LAPORAN JOBSHEET 2

KONSEP DASAR PEMROGRAMAN

Mata Kuliah : Algoritma dan Struktur Data

Dosen : **Mungki Astiningrum, S.T., M.Kom.**

Alfreda Dhaifullah Mahezwara
244107020219
Kelas : 1A
Absen : 04

PROGRAM STUDI TEKNIK INFORMATIKA JURUSAN TEKNOLOGI INFORMASI POLITEKNIK NEGERI MALANG TAHUN 2025

1. Percobaan 1

```
package Pertemuan_12;

public class Mahasiswa004 {
    String nim;
    String nama;
    String kelas;
    double ipk;

    Mahasiswa004 () {

    }

    Mahasiswa004 (String nm, String name, String kls, double ip) {
        nim = nm;
        nama = name;
        kelas = kls;
        ipk = ip;
    }

    void tampilInformasi () {
        System.out.println("NIM: " + nim + ", Nama: " + nama + ", Kelas: " + kelas + ", IPK: " +
ipk);

    }
}
```

```
package Pertemuan_12;

public class Node04 {
    Mahasiswa004 data;
    Node04 next;

    public Node04 (Mahasiswa004 data, Node04 next) {
        this.data = data;
        this.next = next;

    }
}
```

```
package Pertemuan_12;

public class SingleLindeklist04 {
    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 (Mahasiswa004 input) {
        Node04 ndInput = new Node04(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }

    public void addLast (Mahasiswa004 input) {
        Node04 ndInput = new Node04(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
```

```
        } else {
            tail.next= ndInput;
            tail = ndInput;
        }
    }

    // untuk memasukan node yang memiliki data input setelah node yang memiliki data key
    public void insertAfter (String key, Mahasiswa004 input) {
        Node04 ndimput = new Node04(input, null);
        Node04 temp = head;
        do {
            if (temp.data.nama.equalsIgnoreCase(key)) {
                ndimput.next = temp.next;
                temp.next = ndimput;
                if (ndimput.next == null) {
                    tail = ndimput;
                }
                break;
            }
        } while (temp != null);
    }

    // method untuk menambahkan node pada indeks tertentu
    public void insertAt(int index, Mahasiswa004 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;
            }
        }
    }
}
```

```
package Pertemuan_12;

public class SLLMain04 {
    public static void main (String[] args) {
        // Membuat objek SingleLinkedList
        SingleLindeklist04 linkedList = new SingleLindeklist04();

        // Membuat empat objek Mahasiswa
        Mahasiswa004 mhs1 = new Mahasiswa004("123", "Alice", "A", 3.8);
        Mahasiswa004 mhs2 = new Mahasiswa004("124", "Bob", "B", 3.5);
        Mahasiswa004 mhs3 = new Mahasiswa004("125", "Charlie", "C", 3.9);
        Mahasiswa004 mhs4 = new Mahasiswa004("126", "Diana", "D", 3.7);

        // Menambahkan data ke linked list dan mencetak perubahan
        System.out.println("Menambahkan data mhs1:");
        linkedList.addFirst(mhs1);
        linkedList.print();

        System.out.println("\nMenambahkan data mhs2:");
        linkedList.addLast(mhs2);
        linkedList.print();

        System.out.println("\nMenambahkan data mhs3 setelah mhs1:");
        linkedList.insertAfter("Alice", mhs3);
        linkedList.print();

        System.out.println("\nMenambahkan data mhs4 ke akhir:");
        linkedList.addLast(mhs4);
        linkedList.print();
    }
}
```

```
n_12.SLLMain04
Menambahkan data mhs1:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8


Menambahkan data mhs2:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5


Menambahkan data mhs3 setelah mhs1:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8
NIM: 125, Nama: Charlie, Kelas: C, IPK: 3.9
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5


Menambahkan data mhs4 ke akhir:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8
NIM: 125, Nama: Charlie, Kelas: C, IPK: 3.9
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5
NIM: 126, Nama: Diana, Kelas: D, IPK: 3.7
```

