LAPORAN HASIL PRAKTIKUM ALGORITMA DAN STRUKTUR DATA
JOBSHEET 12

NAMA :MOHAMAT FAUZI ROHMAN

NIM: 244107020067

KELAS : TI_1E

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2025

JOBSHEET XII

DOUBLE LINKED LIST

12. Praktikum

12.1 Percobaan 1

12.1.1 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

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

```
public class Mahasiswa01 {

    public String nim;
    public String nama;
    public String kelas;
    public double ipk;

public 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

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

    public Node01(Mahasiswa01 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

```
public class DoubleLinkedList01 {
    Node01 head;
    Node01 tail;
```

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

```
public DoubleLinkedList01(){
    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);
    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. 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.

```
import java.util.Scanner;
public class DLLMain {
    public static void main(String[] args) {
    DoubleLinkedList01 list = new DoubleLinkedList01();
    Scanner scan = new Scanner(System.in);
    int pilihan;
```

13. Buatlah menu pilihan pada class main

```
do{
    System.out.println("\nMenu Double Linked List Mahasiswa");
    System.out.println("1. Tambah di awal");
    System.out.println("2. Tambah di akhir");
    System.out.println("3. Hapus di awal");
    System.out.println("4. Hapus di akhir");
    System.out.println("5. Tampilkan data");
    System.out.println("6. Cari Mahasiswa berdasarkan NIM");
    System.out.println("0. Keluar");
    System.out.print("Pilih Menu: ");
    pilihan = scan.nextInt();
    scan.nextLine();
```

14. Tambahkan switch case untuk menjalankan menu pilihan di atas

```
    switch (pilihan) {
        case 1 ->{
            Mahasiswa01 mhs = list.inputMahasiswa(scan);
            list.addFirst(mhs);
        }

        case 2 ->{
            Mahasiswa01 mhs = list.inputMahasiswa(scan);
            list.addLast(mhs);
        }

        case 3 -> list.removeFirst();
        case 4 -> list.removeLast();
        case 5 -> list.print();
        case 6 -> {
            System.out.print("Masukkan NIM yang dicari: ");
            String nim = scan.nextLine();
            Node01 found = list.search(nim);
            if (found != null) {
                System.out.println("Data ditemukan");
                found.data.tampil();
            } else{
                System.out.println("Data tidak ditemukan");
            }
        }

        case 0 -> System.out.println("Keluar dari program.");
        default -> System.out.println("Pilihan tidak valid!");
        }
```

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

Tambahkan method inputMahasiswa agar dapat memasukkan data mahasiswa

```
public static Mahasiswa01 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();
    return new Mahasiswa01(nim, nama, kelas, ipk);
}
```

12.1.2 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
```

12.1.3 Pertanyaan

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

   : Perbedaan yang paling signifikan ialah ketika single linked list hanya bisa digunakan satu arah (dari depan ke belakang), sedangkan double linked list dapat digunakan 2 arah (dari depan ke belakang atau dari belakang ke depan)

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

   : Atribut next digunakan untuk menunjuk pada node setelahnya, sedangkan prev digunakan untuk menunjukan pada node sebelumnya

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

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

: Konstruktor tersebut digunakan untuk mengatur head dan tail ke null

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

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

: Digunakan apabila double linked list kosong, maka node input akan dijadikan sebagai head sekaligus tail

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

: Artinya pointer prev pada head digunakan untuk menunjuk ke node baru

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

: Tambahkan method isEmpty di dalam method print

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

Hasil

```
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
Linked List masih kosong!
```

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

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

: Maksud kode tersebut ialah node setelah current pada bagian pointer prev menunjuk pada node baru

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

: Tambahkan pilihan ke-7 untuk insert after pada menu

```
System.out.println(x:"7. Sisipkan Data Baru");
```

Isi case 7 dengan kode berikut

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

Hasil

```
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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 1
Masukkan NIM: 111
Masukkan Nama: Agus
Masukkan Kelas: 1A
Masukkan IPK: 3.5

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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 2
Masukkan NIM: 333
Masukkan Nama: Roy
Masukkan Kelas: 1G
Masukkan IPK: 3.4
```

