

**Laporan Hasil Praktikum Algoritma Dan
Struktur Data
Jobsheet 11**



Nama : Arifah Zhafirah Wikananda

NIM : 244107020188

Kelas : 1E

**Program Studi Teknologi
Informasi Jurusan Teknik
Informatika Politeknik Negeri
Malang
2025**

2.1 Percobaan 1 : Pembuatan Single Linked List

```
package jobsheet11;

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

    Mahasiswa04(){

    }

    Mahasiswa04(String name, String nm, String kls, double ip){
        nama = name;
        nim = nm;
        kelas = kls;
        ipk = ip;
    }
}
```

```
package jobsheet11;

public class Node04 {
    Mahasiswa04 data;
    Node04 next;

    public Node04(Mahasiswa04 data, Node04 next){
        this.data = data;
        this.next = next;
    }
}
```

```

package jobsheet11;

public class SingleLinkedList04 {
    Node04 head;
    Node04 tail;

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

    public void print() {
        if(!isEmpty()){
            Node04 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(Mahasiswa04 input) {
        Node04 ndInput = new Node04(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        }else {
            ndInput.next = head;
            head = ndInput;
        }
    }

    public void addLast(Mahasiswa04 input) {
        Node04 ndInput = new Node04(input, null);
        if(isEmpty()) {
            head = ndInput;
            tail = ndInput;
        }else {
            tail.next = ndInput;
            tail = ndInput;
        }
    }

    public void insertAfter(String key, Mahasiswa04 input) {
        Node04 ndInput = new Node04(input, null);
        Node04 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, Mahasiswa04 input) {
    if(index < 0) {
        System.out.println("indeks salah");
    }else if (index == 0) {
        addFirst(input);
    }else {
        Node04 temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new Node04(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}
}
}

```

Outputnya:

```

c:\Praktikum ASD>
c:\Praktikum ASD> c: && cd "c:\Praktikum ASD" && cmd /C ""C:\Program
Files\Java\jdk-21\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessag
es -cp "C:\Users\WINDOWS 11\AppData\Roaming\Code\User\workspaceStorag
e\9ab0f952d31733a7c7e6cca34608067d\redhat.java\jdt_ws\Praktikum ASD_c
51dcdc\bin" jobsheet11.SLLMain04 "
Linked list kosong
Isi Linked List:
Dirga  21212203      4D      3.6

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

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

```

2.1.3. Pertanyaan

1. Mengapa hasil compile kode program di baris pertama menghasilkan “Linked List Kosong”?

Jawab: Pada file SLLMain04.java, baris pertama dalam main menjalankan perintah: Namun, saat print() dipanggil, belum ada data yang dimasukkan ke dalam SingleLinkedList04. Maka, head masih bernilai null. Di dalam method print() (pada SingleLinkedList04.java) terdapat pengecekan:

Karena head == null, maka isEmpty() mengembalikan true, sehingga muncul output **“Linked List Kosong”**.

2. Jelaskan kegunaan variable temp secara umum pada setiap method!

Jawab: Variabel temp digunakan sebagai *pointer bantu* atau *penunjuk sementara* untuk menyusuri atau memanipulasi node pada linked list. Secara umum kegunaannya:

- **Traversal (penelusuran):** temp dipakai untuk menelusuri node satu per satu mulai dari head.
- **Menghindari perubahan head:** Supaya head tidak langsung dimodifikasi saat traversing atau manipulasi node, maka digunakan temp.
- **Digunakan dalam method** seperti print(), removeLast(), remove(key), dll untuk membaca atau mengubah pointer antar node tanpa kehilangan referensi ke head.

