Nama : Siti Nikmatus Sholihah

NIM : 244107020014

Kelas : TI1B

JOBSHEET12

DOUBLE LINKED LIST

A. Percobaan 1

1. Perhatikan diagram class Mahasiswa22.java, Node22.java, dan LinkedList22.java dibawah ini! Diagram class dibawah akan dijadikan sebagai acuan dalam membuat kode program nantinya.

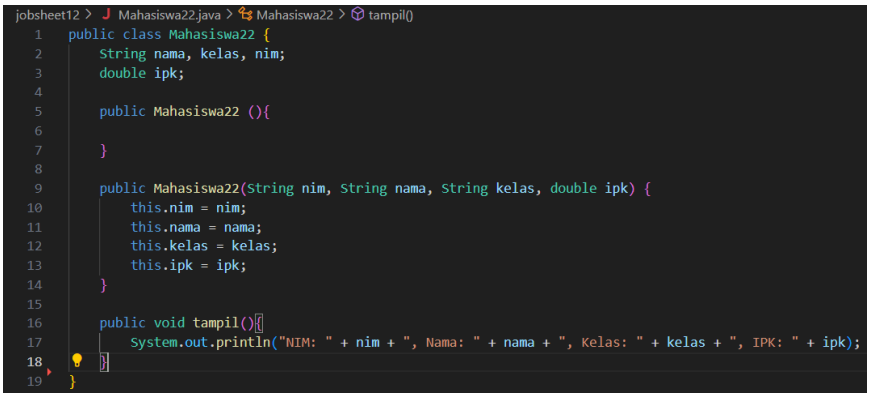
Mahasiswa22.java
nim : String nama : String kelas : String ipk : double
Mahasiswa22(String nim, String nama, String kelas, double ipk) tampil()

Node22.java
data : Mahasiswa22 prev : Node22 next : Node22
Node22(prev:null, data:Mahasiswa22 data, next: null)

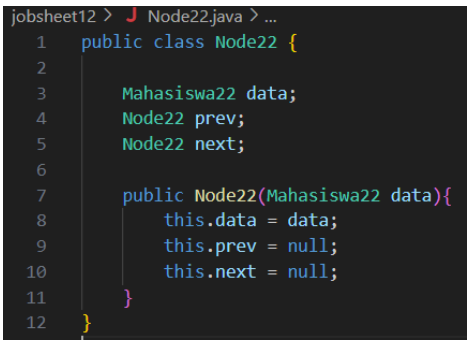
DoubleLinkedList22.java
head : Node22 tail : Node22
DoubleLinkedList() isEmpty() addFirst()

addLast() add(item: int, index: int): void

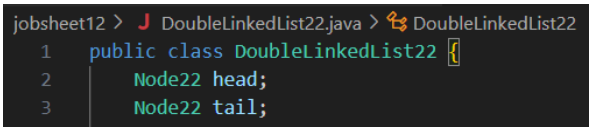
2. Buat folder baru dengan nama jobsheet12 didalam repository Praktikum-ASD.
3. Buat class baru dengan nama Mahasiswa22.java, lalu deklarasikan atribut sesuai dengan diagram diatas. Tambahkan juga konstruktor dan methodnya.



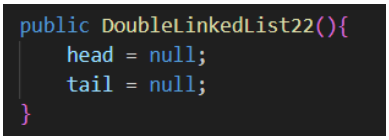
4. Buat class baru dengan nama Node22.java, lalu deklarasikan pula atributnya sesuai dengan diagram diatas. Lalu tambahkan konstruktor dan methodnya.



5. Buat class baru dengan nama DoubleLinkedList22.java didalam jobsheet12, lalu deklarasikan pula atributnya sesuai dengan diagram yang ada diatas.



