

LAPORAN PRAKTIKUM
POSTTEST 6
PEMOGRAMAN BERBASIS
OBJEK



Disusun oleh:
Muhammad Nashrul Fakhri (2309106074)
Kelas (B2 '23)

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA

2025

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

```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 // Interface
6 interface Schedulable {
7     String scheduleInfo();
8     int flightDuration(); // dalam menit
9 }
10
11 // Abstract class
12 abstract class Flight {
13     private String departureTime;
14     private String arrivalTime;
15     private String airlineName;
16     protected final String airlineType; // final attribute
17     private String departureLocation;
18     private String arrivalLocation;
19
20     public Flight(String departureTime, String arrivalTime, String airlineName, String airlineType, String departureLocation, String arrivalLocation) {
21         this.departureTime = departureTime;
22         this.arrivalTime = arrivalTime;
23         this.airlineName = airlineName;
24         this.airlineType = airlineType;
25         this.departureLocation = departureLocation;
26         this.arrivalLocation = arrivalLocation;
27     }
28
29     public String getDepartureTime() {
30         return departureTime;
31     }
32
33     public void setDepartureTime(String departureTime) {
34         this.departureTime = departureTime;
35     }
36
37     public String getArrivalTime() {
38         return arrivalTime;
39     }
40
41     public String getAirlineName() {
42         return airlineName;
43     }
44
45     public void setAirlineName(String airlineName) {
46         this.airlineName = airlineName;
47     }
48
49     public final String getAirlineType() { // final method
50         return airlineType;
51     }
52
53     public String getDepartureLocation() {
54         return departureLocation;
55     }
56
57     public String getArrivalLocation() {
58         return arrivalLocation;
59     }
60
61     public abstract String getFlightType(); // abstract method
62
63     @Override
64     public String toString() {
65         return "Jadwal Penerbangan: " + departureTime + " - " + arrivalTime + "\n" +
66             "Pesawat: " + airlineName + " (" + airlineType + ")\n" +
67             "Route: " + departureLocation + " -> " + arrivalLocation;
68     }
69 }
70
71 // Class CommercialFlight yang mengimplementasikan Interface Schedulable
72 class CommercialFlight extends Flight implements Schedulable {
73     private int passengerCapacity;
74
75     public CommercialFlight(String departureTime, String arrivalTime, String airlineName, String airlineType, String departureLocation, String arrivalLocation, int passengerCapacity) {
76         super(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation);
77         this.passengerCapacity = passengerCapacity;
78     }
79
80     @Override
81     public String getFlightType() {
82         return "Komersial";
83     }
84
85     @Override
86     public String toString() {
87         return super.toString() + "\nType: " + getFlightType() + "\nKapasitas Penumpang: " + passengerCapacity;
88     }
89
90     @Override
91     public String scheduleInfo() {
92         return "Penerbangan komersial dari " + getDepartureLocation() + " ke " + getArrivalLocation();
93     }
94
95     @Override
96     public int flightDuration() {
97         // Contoh simple: asumsi penerbangan komersial 2 jam
98         return 120;
99     }
100 }
101
102 // Class PrivateFlight yang juga mengimplementasikan Interface Schedulable
103 class PrivateFlight extends Flight implements Schedulable {
104     private String owner;
105
106     public PrivateFlight(String departureTime, String arrivalTime, String airlineName, String airlineType, String departureLocation, String arrivalLocation, String owner) {
107         super(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation);
108         this.owner = owner;
109     }
110 }
```

```

110
111     @Override
112     public String getFlightType() {
113         return "Pribadi";
114     }
115
116     @Override
117     public String toString() {
118         return super.toString() + "\nType: " + getFlightType() + "\nPemilik: " + owner;
119     }
120
121     @Override
122     public String scheduleInfo() {
123         return "Penerbangan pribadi milik " + owner + " dari " + getDepartureLocation() + " ke " + getArrivalLocation();
124     }
125
126     @Override
127     public int flightDuration() {
128         // Contoh simple: penerbangan pribadi 90 menit
129         return 90;
130     }
131 }
132
133 // Final class
134 public final class FlightManagement {
135     static ArrayList<Flight> flights = new ArrayList<>();
136     static Scanner scanner = new Scanner(System.in);
137
138     Run main | Debug main | Run | Debug
139     public static void main(String[] args) {
140         while (true) {
141             System.out.println(x:"\nManajemen Penerbangan Pesawat");
142             System.out.println(x:"1. Tambah Jadwal Pesawat");
143             System.out.println(x:"2. Lihat Semua Jadwal");
144             System.out.println(x:"3. Lihat Jadwal Tertentu (Polymorphism Overload)");
145             System.out.println(x:"4. Hapus Jadwal Pesawat");
146             System.out.println(x:"5. Keluar");
147             System.out.print(s:"Pilih menu: ");
148
149             try {
150                 int choice = scanner.nextInt();
151                 scanner.nextLine();
152
153                 switch (choice) {
154                     case 1:
155                         addFlight();
156                         break;
157                     case 2:
158                         displayFlightInfo();
159                         break;
160                     case 3:
161                         System.out.print(s:"Masukkan nomor jadwal yang ingin dilihat: ");
162                         int idx = scanner.nextInt();
163                         scanner.nextLine();

