LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR
DOUBLE LINKED LIST

ABIM MUSTAWA

244107020078

KELAS TI-1B

PRODI D-IV TEKNIK

INFORMATIKA JURUSAN

TEKNOLOGI INFORMASI

POLITEKNIK NEGERI

MALANG 2025

1. Percobaan Praktikum

1.1 Praktikum 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)

Langkah-langkah Percobaan

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.

   Mahasiswa01
   - nim: String
   - nama: String
   - kelas: String
   - ipk: Double
   - Mahasiswa01(String nim, String nama, String kelas, Double IPK)
   - tampil()

   Node01
   - data: Mahasiswa01
   - prev: Node01
   - next: Node01
   - Node01(prev:null, data: Mahasiswa01 data, next:null)

   DoubleLinkedLists
   - head: Node01
   - tail : Node01
   - DoubleLinkedLists()
   - isEmpty(): boolean
   - addFirst (): void
   - addLast(): void
   - add(item: int, index:int): void
   - print(): void
   - removeFirst(): void
   - removeLast(): void
   - search(): null
   - insertAfter: void
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
public class Mahasiswa01 {
    String nim, nama, kelas;
    double ipk;

    Mahasiswa01(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
public class Node01 {
    Mahasiswa01 data;
    Node01 next;
    Node01 prev;

    Node01(Mahasiswa01 data){
        this.data = data;
        this.next = null;
        this.prev = 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
public class DoublelinkedLists01 {
    Node01 head;
    Node01 tail;
```

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

```
public DoublelinkedLists01(){
    head = null;
    tail = null;
}
```

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

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

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

```
public void addFirst(Mahasiswa01 data){
    Node01 newNode = new Node01(data);
    if (isEmpty()) {
        head = tail = newNode;
    }else {
        newNode.next = head;
        head.prev = newNode;
        head = newNode;
    }
}
```

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

```
public void addLast(Mahasiswa01 data) {
    Node01 newNode = new Node01(data);
    if (isEmpty()) {
        head = tail = newNode;
    }else {
        tail.next = newNode;
        newNode.prev = tail;
        tail = newNode;
    }
}
```

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

```
public void insertAfter(String keyNim, Mahasiswa01 data){
    Node01 current = head;
    while (current != null && !current.data.nim.equals(keyNim)) {
        current = current.next;
    }
    if (current == null) {
        System.out.println("Node dengan NIM " + keyNim + " tidak ditemukan");
        return;
    }
    Node01 newNode = new Node01(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);
}
```

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

```
public void print(){
    Node01 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.

```
public static void main(String[] args) {
    DoublelinkedLists01 list = new DoublelinkedLists01();
    Scanner scan = new Scanner(System.in);
    int pilihan;
```

13. Buatlah menu pilihan pada class main

```
do {
    System.out.println(x:"\nMenu 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. Cari Mahasiswa Berdasarkan NIM");
    System.out.println(x:"0. keluar");
    System.out.print(s:"Pilih menu: ");
    pilihan = scan.nextInt();
    scan.nextLine();
```

14. Tambahkan switch case untuk menjalankan menu pilihan di atas

```
switch (pilihan) {
    case 1:
        Mahasiswa01 mhs = inputMahasiswa(scan);
        list.addFirst(mhs);
        break;
    case 2:
        Mahasiswa01 mhs1 = inputMahasiswa(scan);
        list.addLast(mhs1);
        break;
    case 3:
        list.removeFirst();
        break;
    case 4:
        list.removeLast();
        break;
    case 5:
        list.print();
        break;
    case 6:
        System.out.print(s:"Masukkan NIM yang dicari: ");
        String nim = scan.nextLine()
        ;
        Node01 found = list.search(nim);
        if (found != null) {
            System.out.println(x:"Data ditemukan: ");
            found.data.tampil();
        }else {
            System.out.println(x:"Data tidak ditemukan");
        }
        break;
    case 0:
        System.out.println(x:"Keluar dari program!");
        break;
    default:
        System.out.println(x:"Pilihan tidak valid!");
        break;
```

