LAPORAN HASIL KUIS 2
ALGORITMA DAN STRUKTUR DATA
PRAKTIKUM

Oleh:

RANDA HERU KUSUMA

NIM. 2341760009

SIB-1F / 25

D-IV SISTEM INFORMASI BISNIS JURUSAN

TEKNOLOGI INFORMASI POLITEKNIK

NEGERI MALANG

Link github: https://github.com/randaheru/Kuis2.git

Program 1

Class Mahasiswa25

Program ini adalah aplikasi sederhana untuk manajemen data mahasiswa yang menggunakan LinkedList sebagai struktur data untuk menyimpan objek Mahasiswa25. Program ini menyediakan fungsi untuk menambah, menghapus, mencari, dan menampilkan data mahasiswa.

1. ***Struktur Program***
2. ***Class Mahasiswa25: Mewakili objek mahasiswa dengan atribut nim, nama, dan tglLahir.***
3. ***Class LinkedListNode25: Mewakili node dalam linked list yang berisi objek Mahasiswa25 dan referensi ke node berikutnya.***
4. ***Class LinkedList25: Implementasi linked list yang menyediakan operasi dasar seperti menambah, menghapus, mencari, dan menampilkan node.***
5. ***Class MainLinkedList25: Berisi metode main yang mengelola operasi utama program menggunakan LinkedList25 untuk menyimpan objek Mahasiswa25***

```
public Mahasiswa25() {
}

public Mahasiswa25(String nim, String nama, String tglLahir) {
    this.nim = nim;
    this.nama = nama;
    this.tglLahir = tglLahir;
}
```

Mahasiswa25() Konstruktor default tanpa parameter. Menginisialisasi objek Mahasiswa25 tanpa nilai awal untuk atribut.

Mahasiswa25(String nim, String nama, String tglLahir) Konstruktor dengan parameter. Menginisialisasi objek Mahasiswa25 dengan nilai parameter yang diberikan untuk nim, nama, dan tanggal lahir.

```
public String getNim() {
    return nim;
}
```

getNim() Metode publik yang mengembalikan nilai atribut nim.

```
public void setNim(String nim) {
    this.nim = nim;
}
```

setNim(String nim): Metode publik yang mengubah nilai atribut nim dengan parameter nim yang diberikan.

```
public String getNama() {
    return nama;
}
```

getNama() Metode publik yang mengembalikan nilai atribut nama.

```
public void setNama(String nama) {
    this.nama = nama;
}
```

setNama(String nama) Metode publik yang mengubah nilai atribut nama dengan parameter nama yang diberikan.

```
public String getTglLahir() {
    return tglLahir;
}
```

getTglLahir() Metode publik yang mengembalikan nilai atribut tglLahir.

```
public void setTglLahir(String tglLahir) {
    this.tglLahir = tglLahir;
}
```

setTglLahir(String tglLahir) Metode publik yang mengubah nilai atribut tglLahir dengan parameter tglLahir yang diberikan.

```
@Override
public String toString() {
    return "Mahasiswa25{" + "nim='" + nim + '\'' + ", nama='" + nama + '\'' + ", tglLahir='" + tglLahir + '\'' + '}';
}
```

toString() Metode ini mengoverride fungsi bawaan toString() dari class Object. Mengembalikan representasi String dari objek Mahasiswa25, menampilkan nilai dari nim, nama, dan tanggal lahir.

Class LinkedListNode25

```
public class LinkedListNode25 {
```

Membuat class LinkedListNode25

```
Mahasiswa25 data;
LinkedListNode25 next;
```

Data merupakan objek Mahasiswa25 yang menyimpan informasi tentang seorang mahasiswa.

```
Mahasiswa25 data;
LinkedListNode25 next;
```

Next merupakan referensi ke node berikutnya dalam linked list.

```
public LinkedListNode25(Mahasiswa25 data) {
    this.data = data;
    this.next = null;
```

LinkedListNode25(Mahasiswa25 data): Konstruktor ini menginisialisasi node baru dengan objek Mahasiswa25 yang diberikan sebagai parameter. Nilai next diatur ke null karena node ini dianggap sebagai node terakhir pada awalnya.

