

LAPORAN HASIL PRAKTIKUM
Algoritma Struktur Data
JOBSHEET 11



Muhammad Fitra Adhim Nurrochman
2441007020089

TI 1E

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

PERCOBAAN 1

Single Linked List

1. Kode Program

a. Mahasiswa19.java

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

    Mahasiswa19() {

    }
    public Mahasiswa19(String nm, String name, String kls, double ip) {
        nim = nm;
        nama = name;
        kelas = kls;
        ipk = ip;
    }
    public void tampilInformasi() {
        System.out.println(nama + " - " + nim + " - " + kelas + " - " +
ipk);
    }
}
```

b. NodeMahasiswa19.java

```
public class NodeMahasiswa19 {
    Mahasiswa19 data;
    NodeMahasiswa19 next;

    public NodeMahasiswa19(Mahasiswa19 data, NodeMahasiswa19 next) {
        this.data = data;
        this.next = next;
    }
}
```

c. SingleLinkedList19.java

```

public class SingleLinkedList19 {
    NodeMahasiswa19 head;
    NodeMahasiswa19 tail;

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

    public void print() {
        if(!isEmpty()) {
            NodeMahasiswa19 tmp = head;
            System.out.println("Isi Linked List:\t");
            while ( tmp != null) {
                tmp.data.tampilInformasi();
                tmp = tmp.next;
            }
            System.out.println("");
        } else {
            System.out.println("Linked List Kosong");
        }
    }

    public void addFirst(Mahasiswa19 input) {
        NodeMahasiswa19 ndInput = new NodeMahasiswa19(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }

    public void addLast(Mahasiswa19 input) {
        NodeMahasiswa19 ndInput = new NodeMahasiswa19(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            tail.next = ndInput;
            tail = ndInput;
        }
    }

    public void insertAfter(String key, Mahasiswa19 input) {
        NodeMahasiswa19 ndInput = new NodeMahasiswa19(input, null);
        NodeMahasiswa19 temp = head;
        do {
            if (temp.data.nama.equalsIgnoreCase(key)) {
                ndInput.next = temp.next;
                temp.next = ndInput;
                if (ndInput.next == null) {
                    tail = ndInput;
                }
                break;
            }
            temp = temp.next;
        } while (temp != null);
    }

    public void insertAt(int index, Mahasiswa19 input) {
        if (index < 0) {
            System.out.println("indeks salah");
        } else if (index == 0) {
            addFirst(input);
        } else {
            NodeMahasiswa19 temp = head;
            for (int i = 0; i < index -1; i++) {
                temp = temp.next;
            }
            temp.next = new NodeMahasiswa19(input, temp.next);
            if (temp.next.next == null) {
                tail = temp.next;
            }
        }
    }
}

```

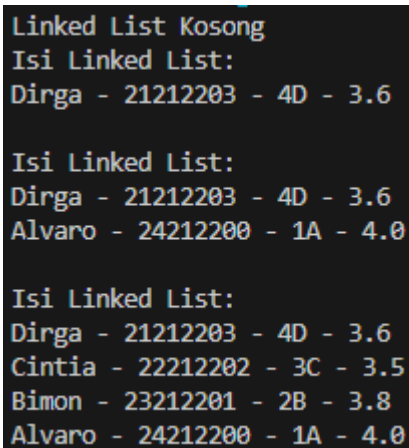
d. SLLMain19.java

```
public class SLLMain19 {
    public static void main(String[] args) {
        SingleLinkedList19 sll = new SingleLinkedList19();
        Mahasiswa19 mhs1 = new Mahasiswa19("24212200", "Alvaro", "1A",
4.0);
        Mahasiswa19 mhs2 = new Mahasiswa19("23212201", "Bimon", "2B",
3.8);
        Mahasiswa19 mhs3 = new Mahasiswa19("22212202", "Cintia", "3C",
3.5);
        Mahasiswa19 mhs4 = new Mahasiswa19("21212203", "Dirga", "4D",
3.6);

        sll.print();
        sll.addFirst(mhs4);
        sll.print();
        sll.addLast(mhs1);
        sll.print();
        sll.insertAfter("Dirga", mhs3);
        sll.insertAt(2, mhs2);
        sll.print();

