LAPORAN HASIL PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

Jobsheet 12

Adi Luhung

244107020088

Kelas 1E

Program Studi Teknik Informatika

Jurusan Teknologi Informasi

Politeknik Negeri Malang

2025

1. Praktikum
   1.1 Percobaan 1
      1. Membuat class Mahasiswa01 yang berisi atribut, konstruktor dan method tampil().

```
package Jobsheet12;

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);
    }
}
```

      2. Membuat class Node01 yang berisi object dan konstruktor.

```
package Jobsheet12;

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

    public Node01(Mahasiswa01 data) {
        this.data = data;
        this.next = null;
        this.prev = null;
    }
}
```

3. Membuat class DoubleLinkedList01 yang berisi method-method untuk algoritma Double Linked List.

```
package Jobsheet12;
public class DoubleLinkedList01 {
    Node01 head;
    Node01 tail;

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

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

4. Membuat class DLLMain01 untuk memanggil method-method yang sudah dibuat.

```
package Jobsheet12;
import java.util.Scanner;
public class DLLMain01 {
    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);
    }
  public static void main(String[] args) {
    DoubleLinkedList01 list = new DoubleLinkedList01();
    Scanner scan = new Scanner (System.in);
    int pilihan;

    do {
        System.out.println("\n=== Menu Antrian layanan Akademik ===");
        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("7. Cari mahasiswa berdasarkan NIM");
        System.out.println("0. Keluar");
        System.out.print("Pilih menu: ");
        pilihan = scan.nextInt();
        scan.nextLine();
```

```
        switch (pilihan) {
            case 1 -> {
                Mahasiswa01 mhs = inputMahasiswa(scan);
                list.addFirst(mhs);
            }
            case 2 -> {
                Mahasiswa01 mhs = inputMahasiswa(scan);
                list.addLast(mhs);
            }
            // case 3 -> list.removeFirst();
            // case 4 -> list.removeLast();
            case 5 -> list.print();
            case 7 -> {
                System.out.println("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!");
        }
    } while (pilihan != 0);
  }
}
```

5. Hasil running kode programz

```
=== Menu Antrian layanan Akademik ===
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: Griffindor
Masukkan IPK: 4,0

=== Menu Antrian layanan Akademik ===
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 Griffindor, IPK: 4.0
```

1.1.1 Pertanyaan

1. Perbedaan Single Linked List dengan Double Linked List adalah pada isi node nya. Pada Single Linked List hanya terdapat data dan next. Tetapi pada Double Linked List, terdapat data dan dua pointer yaitu next dan prev.

2. Pada class Node01 terdapat atribut next dan prev, yang dimana next adalah pointer yang menujuk ke node selanjutnya, dan prev adalah pointer yang menunjuk ke node sebelumnya.

3. Fungsi konstruktor tersebut adalah saat objek DoubleLinkedList01 dibuat, linked list langsung dalam keadaan kosong dan siap digunakan.

4. pada class DoubleLinkedList01 method addFirst()

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

Kode teersebut berfungsi saat linked list dalam keadaan kosong, maka data baru yang akan ditambahkan akan langsung di tandai sebagai head dan tail.

5. Pada class DoubleLinkedList01 method addFirst()

```
head.prev = newNode;
```

Fungsi statement tersebut adalah untuk mengubah pointer prev dari node head menjadi newNode (menunjuk ke nodenewNode) yang semula adalah null.

6. Jika maksud dari pertanyaannya adalah memberi peringatan saat linked list masih kosong, maka modifikasinya adalah seperti ini:

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

7. Pada class DoubleLinkedList01 method addFirst()

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

Maksud dari kode tersebut adalah untuk mengubah atribut prev pada node setelah node current menjadi newNode (menunjuk ke newNode).

8. Memodifikasi class DLLMain pada perulangan Do While agar method insertAfter() berguna.

