

PRACTICAL ASSESSMENT

Employee Name:Seneha.S

Employee ID:seneha selvaraj

Problem Statement:

Develop a **menu-driven console application** in Java that performs **CRUD (Create, Read, Update, Delete)** operations on airline flight records. The application should use **Data Structure** as the data store instead of a database.

Entity to be Used: **Flight**

Fields:

- flightNumber (String) – unique identifier
- airlineName (String)
- source (String)
- destination (String)
- departureTime (String)
- arrivalTime (String)

Tasks:

- **Create:** Add a new flight record.
- **Read:**
 - o Display all flight records.
 - o Search for a flight by flightNumber.
- **Update:** Modify details of an existing flight by flightNumber.
- **Delete:** Remove a flight record by flightNumber.

PRACTICAL ASSESSMENT

Solution :

AirLine class:

```
import java.util.Comparator;

public class AirLine {
    int fno;
    String fname;
    String source;
    String destination;
    String departuretime;
    String arrivaltime;
    public AirLine(int fno, String fname, String source, String destination, String
    departuretime, String arrivaltime) {
        super();
        this.fno = fno;
        this.fname = fname;
        this.source = source;
        this.destination = destination;
        this.departuretime = departuretime;
        this.arrivaltime = arrivaltime;
    }
    @Override
    public String toString() {
        return "AirLine [fno=" + fno + ", fname=" + fname + ", source=" + source +",
        destination=" + destination
        + ", departuretime=" + departuretime + ", arrivaltime=" + arrivaltime + "]";
    }
}
```

MainClass

PRACTICAL ASSESSMENT

```
import java.util.LinkedList;
import java.util.List;
import java.util.Scanner;

public class MainMethod {
    static Scanner sc=new Scanner(System.in);
    static List<AirLine>Flights=new LinkedList();
    public static void add() {
        System.out.println("Enter how many flight want to add in the database ");
        int size=sc.nextInt();
        for(int i=0;i<size;i++) {
            System.out.println("Enter flight number:");
            int fno=sc.nextInt();
            System.out.println("Enter flight name:");
            String fname=sc.next();
            System.out.println("Enter Source of the flight:");
            String source=sc.next();
            System.out.println("Enter the destination:");
            String destination=sc.next();
            System.out.println("Enter the depature time:");
            String dt=sc.next();
            System.out.println("Enter the arrival time:");
            String at=sc.next();
            Flights.add(new AirLine(fno,fname,source,destination,dt,at));
        }
        System.out.println(size+"Flights addedd successfully");
        view();
    }
    public static void view() {
        if(Flights.isEmpty())
            System.out.println("Yet Flights are not added to databases");
        else {
            System.out.println("Flight no | Flight name | Source |
                Destination | Departure Time | Arrival Time | ");
            for(AirLine f:Flights) {
                System.out.println( f.fno+"\t\t"+f.fname+"\t\t"+f.source+"\t\t"+f.destination+
                    "\t\t"+f.departuretime+"\t\t"+f.arrivaltime);
            }
        }
    }

    public static void searchbyno() {
        if(Flights.isEmpty()) {
            System.out.println("Yet Flights are not added to databases");
            add();
        }
        else {
            System.out.println("Enter the flight number to be searched:");
        }
    }
}
```