3. Lakukan modifikasi agar data dapat ditambahkan dari keyboard!

Jawab:

```
import java.util.Scanner;

public class SLLMain04 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Masukkan jumlah data:");
        int n = sc.nextInt();
        sc.nextLine(); // Consume newline character

        for(int i = 0; i < n; i++) {
            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(); // Consume newline character

            Mahasiswa04 mhs = new Mahasiswa04(nama, nim, kelas, ipk);

        }
    }
}
```

```
public void addFirst(String nama, String nim, String kelas, double ipk) {
    Node04 ndInput = new Node04(nama, nim, kelas, ipk, null);
    if (isEmpty()) {
        head = ndInput;
    }else {
        ndInput.next = head;
        head = ndInput;
    }
}

public void addLast(Mahasiswa04 input) {
    Node04 ndInput = new Node04(input, null);
    if(isEmpty()) {
        head = ndInput;
        tail = ndInput;
    }else {
        tail.next = ndInput;
        tail = ndInput;
    }
}
}
```

2.2. Percobaan 2 : Modifikasi Elemen pada Single Linked List

```
package jobsheet11;

public class SingleLinkedList04 {
    Node04 head;
    Node04 tail;

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

    public void print() {
        if(!isEmpty()){
            Node04 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(Mahasiswa04 input) {
        Node04 ndInput = new Node04(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        }else {
            ndInput.next = head;
            head = ndInput;
        }
    }

    public void addLast(Mahasiswa04 input) {
        Node04 ndInput = new Node04(input, null);
        if(isEmpty()) {
            head = ndInput;
            tail = ndInput;
        }else {
            tail.next = ndInput;
            tail = ndInput;
        }
    }
}
```

```

public void insertAfter(String key, Mahasiswa04 input) {
    Node04 ndInput = new Node04(input, null);
    Node04 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, Mahasiswa04 input) {
    if(index < 0) {
        System.out.println("indeks salah");
    }else if (index == 0) {
        addFirst(input);
    }else {
        Node04 temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new Node04(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}

public void getData(int index) {
    Node04 tmp = head;
    for (int i = 0; i < index; i++){
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}

public int indexOf(String key) {
    Node04 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!");
    }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 {
        Node04 temp = head;
        while (temp.next != tail) {
            temp = temp.next;
        }
        temp.next = null;
        tail = temp;
    }
}

public void remove(String key) {
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat dihapus!");
    }else {
        Node04 temp = head;
        while (temp != null) {
            if ((temp.data.nama.equalsIgnoreCase(key)) && (temp == head)) {
                this.removeFirst();
                break;
            }else if (temp.data.nama.equalsIgnoreCase(key)) {
                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 {
        Node04 temp = head;
        for (int i = 0; i < index; i++) {
            temp = temp.next;
        }
        temp.next = temp.next.next;
        if (temp.next == null) {
            tail = temp;
        }
    }
}
}

```



```

package jobsheet11;

public class SLLMain04 {
    public static void main(String[] args) {
        SingleLinkedList04 sll = new SingleLinkedList04();
        Mahasiswa04 mhs1 = new Mahasiswa04("Alvaro", "24212200", "1A", 4.0 );
        Mahasiswa04 mhs2 = new Mahasiswa04("Bimon", "23212201", "2B", 3.8);
        Mahasiswa04 mhs3 = new Mahasiswa04("Cintia", "22212202", "3C", 3.5);
        Mahasiswa04 mhs4 = new Mahasiswa04("Dirga", "21212203", "4D", 3.6);

        sll.print();
        sll.addFirst(mhs4);
        sll.print();
        sll.addFirst(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();

    }
}

```

Outputnya:

```

data index 1 :
Dirga    21212203        4D        3.6
data mahasiswa an Bimon berada pada index : 2

Isi Linked List:
Dirga    21212203        4D        3.6
Bimon    23212201        2B        3.8

Isi Linked List:
Bimon    23212201        2B        3.8

```

Activate V

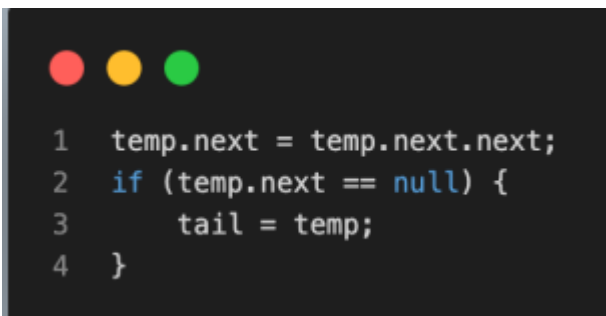
2.2.3 Pertanyaan

1. Mengapa digunakan keyword break pada fungsi remove? Jelaskan!

Jawab: Keyword ****break**** digunakan dalam method remove(key) (biasanya berupa loop seperti while atau for) untuk **menghentikan pencarian** setelah node yang dicari berhasil ditemukan dan dihapus. Tujuannya:

- **Efisiensi:** Tidak perlu melanjutkan traversal setelah node target ditemukan dan dihapus.
- **Keamanan:** Mencegah perubahan tidak disengaja pada node-node lain.
- **Menghindari kesalahan:** Jika tidak dihentikan, operasi berikutnya bisa mengakses node yang sudah tidak relevan.

