# LAPORAN HASIL PRAKTIKUM
# Algoritma Struktur Data
# JOBSHEET 12


Muhammad Fitra Adhim Nurrochman
2441007020089
TI 1E

**PROGRAM STUDI D-IV TEKNIK INFORMATIKA**
**JURUSAN TEKNOLOGI INFORMASI**
**POLITEKNIK NEGERI MALANG**

**PERCOBAAN 1**

1. Kode program
    a. Mahasiswa19.java

```
public class Mahasiswa19 {
    String nim, nama, kelas;
    double ipk;

    public Mahasiswa19(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);
    }
}
```

    b. Node19.java

```
public class Node19 {
    Mahasiswa19 data;
    Node19 next;
    Node19 prev;

    public Node19(Mahasiswa19 data) {
        this.data = data;
        this.next = null;
        this.prev = null;
    }
}
```

    c. DoubleLinkedList19.java

```
public class DoubleLinkedList19 {
    Node19 head;
    Node19 tail;

    public DoubleLinkedList19() {
        head = null;
        tail = null;
    }

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

    public void addFirst(Mahasiswa19 data) {
        Node19 newNode = new Node19(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
    }

    public void addLast(Mahasiswa19 data) {
        Node19 newNode = new Node19(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
    }

    public void insertAfter(String keyNim, Mahasiswa19 data) {
        Node19 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;
        }
        Node19 newNode = new Node19(data);

        //jika current adalah tail, cukup tambahkan di akhir
        if(current == tail) {
            current.next = newNode;
            newNode.prev = current;
            tail = newNode;
        } else {
```

```
            //sisipkan di tengah
            newNode.next = current.next;
            newNode.prev = current;
            current.next.prev = newNode;
            current.next = newNode;
        }
    }

    public void print() {
        Node19 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }
    // tambahan nomer 16
    public void removeFirst() {
        if (isEmpty()) {
            System.out.println("List kosong!");
        } else if (head == tail) {
            head = tail = null;
        } else {
            head = head.next;
            head.prev = null;
        }
    }

    public void removeLast() {
        if (isEmpty()) {
            System.out.println("List kosong!");
        } else if (head == tail) {
            head = tail = null;
        } else {
            tail = tail.prev;
            tail.next = null;
        }
    }

    public Node19 search(String nim) {
        Node19 current = head;
        while (current != null) {
            if (current.data.nim.equals(nim)) {
                return current;
            }
            current = current.next;
        }
        return null;
    }
}
```

d. DLLMain.java

```
import java.util.Scanner;

public class DLLMain {
    public static void main(String[] args) {
        DoubleLinkedList19 list = new DoubleLinkedList19();
        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 di awal");
        System.out.println("4. Hapus di akhir");
        System.out.println("5. Tampilkan data");
        System.out.println("6. Cari Mahasiswa berdasarkan NIM");
        System.out.println("0. Keluar");
        System.out.print("Pilih menu: ");
        pilihan = scan.nextInt();
        scan.nextLine();

        switch (pilihan) {
            case 1 -> {
                Mahasiswa19 mhs = inputMahasiswa(scan);
                list.addFirst(mhs);
            }
            case 2 -> {
                Mahasiswa19 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 yang dicari: ");
                String nim = scan.nextLine();
                Node19 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);
}
        //penambahan nomer 16
    public static Mahasiswa19 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 Mahasiswa19(nim, nama, kelas, ipk);
    }
}
```

2. Hasil Verifikasi kode Program

```
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
```

3. Pertanyaan Percobaan
   a. Jelaskan perbedaan antara single linked list dengan double linked list!
   b. Perhatikan class node01, di dalamnyua terdapat atribut next dan prev. Untuk apakah atribut tersebut?
   c. Perhatikan konstruktor pada class DoubleLinkedList. Apa kegunaan dari konstruktor tersebut?

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

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

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

   e. Perhatikan pada method addFirst(). Apakah arti statement head.prev = newNode?
   f. Modifikasi code pada fungsi print() agar dapat menampilkan warning/ pesan bahwa linked list masih dalam kondisi
   g. Pada insertAfter(), apa maksud dari kode berikut? “current.next.prev = newNode;”
   h. Modifikasi menu pilihan dan switch-case agar fungsi insertAfter() masuk ke dalam menu pilihan dan dapat berjalan dengan baik

4. Jawaban pertanyaan
    a. Single linked list hanya memiliki next untuk menuju ke node berikutnya, dan hanya bisa satu arah(maju) sedangkan double linked list memiliki 2 arah unutk menuju ke node berikutnya dan sebelumnya(next dan prev) dan bisa dua arah(maju dan mundur), double linked list lebih fleksibel jika memiliki banyak data
    b. Next = untuk menyimpan alamat ke node berikutnya, untuk bergerak maju
    Prev = unutk menyimpan alamat ke ndoe sebelumnya, untuk bergerak mundur
    c. Unutkk menginisialisasi nilai awal dari atribut head dan tail
    d. If isempty, jika kondisi masih kosong maka head dan tail akan menjadi newNode(node baru)
    e. Untuk menambahkan node baru di depan/awal double linked list, head.prev berarti alamat sebelum head, newNode berarti node baru, node baru ditambahkan di alamat sebelum head
    f. Modifikasi agar dapat menampilkan warning

```
public void print() {
    if (isEmpty()) {
        System.out.println("Warning!: List masih kosong");
        return;
    }
    Node19 current = head;
    while (current != null) {
        current.data.tampil();
        current = current.next;
    }
```

