ALGORITMA DAN STRUKTUR DATA

“Laporan hasil Praktikum pada Jobsheet 12 “Double Linked List” ”

Oleh:

Hafiz Rizqi Hernanda

NIM (244107020154)

Jurusan Teknologi informasi

Teknik Informatika

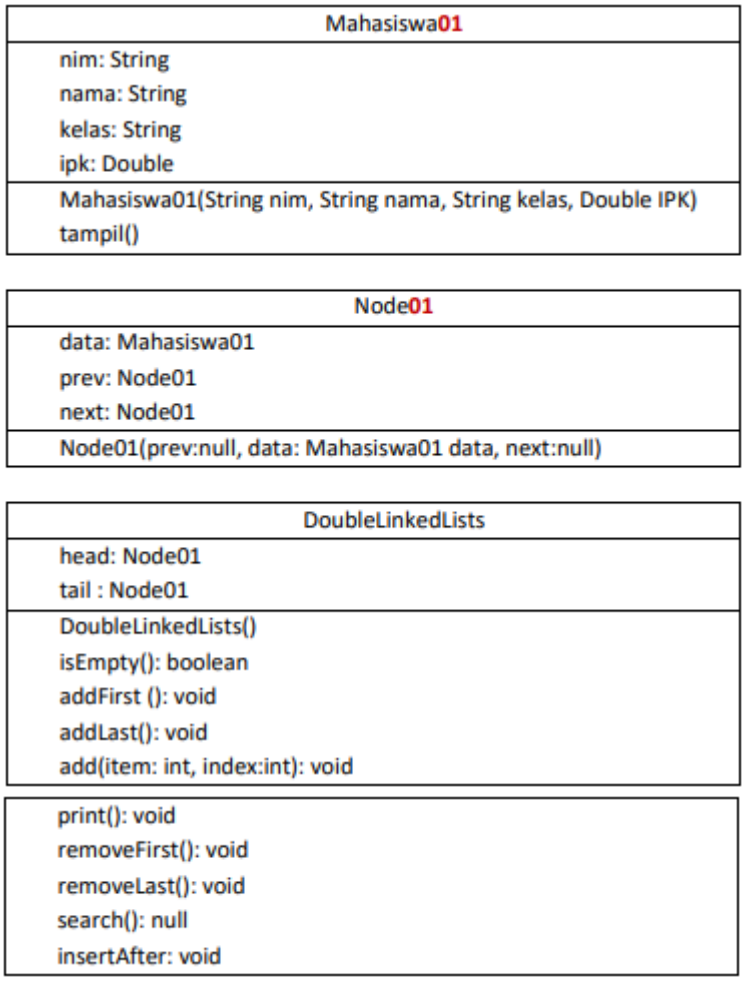
Politeknik Negeri Malang

12.2 Kegiatan Praktikum 1

12.2.1 Percobaan 1

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

1. Perhatikan diagram class Mahasiswa01, Node01 dan class DoublelinkedLists di bawah ini! Diagram class ini yang selanjutnya akan dibuat sebagai acuan dalam membuat kode program DoubleLinkedLists.
   Ganti 01 sesuai dengan nomor absen Anda.



2. Pada Project yang sudah dibuat pada Minggu sebelumnya, buat folder atau package baru bernama Jobsheet12 di dalam repository Praktikum ASD.
3. Buat class di dalam paket tersebut dengan nama Mahasiswa01. Di dalam class tersebut, deklarasikan atribut sesuai dengan diagram class di atas. Tambahkan juga konstruktor dan method sesuai diagram di atas

```java
package Jobsheet12;
public class Mahasiswa11 {
    public String nim;
    public String nama;
    public String kelas;
    public double ipk;

    public Mahasiswa11(String nim, String nama, String kelas, double ipk) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.ipk = ipk;
    }

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

}
```