    }
}
```

2. Verifikasi Hasil Percobaan



```
Linked List Kosong
Isi Linked List:
Dirga - 21212203 - 4D - 3.6

Isi Linked List:
Dirga - 21212203 - 4D - 3.6
Alvaro - 24212200 - 1A - 4.0

Isi Linked List:
Dirga - 21212203 - 4D - 3.6
Cintia - 22212202 - 3C - 3.5
Bimon - 23212201 - 2B - 3.8
Alvaro - 24212200 - 1A - 4.0
```

3. Pertanyaan

- Mengapa hasil compile kode program di baris pertama menghasilkan "Linked List Kosong?"
- Jelaskan kegunaan variable temp secara umum pada setiap method
- Lakukan modifikasi agar data dapat ditambahkan dari keyboard

4. Jawaban

- Karena saat dipanggil sll.print yang pertama, linked list masih kosong sehingga menghasilkan "Linked List Kosong", setelah di addFirst(mhs4) baru Linked List terisi
- Digunakan untuk menduplikat dan menyimpan data utama secara sementara/menjadikan penunjuk untuk data utama agar dapat menelusuri dan mengakses node pada linked list, dengan adanya temp kita dapat mencari posisi data tertentu dan dapat melakukan operasi pada data tersebut tanpa mengubah data asli, pada method print temp digunakan untuk menelusuri dan mencetak semua data node

satu persatu, pada method insertAfter dan insertAt temp digunakan untuk mencari node yang menjadi target penambahan kode baru

c. Memodifikasi SLLMain.java

```
1  import java.util.Scanner;
2
3  public class SLLMain19 {
4      public static void main(String[] args) {
5          Scanner input = new Scanner(System.in);
6          SingleLinkedList19 sll = new SingleLinkedList19();
7          System.out.print(s: "Masukkan jumlah mahasiswa: ");
8          int jumlahMhs = input.nextInt();
9          input.nextLine();
10
11         for (int i = 0; i < jumlahMhs; i++) {
12             System.out.println("Data Mahasiswa ke-" + (i+1));
13             System.out.print(s: "NIM    : ");
14             String nim = input.nextLine();
15             System.out.print(s: "Nama  : ");
16             String nama = input.nextLine();
17             System.out.print(s: "Kelas : ");
18             String kelas = input.nextLine();
19             System.out.print(s: "IPK   : ");
20             double ipk = input.nextDouble();
21             input.nextLine();
22
23             Mahasiswa19 mhs = new Mahasiswa19(nim, nama, kelas, ipk);
24             sll.addLast(mhs);
25         }
26         System.out.println(x: "\nData Mahasiswa dalam Linked List:");
27         sll.print();
28     }
29 }
```

Hasil dari modifikasi

```
Masukkan jumlah mahasiswa: 4
Data Mahasiswa ke-1
NIM    : 123
Nama   : adhim
Kelas : 1e
IPK    : 3.0
Data Mahasiswa ke-2
NIM    : 234
Nama   : perls
Kelas : 1e
IPK    : 3.5
Data Mahasiswa ke-3
NIM    : 345
Nama   : sari
Kelas : 1e
IPK    : 3.4
Data Mahasiswa ke-4
NIM    : 456
Nama   : qerta
Kelas : 1e
IPK    : 3.7

Data Mahasiswa dalam Linked List:
Isi Linked List:
adhim - 123 - 1e - 3.0
perls - 234 - 1e - 3.5
sari - 345 - 1e - 3.4
qerta - 456 - 1e - 3.7
```

PERCOBAAN 2

Modifikasi Elemen pada Single Linked List

1. Penambahan kode program

a. SLLMain.java

```
public class SLLMain19 {
    public static void main(String[] args) {
        SingleLinkedList19 sll = new SingleLinkedList19();
        Mahasiswa19 mhs1 = new Mahasiswa19("24212200", "Alvaro", "1A",
4.0);
        Mahasiswa19 mhs2 = new Mahasiswa19("23212201", "Bimon", "2B",
3.8);
        Mahasiswa19 mhs3 = new Mahasiswa19("22212202", "Cintia", "3C",
3.5);
        Mahasiswa19 mhs4 = new Mahasiswa19("21212203", "Dirga", "4D",
3.6);

        sll.print();
        sll.addFirst(mhs4);
        sll.print();
        sll.addLast(mhs1);
        sll.print();
        sll.insertAfter("Dirga", mhs3);
        sll.insertAt(2, mhs2);
        sll.print();
        System.out.println("data index 1 : ");
        sll.getData(1);

        System.out.println("data mahasiswa an Bimon berada pada index :
" + sll.indexOf("bimon"));
        System.out.println();

        sll.removeFirst();
        sll.removeLast();
        sll.print();
        sll.removeAt(0);
        sll.print();

    }
}
```

b. SingleLinkedList.java

```
public void getData(int index) {
    NodeMahasiswa19 tmp = head;
    for (int i = 0; i < index; i++) {
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}
public int indexOf(String key) {
    NodeMahasiswa19 tmp = head;
    int index = 0;
    while (tmp != null && !tmp.data.nama.equalsIgnoreCase(key)) {
        tmp = tmp.next;
        index++;
    }
    if (tmp == null) {
        return -1;
    } else {
        return index;
    }
}
public void removeFirst() {
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat
dihapus!");
    } else if (head == tail) {
        head = tail = null;
    } else {
        head = head.next;
    }
}
public void removeLast() {
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat
dihapus!");
    } else if (head == tail) {
        head = tail = null;
    } else {
        NodeMahasiswa19 tmp = head;
        while (tmp.next != tail) {
            tmp = tmp.next;
        }
        tmp.next = null;
        tail = tmp;
    }
}
```