Class LinkedList25

```
public class LinkedList25 {
```

Membuat Class LinkedList25

```
    private LinkedListNode25 head;

    public LinkedList25() {
        this.head = null;
    }
```

private LinkedListNode25 head merupakan referensi ke node pertama (head) dalam linked list. Nilai awal head adalah null jika linked list masih kosong

```
public LinkedList25() {
    this.head = null;
}
```

public LinkedList25(): Konstruktor default yang menginisialisasi linked list kosong dengan head bernilai null.

```
    public void add(Mahasiswa25 data) {
        LinkedListNode25 newNode = new LinkedListNode25(data);
        if (head == null) {
            head = newNode;
        } else {
            LinkedListNode25 temp = head;
            while (temp.next != null) {
                temp = temp.next;
            }
            temp.next = newNode;
        }
    }
```

public void add(Mahasiswa25 data): Metode untuk menambahkan node baru berisi objek Mahasiswa25 ke akhir linked list.

```
    public void delete(String nim) {
        if (head == null) return;

        if (head.data.getNim().equals(nim)) {
            head = head.next;
            return;
        }
```

public void delete(String nim): Metode untuk menghapus node dari linked list berdasarkan nilai nim mahasiswa.

```
public Mahasiswa25 get(String nim) {
    LinkedListNode25 temp = head;
    while (temp != null) {
        if (temp.data.getNim().equals(nim)) {
            return temp.data;
        }
        temp = temp.next;
    }
    return null;
}
```

public Mahasiswa25 get(String nim): Metode untuk mencari dan mengembalikan objek Mahasiswa25 dari linked list berdasarkan nilai nim. Mengembalikan null jika mahasiswa tidak ditemukan.

```
    public void printAll() {
        LinkedListNode25 temp = head;
        while (temp != null) {
            System.out.println(temp.data);
            temp = temp.next;
        }
    }
}
```

public void printAll(): Metode untuk mencetak informasi seluruh mahasiswa yang ada di dalam linked list.

Class Main25

```
public class Main25 {
```

Membuat Class Main25

```
LinkedList25 list = new LinkedList25();
```

Membuat objek LinkedList25 bernama list untuk menyimpan data mahasiswa.

```
// Adding initial data
list.add(new Mahasiswa25(nim:"001", nama:"Alice", tglLahir:"01-01-2000"));
list.add(new Mahasiswa25(nim:"002", nama:"Bob", tglLahir:"02-02-2001"));
list.add(new Mahasiswa25(nim:"003", nama:"Charlie", tglLahir:"03-03-2002"));
list.add(new Mahasiswa25(nim:"004", nama:"David", tglLahir:"04-04-2003"));
```

Menambahkan beberapa data mahasiswa awal (opsional) ke dalam list.

```
Scanner scanner = new Scanner(System.in);
int choice;
```

Membuat objek Scanner untuk membaca input pengguna dari keyboard.

```
do {
    System.out.println(x:"\nMenu:");
    System.out.println(x:"1. Tambah Mahasiswa");
    System.out.println(x:"2. Hapus Mahasiswa");
    System.out.println(x:"3. Lihat Mahasiswa");
    System.out.println(x:"4. Cetak Semua Mahasiswa");
    System.out.println(x:"5. Keluar");
    System.out.print(s:"Pilih opsi: ");
    choice = scanner.nextInt();
    scanner.nextLine();  // Consume newline
```

Program menampilkan menu dengan pilihan: Tambah Mahasiswa, Hapus Mahasiswa, Lihat Mahasiswa, Cetak Semua Mahasiswa, dan Keluar. Menggunakan loop do-while untuk menampilkan menu berulang kali sampai pengguna memilih opsi keluar (5).

Dalam setiap iterasi loop: Membaca pilihan pengguna dan menyimpannya di choice.

