Laporan Hasil Praktikum Algoritma Struktur Data

Jobsheet 12

Febryan Akhmad Taajuddin

244107020180

Kelas 1E

Program Studi Teknik Informatika

Jurusan Teknologi Informasi

Politeknik Negeri Malang

2025

Percobaan 1

1. Class Mahasiswa10

```
public class Mahasiswa10 {
    String nim;
    String nama;
    String kelas;
    double ipk;

    public Mahasiswa10(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. Class Node10

```
public class Node10 {
    Mahasiswa10 data;
    Node10 prev;
    Node10 next;

    public Node10(Mahasiswa10 data) {
        this.data = data;
        this.prev = null;
        this.next = null;
    }
}
```

3. Class DoubleLinkedList10

```
public class DoubleLinkedList10 {
    Node10 head;
    Node10 tail;

    public DoubleLinkedList10(){
        head = null;
        tail = null;
    }
    public boolean isEmpty(){
        return head == null;
    }
    public void addFirst(Mahasiswa10 data){
        Node10 newNode = new Node10(data);
        if (isEmpty()) {
            head = tail = newNode;
        }else{
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
    }
```

```
    public void addLast(Mahasiswa10 data){
        Node10 newNode = new Node10(data);
        if (isEmpty()) {
            head = tail = newNode;
        }else{
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
    }
    public void insertAfter(String keyNim, Mahasiswa10 data){
        Node10 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;
        }
        Node10 newNode = new Node10(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(){
        Node10 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }
```

```
    public Node10 search(String nim) {
        Node10 current = head;
        while (current != null) {
            if (current.data.nim.equals(nim)) {
                return current;
            }
            current = current.next;
        }
        return null;
    }
}
```

4. Class DLLMain10

```
import java.util.Scanner;
public class DLLMain10 {

    public static Mahasiswa10 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();
        System.out.println();
        return new Mahasiswa10(nim, nama, kelas, ipk);
    }

    public static void main(String[] args) {
        DoubleLinkedList10 list = new DoubleLinkedList10();
        Scanner scan = new Scanner(System.in);
        int pilihan;
        do {
            System.out.println("Menu 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();
            switch (pilihan) {
                case 1 -> {
                    scan.nextLine();
                    Mahasiswa10 mhs = inputMahasiswa(scan);
                    list.addFirst(mhs);
                }
                case 2 -> {
                    scan.nextLine();
                    Mahasiswa10 mhs = inputMahasiswa(scan);
                    list.addLast(mhs);
                }
                //case 3 -> list.removeFirst();
                //case 4 -> list.removeLast();
                case 5 -> list.print();
```

```
                case 6 -> {
                    scan.nextLine();
                    System.out.print("Masukkan NIM yang dicari: ");
                    String nim = scan.nextLine();
                    Node10 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);
        scan.close();
    }
}
```

5. Run kode progam

```
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 percobaan 1

1. Jelaskan perbedaan antara single linked list dengan double linked lists!
2. Perhatikan class Node01, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?
3. Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan dari konstruktor tersebut?

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

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

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

5. Perhatikan pada method addFirst(). Apakah arti statement head.prev = newNode ?
6. Modifikasi code pada fungsi print() agar dapat menampilka warning/pesan bahwa linked lis masih dalam kondisi.
7. Pada insertAfter(), apa maksud dari kode berikut ? current.next.prev = newNode;
8. Modifikasi menu pilihan dan switch-case agar fungsi insertAfter() masuk ke dalam menu pilihan dan dapat berjalan dengan baik.

Jawaban percobaan 1

1. Perbedaan SLL dengan DLL adalah struktur nodenya. SLL mempunyai node data dan next, sedangkan DLL mempunyai data dan 2 pointer yaitu next dan prev
2. Atribut next untuk menunjuk node berikutnya dalam list, sedangkan prev untuk menunjuk node sebelumnya dalam list
3. Untuk membuat keadaan Linked List menjadi kosong dan bisa digunakan
4. Jika Liked List kosong, maka data baru (newNode) akan di tandai dengan head dan tail
5. Menunjuk prevnya head menjadi node baru. Hal ini dilakukan agar kedua node masih berkesinambungan dengan data yang ditambahkan di depan
6. Modifikasi kode program

```
    public void print(){
        if (!isEmpty()) {
            Node10 current = head;
            while (current != null) {
                current.data.tampil();
                current = current.next;
            }
        }else{
            System.out.println("Linked List kosong");
        }
    }
```

