LAPORAN PRAKTIKUM

POSTTEST 5

PEMOGRAMAN BERBASIS OBJEK

Disusun oleh:

Muhammad Nashrul Fakhri (2309106074)

Kelas (B2 ‘23)

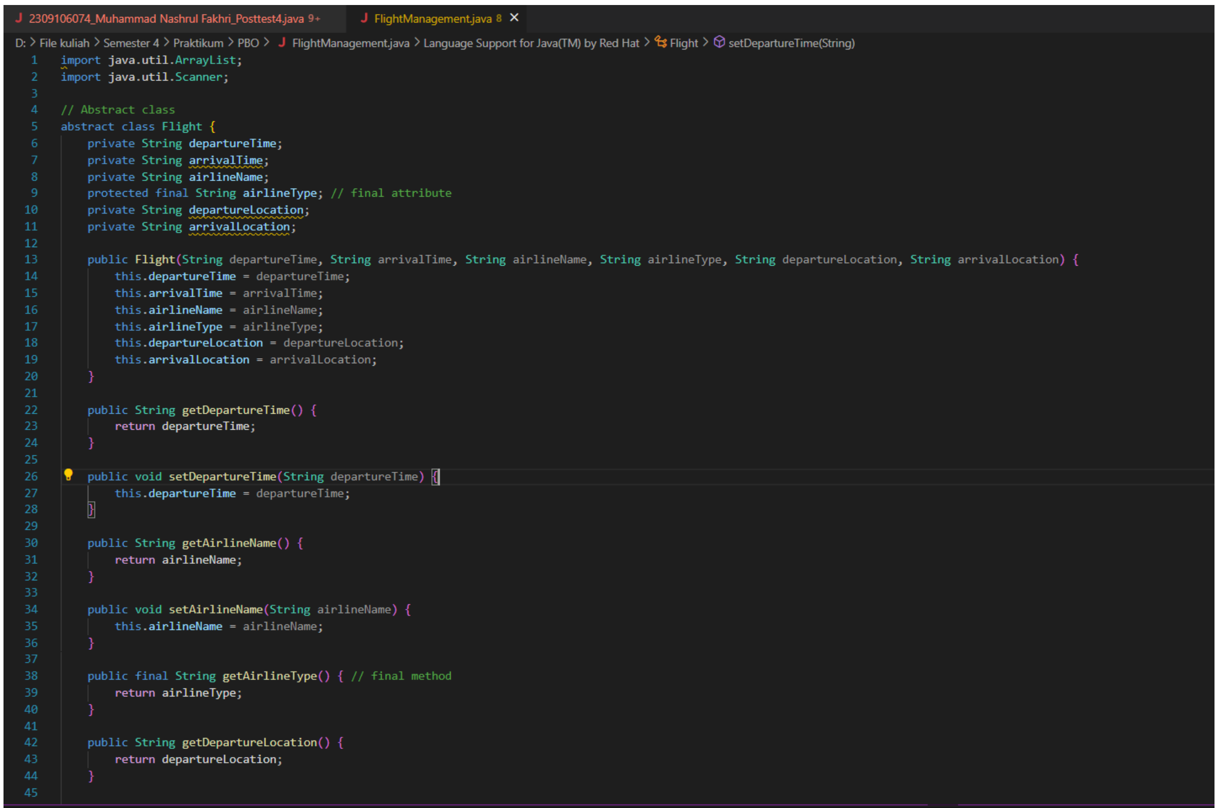
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