Berdasarkan nilai choice, program melakukan aksi yang sesuai menggunakan switch statement.

```
    switch (choice) {
        case 1:
            System.out.print(s:"Masukkan NIM: ");
            String nim = scanner.nextLine();
            System.out.print(s:"Masukkan Nama: ");
            String nama = scanner.nextLine();
            System.out.print(s:"Masukkan Tanggal Lahir: ");
            String tglLahir = scanner.nextLine();
            list.add(new Mahasiswa25(nim, nama, tglLahir));
            break;
```

Meminta pengguna memasukkan informasi mahasiswa baru (NIM, nama, tanggal lahir).

Membuat objek Mahasiswa25 baru dengan informasi tersebut.

Memanggil method add pada list untuk menambahkan objek mahasiswa baru ke dalam linked list.

```
    case 2:
        System.out.print(s:"Masukkan NIM yang akan dihapus: ");
        String nimToDelete = scanner.nextLine();
        list.delete(nimToDelete);
        break;
```

Meminta pengguna memasukkan NIM mahasiswa yang ingin dihapus.

Memanggil method delete pada list untuk menghapus mahasiswa dengan NIM tersebut dari linked list.

```
    case 3:
        System.out.print(s:"Masukkan NIM yang dicari: ");
        String nimToFind = scanner.nextLine();
        Mahasiswa25 mhs = list.get(nimToFind);
        if (mhs != null) {
            System.out.println(mhs);
        } else {
            System.out.println(x:"Mahasiswa tidak ditemukan");
        }
        break;
```

Meminta pengguna memasukkan NIM mahasiswa yang ingin dicari.

Memanggil method get pada list untuk mencari mahasiswa dengan NIM tersebut.
Jika mahasiswa ditemukan, program menampilkan informasinya. Jika tidak ditemukan, program menampilkan pesan "Mahasiswa tidak ditemukan".

```
case 4:
    list.printAll();
    break;
```

Memanggil method printAll pada list untuk menampilkan informasi seluruh mahasiswa yang tersimpan di dalam linked list.

```
case 5:
    System.out.println(x:"Keluar");
    break;
```

Menampilkan pesan "Keluar" dan program berhenti.

```
default:
    System.out.println(x:"Opsi tidak valid");
```

Menampilkan pesan "Opsi tidak valid" jika pengguna memasukkan pilihan selain 1 sampai 5.

```
    } while (choice != 5);

    scanner.close();
}
```

Setelah loop menu selesai, program menutup objek Scanner untuk menghindari kebocoran resource.

```
Menu:
1. Tambah Mahasiswa
2. Hapus Mahasiswa
3. Lihat Mahasiswa
4. Cetak Semua Mahasiswa
5. Keluar
Pilih opsi:
```

Output program ini menunjukkan hasil dari operasi-operasi yang dilakukan pada linked list, termasuk penambahan, penghapusan, pencarian, dan pencetakan data mahasiswa. Program ini interaktif, sehingga outputnya bergantung pada input yang diberikan oleh pengguna saat memilih opsi menu. Setiap operasi memberikan umpan balik yang sesuai untuk membantu pengguna memahami hasil dari tindakan yang mereka lakukan.

Program 2

Class Mahasiswa25

Program ini merupakan aplikasi manajemen data mahasiswa yang menggunakan dua implementasi struktur data berbeda: ArrayList dan LinkedList. Program ini memungkinkan pengguna untuk menambah, menghapus, mencari, dan menampilkan data mahasiswa. Setiap mahasiswa diwakili oleh objek Mahasiswa25 yang memiliki atribut nim (Nomor Induk Mahasiswa), nama, dan tglLahir (Tanggal Lahir).

