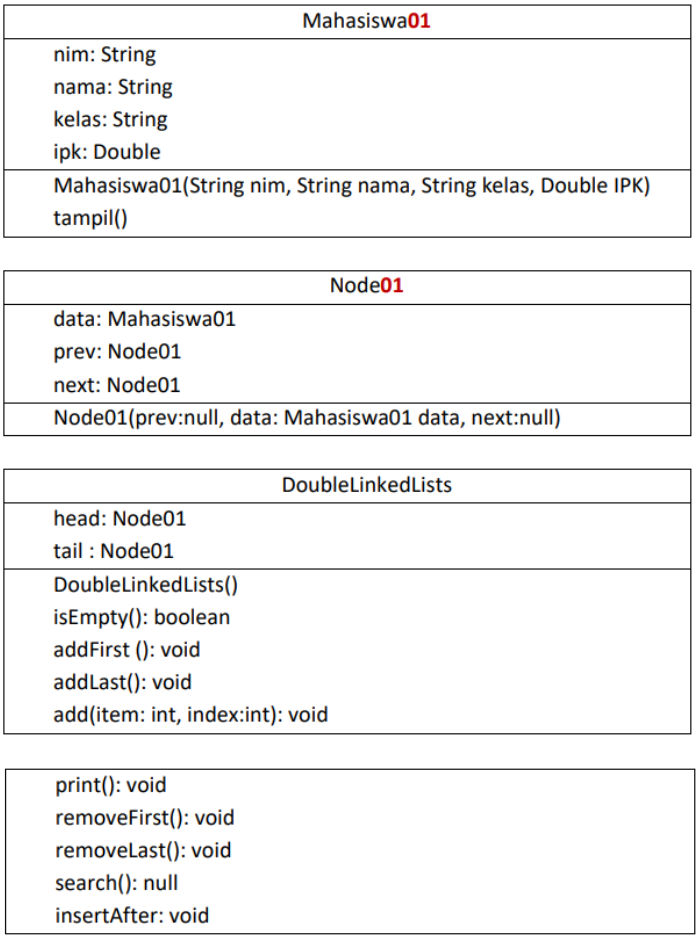
Nama : Dicky Darmawan
Kelas : TI 1B
No. Absen : 08
NIM : 244107020037

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
public class Mahasiswa08 {

    String nim, nama, kelas;
    double ipk;

    public Mahasiswa08(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: "
    }
}
```

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 Node08 {

    Mahasiswa08 data;
    Node08 next, prev;

    public Node08(Mahasiswa08 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 DoubleLinkedLists {

    Node08 head, tail;
```

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

```java
public DoubleLinkedLists() {
    head = null;
    tail = null;
}
```

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

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

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

```java
public void addFirst(Mahasiswa08 data) {
    Node08 newNode = new Node08(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.

```java
public void addLast(Mahasiswa08 data) {
    Node08 newNode = new Node08(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

```java
public void insertAfter(String keyNim, Mahasiswa08 data) {
    Node08 current = head;

    // Mencari node berdasarkan NIM
    while (current != null && !current.data.nim.equalsIgnoreCase(keyNim)) {
        current = current.next;
    }

    if (current == null) {
        System.out.println("Node dengan NIM " + keyNim + " tidak ditemukan.");
        return;
    }

    Node08 newNode = new Node08(data);

    // Current adalah tail, cukup menambahkan 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() {
    Node08 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.

```
import java.util.Scanner;

public class DLLMain {

    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        DoubleLinkedLists list = new DoubleLinkedLists();
        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 : {
        Mahasiswa08 mhs = inputMahasiswa(scan);
        list.addFirst(mhs);
        break;
    }
    case 2 : {
        Mahasiswa08 mhs = inputMahasiswa(scan);
        list.addLast(mhs);
        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();
        Node08 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;
```

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.

    Kurangnya break pada kondisi switch-case

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   : 1245789630
Masukkan Nama  : Ronald Weasley
Masukkan Kelas : Gryffindor
Masukkan IPK   : 3,8

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: 1245789630, Nama: Ronald Weasley, Kelas: Gryffindor, IPK: 3.8
```