2. Percobaan 2. Memodifikasi elemen pada single linked list

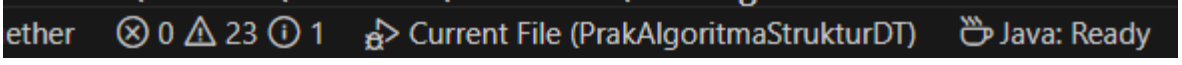
```
inkedList.print();


        System.out.println("data index 1 : ");

        linkedList.getData(1);


        System.out.println("data mahasisma an Binon berada pada index :
"+linkedList.indexOf("binon"));

        System.out.println();


        linkedList.removeFirst();

        linkedList.removeLast();

        linkedList.print();
```

```
        linkedList.removeAt(0);
        linkedList.print();

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 dibapus!");
        } else if (head == tail) {
            head = tail = null;
        } else {
            head = head.next;
        }
    }

    public void removeLast() {
        if (isEmpty()) {
            System.out.println("Linked List mashh 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 dibapus!");
        } 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 - 1; i++) {
                temp = temp.next;
            }
```

```
            temp.next = temp.next.next;
            if (temp.next == null) {
                tail = temp;
            }
        }
    }
```

```
Menambahkan data mhs1:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8


Menambahkan data mhs2:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5


Menambahkan data mhs3 setelah mhs1:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8
NIM: 125, Nama: Charlie, Kelas: C, IPK: 3.9
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5


Menambahkan data mhs4 ke akhir:
Isi linked list:
NIM: 123, Nama: Alice, Kelas: A, IPK: 3.8
NIM: 125, Nama: Charlie, Kelas: C, IPK: 3.9
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5
NIM: 126, Nama: Diana, Kelas: D, IPK: 3.7

data index 1 :
NIM: 125, Nama: Charlie, Kelas: C, IPK: 3.9
data mahasisma an Binon berada pada index : -1

Isi linked list:
NIM: 125, Nama: Charlie, Kelas: C, IPK: 3.9
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5

Isi linked list:
NIM: 124, Nama: Bob, Kelas: B, IPK: 3.5

PS D:\Kuliahh\kuliahhh\Semester2\PrakAlgoritmaStrukturDT>
```



TUGAS

```
package Pertemuan_12;
import java.util.Scanner;

class Mahasiswa {
    String nim;
    String nama;
    String jurusan;

    public Mahasiswa(String nim, String nama, String jurusan) {
        this.nim = nim;
        this.nama = nama;
        this.jurusan = jurusan;
    }

    void tampilData() {
        System.out.println("NIM: " + nim);
        System.out.println("Nama: " + nama);
        System.out.println("Jurusan: " + jurusan);
    }
}

class Node {
    Mahasiswa data;
    Node next;

    public Node(Mahasiswa data) {
        this.data = data;
        this.next = null;
    }
}

class AntrianKemahasiswaan {
    private Node front;
    private Node rear;
    private int size;
    private final int MAX_SIZE = 100; // Batas maksimal antrian

    public AntrianKemahasiswaan() {
```

