LAPORAN HASIL PRAKTIKUM ALGORITMA DAN STRUKTUR DATA
JOBSHEET 11

NAMA :MOHAMAT FAUZI ROHMAN

NIM: 244107020067

KELAS : TI_1E

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2025

JOBSHEET XI

LINKED LIST

11. Praktikum

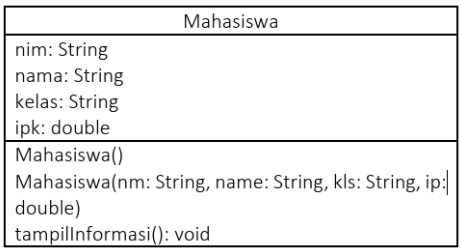
11.1 Percobaan 1: Pembuatan Single Linked List

11.1.1 Langkah-langkah Percobaan

1. Pada Project yang sudah dibuat pada Minggu sebelumnya. Buat folder atau package baru bernama Jobsheet11 di dalam repository Praktikum ASD.

2. Tambahkan class-class berikut:
   a. Mahasiswa00.java
   b. Node00.java
   c. SingleLinkedList00.java
   d. SLLMain00.java
   Ganti 00 dengan nomer Absen Anda

3. Implementasikan Class Mahasiswa00 sesuai dengan diagram class berikut ini :



4. Implementasi class Node seperti gambar berikut ini

```
public class NodeMahasiswa16 {
    Mahasiswa16 data;
    NodeMahasiswa16 next;

    public NodeMahasiswa16(Mahasiswa16 data, NodeMahasiswa16 next){
        this.data = data;
        this.next = next;
    }
}
```

5. Tambahkan attribute head dan tail pada class SingleLinkedList

```
public class SingleLinkedList16 {
    NodeMahasiswa16 head;
    NodeMahasiswa16 tail;
```

6. Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada SingleLinkedList.
7. Tambahkan method isEmpty().

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

8. Implementasi method untuk mencetak dengan menggunakan proses traverse.

```
public void print(){
    if (!isEmpty()) {
        NodeMahasiswa16 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");
    }
}
```

9. Implementasikan method addFirst().

```
public void addFirst(Mahasiswa16 input){
    NodeMahasiswa16 ndInput = new NodeMahasiswa16(input, null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else{
        ndInput.next = head;
        head = ndInput;
    }
}
```

10. Implementasikan method addLast().

```
public void addLast(Mahasiswa16 input){
    NodeMahasiswa16 ndInput = new NodeMahasiswa16(input, null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else{
        tail.next = ndInput;
        tail = ndInput;
    }
}
```

11. Implementasikan method insertAfter, untuk memasukkan node yang memiliki data input setelah node yang memiliki data key.

```
public void insertAfter(String key, Mahasiswa16 input){
    NodeMahasiswa16 ndInput = new NodeMahasiswa16(input, null);
    NodeMahasiswa16 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);
}
```

12. Tambahkan method penambahan node pada indeks tertentu.

```
public void insertAt (int index, Mahasiswa16 input){
    if (index < 0) {
        System.out.println("Indeks Salah");
    } else if (index == 0){
        addFirst(input);
    } else{
        NodeMahasiswa16 temp = head;
        for (int i = 0; i < index - 1; i++){
            temp = temp.next;
        }
        temp.next = new NodeMahasiswa16(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}
```

13. Pada class SLLMain00, buatlah fungsi main, kemudian buat object dari class SingleLinkedList.
14. Buat empat object mahasiswa dengan nama mhs1, mhs2, mhs3, mhs4 kemudian isi data setiap object melalui konstruktor.
15. Tambahkan Method penambahan data dan pencetakan data di setiap penambahannya agar terlihat perubahannya.

```
public class SLLMain16 {
    public static void main(String[] args) {

SingleLinkedList16 sll = new SingleLinkedList16();

Mahasiswa16 mhs1 = new Mahasiswa16("24212200", "Alvaro", "1A", 4.0);
Mahasiswa16 mhs2 = new Mahasiswa16("23212201", "Bimon", "2B", 3.8);
Mahasiswa16 mhs3 = new Mahasiswa16("22212202", "Cintia", "3C", 3.5);
Mahasiswa16 mhs4 = new Mahasiswa16("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();
```

11.1.2 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
```

11.1.3 Pertanyaan

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

: Karena kita melakukan perintah print, sedangkan data di linked list masih kosong. Data baru diisi ketika perintah addFirst pada mahasiswa4, sehingga perintah print yang kedua maka hasil outputnya adalah mahasiswa4

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

: - Pada method print, temp digunakan untuk menelusuri seluruh linked list dan mencetak data dari setiap node.

- Sedangkan di method insertAfter, temp digunakan untuk menemukan posisi yang tepat di mana elemen baru harus ditambahkan.

- Dan di method insertAt digunakan untuk menelusuri linked list untuk mencari posisi yang tepat di mana data mahasiswa2 akan disisipkan

3. Lakukan modifikasi agar data dapat ditambahkan dari keyboard!