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

```java
package Jobsheet12;

public class Node11 {
    Mahasiswa11 data;
    Node11 prev;
    Node11 next;

    public Node11(Mahasiswa11 data) {
        this.data = data;
        this.prev = null;
        this.next = null;
    }

}
```

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

```java
package Jobsheet12;

public class DoubleLinkedList11 {
    Node11 head;
    Node11 tail;
```

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

```
7       public DoubleLinkedList11() {
8           head = null;
9           tail = null;
10      }
```

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

```
12      public boolean isEmpty() {
13          return head == null;
14      }
```

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

```
16      public void addFirst(Mahasiswa11 data) {
17          Node11 newNode = new Node11(data);
18          if (isEmpty()) {
19              head = tail = newNode;
20          } else {
21              newNode.next = head;
22              head.prev = newNode;
23              head = newNode;
24          }
25      }
```

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

```
26      public void addLast(Mahasiswa11 data) {
27          Node11 newNode = new Node11(data);
28          if (isEmpty()) {
29              head = tail = newNode;
30          } else {
31              tail.next = newNode;
32              newNode.prev = tail;
33              tail = newNode;
34          }
35      }
```

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

```java
    public void insertAfter(String keyNim, Mahasiswa11 data) {
        Node11 current = head;

        // Cari node dengan nim = keyNim
        while (current != null && !current.data.nim.equals(keyNim)) {
            current = current.next;
        }
        if (current == null) {
            System.out.println("Node dengan NIM " + keyNim + " tidak ditemukan.");
            return;
        }

        Node11 newNode = new Node11(data);

        // Jika Current adalah tail, cukup tambahkan di akhir
        if (current == tail) {
            current.next = newNode;
            newNode.prev = current;
            tail = newNode;
        } else {
            // Sisipkan di tengah
            newNode.next = current.next;
            newNode.prev = current;
            current.next.prev = newNode;
            current.next = newNode;
        }
        System.out.println("Node berhasil disisipkan setelah NIM " + keyNim);
    }
    public void print () {
        Node11 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }
```

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

```java
    public void print () {
        Node11 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }
```

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

```
Jobsheet12 > DLLMain.java > DLLMain > inputMahasiswa(Scanner)
1   package Jobsheet12;
2   import java.util.Scanner;
3   public class DLLMain {
4
        Run | Debug
5       public static void main(String[] args) {
6           DoubleLinkedList11 list = new DoubleLinkedList11();
7           Scanner scan = new Scanner(System.in);
8           int pilihan;
```

13. Buatlah menu pilihan pada class main

```
9           do {
10              System.out.println(x:"Menu Double Linked List Mahasiswa");
11              System.out.println(x:"1. Tambah di Awal");
12              System.out.println(x:"2. Tambah di Akhir");
13              System.out.println(x:"3.  Hapus di Awal");
14              System.out.println(x:"4. Hapus di Akhir");
15              System.out.println(x:"5. Tampilkan Data");
16              System.out.println(x:"7. Cari Mahasiswa Berdasarkan NIM");
17              System.out.println(x:"0.Keluar");
18              System.out.print(s:"Pilih menu: ");
19              pilihan = scan.nextInt();
20              scan.nextLine();
```

14. Tambahkan switch case untuk menjalankan menu pilihan di atas

```
23              switch (pilihan) {
24                  case 1 -> {
25                      Mahasiswa11 mhs = inputMahasiswa(scan);
26                      list.addFirst(mhs);
27                  }
28                  case 2 -> {
29                      Mahasiswa11 mhs = inputMahasiswa(scan);
30                      list.addLast(mhs);
31                  }
32                  case 3 -> list.removeFirst();
33                  case 4 -> list.removeLast();
34                  case 5 -> list.print();
35                  case 7 -> {
36                      System.out.print(s:"Masukkan NIM yang dicari: ");
37                      String nim = scan.nextLine();
38                      Node11 found = list.search(nim);
39                      if (found != null) {
40                          System.out.println(x:"Data Mahasiswa Ditemukan:");
41                          found.data.tampil();
42                      } else {
43                          System.out.println(x:"Data tidak ditemukan.");
44                      }
45                  }
46                  case 0 -> System.out.println(x:"Keluar dari program.");
47                  default -> System.out.println(x:"Pilihan tidak valid. Silakan coba lagi.");
48              }
```

