Laporan Hasil Praktikum Algoritma Dan Struktur Data

Jobsheet 12

Nama : Arifah Zhafirah Wikananda

NIM : 244107020188

Kelas : 1E

Program Studi Teknologi

Informasi Jurusan Teknik

Informatika Politeknik Negeri

Malang

2025

12.2 Kegiatan Praktikum 1

12.2.1 Percobaan 1

```
package jobsheet12;

public class Mahasiswa04 {
    String nim;
    String nama;
    String kelas;
    double ipk;

    Mahasiswa04(String nim, String nama, String kelas, double ipk) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.ipk = ipk;
    }

    void tampil() {
        System.out.println("Nim: " + nim + ", Nama: " + nama + ", Kelas: " + kelas + ",
IPK: " + ipk); ;
    }
}
```

```
package jobsheet12;

public class Node04 {
    Mahasiswa04 data;
    Node04 prev;
    Node04 next;

    Node04(Mahasiswa04 data) {
        this.data = data;
        this.prev = null;
        this.next = null;
    }
}
```

```
package jobsheet12;

public class DoubleLinkedLists04 {
  Node04 head;
  Node04 tail;

    public DoubleLinkedLists04() {
    head = null;
    tail = null;
  }

    public boolean isEmpty() {
        return head == null;
  }
    public void addFirst(Mahasiswa04 data) {
        Node04 newNode = new Node04(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
    }

    public void addLast(Mahasiswa04 data) {
        Node04 newNode = new Node04(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
    }

    public void insertAfter(String keyNim, Mahasiswa04 data) {
        Node04 current = head;

        //Cari node dengan nim = keyNim
        while(current != null && !current.data.nim.equals(keyNim)) {
            current = current.next;
        }

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

        Node04 newNode04 = new Node04(data);
```

```
        //Jika current adalah tail, cukup tambahkan di akhir
        if(current == tail) {
            current.next = newNode04;
            newNode04.prev = current;
            tail = newNode04;
        }else {
            //Sisispkan di tengah
            newNode04.next = current.next;
            newNode04.prev = current;
            current.next.prev = newNode04;
            current.next = newNode04;
        }

        System.out.println("Node berhasil disisipkan setelah NIM " + keyNim);
    }

    public void print() {
        Node04 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }
```

```
package jobsheet12;

import java.util.Scanner;
public class DLLMain04 {
    public static void main(String[] args) {
        DoubleLinkedLists04 list = new DoubleLinkedLists04();
        Scanner scan = new Scanner(System.in);
        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 dari awal");
            System.out.println("4. Hapus dari 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();

            switch (pilihan) {
                case 1:
                Mahasiswa04 mhs = inputMahasiswa(scan);
                list.addFirst(mhs);
                break;

                case 2:
                Mahasiswa04 mhs2 = inputMahasiswa(scan);
                list.addLast(mhs2);
                break;

                // case 3:
                // list.removeFirst();
                // break;
```

```
                // case 4 :
                // list.removeLast();
                // break;

                case 5:
                list.print();
                break;

                // case 6:
                // System.out.println("Masukkan NIM yang dicari: ");
                // String nim = scan.nextLine();
                // Node04 found = list.search(nim);
                // if(found != null) {
                //     System.out.println("Data Mahasiswa ditemukan: ");
                //     found.data.tampil();
                // } else {
                //     System.out.println("Data Mahasiswa tidak ditemukan.");
                // }
                // break;

                case 0:
                System.out.println("Keluar dari program.");
                default :
                System.out.println("Pilihan tidak valid!");
            }
        } while (pilihan != 0);
        scan.close();
    }

    public static Mahasiswa04 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(); // consume newline
        return new Mahasiswa04(nim, nama, nama, ipk);
    }
}
```

Outputnya:

```
c:\Praktikum ASD>
c:\Praktikum ASD> c: && cd "c:\Praktikum ASD" && cmd /C ""C
onMessages -cp "C:\Users\WINDOWS 11\AppData\Roaming\Code\Us
\Praktikum ASD_c51dcdc\bin" jobsheet12.DLLMain04 "

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
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 dari awal
4. Hapus dari akhir
5. tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
pilih menu: 5
Nim: 20304050, Nama: Hermione, Kelas: Hermione, IPK: 4.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
```

Pertanyaan:

1. Jelaskan perbedaan antara single linked list dengan double linked lists!
Jawab: Double linked list memiliki 2 pointer yaitu prev dan next, sedangkan single linked list hanya memiliki 1 pointer yaitu next.

2. Perhatikan class Node01, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?
Jawab: Atribut next untuk pointer yang akan menunjuk ke node berikutnya dari node saat ini, sedangkan atribut prev untuk pointer ke node sebelumnya dari node saat ini.