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

object scanner

```
} while (pilihan != 0);
scan.close();
```

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

```
public static Mahasiswa01 inputMahasiswa(Scanner scan) {
    System.out.print(s:"Masukkan NIM: ");
    String nim = scan.nextLine();
    System.out.print(s:"Masukkan Nama: ");
    String nama = scan.nextLine();
    System.out.print(s:"Masukkan Kelas: ");
    String kelas = scan.nextLine();
    System.out.print(s:"Masukkan IPK: ");
    double ipk = scan.nextDouble();
    return new Mahasiswa01(nim, nama, kelas, ipk);
}
```

```
public Node01 search(String nim) {
    Node01 current = head;
    while (current != null) {
        if (current.data.nim.equals(nim)) {
            return current;
        }
        current = current.next;
    }
    return null;
}
```

Verifikasi Hasil Percobaan

```
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
```

Pertanyaan

1. Jelaskan perbedaan antara single linked list dengan double linked lists!
   Jawab:
   Single Linked List hanya memiliki satu pointer yaitu pointer next, sedangkan Double Linked List memiliki 2 pointer yaitu next dan prev.

2. Perhatikan class Node01, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?
   Jawab:
   Next berfungsi untuk menujuk ke node berikutnya, sedangkan Prev berfungsi untuk menunjuk ke node sebelumnya.

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

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

Jawab:

Konstruktor ini berfungsi menginisialisasi linked list agar kosong saat pertama kali dibuat dengan head dan tail bernilai null.

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

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

Jawab:

Jika list masih kosong maka node yang baru ditambahkan akan menjadi satu satunya elemen, maka head dan tail akan merujuk ke node tersebut.

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

Jawab:

Artinya node pertama yang lama (head) sekarang memiliki node di sebelumnya yaitu newNode.

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

Jawab:

```java
public void print(){
    Node01 current = head;
    if (isEmpty()) {
        System.out.println(x:"Warning! List masih kosong.");
    }
    while (current != null) {
        current.data.tampil();
        current = current.next;
    }
}
```

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

Jawab:

Maksudnya node baru (newNode) akah disisipkan setelah node current. Maka, node yang setelah current yaitu current.next perlu diperbarui pointer prevnya agar merujuk ke newNode.

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

Jawab:

```
case 7:
    System.out.print(s:"Masukkan NIM yang dicari: ");
    String cariNIM = scan.nextLine();
    Mahasiswa01 mhsNIM = inputMahasiswa(scan);
    list.insertAfter(cariNIM, mhsNIM);
    break;
case 0:
```

1.2 Praktikum 2

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.