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

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

```java
    static Mahasiswa11 inputMahasiswa(Scanner scan) {
        System.out.print("Masukkan NIM: ");
        String nim = scan.nextLine();
        System.out.print("Masukkan Nama: ");
        String nama = scan.nextLine();
        System.out.print("Masukkan Kelas: ");
        String kelas = scan.nextLine();
        System.out.print("Masukkan IPK: ");
        double ipk = scan.nextDouble();
        scan.nextLine();  // Consume newline
        return new Mahasiswa11(nim, nama, kelas, ipk);
    }

}
```

```java
    public Node11 search(String nim) {
        Node11 current = head;
        while (current != null) {
            if (current.data.nim.equals(nim)) {
                return current;
            }
            current = current.next;
        }
        return null;
    }
```

12.2.2 Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini

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

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

```
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu: 1
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4,0
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu: 5
NIM: 20304050, Nama: Hermione, Kelas: Gryffindor, IPK: 4.0
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu:
```

12.2.3 Pertanyaan Percobaan

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

- Single linked list hanya memiliki pointer next, sedangkan double linked list hanya memiliki pointer prev dan next.
- Single linked list dalam traversal hanya bisa dari head ke tail (satu arah), sedangkan double linked list Bisa traversal dari head ke tail maupun tail ke head (dua arah).
- Single linked list Untuk menghapus node di tengah, perlu akses ke node sebelumnya secara manual. Sedangkan double linked list Bisa hapus node di tengah lebih mudah karena ada prev, jadi bisa langsung akses node sebelumnya.

2. Perhatikan class Node01, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?

Next untuk menuju node berikutnya dan prev untuk menuju node sebelumnya

3. Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan dari konstruktor tersebut?

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

Menginisialisasikan bahwa tidak ada node pertama dan node di akhir

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

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

Jika double linked list nya kosong saat mengisi node akan langsung di head dan tail.

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

Menunjuk pointer head ke node yang baru(dibuat)

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

```
65      public void print () {
66          if (isEmpty()) {
67              System.out.println(x:"Warning!. List Kosong");
68              return;
69          }
```