- Output

```
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 kosong
```

7. Menghubungkan node setelah current ke node baru yang diinput, jadi node setelah current tetap berada setelah current, tetapi sekarang prevnya dalah newNode
8. Modifikasi kode program

```
            System.out.println("6. Tambah setelah NIM");

                case 6 -> {
                    System.out.print("Masukkan NIM: ");
                    scan.nextLine();
                    String cari = scan.nextLine();
                    Mahasiswa10 mhs = inputMahasiswa(scan);
                    list.insertAfter(cari, mhs);
                }
```

Percobaan 2

1. Method removeFirst()

```
    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. Method removeLast()

```
    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;
        }
    }
```

3. Output

```
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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 2
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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 5
Linked List kosong
```

Pertanyaan

1. Apakah maksud statement berikut pada method removeFirst()?
   head = head.next;
   head.prev = null;
2. Modifikasi kode program untuk menampilkan pesan “Data sudah berhasil dihapus. Data yang terhapus adalah … “

Jawaban

1. Head = head.next berfungsi untuk memindahkan head ke node selanjutnya , head.prev = null berfungsi untuk mengubah prev dari head yang baru menjadi null, sehingga head lama tidak terhubung atau dinyatakan dihapus

2. Modifikasi kode program

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

- Output

```
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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 1
Masukkan NIM: 1
Masukkan Nama: roni
Masukkan Kelas: 1e
Masukkan IPK: 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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 2
Masukkan NIM: 2
Masukkan Nama: drajat
Masukkan Kelas: 2e
Masukkan IPK: 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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 1
Masukkan NIM: 3
Masukkan Nama: sakroni
Masukkan Kelas: 3e
Masukkan IPK: 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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 3
Data berhasil dihapus, data yang dihapus adalah:
NIM: 3, Nama: sakroni, Kelas: 3e, 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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
```

```
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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 4
Data berhasil dihapus, data yang dihapus adalah:
NIM: 2, Nama: drajat, Kelas: 2e, 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. Tambah setelah NIM
7. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 5
NIM: 1, Nama: roni, Kelas: 1e, IPK: 4.0
```

Latihan Praktikum

1. Berikut modifikasi kode programnya

a. Class DoubleLinkedList10

```
    public void add(Mahasiswa10 data, int index) {
    if (index < 0 || index > size) {
        System.out.println("Indeks di luar batas!");
        return;
    }
    if (index == 0) {
        addFirst(data);
    }else if (index == size) {
        addLast(data);
    }else{
        Node10 newNode = new Node10(data);
        Node10 current = head;
        for (int i = 0; i < index - 1; i++) {
            current = current.next;
        }
        newNode.next = current.next;
        newNode.prev = current;
        current.next.prev = newNode;
        current.next = newNode;
        size++;
    }
}

    public void removeAfter(String keyNim) {
        Node10 current = head;
        while (current != null && !current.data.nim.equals(keyNim)) {
            current = current.next;
        }
        if (current == null || current.next == null) {
            System.out.println("Node setelah NIM " + keyNim + " tidak
ditemukan.");
            return;
        }
        Node10 toRemove = current.next;
        System.out.println("Data berhasil dihapus, data yang dihapus
adalah: ");
        toRemove.data.tampil();
        if (toRemove == tail) {
            tail = current;
            current.next = null;
        }else{
            current.next = toRemove.next;
            toRemove.next.prev = current;
        }
        size--;
    }
```