SAMARINDA

2025

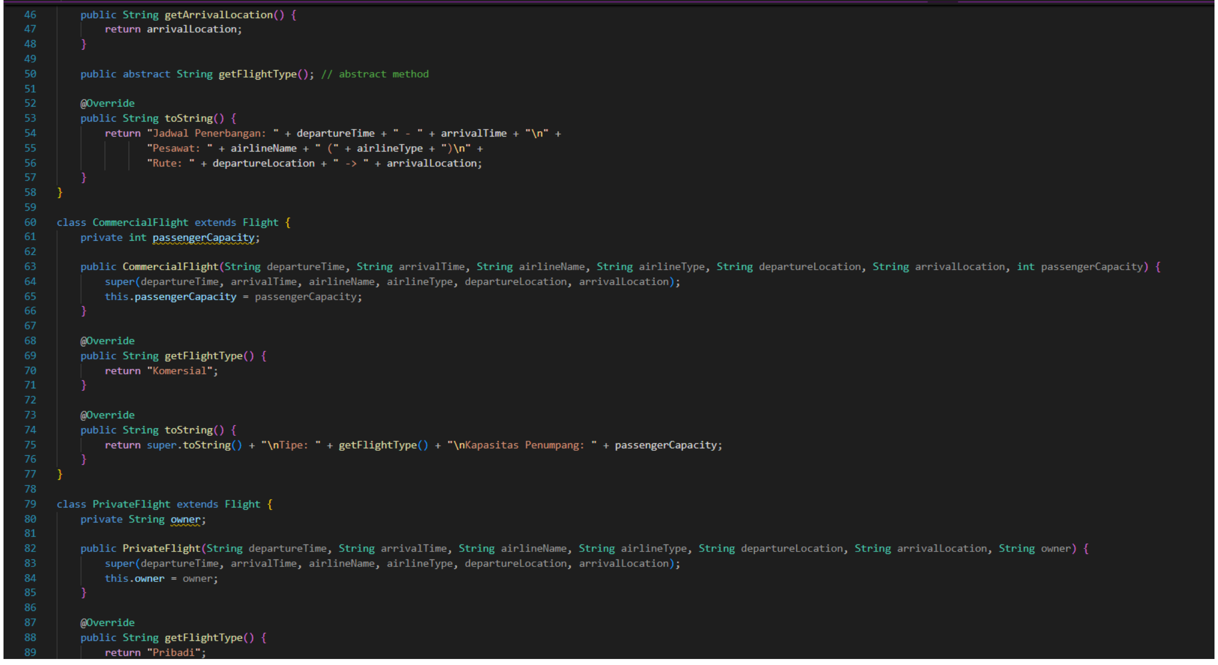
Ss Program Abstrack Class



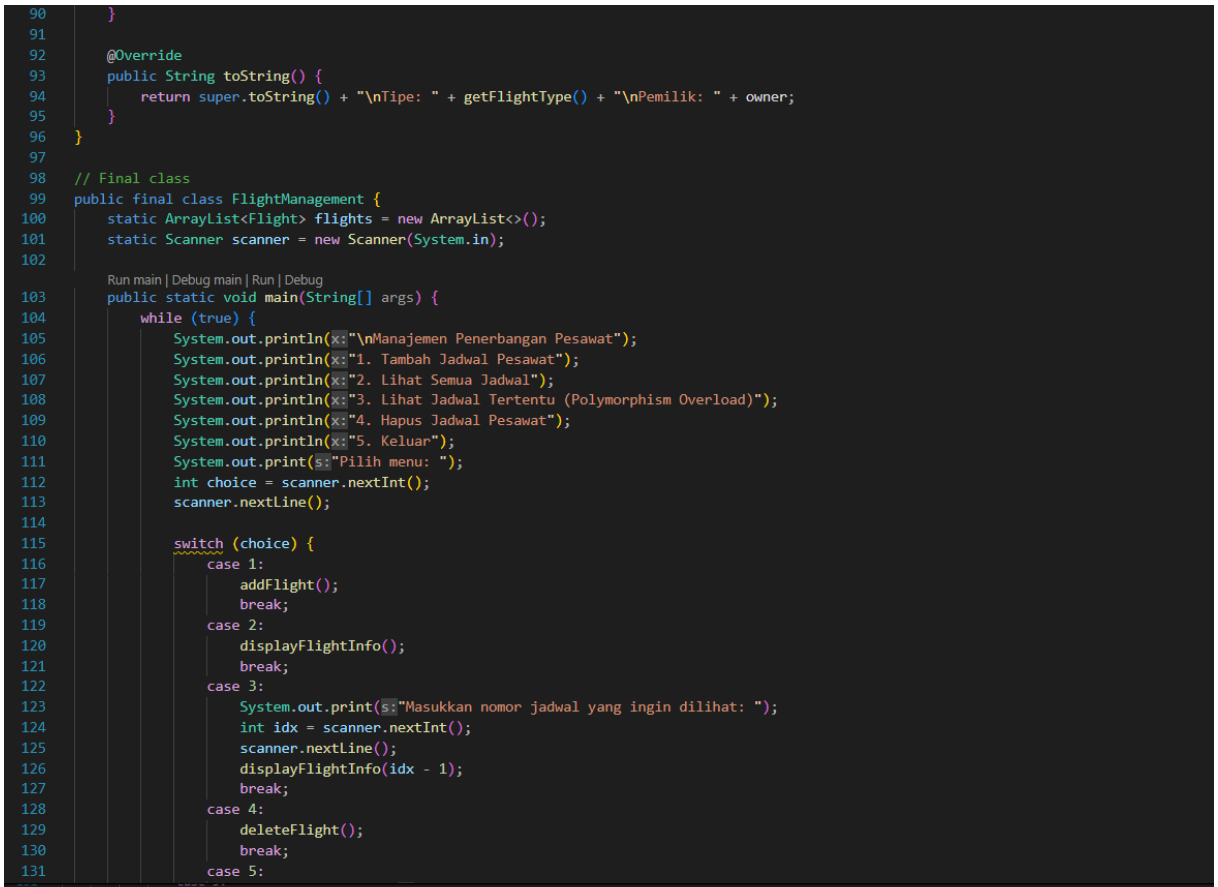
2309106074_Muhammad Nashrul Fakhri_Posttest4.java 9+ | FlightManagement.java 8

D: > File kuliah > Semester 4 > Praktikum > PBO > FlightManagement.java > Language Support for Java(TM) by Red Hat > Flight > setDepartureTime(String)