Struktur Data yang Digunakan

1. ***ArrayList: Struktur data yang digunakan dalam program utama untuk menyimpan objek Mahasiswa25. ArrayList adalah struktur data dinamis yang memungkinkan penyimpanan elemen-elemen secara berurutan dan akses elemen secara cepat menggunakan indeks.***
2. ***LinkedList: Struktur data alternatif yang disediakan dalam implementasi tambahan untuk menyimpan objek Mahasiswa25. LinkedList adalah struktur data yang terdiri dari node-node yang terhubung satu sama lain, di mana setiap node berisi data dan referensi ke node berikutnya.***

```
public class Mahasiswa25 {
    private String nim;
    private String nama;
    private String tglLahir;
```

Membuat class Bernama Mahasiswa25

private String nim: Menyimpan nomor induk mahasiswa dengan tipe data String.

private String nama: Menyimpan nama mahasiswa dengan tipe data String.

private String tglLahir: Menyimpan tanggal lahir mahasiswa dengan tipe data String.

```
public Mahasiswa25() {
}

public Mahasiswa25(String nim, String nama, String tglLahir) {
    this.nim = nim;
    this.nama = nama;
    this.tglLahir = tglLahir;
}
```

Mahasiswa25() Konstruktor default tanpa parameter. Menginisialisasi objek Mahasiswa25 tanpa nilai awal untuk atribut.

Mahasiswa25(String nim, String nama, String tglLahir) Konstruktor dengan parameter. Menginisialisasi objek Mahasiswa25 dengan nilai parameter yang diberikan untuk nim, nama, dan tanggal lahir.

```
public String getNim() {
    return nim;
}
```
getNim() Metode publik yang mengembalikan nilai atribut nim.

```
public void setNim(String nim) {
    this.nim = nim;
}
```
setNim(String nim): Metode publik yang mengubah nilai atribut nim dengan parameter nim yang diberikan.

```
public String getNama() {
    return nama;
}
```
getNama() Metode publik yang mengembalikan nilai atribut nama.

```
public void setNama(String nama) {
    this.nama = nama;
}
```
setNama(String nama) Metode publik yang mengubah nilai atribut nama dengan parameter nama yang diberikan.

```
public String getTglLahir() {
    return tglLahir;
}
```
getTglLahir() Metode publik yang mengembalikan nilai atribut tglLahir.

```
public void setTglLahir(String tglLahir) {
    this.tglLahir = tglLahir;
}
```
setTglLahir(String tglLahir) Metode publik yang mengubah nilai atribut tglLahir dengan parameter tglLahir yang diberikan.

```
@Override
public String toString() {
    return "Mahasiswa25{" + "nim='" + nim + '\'' + ", nama='" + nama + '\'' + ", tglLahir='" + tglLahir + '\'' + '}';
}
```
toString() Metode ini mengoverride fungsi bawaan toString() dari class Object. Mengembalikan representasi String dari objek Mahasiswa25, menampilkan nilai dari nim, nama, dan tanggal lahir.

Main25

```
public class Main25 {
```

Membuat class Bernama Main25

```
public static void main(String[] args) {
    ArrayList<Mahasiswa25> list = new ArrayList<>();
```

Metode main adalah titik masuk program.

```
    ArrayList<Mahasiswa25> list = new ArrayList<>();
```

Membuat objek ArrayList bernama list untuk menyimpan objek Mahasiswa25.

```
// Adding initial data
list.add(new Mahasiswa25(nim:"001", nama:"Alice", tglLahir:"01-01-2000"));
list.add(new Mahasiswa25(nim:"002", nama:"Bob", tglLahir:"02-02-2001"));
list.add(new Mahasiswa25(nim:"003", nama:"Charlie", tglLahir:"03-03-2002"));
list.add(new Mahasiswa25(nim:"004", nama:"David", tglLahir:"04-04-2003"));
```

Menambahkan beberapa objek Mahasiswa25 ke dalam list sebagai data awal (opsional).

```
Scanner scanner = new Scanner(System.in);
int choice;
```

Membuat objek Scanner bernama scanner untuk membaca input pengguna. Dan variabel choice untuk menyimpan pilihan pengguna dari menu.

```
do {
    System.out.println(x:"\nMenu:");
    System.out.println(x:"1. Tambah Mahasiswa");
    System.out.println(x:"2. Hapus Mahasiswa");
    System.out.println(x:"3. Lihat Mahasiswa");
    System.out.println(x:"4. Cetak Semua Mahasiswa");
    System.out.println(x:"5. Keluar");
    System.out.print(s:"Pilih opsi: ");
    choice = scanner.nextInt();
    scanner.nextLine();  // Consume newline
```

Program menampilkan menu dengan pilihan: Tambah Mahasiswa, Hapus Mahasiswa, Lihat Mahasiswa, Cetak Semua Mahasiswa, dan Keluar. Menggunakan loop do-while untuk menampilkan menu berulang kali sampai pengguna memilih opsi keluar (5).