```
    do {
        System.out.println("\n=== Menu Antrian layanan Akademik ===");
        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. Tambah setelah NIM");
        System.out.println("7. Cari berdasarkan NIM");
        System.out.println("0. Keluar");
        System.out.print("Pilih menu: ");
        pilihan = scan.nextInt();
        scan.nextLine();

        switch (pilihan) {
            case 1 -> {
                Mahasiswa01 mhs = inputMahasiswa(scan);
                list.addFirst(mhs);
            }
            case 2 -> {
                Mahasiswa01 mhs = inputMahasiswa(scan);
                list.addLast(mhs);
            }
            // case 3 -> list.removeFirst();
            // case 4 -> list.removeLast();
            case 5 -> list.print();
            case 6 -> {
                System.out.print("Masukkan NIM: ");
                String keyNim = scan.nextLine();
                Mahasiswa01 mhs = inputMahasiswa(scan);
                list.insertAfter(keyNim, mhs);
            }
            case 7 -> {
                System.out.println("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!");
        }
    } while (pilihan != 0);
  }
}
```

1.2 Percobaan 2

1. Menambahkan method removeFirst() dan removeLast() pada class DoubleLinkedList01.

```
    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;
        }
    }
    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;
        }
    }
}
```

2. Hasil running kode program

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 125, Nama: Erick, Kelas 1A, IPK: 3.6
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
NIM: 124, Nama: Budi, Kelas 1E, IPK: 3.7


=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 3
```

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 4

=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
```

1.2.1 Pertanyaan

1. Pada class DoubleLinkedList01 method addFirst()

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

Baris pertama adalah untuk memindahkan tanda head ke node selanjutnya, kemudian untuk beris kedua, atribut prev dari head yang baru diubah menjadi null. Sehingga head lama sudah tidak terhubung dengan linked list.

2. Memodifikasi kode program tepatnya pada method removeFirst() dan removeLast() agar dapat menampilkan pesan saat data dihapus

```
public void removeFirst() {
    if (isEmpty()) {
        System.out.println("List kosong, tidak bisa dihapus.");
        return;
    }
    System.out.print("Data sudah berhasil dihapus. Data yang terhapus adalah: ");
    head.data.tampil();
    if (head == tail) {
        head = tail = null;

    } else {
        head = head.next;
        head.prev = null;
    }
}
```

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

Hasil kode program

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 125, Nama: Erick, Kelas 1A, IPK: 3.6
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
NIM: 124, Nama: Budi, Kelas 1E, IPK: 3.7


=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 3
Data sudah berhasil dihapus.
Data yang terhapus adalah: NIM: 125, Nama: Erick, Kelas 1A, IPK: 3.6
```

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 4
Data sudah berhasil dihapus.
Data yang terhapus adalah: NIM: 124, Nama: Budi, Kelas 1E, IPK: 3.7

=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
```

2. Tugas
    1. Memodifikasi kode program dengan menambahkan method add() pada class DoubleLinkedList01() dan menambahkan pilihan menu pada class DLLMain01.

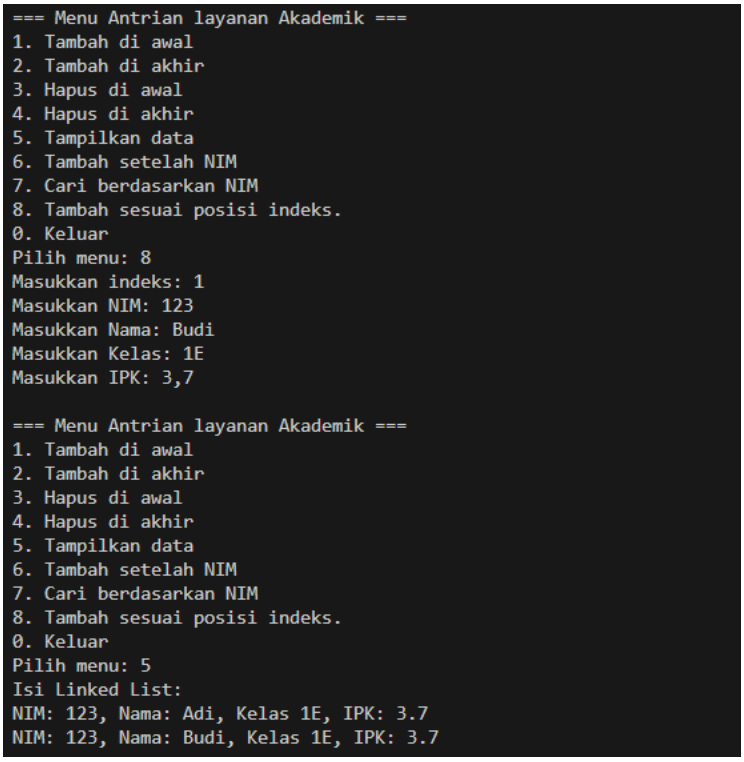
```
    public void add(int index, Mahasiswa01 data) {
        if (index < 0 || index > size) {
            System.out.println("Index di luar batas.");
            return;
        }
        if (index == 0) {
            addFirst(data);
        } else if (index == size) {
            addLast(data);
        } else {
            Node01 current = head;
            for (int i = 0; i < index; i++) {
                current = current.next;
            }
            Node01 newNode = new Node01(data);
            newNode.prev = current.prev;
            newNode.next = current;
            current.prev.next = newNode;
            current.prev = newNode;
        }
    }
```

