.out.println(" 8. Cari ID Film");
18        System.out.println(" 9. Urut Data Rating Film DESC");
19        System.out.println(" 10. Keluar");
20        System.out.println("-----");
21    }
22
23    public static void main(String[] args) throws Exception {
24        Scanner sc04 = new Scanner(System.in);
25        Scanner sd04 = new Scanner(System.in);
26        Scanner sb04 = new Scanner(System.in);
27
28        DoubleLinkedListsFilm04 dll = new DoubleLinkedListsFilm04();
29
30        int pilih;
31        do {
32            menu();
33            pilih = sc04.nextInt();
34            sc04.nextLine();
35
36            switch (pilih) {
37                case 1:
38                    System.out.println("Masukkan Data Film Posisi Awal");
39                    System.out.print("ID Film\t\t: ");
40                    int id = sd04.nextInt();
41                    System.out.print("Judul Film\t: ");
42                    String judulFilm = sc04.nextLine();
43                    System.out.print("Rating Film\t: ");
44                    double rating = sb04.nextDouble();
45                    dll.addFirst(id, judulFilm, rating);
46                    sc04.nextLine();
47                    break;
48
49                case 2:
50                    System.out.println("Masukkan Data Film Posisi Akhir");
51                    System.out.print("ID Film\t\t: ");
52                    int id0 = sd04.nextInt();
53                    System.out.print("Judul Film\t: ");
54                    String judulFilm0 = sc04.nextLine();
55                    System.out.print("Rating Film\t: ");
56                    double rating0 = sb04.nextDouble();
57                    dll.addLast(id0, judulFilm0, rating0);
58                    sc04.nextLine();
59                    break;
60
61                case 3:
62                    System.out.println("Masukkan Data Film Posisi yang diinginkan");
63                    System.out.print("Urutan ke - ");
64                    int idx = sd04.nextInt();
65                    if (idx < 0 || idx > dll.size()) {
66                        System.out.println("Nilai indeks di luar batas. Indeks yang valid adalah 0 hingga " + dll.size());
67                    } else {
68                        System.out.print("ID Film\t\t: ");
69                        int id00 = sd04.nextInt();
70                        sc04.nextLine(); // consume the newline
71                        System.out.print("Judul Film\t: ");
72                        String judulFilm00 = sc04.nextLine();
73                        System.out.print("Rating Film\t: ");
74                        double rating00 = sb04.nextDouble();
75                        dll.add(id00, judulFilm00, rating00, idx);
76                        sc04.nextLine();
77                    }
78                    break;
79
80                case 4:
81                    dll.removeFirst();
82                    dll.print();
83                    break;
84
85                case 5:
86                    dll.removeLast();
87                    dll.print();
88                    break;
89
90                case 6:
91                    System.out.println("Hapus Data Film Posisi yang diinginkan");
92                    System.out.print("Urutan ke - ");
93                    int index = sd04.nextInt();
94                    if (index < 0 || index >= dll.size()) {
95                        System.out.println("Nilai indeks di luar batas. Indeks yang valid adalah 0 hingga " + (dll.size() - 1));
96                    } else {
97                        dll.remove(index);
98                        dll.print();
99                    }
100                    break;
101
102                case 7:
103                    dll.print();
104                    break;
105
106                case 8:
107                    System.out.println("Cari ID Film Yang ingin dicari");
108                    System.out.print("Masukkan ID\t: ");
109                    int cari = sd04.nextInt();
110                    int idF = dll.findSeqSearch(cari);
111                    dll.tampilPosisi(cari, idF);
112                    break;
113
114                case 9:
115                    System.out.println("Data Akan diurut secara DESC");
116                    dll.sort();
117                    break;
118
119                case 10:
120                    System.exit(0);
121                    break;
122
123                default:
124                    System.out.println("Pilihan tidak valid. Silakan pilih antara 1 hingga 10.");
125                    break;
126            }
127        } while (pilih >= 1 && pilih <= 10);
128    }
129 }
130

```

Berikut merupakan hasil running

```
=====
Data 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film DESC
10. Keluar
-----
1
Masukkan Data Film Posisi Awal
ID Film      : 1222
Judul Film   : Spider-Man: No Way Home
Rating Film  : 8.7
```

```
=====
Data 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film DESC
10. Keluar
-----
2
Masukkan Data Film Posisi Akhir
ID Film      : 1346
Judul Film   : Uncharted
Rating Film  : 6.7
```

```
=====
Data 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film DESC
10. Keluar
-----
3
Masukkan Data Film Posisi yang diinginkan
Urutan ke - 1
ID Film      : 1234
Judul Film   : Death on the Nile
Rating Film  : 6.6
```

```
=====
Data 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film DESC
10. Keluar
-----
7
-----
Data Saat Ini Menjadi
ID      : 1222
Judul Film : Spider-Man: No Way Home
Rating   : 8.7

ID      : 1234
Judul Film : Death on the Nile
Rating   : 6.6

ID      : 1346
Judul Film : Uncharted
Rating   : 6.7
```

```
=====
Data 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
7. Cetak
8. Cari ID Film
9. Urut Data Rating Film DESC
10. Keluar
-----
8
Cari ID Film Yang ingin dicari
Masukkan ID      : 1222
ID               : 1222 ditemukan pada indeks 0
```