```
import java.util.Scanner;
public class SLLMain16 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

SingleLinkedList16 sll = new SingleLinkedList16();

for (int i = 0; i < 4; i++) {
    System.out.println("Data Mahasiswa ke-" + (i + 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();
    System.out.println();

    Mahasiswa16 mhs = new Mahasiswa16(nim, nama, kelas, ipk);
    sll.addFirst(mhs);
    sll.print();
}
```

Hasil :

```
Data Mahasiswa ke-1
NIM: 111
Nama: Agus
Kelas: 1A
IPK: 3.3

Isi Linked List:
Agus            111     1A      3.3

Data Mahasiswa ke-2
NIM: 222
Nama: Bella
Kelas: 1B
IPK: 3.4

Isi Linked List:
Bella           222     1B      3.4
Agus            111     1A      3.3
```

```
Data Mahasiswa ke-3
NIM: 333
Nama: Citra
Kelas: 1C
IPK: 3.5

Isi Linked List:
Citra           333     1C      3.5
Bella           222     1B      3.4
Agus            111     1A      3.3

Data Mahasiswa ke-4
NIM: 444
Nama: Doni
Kelas: 1D
IPK: 3.6

Isi Linked List:
Doni            444     1D      3.6
Citra           333     1C      3.5
Bella           222     1B      3.4
Agus            111     1A      3.3
```

11.2 Percobaan 2: Modifikasi Elemen pada Single Linked List

11.2.1 Langkah-langkah Percobaan

1. Implementasikan method untuk mengakses data dan indeks pada linked list

2. Tambahkan method untuk mendapatkan data pada indeks tertentu pada class Single Linked List

```
public void getData(int index){
    NodeMahasiswa16 tmp = head;
    for (int i = 0; i < index; i++){
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}
```

3. Implementasikan method indexOf.

```
public int indexOf(String key){
    NodeMahasiswa16 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;
    }
}
```

4. Tambahkan method removeFirst pada class SingleLinkedList

```
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;
    }
}
```

5. Tambahkan method untuk menghapus data pada bagian belakang pada class SingleLinkedList

```
public void removeLast(){
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat dihapus!");
    } else if (head == tail) {
        head = tail = null;
    } else{
        NodeMahasiswa16 temp = head;
        while (temp.next != tail) {
            temp = temp.next;
        }
        temp.next = null;
        tail = temp;
    }
}
```

6. Sebagai langkah berikutnya, akan diimplementasikan method remove

```
public void remove(String key){
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat dihapus!");
    } else{
        NodeMahasiswa16 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;
        }
    }
}
```

7. Implementasi method untuk menghapus node dengan menggunakan index.

```
public void RemoveAt(int index){
    if (index == 0) {
        removeFirst();
    } else{
        NodeMahasiswa16 temp = head;
        for (int i = 0; i < index - 1; i++){
            temp = temp.next;
        }
        temp.next = temp.next.next;
        if (temp.next == null) {
            tail = temp;
        }
    }
}
}
```

8. Kemudian, coba lakukan pengaksesan dan penghapusan data di method main pada class SLLMain dengan menambahkan kode berikut

```
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();
}
}
```

9. Jalankan class SLLMain

11.2.2 Verifikasi Hasil Percobaan

```
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
```

11.2.3 Pertanyaan

1. Mengapa digunakan keyword break pada fungsi remove? Jelaskan!
   : break pada method remove digunakan untuk menghentikan pencarian dan penghapusan setelah node yang sesuai ditemukan

2. Jelaskan kegunaan kode dibawah pada method remove

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

: Kode tersebut digunakan untuk menghapus node dari linked list dan memperbarui referensi tail

11.3 Tugas

Class Mahasiswa

```
public class Mahasiswa {
    String nim;
    String nama;
    String prodi;