    g. current.next.prev adalah untuk menghubungkan node setelah current ke newNode sebagai node berikutnya
    h. modifikasi

```
case 7 -> {
    System.out.print("Masukkan NIM yang akan ditambahkan setelah: ");
    String keyNim = scan.nextLine();
    System.out.println("Masukkan data mahasiswa baru:");
    Mahasiswa19 newMhs = inputMahasiswa(scan);
    list.insertAfter(keyNim, newMhs);
}
```

```
        System.out.println("6. Cari Mahasiswa berdasarkan NIM");
        System.out.println("7. Tambah data setelah data yang dimasukkan");
        System.out.println("0. Keluar");
        System.out.print("Pilih menu: ");
        pilihan = scan.nextInt();
```

**PERCOBAAN 2**

1. tambahan kode program

```
76      public void removeFirst() {
77          if (isEmpty()) {
78              System.out.println("List kosong, tidak bisa dihapus.");
79          }
80          if (head == tail) {
81              head = tail = null;
82          } else {
83              head = head.next;
84              head.prev = null;
85          }
86      }
87
88      public void removeLast() {
89          if (isEmpty()) {
90              System.out.println("List kosong, tidak bisa dihapus.");
91              return;
92          }
93          if (head == tail) {
94              head = tail = null;
95          } else {
96              tail = tail.prev;
97              tail.next = null;
98          }
99      }
```

2. hasil verifikasi kode program

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

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Tambah data setelah data yang dimasukkan
0. Keluar
Pilih menu: 5
Warning!: List masih kosong
```

3. Pertanyaan Percobaan

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 … "

4. Jawaban pertanyaan
    1. Head = head.next adalah head dipindahkan ke node berikutnya setelah head, sehingga node sebelumnya dihapus, head.prev = null artinya adalah node head yang baru tidak memiliki node sebelumnya, sehingga prev pada node tersebut tidak memilki isi/di set null
    2. Modifikasi

```
        } else {
            Node19 temp = head;
            if (head == tail) {
                head = tail = null;
            } else {
                head = head.next;
                head.prev = null;
            }
            System.out.print("Data sudah berhasil dihapus. Data yang terhapus adalah: ");
            temp.data.tampil();
        }
    }
```

```
Pilih menu: 3
Data sudah berhasil dihapus. Data yang terhapus adalah: NIM: 123, Nama: adhim, Kelas: 1e, IPK: 2.1
```

**TUGAS**

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

1. Tambahan

```
public class DoubleLinkedList19 {
    Node19 head;
    Node19 tail;
    int size = 0;

    public DoubleLinkedList19() {
        head = null;
        tail = null;
        size = 0;
    }
```

Menambahkan size untuk menghitung jumlah data

2. Tambahan

```
public void add(int index, Mahasiswa19 data) {
    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 {
        Node19 newNode = new Node19(data);
        Node19 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++;
    }
}
```

Menambahkan method add

3. Tambahan

```java
public void removeAfter(String keyNim) {
    Node19 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 atau tidak ada.");
        return;
    }
    Node19 toDelete = current.next;
    if (toDelete == tail) {
        tail = current;
        current.next = null;
    } else {
        current.next = toDelete.next;
        toDelete.next.prev = current;
    }
    System.out.print("Data setelah NIM " + keyNim + " sudah dihapus. Data yang dihapus: ");
    toDelete.data.tampil();
    size--;
}
```

Menambahkan method removeAfter()

4. Tambahan

```java
public void remove(int index) {
    if (isEmpty() || index < 0 || index >= size) {
        System.out.println("Indeks di luar batas atau list kosong!");
        return;
    }
    if (index == 0) {
        removeFirst();
    } else if (index == size - 1) {
        removeLast();
    } else {
        Node19 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 pada indeks " + index + " sudah dihapus. Data yang dihapus: ");
        current.data.tampil();
        size--;
    }
```

Menambahkan method remove()

5. Tambahan

```
192     public void getFirst() {
193         if (isEmpty()) {
194             System.out.println("List kosong.");
195         } else {
196             System.out.print("Data pada head: ");
197             head.data.tampil();
198         }
199     }
200 
201     public void getLast() {
202         if (isEmpty()) {
203             System.out.println("List kosong.");
204         } else {
205             System.out.print("Data pada tail: ");
206             tail.data.tampil();
207         }
208     }
209 
210     public void getIndex(int index) {
211         if (isEmpty() || index < 0 || index >= size) {
212             System.out.println("Indeks di luar batas atau list kosong!");
213             return;
214         }
215         Node19 current = head;
216         for (int i = 0; i < index; i++) {
217             current = current.next;
218         }
219         System.out.print("Data pada indeks " + index + ": ");
220         current.data.tampil();
221     }
```

Menambahkan method getfirst, getlast dan getindex

6. Tambahan method untuk membaca jumlah data

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