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

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

Jawab: Konstruktor tersebut digunakan untuk menginisialisasi linked list dalam keadaan kosong, yaitu dengan mengatur head dan tail bernilai null. Ini menandakan bahwa belum ada node yang ditambahkan ke dalam list saat pertama kali dibuat.

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

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

Jawab: Kode tersebut memeriksa apakah linked list kosong dengan isEmpty(). Jika benar (list kosong), maka node baru (newNode) menjadi node pertama dan sekaligus node terakhir dalam list, sehingga head dan tail menunjuk ke node yang sama.

5. Perhatikan pada method addFirst(). Apakah arti statement head.prev = newNode ?
Jawab: Pernyataan head.prev = newNode digunakan untuk meyambungkan node baru ke node pertama yang sebelumnya ada. Setelah newNode ditambahkan di depan, node yang sebelumnya menjadi head sekarang memiliki node sebelumnya (prev) yaitu newNode. Ini menjaga struktur dua arah dari double linked list.

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("List kosong.");
            return;
        }else {
            Node04 current = head;
            System.out.println("Isi Double Linked List:");
            while (current != null) {
                System.out.println(current.data + " ");
                current = current.next;
            }
            System.out.println();
        }
    }
```

12.3 Kegiatan Praktikum 2

12.3.1 Tahapan Percobaan

```
 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;
        }
    }

    public Node04 search(String nim){
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dicari.");
            return null;
        }
        Node04 current = head;
        while (current != null) {
            if (current.data.nim.equals(nim)) {
                return current;
            }
            current = current.next;
        }
        return null; // Jika tidak ditemukan
    }
}
```

Outputnya:

```
C:\Praktikum ASD> cmd /C ""C:\Program Fil
S 11\AppData\Roaming\Code\User\workspaceS
sheet12.DLLMain04 "

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
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 dari awal
4. Hapus dari akhir
5. tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
pilih menu: 3

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
```

Pertanyaan:

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

head = head.next;

head.prev = null;

Jawab: Data yang sebelumnya menjadi head diubah menjadi data yang ada di setelahnya, lalu data yang sebelumnya head diubah menjadi null sehingga data yang sebelumnya menjadi head akan terhapus.

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.");
        }else {
            Mahasiswa04 dataDihapus = head.data;
            if(head == tail) {
                head = tail = null; // Jika hanya ada satu elemen
            } else {
                head = head.next; // Pindahkan head ke node berikutnya
                head.prev = null; // Set prev dari head baru ke null
            }
            System.out.println("Data sudah berhasil dihapus. Data yang terhapus
adalah: " );
            dataDihapus.tampil(); // Tampilkan data yang dihapus
        }
    }
```