Langkah-langkah Percobaan

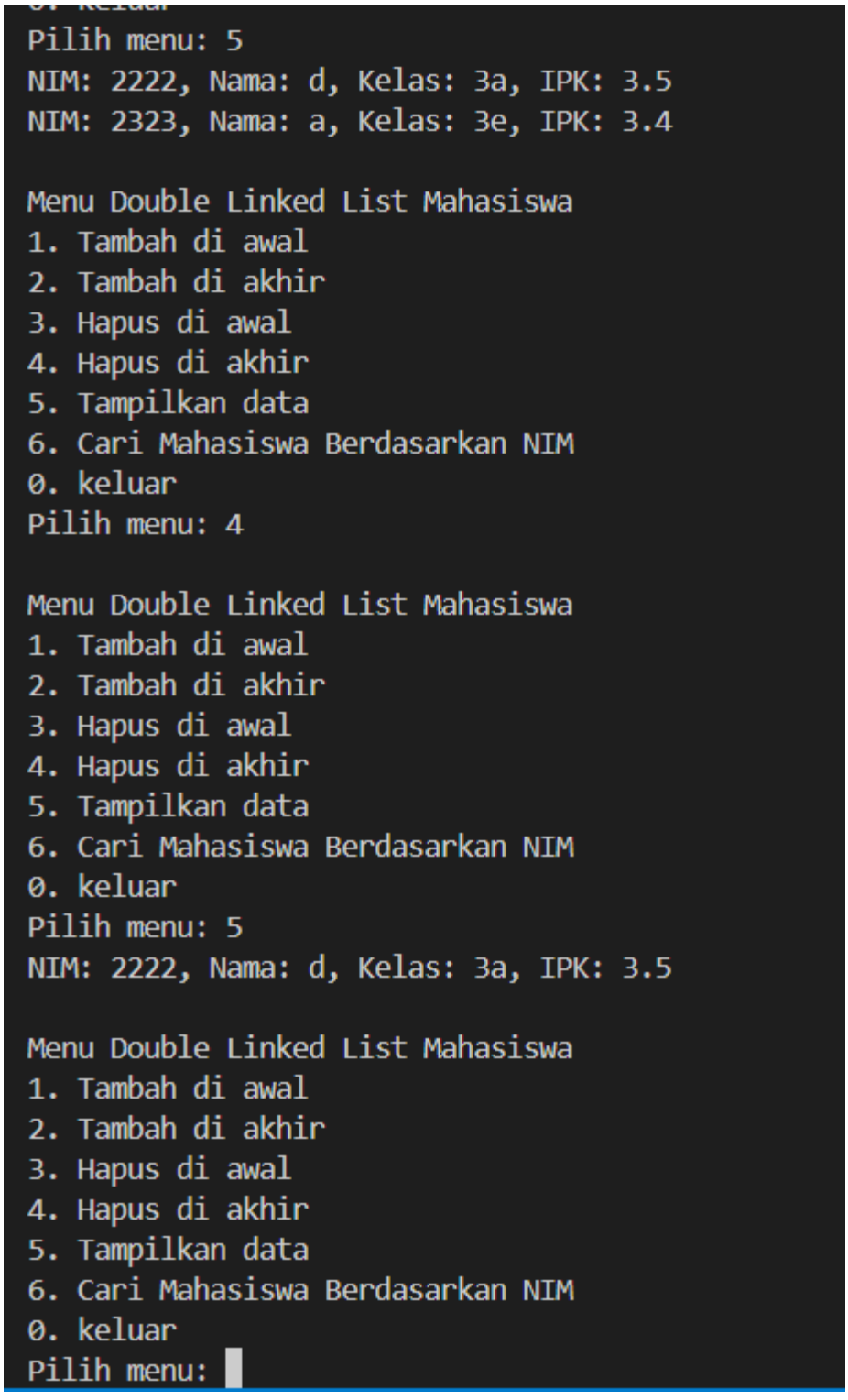
1. Buatlah method removeFirst() di dalam class DoubleLinkedLists.

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

Verifikasi Hasil Percobaan



Pertanyaan

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

Head = head.next, kode ini berfungsi untuk memindahkan pointer head ke node berikutnya
Head.prev = null, kode ini berfungsi memutus hubungan antara node pertama yang baru dengan node sebelumnya, sehingga node lama akan terhapus.

2. Modifikasi kode program untuk menampilkan pesan "Data sudah berhasil dihapus. Data yang terhapus adalah … "
Jawab:

```
public void removeFirst(){
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }
    if (head == tail) {
        System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah " + head.data.nama);
        head = tail = null;
    }else {
        System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah " + head.data.nama);
        head = head.next;
        head.prev = null;
    }
}
public void removeLast(){
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }
    if (head == tail) {
        System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah " + tail.data.nama);
        head = tail = null;
    }else {
        System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah " + tail.data.nama);
        tail = tail.prev;
        tail.next = null;
    }
}
```

2. Latihan Praktikum

2.1 Tugas Praktikum

a. Tambahkan fungsi add() pada kelas DoubleLinkedList untuk menambahkan node pada indeks tertentu
b. Tambahkan removeAfter() pada kelas DoubleLinkedList untuk menghapus node setelah data key.
c. Tambahkan fungsi remove() pada kelas DoubleLinkedList untuk menghapus node pada indeks tertentu.
d. Tambahkan fungsi getFirst(), getLast() dan getIndex() untuk menampilkan data pada node head, node tail dan node pada indeks tertentu.
e. tambahkan kode program dan fungsi agar dapat membaca size/ jumlah data pada Double Linked List