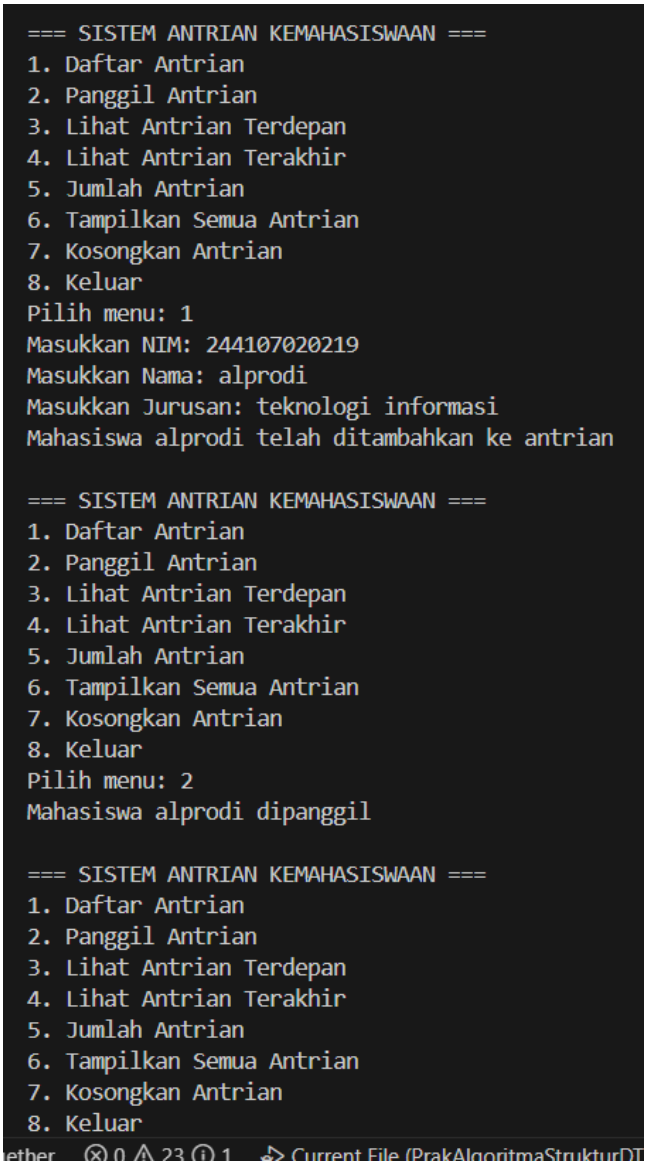
```
    front = null;
    rear = null;
    size = 0;
}

// c. Cek antrian kosong
public boolean isEmpty() {
    return front == null;
}

// d. Cek antrian penuh
public boolean isFull() {
    return size == MAX_SIZE;
}

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

// e. Menambahkan antrian
public void enqueue(Mahasiswa mhs) {
    if (isFull()) {
        System.out.println("Antrian penuh! Tidak bisa menambahkan lagi.");
        return;
    }

    Node newNode = new Node(mhs);
    if (isEmpty()) {
        front = newNode;
        rear = newNode;
    } else {
        rear.next = newNode;
        rear = newNode;
    }
    size++;
    System.out.println("Mahasiswa " + mhs.nama + " telah ditambahkan ke antrian");
```