```
case 8 -> {
    System.out.print("Masukkan indeks: ");
    int idx = scan.nextInt(); scan.nextLine();
    Mahasiswa01 mhs = inputMahasiswa(scan);
    list.add(idx, mhs);
}
```

Hasil running kode program

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
0. Keluar
Pilih menu: 1
Masukkan NIM: 123
Masukkan Nama: Adi
Masukkan Kelas: 1E
Masukkan IPK: 3,7

=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.7
```



2. Menambahkan method removeAfter() pada class DoubleLinkedList01 untuk menghapus node setelah kata kunci tertentu. Dan juga menambahkan pilihan menu baru pada class DLLMain01.

```
public void removeAfter(String keyNim) {
    Node01 current = head;
    while (current != null && !current.data.nim.equals(keyNim)) {
        current = current.next;
    }
    if (current == null || current.next == null) {
        System.out.println("Node setelah " + keyNim + " tidak ditemukan.");
        return;
    }
    Node01 toRemove = current.next;
    if (toRemove == tail) {
        tail = current;
        current.next = null;
    } else {
        current.next = toRemove.next;
        toRemove.next.prev = current;
    }
    System.out.println("Node setelah " + keyNim + " berhasil dihapus.");
}
```

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

Hasil running kode program.

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
NIM: 124, Nama: Budi, Kelas 1W, IPK: 3.7
```

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
0. Keluar
Pilih menu: 9
Masukkan NIM: 123
Node setelah 123 berhasil dihapus.
```

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
```

3. Menambahkan method remove() pada class DoubleLinkedList01 agar dapat menghapus node pada indeks yang diinginkan. Dan menambahkan pilihan menu pada class DDLMain01.

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

```
    public void remove(int index) {
        if (index < 0 || index >= size) {
            System.out.println("Index di luar batas.");
            return;
        }
        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;
            System.out.print("Data yang dihapus: ");
            current.data.tampil();
        }
    }
```

Hasil running kode program.

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
10. Hapus sesuai posisi indeks
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
NIM: 124, Nama: budi, Kelas 1E, IPK: 3.8


=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
10. Hapus sesuai posisi indeks
0. Keluar
Pilih menu: 10
Masukkan indeks: 0
Data sudah berhasil dihapus.
Data yang terhapus adalah: NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
```

