LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN STRUKTUR DATA

JOBSHEET 12

ATHAULLA HAFIZH

244107020030

TI 1 E

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

2. Praktikum

Percobaan 1

Kode program class Mahasiswa05

```
package Jobsheet12;


public class Mahasiswa05 {
    public String nim;
    public String nama;
    public String kelas;
    public double ipk;


    public Mahasiswa05(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);
    }
}
```

Kode program class Node05

```
package Jobsheet12;


public class Node05 {
    Mahasiswa05 data;
    Node05 prev;
    Node05 next;
```

```
    public Node05(Mahasiswa05 data) {
        this.data = data;
        this.prev = null;
        this.next = null;
    }
}
```

Kode program class DoubleLinkedLists05

```
package Jobsheet12;
public class DoubleLinkedLists05 {
    Node05 head;
    Node05 tail;

    public DoubleLinkedLists05() {
        head = null;
        tail = null;
    }

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

    public void addFirst(Mahasiswa05 data) {
        Node05 newNode = new Node05(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
    }
```

```
    public void addLast(Mahasiswa05 data) {
        Node05 newNode = new Node05(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
    }

    public void insertAfter(String keyNim, Mahasiswa05 data) {
        Node05 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;
        }

        Node05 newNode = new Node05(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() {
        Node05 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }


    public void removeFirst() {
        if (isEmpty()) {
            System.out.println("Linked list masih kosong, tidak
dapat dihapus!");
        }
        if (head == tail) {
            head = tail = null;
        } else {
            head = head.next;
            head.prev = null;
        }
    }


    public void removeLast() {
        if (isEmpty()) {
            System.out.println("Linked list masih kosong, tidak
dapat dihapus!");
            return;
        }
        if (head == tail) {
            head = tail = null;
        } else {
            tail = tail.prev;
```

```
            tail.next = null;
        }
    }

    public Node05 search(String cariNim) {
        Node05 current = head;
        while (current != null) {
            if (current.data.nim.equals(cariNim)) {
                return current;
            }
            current = current.next;
        }
        return null;
    }
}
```

Kode program class DLLMain

```
package Jobsheet12;
import java.util.Scanner;
public class DLLMain {
    static Scanner sc = new Scanner(System.in);
    public static Mahasiswa05 inputMahasiswa() {
        System.out.print("Masukkan NIM: ");
        String nim = sc.nextLine();
        System.out.print("Masukkan Nama: ");
        String nama = sc.nextLine();
        System.out.print("Masukkan Kelas: ");
        String kelas = sc.nextLine();
        System.out.print("Masukkan IPK: ");
        double ipk = sc.nextDouble();
        sc.nextLine();
        Mahasiswa05 mhs = new Mahasiswa05(nim, nama, kelas, ipk);
        return mhs;
    }
```

```
    public static void main(String[] args) {
        DoubleLinkedLists05 list = new DoubleLinkedLists05();
        int pilihan;

            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 = sc.nextInt();
                sc.nextLine();

                switch (pilihan) {
                    case 1:
                        Mahasiswa05 mhs1 = inputMahasiswa();
                        list.addFirst(mhs1);
                        break;
                    case 2:
                        Mahasiswa05 mhs2 = inputMahasiswa();
                        list.addLast(mhs2);
                        break;
                    case 3:
                        list.removeFirst();
                        break;
                    case 4:
                        list.removeLast();
                        break;
                    case 5:
                        list.print();
```

```
                        break;
                    case 6:
                        System.out.print("Masukkan NIM yang ingin
dicari: ");
                        String nimCari = sc.nextLine();
                        Node05 found = list.search(nimCari);
                        if (found != null) {
                            System.out.println("Data Mahasiswa
ditemukan:");
                            found.data.tampil();
                        } else {
                            System.out.println("Data Mahasiswa
dengan NIM " + nimCari + " tidak ditemukan.");
                        }
                        break;
                    case 0:
                        System.out.println("Keluar dari program.");
                        break;
                    default:
                        System.out.println("Pilihan tidak valid!");
                        break;
                }
            } while (pilihan != 0);

        sc.close();
    }
}
```