```

```

163         displayFlightInfo(idx - 1);
164         break;
165     case 4:
166         deleteFlight();
167         break;
168     case 5:
169         System.out.println("Anda telah keluar dari program.");
170         return;
171     default:
172         System.out.println("Pilihan tidak valid. Coba lagi.");
173     }
174 } catch (InputMismatchException e) {
175     System.out.println("Input harus berupa angka!");
176     scanner.nextLine(); // membersihkan buffer
177 }
178 }
179 }
180
181 static void addFlight() {
182     try {
183         System.out.println("Pilih jenis penerbangan:");
184         System.out.println("1. Komersial");
185         System.out.println("2. Pribadi");
186         System.out.print("Pilihan: ");
187         int typeChoice = scanner.nextInt();
188         scanner.nextLine();
189
190         System.out.print("Masukkan Jam Berangkat: ");
191         String departureTime = scanner.nextLine();
192         System.out.print("Masukkan Jam Tiba: ");
193         String arrivalTime = scanner.nextLine();
194         System.out.print("Masukkan Nama Pesawat: ");
195         String airlineName = scanner.nextLine();
196         System.out.print("Masukkan Tipe Pesawat: ");
197         String airlineType = scanner.nextLine();
198         System.out.print("Masukkan Lokasi Keberangkatan: ");
199         String departureLocation = scanner.nextLine();
200         System.out.print("Masukkan Lokasi Kedatangan: ");
201         String arrivalLocation = scanner.nextLine();
202
203         if (typeChoice == 1) {
204             System.out.print("Masukkan Kapasitas Penumpang: ");
205             int passengerCapacity = scanner.nextInt();
206             flights.add(new CommercialFlight(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation, passengerCapacity));
207         } else if (typeChoice == 2) {
208             System.out.print("Masukkan Nama Pemilik: ");
209             String owner = scanner.nextLine();
210             flights.add(new PrivateFlight(departureTime, arrivalTime, airlineName, airlineType, departureLocation, arrivalLocation, owner));
211         } else {
212             System.out.println("Pilihan tidak valid!");
213         }
214     }
215     System.out.println("Jadwal pesawat berhasil ditambahkan.");
216 } catch (InputMismatchException e) {
217     System.out.println("Input salah, pastikan angka diisi dengan benar.");
218     scanner.nextLine(); // membersihkan buffer
219 }
220 }
221
222 static void displayFlightInfo() {
223     if (flights.isEmpty()) {
224         System.out.println("Tidak ada jadwal penerbangan.");
225         return;
226     }
227     for (int i = 0; i < flights.size(); i++) {
228         System.out.println((i + 1) + ".\n" + flights.get(i));
229         if (flights.get(i) instanceof Schedulable) {
230             Schedulable s = (Schedulable) flights.get(i);
231             System.out.println("Info Jadwal: " + s.scheduleInfo());
232             System.out.println("Durasi Penerbangan: " + s.flightDuration() + " menit");
233         }
234         System.out.println("-----");
235     }
236 }
237
238 static void displayFlightInfo(int index) {
239     if (index >= 0 && index < flights.size()) {
240         System.out.println("Detail Jadwal:\n" + flights.get(index));
241     } else {
242         System.out.println("Index tidak valid.");
243     }
244 }
245
246 static void deleteFlight() {
247     if (flights.isEmpty()) {
248         System.out.println("Tidak ada jadwal yang bisa dihapus.");
249         return;
250     }
251     displayFlightInfo();
252     System.out.print("Pilih nomor jadwal yang ingin dihapus: ");
253     try {
254         int index = scanner.nextInt() - 1;
255         scanner.nextLine();
256
257         if (index >= 0 && index < flights.size()) {
258             flights.remove(index);
259             System.out.println("Jadwal berhasil dihapus.");
260         } else {
261             System.out.println("Nomor tidak valid.");
262         }
263     } catch (InputMismatchException e) {
264         System.out.println("Input harus berupa angka.");
265         scanner.nextLine(); // bersihkan buffer
266     }
267 }
268 }

```