12.5 Tugas Praktikum

1. Tambahkan fungsi add() pada kelas DoubleLinkedList untuk menambahkan node pada indeks tertentu
2. Tambahkan removeAfter() pada kelas DoubleLinkedList untuk menghapus node setelah data key.
3. Tambahkan fungsi remove() pada kelas DoubleLinkedList untuk menghapus node pada indeks tertentu.
4. Tambahkan fungsi getFirst(), getLast() dan getIndex() untuk menampilkan data pada node head, node tail dan node pada indeks tertentu.
5. tambahkan kode program dan fungsi agar dapat membaca size/ jumlah data pada Double Linked List

```
package jobsheet12;

public class DoubleLinkedLists04 {
  Node04 head;
  Node04 tail;
  int size;

    public DoubleLinkedLists04() {
    head = null;
    tail = null;
    size = 0;
  }

    public boolean isEmpty() {
        return head == null;
  }
    public void addFirst(Mahasiswa04 data) {
        Node04 newNode = new Node04(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
        size++;
    }

    public void addLast(Mahasiswa04 data) {
        Node04 newNode = new Node04(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
        size++;
    }
```

```
public void add(int index, Mahasiswa04 data) {
    if (index < 0 || index > size) {
        System.out.println("Indeks salah, tidak bisa menambahkan data.");
        return;
    }
    if (index == 0) {
        addFirst(data);
        return;
    }else if (index == size) {
        addLast(data);
        return;
    }else {
        Node04 newNode = new Node04(data);
        Node04 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) {
    Node04 current = search(keyNim);
    if (current == null || current.next == null) {
        System.out.println("Tidak ada node setelah NIM " + keyNim);
        return;
    }
    Node04 toRemove = current.next;
    current.next = toRemove.next;
    if (toRemove.next != null) {
        toRemove.next.prev = current;
    } else {
        tail = current;
    }
    size--;
    System.out.println("Data setelah NIM " + keyNim + " berhasil dihapus:");
    toRemove.data.tampil();
}

public void remove(int index) {
    if (index < 0 || index >= size) {
        System.out.println("Indeks tidak valid.");
        return;
    }
    if (index == 0) {
        removeFirst();
    } else if (index == size - 1) {
        removeLast();
    } else {
        Node04 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        current.prev.next = current.next;
        current.next.prev = current.prev;
        System.out.println("Data berhasil dihapus pada indeks " + index + ":");
        current.data.tampil();
        size--;
    }
}
```

```
public Mahasiswa04 getFirst() {
    if (isEmpty()) return null;
    return head.data;
}

public Mahasiswa04 getLast() {
    if (isEmpty()) return null;
    return tail.data;
}

public Mahasiswa04 getIndex(int index) {
    if (index < 0 || index >= size) return null;
    Node04 current = head;
    for (int i = 0; i < index; i++) {
        current = current.next;
    }
    return current.data;
}

public void insertAfter(String keyNim, Mahasiswa04 data) {
    Node04 current = head;

    //Cari node dengan nim = keyNim
    while(current != null && !current.data.nim.equals(keyNim)) {
        current = current.next;
    }
    if(current == null) {
        System.out.println("Node dengan NIM " + keyNim + " tidak ditemukan.");
        return;
    }

    Node04 newNode04 = new Node04(data);
    //Jika current adalah tail, cukup tambahkan di akhir
    if(current == tail) {
        current.next = newNode04;
        newNode04.prev = current;
        tail = newNode04;
    }else {
        //Sisispkan di tengah
        newNode04.next = current.next;
        newNode04.prev = current;
        current.next.prev = newNode04;
        current.next = newNode04;
    }
    size++;
    System.out.println("Node berhasil disisipkan setelah NIM " + keyNim);
}


public void print() {
    if (isEmpty()) {
        System.out.println("List kosong.");
        return;
    }else {
        Node04 current = head;
        System.out.println("Isi Double Linked List:");
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
        System.out.println("Total data: " + size);
    }
}
```

```
    public void removeFirst() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
            return;
        }else {
            Mahasiswa04 dataDihapus = head.data;
            if(head == tail) {
                head = tail = null; // Jika hanya ada satu elemen
            } else {
                head = head.next; // Pindahkan head ke node berikutnya
                head.prev = null; // Set prev dari head baru ke null
            }
            System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah: "
);
            dataDihapus.tampil(); // Tampilkan data yang dihapus
        }
    }

    public void removeLast() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
            return;
        }
        Mahasiswa04 dataDihapus = tail.data;
        if (head == tail) {
            head = tail = null;
        } else {
            tail = tail.prev;
            tail.next = null;
        }
        size--;
        System.out.println("Data sudah berhasil dihapus. Data yang terhapus adalah: ");
        dataDihapus.tampil(); // Tampilkan data yang dihapus
    }

    public Node04 search(String nim){
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dicari.");
            return null;
        }
        Node04 current = head;
        while (current != null) {
            if (current.data.nim.equals(nim)) {
                return current;
            }
            current = current.next;
        }
        return null; // Jika tidak ditemukan
    }
    public int getSize() {
        return size;
    }
}
```

Outputnya:

```
C:\Praktikum ASD> cmd /C ""C:\Program
ser\workspaceStorage\9ab0f952d31733a7c

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
0. Keluar
pilih menu: 1
Masukkan NIM: 123
Masukkan Nama: Arifah
Masukkan Kelas: 1E
Masukkan IPK: 3.9

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
0. Keluar
pilih menu: 2
Masukkan NIM: 456
Masukkan Nama: Angel
Masukkan Kelas: 1E
Masukkan IPK: 3.9
```

```
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
0. Keluar
pilih menu: 5
Isi Double Linked List:
Nim: 123, Nama: Arifah, Kelas: Arifah, IPK: 3.9
Nim: 456, Nama: Angel, Kelas: Angel, IPK: 3.9
Total 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
0. Keluar
pilih menu: 6
Masukkan NIM yang dicari:
123
Data Mahasiswa ditemukan:
Nim: 123, Nama: Arifah, Kelas: Arifah, IPK: 3.9
```

```
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
0. Keluar
pilih menu: 4
Data sudah berhasil dihapus. Data yang terhapus adalah:
Nim: 456, Nama: Angel, Kelas: Angel, IPK: 3.9

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
0. Keluar
pilih menu: 5
List kosong.
```

```
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
0. Keluar
pilih menu: 3
Data sudah berhasil dihapus. Data yang terhapus adalah:
Nim: 123, Nama: Arifah, Kelas: Arifah, IPK: 3.9

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
0. Keluar
pilih menu: 5
Isi Double Linked List:
Nim: 456, Nama: Angel, Kelas: Angel, IPK: 3.9
Total 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
0. Keluar
pilih menu: 0
Keluar dari program.
Pilihan tidak valid!
```