```

public void remove(String key) {
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat
dihapus!");
    } else {
        NodeMahasiswa19 temp = head;
        while (temp != null) {
            if ((temp.data.nama.equalsIgnoreCase(key)) && temp ==
head) {
                temp.next = temp.next.next;
                if (temp.next == null) {
                    tail = temp;
                }
                break;
            }
            temp = temp.next;
        }
    }
}
public void removeAt(int index) {
    if (index == 0) {
        removeFirst();
    } else {
        NodeMahasiswa19 temp = head;
        for (int i = 0; i < index -1; i++) {
            temp = temp.next;
        }
        temp.next = temp.next.next;
        if (temp.next == null) {
            tail = temp;
        }
    }
}
}

```


2. Verifikasi hasil percobaan

```
Linked List Kosong
Isi Linked List:
Dirga - 21212203 - 4D - 3.6

Isi Linked List:
Dirga - 21212203 - 4D - 3.6
Alvaro - 24212200 - 1A - 4.0

Isi Linked List:
Dirga - 21212203 - 4D - 3.6
Cintia - 22212202 - 3C - 3.5
Bimon - 23212201 - 2B - 3.8
Alvaro - 24212200 - 1A - 4.0

data index 1 :
Cintia - 22212202 - 3C - 3.5
data mahasiswa an Bimon berada pada index : 2

Isi Linked List:
Cintia - 22212202 - 3C - 3.5
Bimon - 23212201 - 2B - 3.8

Isi Linked List:
Bimon - 23212201 - 2B - 3.8
```

3. Pertanyaan

- Mengapa digunakan keyword break pada fungsi remove? Jelaskan
- Jelaskan kegunaan kode dibawah method remove

```
1 temp.next = temp.next.next;
2 if (temp.next == null) {
3     tail = temp;
4 }
```

4. Jawaban

- Keyword break digunakan untuk menghentikan proses pencarian dan penghapusan node, setelah node ditemukan akan dihapus, jika tidak diberikan keyword break maka proses pencarian akan terus berjalan ke node berikutnya (jika dalam kasus ini ada nama yang sama atau ada data yang tidak kita inginkan untuk terhapus) , dengan adanya break maka setelah apa yang dicari ditemukan maka proses akan langsung berhenti
- Line 1 menghapus node setelah node temp dengan cara menghubungkan node temp langsung ke node setelah temp.next, sehingga node temp.next tidak lagi terhubung

pada linked list, line 2 dan 3 memeperbarui penunjuk tail, jika node yang dihapus adalah node terakhir, maka node tersebut menjadi node terakhir sehingga tail diberikan ke temp

TUGAS

1. Kode program

a. MhsQueue19.java

```
public class MhsQueue19 {
    String nim, nama, kelas;
    double ipk;

    public MhsQueue19(String nim, String nama,
String kelas, double ipk) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.ipk = ipk;
    }

    public void tampilInformasi() {
        System.out.println(nama + " - " + nim + "
- " + kelas + " - " + ipk);
    }
}
```

b. NodeQueue19.java

```
public class NodeQueue19 {
    MhsQueue19 data;
    NodeQueue19 next;

    public NodeQueue19(MhsQueue19 data,
NodeQueue19 next) {
        this.data = data;
        this.next = next;
    }
}
```

c. QueueLinkedList.java

```
public class QueueLinkedList {
    NodeQueue19 front, rear;
    int size = 0;
    int max = 10;

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

    public boolean isFull() {
        return size == max;
    }

    public void clear() {
        front = rear = null;
        size = 0;
        System.out.println("Antrian dikosongkan.");
    }
}
```

```

public void enqueue(MhsQueue19 mhs) {
    if (isFull()) {
        System.out.println("Antrian penuh!");
        return;
    }
    NodeQueue19 newNode = new NodeQueue19(mhs, null);
    if (isEmpty()) {
        front = rear = newNode;
    } else {
        rear.next = newNode;
        rear = newNode;
    }
    size++;
    System.out.println("Mahasiswa berhasil masuk antrian.");
}

public MhsQueue19 dequeue() {
    if (isEmpty()) {
        System.out.println("Antrian kosong!");
        return null;
    }
    MhsQueue19 mhs = front.data;
    front = front.next;
    if (front == null) rear = null;
    size--;
    System.out.println("Mahasiswa keluar dari antrian.");
    return mhs;
}