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: 1
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
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. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 5
NIM: 20304050, Nama: Hermione, Kelas: Gryffindor, IPK: 4.0
```

Pertanyaan

1. Jelaskan perbedaan antara single linked list dengan double linked lists!
   Single Linked List:
   - Setiap node hanya memiliki satu pointer yang menunjuk ke node berikutnya (next)
   - Traversal hanya dapat dilakukan dalam satu arah (maju / ke kanan)
   - Untuk mencapai node sebelumnya, harus dimulai dari head
   - Menggunakan memori lebih sedikit karena hanya satu pointer per node

   Double Linked List:
   - Setiap node memiliki dua pointer: next (menunjuk ke node berikutnya) dan prev (menunjuk ke node sebelumnya)
   - Traversal dapat dilakukan dalam dua arah (maju dan mundur)
   - Dapat mengakses node sebelumnya secara langsung
   - Menggunakan memori lebih banyak karena dua pointer per node

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

   Atribut next:
- Berfungsi untuk menyimpan referensi/alamat ke node berikutnya dalam linked list
- Memungkinkan traversal dari node saat ini ke node yang ada setelahnya

- Jika node adalah node terakhir, maka next akan bernilai null

Atribut prev:
- Berfungsi untuk menyimpan referensi/alamat ke node sebelumnya dalam linked list
- Memungkinkan traversal mundur dari node saat ini ke node yang ada sebelumnya
- Jika node adalah node pertama, maka prev akan bernilai null

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

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

Konstruktor DoubleLinkedLists() berfungsi untuk:
- Menginisialisasi double linked list dalam keadaan kosong
- Mengatur head dan tail menjadi null, menandakan bahwa belum ada node dalam list

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

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

if (isEmpty()): Mengecek apakah list masih kosong
Jika kosong: head dan tail diatur ke newNode karena node baru adalah satu-satunya node

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

Statement head.prev = newNode berarti:
- Mengatur pointer prev dari node head (yang lama) untuk menunjuk ke node baru
- Menyambungkan koneksi dua arah antara node baru dengan head lama
- Memastikan bahwa head lama dapat mengakses node baru sebagai node sebelumnya

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

```
public void print() {
    if (isEmpty()) {
        System.out.println(x:"Linked list masih kosong!");
        return;
    }
```

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

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

current.next: mengakses node yang ada setelah current
.prev = newNode: mengatur pointer prev dari node setelah current untuk menunjuk ke newNode
Fungsinya adalah menghubungkan node lama yang ditunjuk untuk disisipkan dengan node baru

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

Kode program DLLMain

```
System.out.println(x:"7. Masukkan data setelah NIM tertentu");
case 7:
    System.out.print(s:"Masukkan NIM yang ingin dicari: ");
    String keyNim = sc.nextLine();
    Node05 foundNode = list.search(keyNim);
    if (foundNode != null) {
        System.out.println(x:"Masukkan data Mahasiswa baru:");
        Mahasiswa05 newMhs = inputMahasiswa();
        list.insertAfter(keyNim, newMhs);
    } else {
        System.out.println("NIM " + keyNim + " tidak ditemukan.");
    }
    break;      You, 24 seconds ago • Uncommitted changes
```

Percobaan 2

Buatlah method removeFirst() dan removeLast() di dalam class DoubleLinkedLists.

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

Output

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 2
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 3
```

Pertanyan

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

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

   - head = head.next: Memindahkan pointer head ke node berikutnya, sehingga node pertama lama tidak lagi menjadi head
   - head.prev = null: Mengatur pointer prev dari head yang baru menjadi null, karena head tidak memiliki node sebelumnya

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

```java
public void removeFirst() {
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
    }
    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;
    }
}       You, 29 minutes ago • percobaan1

Windsurf: Refactor | Explain | Generate Javadoc | ×
public void removeLast() {
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }
    if (head == tail) {
        System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah " + tail.data.nama);
        head = tail = null;
    } else {
        System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah " + tail.data.nama);
        tail = tail.prev;
        tail.next = null;
```

Output

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 1
Masukkan NIM: 13141597
Masukkan Nama: Hafizh
Masukkan Kelas: 1E
Masukkan IPK: 3

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 2
Masukkan NIM: 12308194797
Masukkan Nama: Athaulla Hafizh
Masukkan Kelas: 1E
Masukkan IPK: 4

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 3
Data sudah berhasil dihapus. Data yang terhapus adalah Hafizh
```

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 4
Data sudah berhasil dihapus. Data yang terhapus adalah Athaulla Hafizh
```