```
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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 7
Masukkan NIM yang dicari: 111
Masukkan NIM: 222
Masukkan Nama: Bella
Masukkan Kelas: 1C
Masukkan IPK: 4.0
Node berhasil disisipkan setelah NIM 111

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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 5
NIM: 111, Nama: Agus, Kelas: 1A, IPK: 3.5
NIM: 222, Nama: Bella, Kelas: 1C, IPK: 4.0
NIM: 333, Nama: Roy, Kelas: 1G, IPK: 3.4
```

12.2 Percobaan 2

12.2.1 Langkah-langkah Percobaan

1. Buatlah method removeFirst() di dalam class DoubleLinkedLists

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

12.2.2 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
7. Sisipkan Data Baru
0. Keluar
Pilih Menu: 1
Masukkan NIM: 111
Masukkan Nama: Agus
Masukkan Kelas: 1A
Masukkan IPK: 3.3

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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 2
Masukkan NIM: 333
Masukkan Nama: Bella
Masukkan Kelas: 1D
Masukkan IPK: 3.5

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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 5
NIM: 111, Nama: Agus, Kelas: 1A, IPK: 3.3
NIM: 333, Nama: Bella, Kelas: 1D, IPK: 3.5
```

```
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. Sisipkan Data Baru
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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 5
NIM: 333, Nama: Bella, Kelas: 1D, IPK: 3.5
```

12.2.3 Pertanyaan

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

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

: Maksud dari kode tersebut untuk menjadikan node setelah head menjadi head baru dan prev dari head baru di set menjadi null agar head yang lama terputus dengan head yang baru

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

```
public void removeFirst(){
    if (isEmpty()) {
        System.out.println("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;
    }
}
```

Hasil

```
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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 1
Masukkan NIM: 111
Masukkan Nama: Agus
Masukkan Kelas: 1A
Masukkan IPK: 3.4

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. Sisipkan Data Baru
0. Keluar
Pilih Menu: 3
Data sudah berhasil dihapus. Data yang terhapus adalah Agus
```

12.3 Tugas

Data Mahasiswa

```
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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 1
Masukkan NIM: 111
Masukkan Nama: Agus
Masukkan Kelas: 1A
Masukkan IPK: 3.3

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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 2
Masukkan NIM: 222
Masukkan Nama: Boy
Masukkan Kelas: 1C
Masukkan IPK: 3.5
```

```
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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 2
Masukkan NIM: 333
Masukkan Nama: Bella
Masukkan Kelas: 1E
Masukkan IPK: 3.6

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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 5
NIM: 111, Nama: Agus, Kelas: 1A, IPK: 3.3
NIM: 222, Nama: Boy, Kelas: 1C, IPK: 3.5
NIM: 333, Nama: Bella, Kelas: 1E, IPK: 3.6
```

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

Tambahkan method add() pada class DoubleLinkedList

```
public void add(int index, Mahasiswa01 data){
    if (index == 0) {
        addFirst(data);
    } else if (index == size) {
        addLast(data);
    } else {
        Node01 newNode = new Node01(data);
        Node01 current = head;
    for (int i = 0; i < index; i++) {
        current = current.next;
    }
        newNode.prev = current.prev;
        newNode.next = current;
        current.prev.next = newNode;
        current.prev = newNode;
        size++;
        }
    }
```

Tambahkan juga pada class main

```
case 8 -> {
            System.out.print("Masukkan Index: ");
            int index = scan.nextInt();
            scan.nextLine();
            list.add(index, list.inputMahasiswa(scan));
        }
```

Hasil

```
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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 8
Masukkan Index: 1
Masukkan NIM: 666
Masukkan Nama: Via
Masukkan Kelas: 1F
Masukkan IPK: 3.7

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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 5
NIM: 111, Nama: Agus, Kelas: 1A, IPK: 3.3
NIM: 666, Nama: Via, Kelas: 1F, IPK: 3.7
NIM: 222, Nama: Boy, Kelas: 1C, IPK: 3.5
NIM: 333, Nama: Bella, Kelas: 1E, IPK: 3.6
```