public void print() {
    if (isEmpty()) {
        System.out.println("Antrian kosong!");
        return;
    }
    NodeQueue19 temp = front;
    System.out.println("Daftar Mahasiswa dalam Antrian:");
    while (temp != null) {
        temp.data.tampilInformasi();
        temp = temp.next;
    }
}

public void printFront() {
    if (isEmpty()) {
        System.out.println("Antrian kosong!");
    } else {
        System.out.print("Mahasiswa terdepan: ");
        front.data.tampilInformasi();
    }
}

public void printRear() {
    if (isEmpty()) {
        System.out.println("Antrian kosong!");
    } else {
        System.out.print("Mahasiswa paling akhir: ");
        rear.data.tampilInformasi();
    }
}

public int getSize() {
    return size;
}
}

```

d. MainQueueLinkedList.java

```
import java.util.Scanner;

public class MainQueueLinkedList {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        QueueLinkedList queue = new QueueLinkedList();
        int pilih;

        do {
            System.out.println("\n=== Menu Antrian Layanan Kemahasiswaan  

            ===");

            System.out.println("1. Tambah Mahasiswa ke Antrian");
            System.out.println("2. Layani Mahasiswa");
            System.out.println("3. Lihat Semua Antrian");
            System.out.println("4. Kosongkan Antrian");
            System.out.println("5. Lihat Mahasiswa Terdepan");
            System.out.println("6. Lihat Mahasiswa Paling Akhir");
            System.out.println("7. Lihat Jumlah Mahasiswa dalam  

            Antrian");

            System.out.println("8. Keluar");
            System.out.print("Pilih menu: ");
            pilih = sc.nextInt();
            sc.nextLine();

            switch (pilih) {
                case 1:
                    System.out.print("NIM    : ");
                    String nim = sc.nextLine();
                    System.out.print("Nama  : ");
                    String nama = sc.nextLine();
                    System.out.print("Kelas : ");
                    String kelas = sc.nextLine();
                    System.out.print("IPK   : ");
                    double ipk = sc.nextDouble();
                    sc.nextLine();
                    MhsQueue19 mhs = new MhsQueue19(nim, nama, kelas,  

                    ipk);

                    queue.enqueue(mhs);
                    break;
                case 2:
                    queue.dequeue();
                    break;
                case 3:
                    queue.print();
                    break;
                case 4:
                    queue.clear();
                    break;
                case 5:
                    queue.printFront();
                    break;
                case 6:
                    queue.printRear();
                    break;
                case 7:
                    System.out.println("Jumlah mahasiswa dalam antrian:  

                    " + queue.getSize());
                    break;
                case 8:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid!");
            }
        } while (pilih != 0);
    }
}
```

2. Hasil Verifikasi Kode Program

```
=== Menu Antrian Layanan Kemahasiswaan ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar
Pilih menu: 1
NIM : 1234
Nama : Adhim
Kelas : 1E
IPK : 3.0
Mahasiswa berhasil masuk antrian.
```

```
=== Menu Antrian Layanan Kemahasiswaan ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar
Pilih menu: 1
NIM : 2345
Nama : Budi
Kelas : 1P
IPK : 3.4
Mahasiswa berhasil masuk antrian.
```

```
=== Menu Antrian Layanan Kemahasiswaan ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar
Pilih menu: 1
NIM : 8909
Nama : Yuli
Kelas : 3A
IPK : 3.7
Mahasiswa berhasil masuk antrian.
```

```
=== Menu Antrian Layanan Kemahasiswaan ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar
Pilih menu: 3
Daftar Mahasiswa dalam Antrian:
Adhim - 1234 - 1E - 3.0
Budi - 2345 - 1P - 3.4
Yuli - 8909 - 3A - 3.7
```

```
=== Menu Antrian Layanan Kemahasiswaan ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar
Pilih menu: 2
Mahasiswa keluar dari antrian.
```

```
=== Menu Antrian Layanan Kemahasiswaan ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar
Pilih menu: 3
Daftar Mahasiswa dalam Antrian:
Budi - 2345 - 1P - 3.4
Yuli - 8909 - 3A - 3.7
```

=== Menu Antrian Layanan Kemahasiswaan ===

1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar

Pilih menu: 5

Mahasiswa terdepan: Budi - 2345 - 1P - 3.4

=== Menu Antrian Layanan Kemahasiswaan ===

1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar

Pilih menu: 6

Mahasiswa paling akhir: Yuli - 8909 - 3A - 3.7

=== Menu Antrian Layanan Kemahasiswaan ===

1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Semua Antrian
4. Kosongkan Antrian
5. Lihat Mahasiswa Terdepan
6. Lihat Mahasiswa Paling Akhir
7. Lihat Jumlah Mahasiswa dalam Antrian
8. Keluar

Pilih menu: 7

Jumlah mahasiswa dalam antrian: 2