7. Pada insertAfter(), apa maksud dari kode berikut ? current.next.prev = newNode;

Pada node current.next menunjuk pada prev nnya untuk menuju ke newNode

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

```
53  static Mahasiswa11 inputMahasiswa(Scanner scan) {
54      System.out.print(s:"Masukkan NIM: ");
55      String nim = scan.nextLine();
56      System.out.print(s:"Masukkan Nama: ");
57      String nama = scan.nextLine();
58      System.out.print(s:"Masukkan Kelas: ");
59      String kelas = scan.nextLine();
60      System.out.print(s:"Masukkan IPK: ");
61      double ipk = scan.nextDouble();
62      scan.nextLine();  // Consume newline
63      return new Mahasiswa11(nim, nama, kelas, ipk);
64  }
65
66  }
```

```
99      public Node11 search(String nim) {
100         Node11 current = head;
101         while (current != null) {
102             if (current.data.nim.equals(nim)) {
103                 return current;
104             }
105             current = current.next;
106         }
107         return null;
108     }
```

```java
        do {
            System.out.println(x:"Menu Double Linked List Mahasiswa");
            System.out.println(x:"1. Tambah di Awal");
            System.out.println(x:"2. Tambah di Akhir");
            System.out.println(x:"3.  Hapus di Awal");
            System.out.println(x:"4. Hapus di Akhir");
            System.out.println(x:"5. Tampilkan Data");
            System.out.println(x:"6. Sisipkan Setelah NIM Tertentu");
            System.out.println(x:"7. Cari Mahasiswa Berdasarkan NIM");
            System.out.println(x:"0.Keluar");
            System.out.print(s:"Pilih menu: ");
            pilihan = scan.nextInt();
            scan.nextLine();

            switch (pilihan) {
                case 1 -> {
                    Mahasiswa11 mhs = inputMahasiswa(scan);
                    list.addFirst(mhs);
                }
                case 2 -> {
                    Mahasiswa11 mhs = inputMahasiswa(scan);
                    list.addLast(mhs);
                }
                case 3 -> list.removeFirst();
                case 4 -> list.removeLast();
                case 5 -> list.print();
                case 6 -> {
                    System.out.print(s:"Masukkan NIM setelah yang ingin disisipkan: ");
                    String keyNim = scan.nextLine();
                    Mahasiswa11 mhs = inputMahasiswa(scan);
                    list.insertAfter(keyNim, mhs);
                }
                case 7 -> {
                    System.out.print(s:"Masukkan NIM yang dicari: ");
                    String nim = scan.nextLine();
                    Node11 found = list.search(nim);
                    if (found != null) {
                        System.out.println(x:"Data Mahasiswa Ditemukan:");
                        found.data.tampil();
                    } else {
                        System.out.println(x:"Data tidak ditemukan.");
                    }
                }
                case 0 -> System.out.println(x:"Keluar dari program.");
                default -> System.out.println(x:"Pilihan tidak valid. Silakan coba lagi.");
            }
        } while (pilihan != 0);
        scan.close();
```

```
um-ASD_9c9db009\bin' 'Jobsheet12.DLLMain'
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
6. Sisipkan Setelah NIM Tertentu
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu: 1
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4,0
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
6. Sisipkan Setelah NIM Tertentu
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu: 6
Masukkan NIM setelah yang ingin disisipkan: 20304050
Masukkan NIM: 1234
Masukkan Nama: Arief
Masukkan Kelas: TI
Masukkan IPK: 3,8
Node berhasil disisipkan setelah NIM 20304050
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
6. Sisipkan Setelah NIM Tertentu
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu: 5
NIM: 20304050, Nama: Hermione, Kelas: Gryffindor, IPK: 4.0
NIM: 1234, Nama: Arief, Kelas: TI, IPK: 3.8
```

12.3 Kegiatan Praktikum 2

12.3.1 Tahapan Percobaan

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

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

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

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

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

12.3.2 Verifikasi Hasil Percobaan

12.3.2 Verifikasi Hasil Percobaan

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

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

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

```
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
6. Sisipkan Setelah NIM Tertentu
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu: 1
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4,0
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
6. Sisipkan Setelah NIM Tertentu
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu: 3
Menu Double Linked List Mahasiswa
1. Tambah di Awal
2. Tambah di Akhir
3.  Hapus di Awal
4. Hapus di Akhir
5. Tampilkan Data
6. Sisipkan Setelah NIM Tertentu
7. Cari Mahasiswa Berdasarkan NIM
0.Keluar
Pilih menu:
```

12.3.3 Pertanyaan Percobaan

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

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

Head nya berubah menuju ke head pointer next, pada head pointer prev di nullkan agar node sebelum nya tidak di baca

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

```
32              case 3 -> {
33                  System.out.println("Data berhasil di hapus. Data yang dihapus adalah:" + list.head.data.nama);
34                  list.removeFirst();
35              }
36              case 4 -> {
37                  System.out.println("Data berhasil di hapus. Data yang dihapus adalah:" + list.tail.data.nama);
38                  list.removeLast();
39              }
```