PRACTICAL ASSESSMENT

```
int fno=sc.nextInt();
for(AirLine f:Flights) {
if(f.fno==fno) {
System.out.println("Flight found");
System.out.println("Flight no | Flight name | Source |
Destination | Departure Time | Arrival Time | ");
System.out.println( f.fno+"\t\t"+f.fname+"\t\t"+f.source+"\t\t"+f.destination+
"\t\t"+f.departuretime+"\t\t"+f.arrivaltime);
}
}

System.out.println("Flight not found");
return;
}
public static void update() {
boolean found=true;
if(Flights.isEmpty())
System.out.println("Yet database are empty Flights are to be added ");
else {
System.out.println("Enter the flight number to be searched:");
int fno=sc.nextInt();
for(AirLine f:Flights) {
if(f.fno==fno) {
System.out.println("Enter fno");
int fnoo=sc.nextInt();
f.fno=fnoo;
System.out.println("Enter fname");
String fname=sc.next();
f.fname=fname;
System.out.println("Enter source");
String source=sc.next();
f.source=source;
System.out.println("Enter destination");
String destination=sc.next();
f.destination=destination;
System.out.println("Enter departure time");
String dt=sc.next();
f.departuretime=dt;
System.out.println("Enter arrival time");
String at=sc.next();
f.arrivaltime=at;
found=false;
}
}
System.out.println("Flight updated successfully!!!!");
view();
}
}
```

PRACTICAL ASSESSMENT

```
if(found) {
System.out.println("Invalid flight number");
return;
}
}
public static void delete() {
boolean found=true;
if(Flights.isEmpty())
System.out.println("Yet database are empty Flights are to be added");
else {
System.out.println("Enter the flight number to be searched:");
int fno=sc.nextInt();
for(AirLine f:Flights) {
if(f.fno==fno) {
Flights.remove(f);
System.out.println("Flight deleted successfully");
found=false;
view();
}
}
}
}

if(found) {
System.out.println("Invalid flight no");
return;
}
}

public static void main(String[] args) {

System.out.println("-----AirLine Management System-----");
while(true) {
System.out.println("1.Add Flights \n2.View all Flights \n3.Search Flight by
Flightnumber \n4.Update Flight \n5.Delete Flight \n6.Exit ");
int choice=sc.nextInt();
switch(choice) {
case 1: add();break;
case 2: view();break;
case 3:searchbyno();break;
case 4:update();break;
case 5:delete();break;
case 6:System.out.println("Exisiting form the process.....GoodBye!!!!\n-----
----- Have a Nice day!!!!-----");
return;
default:System.out.println("Invalid choice number");break;
}
}
}
```

PRACTICAL ASSESSMENT

}

Output

PRACTICAL ASSESSMENT

The screenshot shows the Eclipse IDE interface with two code editors and a terminal window.

Code Editor 1 (Top): Contains the `MainMethod.java` code for Airline Management System. It includes logic for adding flights, searching by flight number, updating, deleting, and viewing all flights. A switch statement handles user input from 1 to 6.

```
1 MainMethod.java X
2
3 1.Add Flights
4 2.View all Flights
5 3.Search Flight by Flightnumber
6 4.Update Flight
7 5.Delete Flight
8 6.Exit
9
10 System.out.println("Yet database are empty Flig
11
12 else {
13     System.out.println("Enter the flight number to l
14     int fno=sc.nextInt();
15     for(Airline f:flights) {
16         if(f.fno==fno) {
17             flights.remove();
18             System.out.println("Flight deleted succ
19             found=false;
20             view();
21         }
22     }
23 }
24
25
26 public static void main(String[] args) {
27
28     System.out.println("-----Airline Management Sy:
29     while(true) {
30         System.out.println("1.Add Flights \n2.View all Flights \n3.Search Flight by Flightnumber \n4.Update Flight \n5.Delete Flight \n6.Exit ");
31         int choice=sc.nextInt();
32         switch(choice) {
33             case 1: add();break;
34             case 2: view();break;
35             case 3: searchbyname();break;
36             case 4: update();break;
37             case 5: delete();break;
38             case 6:System.out.println("Exiting form the process.....GoodBye!!!!\n----- Have a Nice day!!!!-----");
39             return;
40         default:System.out.println("Invalid choice number");break;
41     }
42 }
43
44
45
46 }
47
48 }
```