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
10. Hapus sesuai posisi indeks
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 124, Nama: budi, Kelas 1E, IPK: 3.8
```

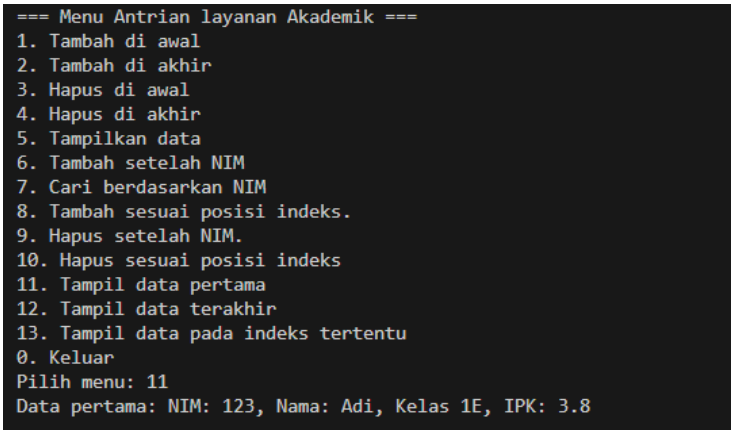
4. Menambahkan method getFirst(), getLast(), getIndex() untuk menampilkan data pada posisi tertentu, dan menambahka pilihan menu baru pada class DLLMain01.

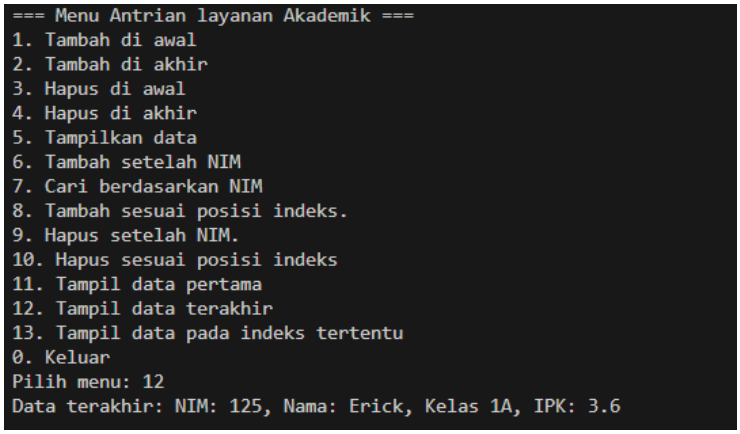
```
    public void getFirst() {
        if (!isEmpty()) {
            System.out.print("Data pertama: ");
            head.data.tampil();
        } else {
            System.out.println("List kosong.");
        }
    }
    public void getLast() {
        if (!isEmpty()) {
            System.out.print("Data terakhir: ");
            tail.data.tampil();
        } else {
            System.out.println("List kosong.");
        }
    }
    public void getIndex(int index) {
        if (index < 0 || index >= size) {
            System.out.println("Index di luar batas.");
            return;
        }
        Node01 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        System.out.print("Data pada index " + index + ": ");
        current.data.tampil();
    }
```

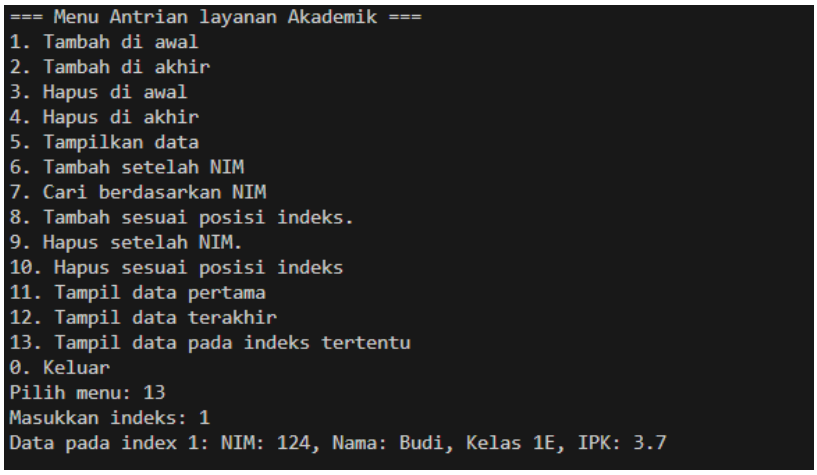
```
case 11 -> list.getFirst();
case 12 -> list.getLast();
case 13 -> {
    System.out.print("Masukkan indeks: ");
    int idx = scan.nextInt(); scan.nextLine();
    list.getIndex(idx);
}
```

Hasil running kode program.

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
10. Hapus sesuai posisi indeks
11. Tampil data pertama
12. Tampil data terakhir
13. Tampil data pada indeks tertentu
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
NIM: 124, Nama: Budi, Kelas 1E, IPK: 3.7
NIM: 125, Nama: Erick, Kelas 1A, IPK: 3.6
```







5. Menambahkan method size() pada class DoubleLinkedList01 untuk menampilkan jumlah data, dan juga memodifikasi method – method sebelumnya yang berfungsi sebagai penambah dan penghapus dengan menambahkan counter size.