12.5 Tugas Praktikum

1. Tambahkan fungsi add() pada kelas DoubleLinkedList untuk menambahkan node pada indeks tertentu

```
114     public void addAt(int index, Mahasiswa11 data) {
115         if (index < 0) {
116             System.out.println(x:"Index tidak valid.");
117             return;
118         }
119         if (index == 0) {
120             addFirst(data);
121         } else {
122             Node11 newNode = new Node11(data);
123             Node11 current = head;
124             for (int i = 0; i < index - 1 && current != null; i++) {
125                 current = current.next;
126             }
127             if (current == null) {
128                 System.out.println(x:"Index melebihi jumlah elemen, menambahkan di akhir.");
129                 addLast(data);
130             } else {
131                 newNode.next = current.next;
132                 newNode.prev = current;
133                 if (current.next != null) {
134                     current.next.prev = newNode;
135                 } else {
136                     tail = newNode;
137                 }
138                 current.next = newNode;
139             }
140         }
141     }
```

2. Tambahkan removeAfter() pada kelas DoubleLinkedList untuk menghapus node setelah data key.

```java
public void removeAfter(String keyNim) {
    if (isEmpty()) {
    System.out.println(x:"List kosong. Tidak ada yang bisa dihapus.");
    return;
}
    Node11 current = head;

    // Cari node dengan nim = keyNim
    while (current != null && !current.data.nim.equals(keyNim)) {
        current = current.next;
    }
    if (current == null) {
        System.out.println("Node dengan NIM " + keyNim + " tidak ditemukan.");
        return;
    }

    // Jika node berikutnya ada, hapus node tersebut
    if (current.next != null) {
        Node11 toRemove = current.next;
        current.next = toRemove.next;
        if (toRemove.next != null) {
            toRemove.next.prev = current;
        } else {
            tail = current;
        }
        System.out.println("Node setelah NIM " + keyNim + " berhasil dihapus.");
    } else {
        System.out.println("Tidak ada node setelah NIM " + keyNim + ".");
    }
}
```

3. Tambahkan fungsi remove() pada kelas DoubleLinkedList untuk menghapus node pada indeks tertentu.

```java
public void removeAt(int index) {
    if (index < 0) {
        System.out.println(x:"Index tidak valid.");
        return;
    }
    if (isEmpty()) {
        System.out.println(x:"List Kosong, tidak ada yang dihapus.");
        return;
    }
    if (index == 0) {
        removeFirst();
    } else {
        Node11 current = head;
        for (int i = 0; i < index && current != null; i++) {
            current = current.next;
        }
        if (current == null) {
            System.out.println(x:"Index melebihi jumlah elemen, tidak ada yang dihapus.");
            return;
        }
        if (current == tail) {
            removeLast();
        } else {
            current.prev.next = current.next;
            if (current.next != null) {
                current.next.prev = current.prev;
            }
        }
    }
}
```

4. Tambahkan fungsi getFirst(), getLast() dan getIndex() untuk menampilkan data pada node head, node tail dan node pada indeks tertentu.

```
205     public void getFirst() {
206         if (isEmpty()) {
207             System.out.println(x:"List Kosong, tidak ada yang ditampilkan.");
208         } else {
209             head.data.tampil();
210         }
211     }
212     public void getLast() {
213         if (isEmpty()) {
214             System.out.println(x:"List Kosong, tidak ada yang ditampilkan.");
215         } else {
216             tail.data.tampil();
217         }
218     }
219     public int getIndex(String nim) {
220         Node11 current = head;
221         int index = 0;
222         while (current != null) {
223             if (current.data.nama.equals(nim)) {
224                 return index;
225             }
226             current = current.next;
227             index++;
228         }
229         return -1;
230     }
```

5. tambahkan kode program dan fungsi agar dapat membaca size/ jumlah data pada Double Linked List

```
232     public int getSize() {
233         Node11 current = head;
234         int size = 0;
235         while (current != null) {
236             size++;
237             current = current.next;
238         }
239         return size;
240     }
241
242
243
244 }
```