2. Jelaskan kegunaan kode dibawah pada method remove



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

Jawab:

1. **temp.next = temp.next.next;**

- Menghapus node **setelah node temp**.
- Menyambungkan node temp ke node setelah temp.next, artinya node temp.next di-*bypass* dan akan dihapus dari list.

2. **if (temp.next == null) {**

- Mengecek apakah node setelah temp **sudah tidak ada** (artinya node yang dihapus adalah node terakhir/tail).

3. **tail = temp;**

- Jika benar, maka node temp sekarang menjadi node terakhir.
- Maka pointer tail perlu diperbarui agar menunjuk ke node temp, agar linked list tetap valid.

LATIHAN

Buatlah implementasi program antrian layanan unit kemahasiswaan sesuai dengan berikut ini :

- a. Implementasi antrian menggunakan Queue berbasis Linked List!
- b. Program merupakan proyek baru bukan modifikasi dari percobaan
- c. Ketika seorang mahasiswa akan mengantri, maka dia harus mendaftarkan datanya
- d. Cek antrian kosong, Cek antrian penuh, Mengosongkan antrian.
- e. Menambahkan antrian f. Memanggil antrian
- g. Menampilkan antrian terdepan dan antrian paling akhir
- h. Menampilkan jumlah mahasiswa yang masih mengantre.

```
public class TugasMahasiswa04 {
    String nim, nama, keperluan;

    public TugasMahasiswa04(String nim, String nama, String keperluan) {
        this.nim = nim;
        this.nama = nama;
        this.keperluan = keperluan;
    }

    void tampilkanInformasi() {
        System.out.println("NIM:" + nim + "\t Nama:" + nama + "\t
Keperluan:" + keperluan);
    }
}
```

```
public class TugasNode04 {
    TugasMahasiswa04 data;
    TugasNode04 next;

    TugasNode04(TugasMahasiswa04 data) {
        this.data = data;
        this.next = null;
    }
}
```

```

public class TugasSingleLingkedList04 {

    TugasNode04 front;
    TugasNode04 rear;
    int size;

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

    public void enqueue(TugasMahasiswa04 data) {
        TugasNode04 newNode = new TugasNode04(data);
        if (isEmpty()) {
            front = newNode;
            rear = newNode;
        } else {
            rear.next = newNode;
            rear = newNode;
        }
        size++;
    }

    public TugasMahasiswa04 dequeue() {
        if (isEmpty()) {
            System.out.println("Queue is empty");
            return null;
        }
        TugasMahasiswa04 data = front.data;
        front = front.next;
        size--;
        if (isEmpty()) {
            rear = null;
        }
        return data;
    }

    public TugasMahasiswa04 peekFront() {
        if (isEmpty()) {
            System.out.println("Queue is empty");
            return null;
        }
        return front.data;
    }

    public TugasMahasiswa04 peekRear() {
        if (isEmpty()) {
            System.out.println("Queue is empty");
            return null;
        }
        return rear.data;
    }

    public int getSize() {
        return size;
    }

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

```

```

public void tampilAntrian() {
    if (isEmpty()) {
        System.out.println("Queue is empty");
        return;
    }
    TugasNode04 current = front;
    System.out.println("Daftar Antrian:");
    while (current != null) {
        current.data.tampilkanInformasi();
        current = current.next;
    }
}
}
}