6. Selanjutnya buat konstruktor pada class DoubleLinkedList.



7. Lalu buat method isEmpty() yang berfungsi untuk mengecek kondisi LinkedList kosong.

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

8. Kemudian buatlah method addFirst() yang berfungsi untuk menambahkan data baru didepan LinkedList.

```
public void addFirst(Mahasiswa22 data){
    Node22 newNode = new Node22(data);
    if (isEmpty()) {
        head = tail = newNode;
    } else {
        newNode.next = head;
        head.prev = newNode;
        head = newNode;
    }
}
```

9. Tambahkan punya method addLast() yang berfungsi untuk menambahkan data baru dibagian belakang LinkedList.

```
public void addLast(Mahasiswa22 data){
    Node22 newNode = new Node22(data);
    if (isEmpty()) {
        head = tail = newNode;
    } else {
        tail.next = newNode;
        newNode.prev = tail;
        tail = newNode;
    }
}
```

10. Untuk menambahkan data baru setelah node yang menyimpan data key, buat juga method insertAfter().

```
public void insertAfter(String keyNim, Mahasiswa22 data){
    Node22 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;
    }

    Node22 newNode = new Node22(data);

    if (current == tail) {
        current.next = newNode;
        newNode.prev = current;
        tail = newNode;
    } else {
        newNode.next = current.next;
        newNode.prev = current;
        current.next.prev = newNode;
        current.next = newNode;
    }

    System.out.println("Node berhasil disisipkan setelah NIM " + keyNim);
}
```

11. Buat method print() untuk menampilkan semua data yang sudah ditambahkan tadi.

```
public void print(){
    Node22 current = head;
    while (current != null) {
        current.data.tampil();
        current = current.next;
    }
}
```

12. Tambahkan class baru dengan nama DLLMain.java yang berfungsi untuk menjalankan semua method yang sudah dibuat sebelumnya.

```
import java.util.Scanner;
public class DLLMain {
        Run | Debug
        public static void main(String[] args) {
        DoubleLinkedList22 list = new DoubleLinkedList22();
        Scanner scan = new Scanner(System.in);
        int pilihan;
```

13. Buat menu pilihan pada class main menggunakan perulangan do while()

```
do {
    System.out.println(x:"\nMenu DoubleLinkedList Mahasiswa");
    System.out.println(x:"1. Tambah diawal");
    System.out.println(x:"2. Tambah diakhir");
    System.out.println(x:"3. Hapus diawal");
    System.out.println(x:"4. Hapus diakhir");
    System.out.println(x:"5. Tampilkan data");
    System.out.println(x:"6. Cari berdasarkan NIM");
    System.out.println(x:"0. Keluar");
    System.out.print(s:"Pilih menu: ");
    pilihan = scan.nextInt();
    scan.nextLine();
```

14. Tambahkan pemilihan switch case untuk menjalankan menu yang ada diatas.

```
    switch (pilihan) {
        case 1 -> {
            Mahasiswa22 mhs = inputMahasiswa(scan);
            list.addFirst(mhs);
        }

        case 2 -> {
            Mahasiswa22 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 yang dicari: ");
            String nim = scan.nextLine();
            Mahasiswa22 found = list.search(nim);
            if (found != null) {
                System.out.println(x:"Data ditemukan:");
                found.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!");
    }
} while (pilihan != 0);
```

15. Tambahkan method inputanMahasiswa didalam class DDLMain yang berfungsi untuk menginputkan data saat menambahkan data di awal LinkedList maupun diakhir LinkedList

```
public static Mahasiswa22 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();
    scan.nextLine(); // Buang newline

    return new Mahasiswa22(nim, nama, kelas, ipk);
}
```

16. Verifikasi hasil percobaan

```
Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
0. Keluar
Pilih menu: 1
Masukkan NIM: 20304050
Masukkan Nama: Athanasia
Masukkan Kelas: 1b
Masukkan IPK: 4.0

Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
0. Keluar
Pilih menu: 5
NIM: 20304050, Nama: Athanasia, Kelas: 1b, IPK: 4.0
```

Pertanyaan

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

   Jawab:

   peredaan antara single linked list dengan double linked list terdapat pada pointernya, jika single linked list hanya memilki 1 pointer yakni next(yang menyimpan link node setelahnya). Sedangkan double linked list mempunyai 2 pointer, yakni prev(yang menyimpan link node sebelumnya) dan next(yang menyimpan link node setelahnya).

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

   Jawab:

   atribut prev dideklarasikan untuk menyimpan link node sebelumnya, sedangkan next dideklarasikan untuk menyimpan link setelahnya.

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

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

   Jawab:

   konstruktor tersebut berfungsi untuk menandai mana bagian head dan tail. Pada konstruktor tersebut dideklarasikan head dan tailnya masih null karena tidak mengarah ke node manapun.