```
    public void remove(int index) {
        if (index < 0 || index >= size) {
            System.out.println("Indeks di luar batas!");
            return;
        }
        if (index == 0) {
            removeFirst();
        }else if (index == size - 1) {
            removeLast();
        }else{
            Node10 current = head;
            for (int i = 0; i < index; i++) {
                current = current.next;
            }
            System.out.println("Data berhasil dihapus, data yang dihapus
adalah: ");
            current.data.tampil();
            current.prev.next = current.next;
            current.next.prev = current.prev;
            size--;
        }
    }

    public void getFirst() {
        if (isEmpty()) {
            System.out.println("List kosong.");
        }else{
            System.out.println("Data pada head:");
            head.data.tampil();
        }
    }
    public void getLast() {
        if (isEmpty()) {
            System.out.println("List kosong.");
        }else{
            System.out.println("Data pada tail:");
            tail.data.tampil();
        }
    }
    public void getIndex(int index) {
        if (index < 0 || index >= size) {
            System.out.println("Indeks di luar batas!");
            return;
        }
        Node10 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        System.out.println("Data pada indeks ke-" + index + ":");
        current.data.tampil();
    }

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

b. Class DLLMain10

```
            System.out.println("7. Tambah data pada indeks tertentu");
            System.out.println("8. Hapus data setelah NIM tertentu");
            System.out.println("9. Hapus data pada indeks tertentu");
            System.out.println("10. Tampilkan data head");
            System.out.println("11. Tampilkan data tail");
            System.out.println("12. Tampilkan data pada indeks tertentu");
            System.out.println("13. Tampilkan jumlah data");

                case 7 -> {
                    System.out.print("Masukkan indeks: ");
                    int idx = scan.nextInt(); scan.nextLine();
                    Mahasiswa10 mhs = inputMahasiswa(scan);
                    list.add(mhs, idx);
                }
                case 8 -> {
                    System.out.print("Masukkan NIM: ");
                    String nim = scan.nextLine();
                    list.removeAfter(nim);
                }
                case 9 -> {
                    System.out.print("Masukkan indeks: ");
                    int idx = scan.nextInt(); scan.nextLine();
                    list.remove(idx);
                }
                case 10 -> list.getFirst();
                case 11 -> list.getLast();
                case 12 -> {
                    System.out.print("Masukkan indeks: ");
                    int idx = scan.nextInt(); scan.nextLine();
                    list.getIndex(idx);
                }
                case 13 -> System.out.println("Jumlah data: " + list.size());
```

- Output

```
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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 1
Masukkan NIM: 1
Masukkan Nama: Roni
Masukkan Kelas: 1E
Masukkan IPK: 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 2
Masukkan NIM: 2
Masukkan Nama: Agus
Masukkan Kelas: 1E
Masukkan IPK: 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 5
NIM: 1, Nama: Roni, Kelas: 1E, IPK: 4.0
NIM: 2, Nama: Agus, Kelas: 1E, 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 6
Masukkan NIM: 1
Masukkan NIM: 3
Masukkan Nama: Dirman
Masukkan Kelas: 1E
Masukkan IPK: 4

Node berhasil disisipkan setelah NIM 1
```

```
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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 7
Masukkan indeks: 1
Masukkan NIM: 4
Masukkan Nama: Munas
Masukkan Kelas: 1E
Masukkan IPK: 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 5
NIM: 1, Nama: Roni, Kelas: 1E, IPK: 4.0
NIM: 4, Nama: Munas, Kelas: 1E, IPK: 4.0
NIM: 3, Nama: Dirman, Kelas: 1E, IPK: 4.0
NIM: 2, Nama: Agus, Kelas: 1E, 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 8
Masukkan NIM: Node setelah NIM  tidak ditemukan.
```

```
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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 9
Masukkan indeks: 2
Data berhasil dihapus, data yang dihapus adalah:
NIM: 2, Nama: Agus, Kelas: 1E, 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 10
Data pada head:
NIM: 1, Nama: Roni, Kelas: 1E, 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 11
Data pada tail:
NIM: 3, Nama: Dirman, Kelas: 1E, 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 12
Masukkan indeks: 1
Data pada indeks ke-1:
NIM: 4, Nama: Munas, Kelas: 1E, 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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 13
Jumlah data: 2
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. Tambah setelah NIM
7. Tambah data pada indeks tertentu
8. Hapus data setelah NIM tertentu
9. Hapus data pada indeks tertentu
10. Tampilkan data head
11. Tampilkan data tail
12. Tampilkan data pada indeks tertentu
13. Tampilkan jumlah data
0. Keluar
Pilih menu: 0
Keluar dari program.
```