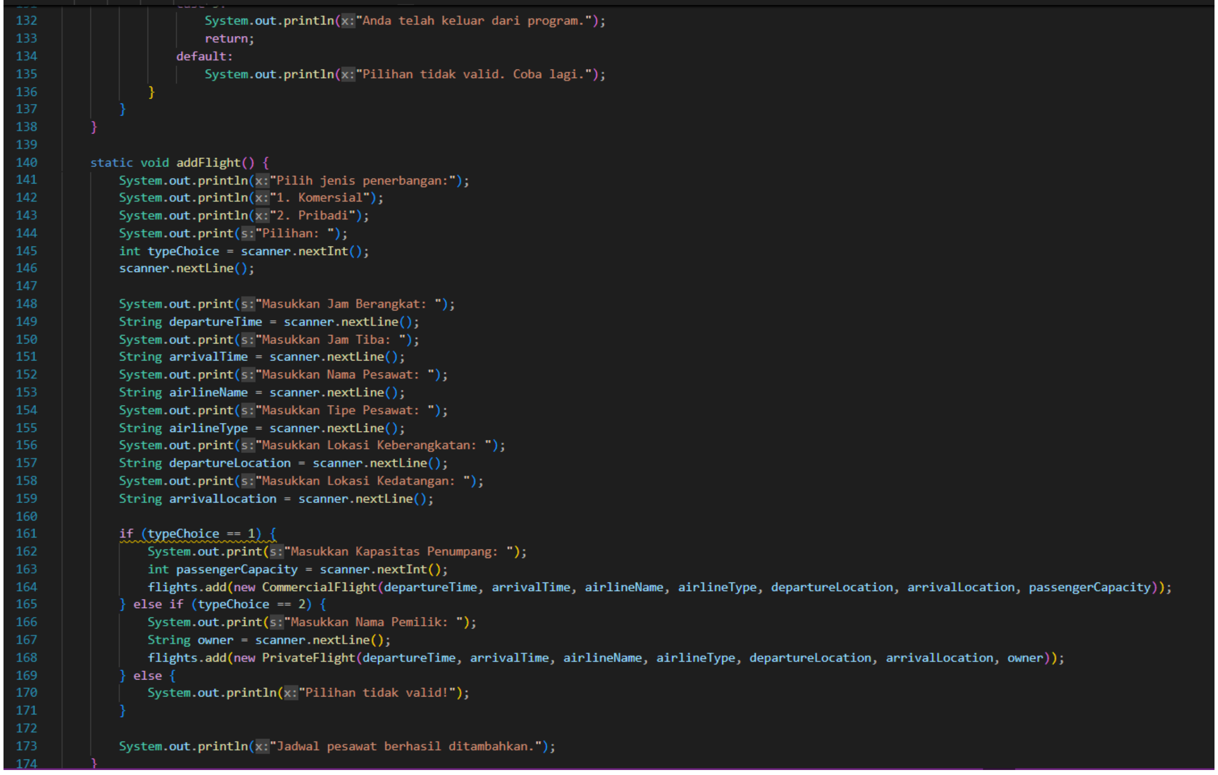
```java
import java.util.ArrayList;
import java.util.Scanner;

// Abstract class
abstract class Flight {
    private String departureTime;
    private String arrivalTime;
    private String airlineName;
    protected final String airlineType; // final attribute
    private String departureLocation;
    private String arrivalLocation;

    public Flight(String departureTime, String arrivalTime, String airlineName, String airlineType, String departureLocation, String arrivalLocation) {
        this.departureTime = departureTime;
        this.arrivalTime = arrivalTime;
        this.airlineName = airlineName;
        this.airlineType = airlineType;
        this.departureLocation = departureLocation;
        this.arrivalLocation = arrivalLocation;
    }

    public String getDepartureTime() {
        return departureTime;
    }

    public void setDepartureTime(String departureTime) {
        this.departureTime = departureTime;
    }

    public String getAirlineName() {
        return airlineName;
    }

    public void setAirlineName(String airlineName) {
        this.airlineName = airlineName;
    }

    public final String getAirlineType() { // final method
        return airlineType;
    }

    public String getDepartureLocation() {
        return departureLocation;
    }