Dalam setiap iterasi loop: Membaca pilihan pengguna dan menyimpannya di choice.

Berdasarkan nilai choice, program melakukan aksi yang sesuai menggunakan switch statement.

```
switch (choice) {
    case 1:
        System.out.print(s:"Masukkan NIM: ");
        String nim = scanner.nextLine();
        System.out.print(s:"Masukkan Nama: ");
        String nama = scanner.nextLine();
        System.out.print(s:"Masukkan Tanggal Lahir: ");
        String tglLahir = scanner.nextLine();
        list.add(new Mahasiswa25(nim, nama, tglLahir));
        break;
```

Meminta pengguna memasukkan NIM, nama, dan tanggal lahir mahasiswa baru.

Membuat objek Mahasiswa25 baru dengan data yang dimasukkan pengguna.
Menambahkan objek mahasiswa baru tersebut ke dalam list.

```
case 2:
    System.out.print(s:"Masukkan NIM yang akan dihapus: ");
    String nimToDelete = scanner.nextLine();
    list.removeIf(mhs -> mhs.getNim().equals(nimToDelete));
    break;
```

Meminta pengguna memasukkan NIM mahasiswa yang ingin dihapus.
Menggunakan method removeIf pada list untuk menghapus objek mahasiswa yang memiliki NIM yang sama dengan input pengguna.

```
case 3:
    System.out.print(s:"Masukkan NIM yang dicari: ");
    String nimToFind = scanner.nextLine();
    Mahasiswa25 mhs = list.stream()
            .filter(mahasiswa -> mahasiswa.getNim().equals(nimToFind))
            .findFirst()
            .orElse(other:null);
    if (mhs != null) {
        System.out.println(mhs);
    } else {
        System.out.println(x:"Mahasiswa tidak ditemukan");
    }
    break;
```

Meminta pengguna memasukkan NIM mahasiswa yang ingin dicari.
Menggunakan method stream, filter, dan findFirst untuk mencari objek mahasiswa di dalam list yang memiliki NIM yang sama dengan input pengguna.
Jika mahasiswa ditemukan, program menampilkan detail mahasiswa tersebut. Jika tidak ditemukan, program menampilkan pesan "Mahasiswa tidak ditemukan".

```
case 4:
    list.forEach(System.out::println);
    break;
```

Menggunakan method forEach pada list untuk mencetak detail seluruh objek mahasiswa yang ada di dalam list ke konsol.

```
case 5:
    System.out.println(x:"Keluar");
    break;
```

Menampilkan pesan "Keluar" dan program berhenti.

```
default:
    System.out.println(x:"Opsi tidak valid");
```

Menampilkan pesan "Opsi tidak valid" jika pengguna memasukkan pilihan selain 1 sampai 5.
Penutupan:

```
        } while (choice != 5);

        scanner.close();
    }
```

Setelah loop menu selesai, program menutup objek Scanner untuk menghindari kebocoran resource.

```
Menu:
1. Tambah Mahasiswa
2. Hapus Mahasiswa
3. Lihat Mahasiswa
4. Cetak Semua Mahasiswa
5. Keluar
Pilih opsi:
```

Program ini memberikan gambaran bagaimana struktur data ArrayList dan LinkedList dapat digunakan untuk mengelola data objek dalam konteks aplikasi manajemen data mahasiswa. Dengan menggunakan kedua struktur data ini, program dapat melakukan operasi dasar seperti menambah, menghapus, mencari, dan mencetak data secara efisien.