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

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

   Jawab:

   kode tersebut mengecek apakah linked list masih kosong atau tidak, jika iya maka newNode akan menjadi satu-satunya data yang ada pada linked list sehingga head dan tail sama-sama menunjuk ke newNode tersebut.

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

   Jawab:

   kode tersebut menyambungkan node baru ke node yang sebelumnya jadi pertama.

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

   Jawab:

```
public void print(){
    if (isEmpty()) {
        System.out.println(x:"Linked list masih kosong!!!");
    } else {
    Node22 current = head;
    while (current != null) {
        current.data.tampil();
        current = current.next;
    }
    }
}
```

7. Pada insertAfter(), apa maksud dari kode berikut ?

   current.next.prev = newNode;

   Jawab:

   Kode tersebut menghubungkan newNode dengan node yang awalnya ada dibelakang current, sehingga mereka saling terhubung 2 arah.

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

   Jawab:

```
public void insertAfter(Node22 prevNode, Mahasiswa22 data) {
    if (prevNode == null) {
        System.out.println(x:"Tidak bisa menambahkan data");
        return;
    }

    Node22 newNode = new Node22(data);

    newNode.next = prevNode.next;
    newNode.prev = prevNode;
    prevNode.next = newNode;

    if (newNode.next != null) {
        newNode.next.prev = newNode;
    }
}
```

```
case 7 -> {
    System.out.print(s:"Masukkan NIM yang akan disisipkan setelahnya: ");
    String keyNim = scan.nextLine();
    Mahasiswa22 mhsBaru = inputMahasiswa(scan);
    list.insertAfter(keyNim, mhsBaru);
}
```

   Output:

```
Pilih menu: 1
Masukkan NIM: 123
Masukkan Nama: Athanasia
Masukkan Kelas: 1B
Masukkan IPK: 3.5

Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
7. Sisipkan data menggunakan key
0. Keluar
Pilih menu: 7
Masukkan NIM yang akan disisipkan setelahnya: 123
Masukkan NIM: 125
Masukkan Nama: Budi
Masukkan Kelas: 1C
Masukkan IPK: 3.7
Node berhasil disisipkan setelah NIM 123

Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
7. Sisipkan data menggunakan key
0. Keluar
Pilih menu: 5
NIM: 123, Nama: Athanasia, Kelas: 1B, IPK: 3.5
NIM: 125, Nama: Budi, Kelas: 1C, IPK: 3.7
```

B. Percobaan 2

1. Tambahkan method removeFirst() yang berfungsi untuk menghapus data yang ada pada bagian depan LinkedList.

```
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 = head;
    }
}
```

2. Tambahkan method removeLast() yang berfungsi untuk menghapus data yang ada pada bagian belakang LinkedList.

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

3. Analisa hasil percobaan

```
Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
0. Keluar
Pilih menu: 1
Masukkan NIM: 20304050
Masukkan Nama: Athanasia
Masukkan Kelas: 1b
Masukkan IPK: 4.0

Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
0. Keluar
Pilih menu: 3
```

Pertanyaan

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

   head = head.next;

   head.prev = null;

   Jawab:

   Maksud dari kode tersebut adalah menggeser head ke node berikutnya dan memutuskan link ke node lama agar node pertama terhapus dengan benar.

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

   Jawab:

```
public String toString() {
    return "NIM: " + nim + ", Nama: " + nama + ", Kelas: " + kelas + ", IPK: " + ipk;
}
```

```
public void removeFirst(){
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }

    Mahasiswa22 dataDihapus = head.data;

    if (head == tail) {
        head = tail = null;
    } else {
        head = head.next;
        head.prev = head;
    }
    System.out.println("Data sudah berhasil dihapus. Data yang berhasil dihapus adalah " + dataDihapus);
}
```

   Output:

```
Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
7. Sisipkan data menggunakan key
0. Keluar
Pilih menu: 5
NIM: 123, Nama: a, Kelas: 1b, IPK: 3.6

Menu DoubleLinkedList Mahasiswa
1. Tambah diawal
2. Tambah diakhir
3. Hapus diawal
4. Hapus diakhir
5. Tampilkan data
6. Cari berdasarkan NIM
7. Sisipkan data menggunakan key
0. Keluar
Pilih menu: 3
Data sudah berhasil dihapus. Data yang berhasil dihapus adalah NIM: 123, Nama: a, Kelas: 1b, IPK: 3.6
```