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

public void tampil(){
    System.out.println(nama + "\t" + nim + "\t" + prodi);
}
}
```

Class Node

```
public class Node {
    Mahasiswa data;
    Node next;

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

Class Queue

```
public class Queue {
Mahasiswa [] data;
Node head;
Node tail;
int size;

public Queue(){
    this.head = null;
    this.tail = null;
    this.size = 0;
}

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

public boolean isFull(){
    return false;
}

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

```
public void tambah(String nim, String nama, String prodi) {
Mahasiswa mahasiswa = new Mahasiswa(nim, nama, prodi);
Node nd = new Node(mahasiswa, null);
    if (isEmpty()) {
            head = tail = nd;
        } else {
            tail.next = nd;
            tail = nd;
        }
        size++;
        System.out.println(nama + " berhasil mendaftar ke antrian.");
    }

public void panggilAntrian(){
    if (isEmpty()) {
        System.out.println("Antrian masih kosong, tidak dapat dipanggil");
        return;
    }
    System.out.println("Mahasiswa " + head.data.nama + " dipanggil");
    head = head.next;
    size--;
}

public void terdepan(){
    if (isEmpty()) {
        System.out.println("Antrian masih kosong");
    } else{
        System.out.println("Antrian Terdepan: ");
        System.out.println("NAMA\tNIM\tPRODI");
        head.data.tampil();
    }
}

public void terakhir(){
    if (isEmpty()) {
        System.out.println("Antrian masih kosong");
    } else{
        System.out.println("Antrian Terakhir: ");
        System.out.println("NAMA\tNIM\tPRODI");
        tail.data.tampil();

    }
}

public void jumlahAntrian() {
        System.out.println("Jumlah Mahasiswa dalam Antrian: " + size);
    }
}
```

Class TugasMain

```
import java.util.Scanner;
public class TugasMain {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

Queue queue = new Queue();
int pilih;
```

```
do{
    System.out.println();
    System.out.println("==== LAYANAN UNIT KEMAHASISWAAN ====");
    System.out.println("1. Daftar Antrian");
    System.out.println("2. Panggil Antrian");
    System.out.println("3. Tampilkan Antrian Terdepan dan Paling Akhir");
    System.out.println("4. Tampilkan Jumlah Antrian");
    System.out.println("5. Kosongkan Antrian");
    System.out.println("0. Keluar");
    System.out.print("Pilih: ");
    pilih = sc.nextInt();
    sc.nextLine();

    switch (pilih) {
        case 1:
            System.out.println();
            System.out.println("Data Mahasiswa: ");
            System.out.print("NIM   : ");
            String nim = sc.nextLine();
            System.out.print("Nama  : ");
            String nama = sc.nextLine();
            System.out.print("Prodi : ");
            String prodi = sc.nextLine();
            queue.tambah(nim, nama, prodi);
            break;

        case 2:
            queue.panggilAntrian();
            break;

        case 3:
            queue.terdepan();
            System.out.println();
            queue.terakhir();
            break;

        case 4:
            queue.jumlahAntrian();
            break;

        case 5:
            queue.clear();
            break;

        case 0:
            System.out.println("Anda telah keluar, Terima Kasih");
            break;
}
} while (pilih != 0);
}
}
```

Output :

```
==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 1

Data Mahasiswa:
NIM   : 111
Nama  : Fauzi
Prodi : Informatika
Fauzi berhasil mendaftar ke antrian.

==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 1

Data Mahasiswa:
NIM   : 222
Nama  : Bella
Prodi : SIB
Bella berhasil mendaftar ke antrian.

==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 3
Antrian Terdepan:
NAMA    NIM     PRODI
Fauzi   111     Informatika

Antrian Terakhir:
NAMA    NIM     PRODI
Bella   222     SIB
```

```
==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 4
Jumlah Mahasiswa dalam Antrian: 2

==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 2
Mahasiswa Fauzi dipanggil

==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 4
Jumlah Mahasiswa dalam Antrian: 1

==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 5
Antrian sudah dikosongkan
```

```
==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 2
Antrian masih kosong, tidak dapat dipanggil

==== LAYANAN UNIT KEMAHASISWAAN ====
1. Daftar Antrian
2. Panggil Antrian
3. Tampilkan Antrian Terdepan dan Paling Akhir
4. Tampilkan Jumlah Antrian
5. Kosongkan Antrian
0. Keluar
Pilih: 0
Anda telah keluar, Terima Kasih
```