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

Tambahkan method removeAfter() pada class DoubleLinkedList

```
public void removeAfter(String keyNim){
    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 remove = current.next;
    if (remove == tail) {
        tail = current;
        current.next = null;
    } else{
        current.next = remove.next;
        remove.next.prev = current;
    }
    size--;
}
```

Tambahkan juga pada class main

```
case 9 -> {
            System.out.print("Masukkan NIM: ");
            String key = scan.nextLine();
            list.removeAfter(key);
        }
```

Hasil

```
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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 9
Masukkan NIM: 111

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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 5
NIM: 111, Nama: Agus, Kelas: 1A, IPK: 3.3
NIM: 222, Nama: Boy, Kelas: 1C, IPK: 3.5
NIM: 333, Nama: Bella, Kelas: 1E, IPK: 3.6
```

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

Tambahkan method remove() pada class DoubleLinkedList

```
public void remove(int index){
    if (index == 0) {
        removeFirst();
    } else if (index == size - 1) {
        removeLast();
    } else {
    Node01 current = head;
    for (int i = 0; i < index; i++) {
        current = current.next;
    }
        current.prev.next = current.next;
        current.next.prev = current.prev;
        size--;
    }
}
```

Tambahkan juga pada class main

```
    case 10 -> {
              System.out.print("Masukkan Indeks: ");
              int index = scan.nextInt();
              list.remove(index);
         }
```

Hasil

```
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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 10
Masukkan Indeks: 0
Data sudah berhasil dihapus. Data yang terhapus adalah Agus

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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 5
NIM: 222, Nama: Boy, Kelas: 1C, IPK: 3.5
NIM: 333, Nama: Bella, Kelas: 1E, IPK: 3.6
```

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

Tambahkan method getFirst(), getLast, dan getIndex pada class DoubleLinkedList

```
public Mahasiswa01 getFirst(){
    if (isEmpty()) {
        System.out.println("Linked list masih kosong");
        return null;
    }
    return head.data;
}

public Mahasiswa01 getLast(){
    if (isEmpty()) {
        System.out.println("Linked list masih kosong");
        return null;
    }
    return tail.data;
}
public Mahasiswa01 getIndex(int index){
    Node01 current = head;
    for(int i = 0; i < index; i++){
        current = current.next;
    }
    return current.data;
}
```

Tambahkan juga pada class main

```
case 11 -> {
            Mahasiswa01 first = list.getFirst();
            if (first != null) {
                System.out.println("Data Mahasiswa Terdepan: ");
                first.tampil();
            }
            System.out.println();
            Mahasiswa01 last = list.getLast();
            if (last != null) {
                System.out.println("Data Mahasiswa Terakhir: ");
                last.tampil();
            }
            System.out.println();
            int indeks = -1;
            System.out.print("Masukkan Indeks: ");
            indeks = scan.nextInt();
            scan.nextLine();
            Mahasiswa01 index = list.getIndex(indeks);
            System.out.println("Data Indeks ke-" + indeks);
            index.tampil();
        }
```

Hasil

```
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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 11
Data Mahasiswa Terdepan:
NIM: 222, Nama: Boy, Kelas: 1C, IPK: 3.5

Data Mahasiswa Terakhir:
NIM: 333, Nama: Bella, Kelas: 1E, IPK: 3.6

Masukkan Indeks: 1
Data Indeks ke-1
NIM: 333, Nama: Bella, Kelas: 1E, IPK: 3.6
```

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

Tambahkan method getSize() pada class DoubleLinkedList

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

Tambahkan juga pada class main

```
        case 12 -> {
            System.out.println("Jumlah Data pada Double Linked List: " + list.getSize());
        }
```

Hasil

```
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. Sisipkan Data Baru
8. Tambah Data di Indeks Tertentu
9. Hapus Data Setelah Data Key
10. Hapus Data di Indeks Tertentu
11. Tampilkan Data di Head, Tail dan Indeks Tertentu
12. Jumlah Data pada DLL
0. Keluar
Pilih Menu: 12
Jumlah Data pada Double Linked List: 2
```