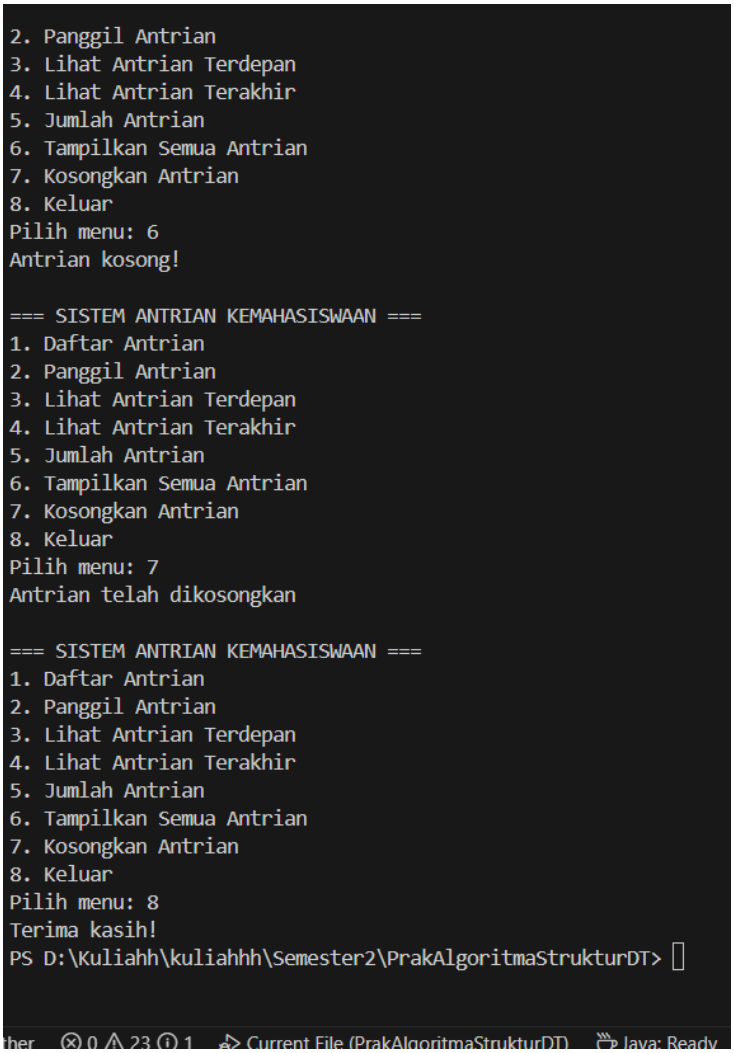
```
}

// f. Memanggil antrian
public Mahasiswa dequeue() {
    if (isEmpty()) {
        System.out.println("Antrian kosong!");
        return null;
    }

    Mahasiswa mhs = front.data;
    front = front.next;
    if (front == null) {
        rear = null;
    }
    size--;
    System.out.println("Mahasiswa " + mhs.nama + " dipanggil");
    return mhs;
}

// g. Menampilkan antrian terdepan
public void peekFront() {
    if (isEmpty()) {
        System.out.println("Antrian kosong!");
        return;
    }
    System.out.println("Mahasiswa terdepan:");
    front.data.tampilData();
}

// g. Menampilkan antrian paling akhir
public void peekRear() {
    if (isEmpty()) {
        System.out.println("Antrian kosong!");
        return;
    }
    System.out.println("Mahasiswa terakhir:");
    rear.data.tampilData();
}

// h. Menampilkan jumlah mahasiswa yang mengantre
```

```
    public void jumlahAntrian() {
        System.out.println("Jumlah mahasiswa yang mengantre: " + size);
    }

    // Menampilkan seluruh antrian
    public void displayQueue() {
        if (isEmpty()) {
            System.out.println("Antrian kosong!");
            return;
        }

        System.out.println("Daftar Antrian:");
        Node current = front;
        int no = 1;
        while (current != null) {
            System.out.print(no++ + ". ");
            current.data.tampilData();
            System.out.println("--------------------");
            current = current.next;
        }
    }
}

 class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        AntrianKemahasiswaan antrian = new AntrianKemahasiswaan();

        while (true) {
            System.out.println("\n=== SISTEM ANTRIAN KEMAHASISWAAN ===");
            System.out.println("1. Daftar Antrian");
            System.out.println("2. Panggil Antrian");
            System.out.println("3. Lihat Antrian Terdepan");
            System.out.println("4. Lihat Antrian Terakhir");
            System.out.println("5. Jumlah Antrian");
            System.out.println("6. Tampilkan Semua Antrian");
            System.out.println("7. Kosongkan Antrian");
            System.out.println("8. Keluar");
            System.out.print("Pilih menu: ");
```

```
int pilihan = scanner.nextInt();
scanner.nextLine(); // consume newline

switch (pilihan) {
    case 1:
        System.out.print("Masukkan NIM: ");
        String nim = scanner.nextLine();
        System.out.print("Masukkan Nama: ");
        String nama = scanner.nextLine();
        System.out.print("Masukkan Jurusan: ");
        String jurusan = scanner.nextLine();

        Mahasiswa mhs = new Mahasiswa(nim, nama, jurusan);
        antrian.enqueue(mhs);
        break;

    case 2:
        antrian.dequeue();
        break;

    case 3:
        antrian.peekFront();
        break;

    case 4:
        antrian.peekRear();
        break;

    case 5:
        antrian.jumlahAntrian();
        break;

    case 6:
        antrian.displayQueue();
        break;

    case 7:
        antrian.clear();
        break;
```

```
                case 8:
                    System.out.println("Terima kasih!");
                    scanner.close();
                    System.exit(0);

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





https://github.com/AlfredaDhaifullah04/Semester-2/tree/master/Pertemuan_12