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

a. Method addFirst()

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

b. Method addLast() dan insertAfter()

```
public void addLast(Mahasiswa01 data) {
    Node01 newNode = new Node01(data);
    if (isEmpty()) {
        head = tail = newNode;
    } else {
        tail.next = newNode;
        newNode.prev = tail;
        tail = newNode;
    }
    size++;
}
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;
    }
    size++;
    System.out.println("Node berhasil disisipkan setelah NIM " + keyNim);
}
```

c. Method removeFirst()

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

d. Method removeLast(), add(), removeAfter().

```
public void removeLast() {
    if (isEmpty()) {
        System.out.println("List kosong, tidak bisa dihapus.");
        return;
    }
    System.out.println("Data sudah berhasil dihapus.");
    System.out.print("Data yang terhapus adalah: ");
    tail.data.tampil();
    if (head == tail) {
        head = tail = null;
    } else {
        tail = tail.prev;
        tail.next = null;
    }
    size--;
}
public void add(int index, Mahasiswa01 data) {
    if (index < 0 || index > size) {
        System.out.println("Index di luar batas.");
        return;
    }
    if (index == 0) {
        addFirst(data);
    } else if (index == size) {
        addLast(data);
    } else {
        Node01 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        Node01 newNode = new Node01(data);
        newNode.prev = current.prev;
        newNode.next = current;
        current.prev.next = newNode;
        current.prev = newNode;
    }
    size++;
}
public void removeAfter(String keyNim) {
    Node01 current = head;
    while (current != null && !current.data.nim.equals(keyNim)) {
        current = current.next;
    }
    if (current == null || current.next == null) {
        System.out.println("Node setelah " + keyNim + " tidak ditemukan.");
        return;
    }
    Node01 toRemove = current.next;
    if (toRemove == tail) {
        tail = current;
        current.next = null;
    } else {
        current.next = toRemove.next;
        toRemove.next.prev = current;
    }
    System.out.println("Node setelah " + keyNim + " berhasil dihapus.");
    size--;
}
```

e. Method remove().

```
public void remove(int index) {
    if (index < 0 || index >= size) {
        System.out.println("Index di luar batas.");
        return;
    }
    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;
        System.out.print("Data yang dihapus: ");
        current.data.tampil();
        size--;
    }
}
```

f. Hasil running kode program.

```
=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
10. Hapus sesuai posisi indeks
11. Tampil data pertama
12. Tampil data terakhir
13. Tampil data pada indeks tertentu
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 5
Isi Linked List:
NIM: 123, Nama: Adi, Kelas 1E, IPK: 3.8
NIM: 124, Nama: Budi, Kelas 1E, IPK: 3.7
NIM: 125, Nama: Erick, Kelas 1A, IPK: 3.6


=== Menu Antrian layanan Akademik ===
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Tambah setelah NIM
7. Cari berdasarkan NIM
8. Tambah sesuai posisi indeks.
9. Hapus setelah NIM.
10. Hapus sesuai posisi indeks
11. Tampil data pertama
12. Tampil data terakhir
13. Tampil data pada indeks tertentu
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 14
Jumlah data: 3
```

6. Push kode program ke github

```
PS C:\Users\luhun\Kuliah1 Semester2\Praktikum_ASD> git add .
PS C:\Users\luhun\Kuliah1 Semester2\Praktikum_ASD> git commit -m "Jobsheet 12"
[main d5d6483] Jobsheet 12
 4 files changed, 338 insertions(+)
 create mode 100644 Jobsheet12/DLLMain01.java
 create mode 100644 Jobsheet12/DoubleLinkedList01.java
 create mode 100644 Jobsheet12/Mahasiswa01.java
 create mode 100644 Jobsheet12/Node01.java
PS C:\Users\luhun\Kuliah1 Semester2\Praktikum_ASD> git push origin main
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 2.67 KiB | 2.67 MiB/s, done.
Total 7 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote: This repository moved. Please use the new location:
remote:   https://github.com/Kumaaan/Praktikum_ASD.git
To https://github.com/Kumaaan/Praktikum-ASD.git
   eb72080..d5d6483  main -> main
```