Tugas

Menambahkan fungsi add(), removeAfter(), remove(), getFirst(), getLast(), getIndex(), getSize()

```
    public void add(Mahasiswa05 data, int index) {
        if (index == 0) {
            addFirst(data);
            return;
        }

        Node05 temp = head;
        for (int i = 0; i < index - 1; i++) {
            if (temp == null) {
                System.out.println("Index melebihi ukuran list.");
                return;
            }
            temp = temp.next;
        }
        if (temp.next == null) {
            addLast(data);
        } else {
            Node05 newNode = new Node05(data);
            temp.next.prev = newNode;
            newNode.next = temp.next;
            temp.next = newNode;
            newNode.prev = temp;
        }
    }
```

```
    public void removeAfter(String key) {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
            return;
        }

        Node05 temp = head;

        while (temp != null &&
!temp.data.nama.equalsIgnoreCase(key)) {
            temp = temp.next;
        }

        if (temp == null || temp.next == null) {
            System.out.println("Tidak ada node setelah " + key + "
atau node tersebut adalah yang terakhir.");
            return;
        }

        temp.next.prev = temp;
        temp.next = temp.next.next;
    }

    public void remove(int index) {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
            return;
        }

        if (index < 0) {
            System.out.println("Index tidak boleh negatif.");
            return;
        }

        if (index == 0) {
            removeFirst();
```

```
            return;
        }

        Node05 temp = head;
        for (int i = 0; i < index - 1; i++) {
            if (temp == null) {
                System.out.println("Index melebihi ukuran list.");
                return;
            }
            temp = temp.next;
        }

        if (temp.next == tail) {
            removeLast();
        }
        temp.next.prev = temp.prev;
        temp.prev.next = temp.next;
    }

    Mahasiswa05 getFirst() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak ada data yang
bisa diambil.");
            return null;
        }
        return head.data;
    }

    Mahasiswa05 getLast() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak ada data yang
bisa diambil.");
            return null;
        }
        return tail.data;
    }
```

```
    Mahasiswa05 getIndex(int index) {
        if (isEmpty()) {
            System.out.println("List kosong, tidak ada data yang
bisa diambil.");
            return null;
        }

        if (index < 0) {
            System.out.println("Index tidak boleh negatif.");
            return null;
        }

        Node05 temp = head;
        for (int i = 0; i < index; i++) {
            if (temp == null) {
                System.out.println("Index melebihi ukuran list.");
                return null;
            }
            temp = temp.next;
        }
        return temp.data;
    }

    int getSize() {
        int counter = 0;

        Node05 temp = head;
        while (temp != null) {
            temp = temp.next;
            counter++;
        }
        return counter;
    }
}
```

Output

Fungsi add()

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 8
Masukkan NIM: 1390154
Masukkan Nama: Athaulla Hafizh
Masukkan Kelas: 1E
Masukkan IPK: 4
Masukkan indeks untuk menambahkan data: 2

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 5
NIM: 0139013, Nama: Jon, Kelas: 1E, IPK: 3.0
NIM: 13131, Nama: Aldi, Kelas: 1E, IPK: 2.0
NIM: 1390154, Nama: Athaulla Hafizh, Kelas: 1E, IPK: 4.0
```

Fungsi removeAfter()

```
NIM: 0813013, Nama: Jon, Kelas: 1E, IPK: 3.0
NIM: 123131, Nama: Aldi, Kelas: 1E, IPK: 2.0
NIM: 1839175, Nama: Athaulla Hafizh, Kelas: 1E, IPK: 4.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 9
Masukkan nama mahasiswa yang ingin dihapus setelahnya: Jon

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 5
NIM: 0813013, Nama: Jon, Kelas: 1E, IPK: 3.0
NIM: 1839175, Nama: Athaulla Hafizh, Kelas: 1E, IPK: 4.0
```

Fungsi remove()

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 10
Masukkan indeks yang ingin dihapus: 0
Data sudah berhasil dihapus. Data yang terhapus adalah Jon

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 5
NIM: 1839175, Nama: Athaulla Hafizh, Kelas: 1E, IPK: 4.0
```

Fungsi getSize()

```
Pilih menu: 5
NIM: 3123919803, Nama: Hafizh, Kelas: 1E, IPK: 3.0
NIM: 1839175, Nama: Athaulla Hafizh, Kelas: 1E, IPK: 4.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 14
Jumlah Mahasiswa : 2
```

Fungsi getFirst(), getLast(), getIndex()

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 11
Data Mahasiswa pertama:
NIM: 3123919803, Nama: Hafizh, Kelas: 1E, IPK: 3.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 12
Data Mahasiswa terakhir:
NIM: 1839175, Nama: Athaulla Hafizh, Kelas: 1E, IPK: 4.0
```

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah Mahasiswa tertentu
8. Tambah data pada indeks tertentu
9. Hapus data setelah Mahasiswa tertentu
10. Hapus data pada indeks tertentu
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data berdasarkan indeks
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 13
Masukkan indeks yang ingin ditampilkan: 0
Data Mahasiswa pada indeks 0:
NIM: 3123919803, Nama: Hafizh, Kelas: 1E, IPK: 3.0
```