```

```

import java.util.Scanner;

public class TugasSllMain04 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        TugasSingleLingkedList04 queue = new TugasSingleLingkedList04();
        int pilih;

        do {
            System.out.println("\n===== MENU ANTRIAN LAYANAN
=====");
            System.out.println("1. Tambah Antrian");
            System.out.println("2. Panggil Antrian");
            System.out.println("3. Lihat Antrian Depan");
            System.out.println("4. Lihat Antrian Belakang");
            System.out.println("5. Lihat Semua Antrian");
            System.out.println("6. Lihat Jumlah Antrian");
            System.out.println("7. Kosongkan Antrian");
            System.out.println("8. Keluar Program");
            System.out.print("Pilih menu: ");
            pilih = sc.nextInt();
            sc.nextLine();

            switch (pilih) {
                case 1:
                    System.out.print("Masukkan NIM: ");
                    String nim = sc.nextLine();
                    System.out.print("Masukkan Nama: ");
                    String nama = sc.nextLine();
                    System.out.print("Masukkan Keperluan: ");
                    String keperluan = sc.nextLine();
                    TugasMahasiswa04 mhs = new TugasMahasiswa04(nim,
nama, keperluan);
                    queue.enqueue(mhs);
                    break;

                case 2:
                    TugasMahasiswa04 keluar = queue.dequeue();
                    if (keluar != null) {
                        System.out.println("Mahasiswa berikut telah
dipanggil:");
                        keluar.tampilkanInformasi();
                    }
                    break;
            }
        } while (true);
    }
}

```

```

        case 3:
            TugasMahasiswa04 depan = queue.peekFront();
            if (depan != null) {
                System.out.println("Antrian Terdepan:");
                depan.tampilkanInformasi();
            }
            break;

        case 4:
            TugasMahasiswa04 belakang = queue.peekRear();
            if (belakang != null) {
                System.out.println("Antrian Terakhir:");
                belakang.tampilkanInformasi();
            }
            break;

        case 5:
            queue.tampilAntrian();
            break;

        case 6:
            System.out.println("Jumlah Mahasiswa Dalam Antrian:
" + queue.getSize());
            break;

        case 7:
            queue.clear();
            break;

        case 8:
            System.out.println("Terima kasih telah menggunakan
layanan.");
            break;

        default:
            System.out.println("Pilihan tidak valid.");
    }

    } while (pilih != 8);

    sc.close();
}

```

Outputnya:

```
c:\Praktikum ASD>
c:\Praktikum ASD> c: && cd "c:\Praktikum ASD" &&
onMessages -cp "C:\Users\WINDOWS 11\AppData\Roam
\Praktikum ASD_c51dcdc\bin" TugasSllMain04 "

===== MENU ANTRIAN LAYANAN =====
1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program
Pilih menu: 1
Masukkan NIM: 1234
Masukkan Nama: Arifah
Masukkan Keperluan: Belajar
```

```
===== MENU ANTRIAN LAYANAN =====
1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program
Pilih menu: 8
Terima kasih telah menggunakan layanan.
```

```
===== MENU ANTRIAN LAYANAN =====
```

1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program

Pilih menu: 7

Antrian telah dikosongkan.

```
===== MENU ANTRIAN LAYANAN =====
```

1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program

Pilih menu: 5

Queue is empty

```
===== MENU ANTRIAN LAYANAN =====
```

1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program

Pilih menu: 4

Antrian Terakhir:

NIM:4321 Nama:Radit Keperluan:Belajar

```
===== MENU ANTRIAN LAYANAN =====
```

1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program

Pilih menu: 6

Jumlah Mahasiswa Dalam Antrian: 1


```
===== MENU ANTRIAN LAYANAN =====
1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program
Pilih menu: 2
Mahasiswa berikut telah dipanggil:
NIM:1234      Nama:Arifah      Keperluan:Belajar

===== MENU ANTRIAN LAYANAN =====
1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program
Pilih menu: 3
Antrian Terdepan:
NIM:4321      Nama:Radit      Keperluan:Belajar
```

```
===== MENU ANTRIAN LAYANAN =====
1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program
Pilih menu: 1
Masukkan NIM: 4321
Masukkan Nama: Radit
Masukkan Keperluan: Belajar

===== MENU ANTRIAN LAYANAN =====
1. Tambah Antrian
2. Panggil Antrian
3. Lihat Antrian Depan
4. Lihat Antrian Belakang
5. Lihat Semua Antrian
6. Lihat Jumlah Antrian
7. Kosongkan Antrian
8. Keluar Program
Pilih menu: 5
Daftar Antrian:
NIM:1234      Nama:Arifah      Keperluan:Belajar
NIM:4321      Nama:Radit      Keperluan:Belajar
```