Pertanyaan

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

   Node pada single linked list hanya berisikan atribut data dan next, sedangkan node pada double linked list berisikan 3 atribut, yakni data, next, dan previous yang lebih memudahkan dalam penggunaan

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

   Atribut next digunakan untuk membaca node selanjutnya, atribut prev untuk membaca node sebelumnya selama node tersebut bukan sebagai head

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

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

Konstruktor ini digunakan sebagai node pertama (head) dan terakhir (tail). Keduanya diberi nilai null karena masih dalam kondisi kosong

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

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

Selama data masih kosong, node yang baru dibuat akan dijadikan sebagai head dan tail sekaligus

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

   Karena data akan ditambahkan pada bagian depan, jadi atribut prev dari head perlu disambungkan terlebih dahulu dengan node baru yang akan ditambahkan

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

```
if (isEmpty()) {
    System.out.println(x:"Linked List masih kosong!");
```

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

   current.next.prev = newNode;

   Potongan kode tersebut merujuk kepada previous dari node setelah current untuk disambungkan dengan newNode

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

```
case 7 : {
    System.out.print(s:"Masukkan NIM: ");
    String keyNim = scan.nextLine();
    Mahasiswa08 mhs = inputMahasiswa(scan);
    list.insertAfter(keyNim, mhs);
    break;
```

Percobaan 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.

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.

```
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

```
Pilih menu: 5
NIM: 234567, Nama: Salazar Slytherin, Kelas: Slytherin, IPK: 4.0
NIM: 123456, Nama: Harry Potter, 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
6. Cari Mahasiswa berdasarkan NIM
7. Tambah setelah key
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. Cari Mahasiswa berdasarkan NIM
7. Tambah setelah key
0. Keluar
Pilih menu: 5
NIM: 123456, Nama: Harry Potter, Kelas: Gryffindor, IPK: 4.0
```

Pertanyaan

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

   head = head.next;

   head.prev = null;

   Atribut head langsung dipindahkan ke node setelahnya dan atribut prev dari head yang baru diset null karena tidak menunjuk node manapun

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

```
case 3 : {
    System.out.println(x:"Data sudah berhasil dihapus, data yang terhapus adalah");
    list.head.data.tampil();
    list.removeFirst();
    break;
}
case 4 : {
    System.out.println(x:"Data sudah berhasil dihapus, data yang terhapus adalah");
    list.tail.data.tampil();
    list.removeLast();
    break;
}
```

   Akan ditampikan terlebih dulu, lalu dihapus

Tugas Praktikum

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

```
public void add(int index, Mahasiswa08 data) {
    if (index == 0) {
        addFirst(data);
    } else if (index >= size) {
        addLast(data);
    } else {
        Node08 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        Node08 newNode = new Node08(current.prev, data, current);
        current.prev.next = newNode;
        current.prev = newNode;
    }
    size++;
}
```

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

```
public void removeAfter(String nim) {
    Node08 current = head;
    while (current != null && !current.data.nim.equals(nim)) {
        current = current.next;
    }

    if (current != null && current.next != null) {
        Node08 toDelete = current.next;
        current.next = toDelete.next;
        if (toDelete.next != null) {
            toDelete.next.prev = current;
        } else {
            tail = current;
        }
        size--;
    } else {
        System.out.println(x:"Tidak ada node setelah NIM tersebut atau
    }
}
```

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

```
public void remove(int index) {
    if (index < 0 || index >= size) {
        System.out.println(x:"Index tidak valid.");
        return;
    }

    if (index == 0) {
        removeFirst();
    } else if (index == size - 1) {
        removeLast();
    } else {
        Node08 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        current.prev.next = current.next;
        current.next.prev = current.prev;
        size--;
    }
}
```

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

```java
public void getFirst() {
    if (head != null) {
        head.data.tampil();
    } else {
        System.out.println(x:"Data kosong.");
    }
}

public void getLast() {
    if (tail != null) {
        tail.data.tampil();
    } else {
        System.out.println(x:"Data kosong.");
    }
}

public void getIndex(int index) {
    if (index < 0 || index >= size) {
        System.out.println(x:"Index tidak valid.");
        return;
    }
    Node08 current = head;
    for (int i = 0; i < index; i++) {
        current = current.next;
    }
    current.data.tampil();
}
```

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

```java
public int getSize() {
return size;
```