```



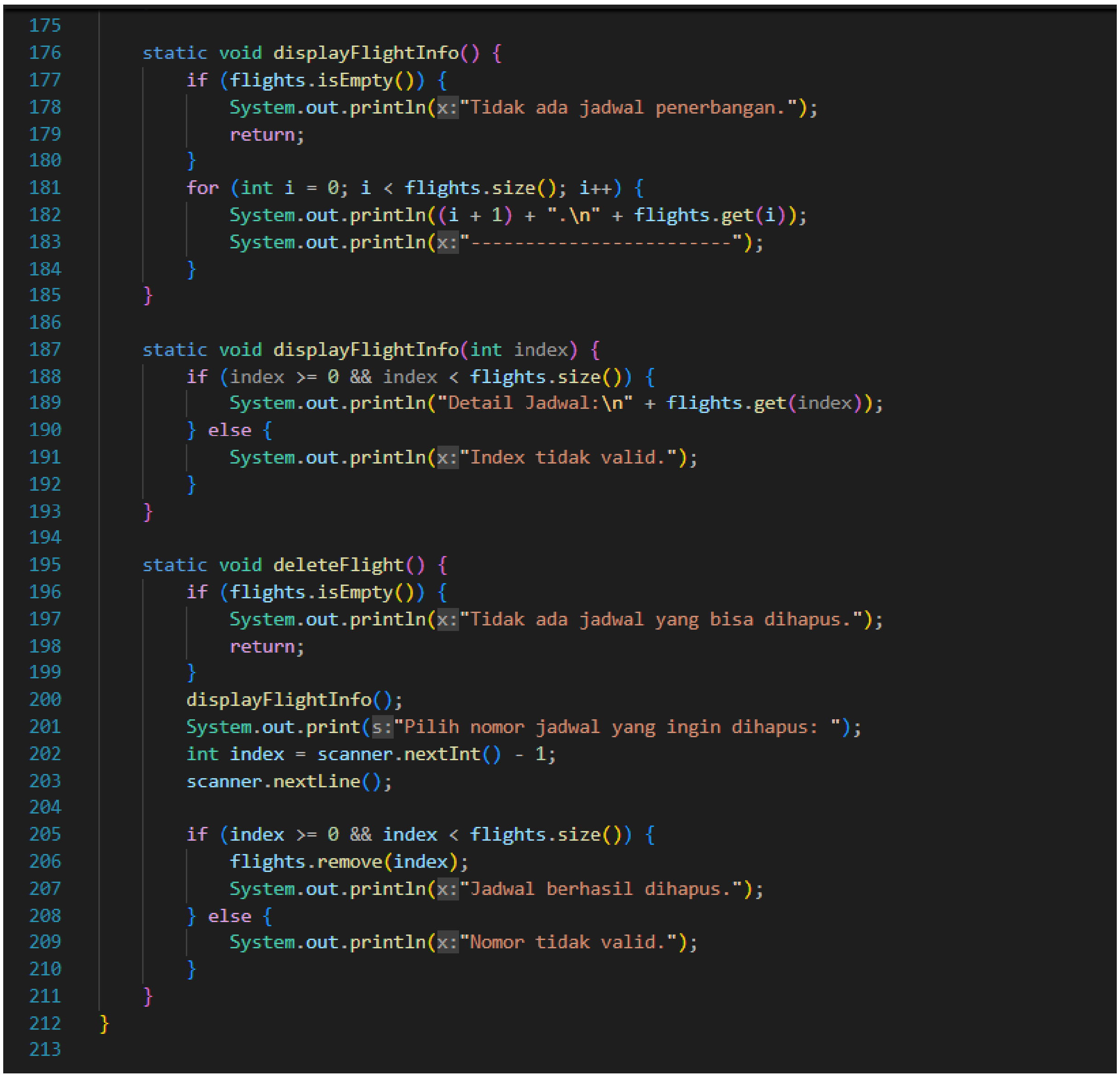
```java
    public String getArrivalLocation() {
        return arrivalLocation;
    }

    public abstract String getFlightType(); // abstract method

    @Override
    public String toString() {
        return "Jadwal Penerbangan: " + departureTime + " - " + arrivalTime + "\n" +
                "Pesawat: " + airlineName + " (" + airlineType + ")\n" +
                "Rute: " + departureLocation + " -> " + arrivalLocation;
    }
}

class CommercialFlight extends Flight {
    private int passengerCapacity;

    public CommercialFlight(String departureTime, String arrivalTime, String airlineName, String airlineType, String departureLocation, String arrivalLocation, int passengerCapacity) {
        super(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation);
        this.passengerCapacity = passengerCapacity;
    }

    @Override
    public String getFlightType() {
        return "Komersial";
    }

    @Override
    public String toString() {
        return super.toString() + "\nTipe: " + getFlightType() + "\nKapasitas Penumpang: " + passengerCapacity;
    }
}

class PrivateFlight extends Flight {
    private String owner;

    public PrivateFlight(String departureTime, String arrivalTime, String airlineName, String airlineType, String departureLocation, String arrivalLocation, String owner) {
        super(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation);
        this.owner = owner;
    }

    @Override
    public String getFlightType() {
        return "Pribadi";
```



```
    }

    @Override
    public String toString() {
        return super.toString() + "\nTipe: " + getFlightType() + "\nPemilik: " + owner;
    }
}

// Final class
public final class FlightManagement {
    static ArrayList<Flight> flights = new ArrayList<>();
    static Scanner scanner = new Scanner(System.in);

    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        while (true) {
            System.out.println(x:"\nManajemen Penerbangan Pesawat");
            System.out.println(x:"1. Tambah Jadwal Pesawat");
            System.out.println(x:"2. Lihat Semua Jadwal");
            System.out.println(x:"3. Lihat Jadwal Tertentu (Polymorphism Overload)");
            System.out.println(x:"4. Hapus Jadwal Pesawat");
            System.out.println(x:"5. Keluar");
            System.out.print(s:"Pilih menu: ");
            int choice = scanner.nextInt();
            scanner.nextLine();

            switch (choice) {
                case 1:
                    addFlight();
                    break;
                case 2:
                    displayFlightInfo();
                    break;
                case 3:
                    System.out.print(s:"Masukkan nomor jadwal yang ingin dilihat: ");
                    int idx = scanner.nextInt();
                    scanner.nextLine();
                    displayFlightInfo(idx - 1);
                    break;
                case 4:
                    deleteFlight();
                    break;
                case 5:
```



```
                    System.out.println(x:"Anda telah keluar dari program.");
                    return;
                default:
                    System.out.println(x:"Pilihan tidak valid. Coba lagi.");
            }
        }
    }

    static void addFlight() {
        System.out.println(x:"Pilih jenis penerbangan:");
        System.out.println(x:"1. Komersial");
        System.out.println(x:"2. Pribadi");
        System.out.print(s:"Pilihan: ");
        int typeChoice = scanner.nextInt();
        scanner.nextLine();

        System.out.print(s:"Masukkan Jam Berangkat: ");
        String departureTime = scanner.nextLine();
        System.out.print(s:"Masukkan Jam Tiba: ");
        String arrivalTime = scanner.nextLine();
        System.out.print(s:"Masukkan Nama Pesawat: ");
        String airlineName = scanner.nextLine();
        System.out.print(s:"Masukkan Tipe Pesawat: ");
        String airlineType = scanner.nextLine();
        System.out.print(s:"Masukkan Lokasi Keberangkatan: ");
        String departureLocation = scanner.nextLine();
        System.out.print(s:"Masukkan Lokasi Kedatangan: ");
        String arrivalLocation = scanner.nextLine();

        if (typeChoice == 1) {
            System.out.print(s:"Masukkan Kapasitas Penumpang: ");
            int passengerCapacity = scanner.nextInt();
            flights.add(new CommercialFlight(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation, passengerCapacity));
        } else if (typeChoice == 2) {
            System.out.print(s:"Masukkan Nama Pemilik: ");
            String owner = scanner.nextLine();
            flights.add(new PrivateFlight(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation, owner));
        } else {
            System.out.println(x:"Pilihan tidak valid!");
        }

        System.out.println(x:"Jadwal pesawat berhasil ditambahkan.");
    }
```


```java

    static void displayFlightInfo() {
        if (flights.isEmpty()) {
            System.out.println(x:"Tidak ada jadwal penerbangan.");
            return;
        }
        for (int i = 0; i < flights.size(); i++) {
            System.out.println((i + 1) + ".\n" + flights.get(i));
            System.out.println(x:"------------------------");
        }
    }

    static void displayFlightInfo(int index) {
        if (index >= 0 && index < flights.size()) {
            System.out.println("Detail Jadwal:\n" + flights.get(index));
        } else {
            System.out.println(x:"Index tidak valid.");
        }
    }

    static void deleteFlight() {
        if (flights.isEmpty()) {
            System.out.println(x:"Tidak ada jadwal yang bisa dihapus.");
            return;
        }
        displayFlightInfo();
        System.out.print(s:"Pilih nomor jadwal yang ingin dihapus: ");
        int index = scanner.nextInt() - 1;
        scanner.nextLine();

        if (index >= 0 && index < flights.size()) {
            flights.remove(index);
            System.out.println(x:"Jadwal berhasil dihapus.");
        } else {
            System.out.println(x:"Nomor tidak valid.");
        }
    }
}

```