Code Editor 2 (Bottom): Contains the `MainMethod.java` code for Airline Management System. It includes logic for adding flights, searching by flight number, updating, deleting, and viewing all flights. A switch statement handles user input from 1 to 6.

```
1 MainMethod.java X
2
3 1.Add Flights
4 2.View all Flights
5 3.Search Flight by Flightnumber
6 4.Update Flight
7 5.Delete Flight
8 6.Exit
9
10 import java.util.LinkedList;
11 import java.util.List;
12 import java.util.Scanner;
13
14 public class MainMethod {
15     static Scanner sc=new Scanner(System.in);
16     static List<Airline>flights=new LinkedList();
17
18     public static void add() {
19
20         System.out.println("Enter how many flight want to add in the database");
21         int size=sc.nextInt();
22         for(int i=0;i<size;i++) {
23             System.out.println("Enter flight number:");
24             int fno=sc.nextInt();
25             System.out.println("Enter flight name:");
26             String fname=sc.next();
27             System.out.println("Enter Source of the flight:");
28             String source=sc.next();
29             System.out.println("Enter the destination:");
30             String destination=sc.next();
31             System.out.println("Enter the departure time:");
32             String dtsc.next();
33             System.out.println("Enter the arrival time:");
34             String atsc.next();
35             flights.add(new Airline(fno,fname,source,destination,dt,at));
36
37         }
38         System.out.println(size+"Flights addedd successfully");
39         view();
40     }
41
42     public static void view() {
43         if(flights.isEmpty())
44             System.out.println("Yet Flights are not added to databases");
45
46         else {
47             System.out.println("Flight no | Flight name | Source | Destination | Departure Time | Arrival Time | ");
48             for(Airline f:flights) {
49                 System.out.println( f.fno+"\t"+f.fname+"\t"+f.source+"\t"+f.destination+"\t"+f.departuretime+"\t"+f.arrivaltime);
50             }
51         }
52     }
53 }
```

Terminal Window: Shows the execution of the Java application. The user enters flight details and chooses option 1 to add flights. The application then lists all added flights.

```
MainMethod [Java Application] D:\eclipse-ide-2025-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v20250724-1412\re\bin\javaw.exe (17-N)
-----Airline Management System-----
1.Add Flights
2.View all Flights
3.Search Flight by Flightnumber
4.Update Flight
5.Delete Flight
6.Exit
1
Enter how many flight want to add in the database
1
Enter flight number:
001
Enter flight name:
Kingfisher
Enter Source of the flight:
Chennai
Enter the destination:
Madurai
Enter the depature time:
10am
Enter the arrival time:
12pm
[Flights addedd successfully
Flight no | Flight name | Source | Destination | Departure Time | Arrival Time |
1 | Kingfisher | Chennai | Madurai | 10am | 12pm
1.Add Flights
2.View all Flights
3.Search Flight by Flightnumber
4.Update Flight
5.Delete Flight
6.Exit
```

PRACTICAL ASSESSMENT

eclipse-workspace - PostAssessment/src/MainMethod.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

AirLineJava MainMethod.java

```
59     }
60   }
61 }
62 }  
63 if(Found) []
64 System.out.println("Flight not found");
65 return;
66 }
67 }
68 public static void update() {
69   boolean Found=true;
70   if(Flights.isEmpty())
71     System.out.println("Yet database are empty Flights are to be added ");
72   else
73     System.out.println("Enter the flight number to be searched:");
74   int fno=sc.nextInt();
75   for(Airline f:Flights) {
76     if(f.fno==fno) {
77       System.out.println("Enter fno");
78       int fno=sc.nextInt();
79       f.fno=fno;
80       System.out.println("Enter fname");
81       String fname=sc.next();
82       f.fname=fname;
83       System.out.println("Enter source");
84 }
```

Problems Javadoc Declaration Console × ① Install Java 25 Support ② Eclipse IDE for Java Developers 2025-12 M2
MainMethod [Java Application] D:\eclipse\java-2025-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jst\openjdk\hotspot\jre\full\win32\x86_64_21.0.8.v20250724-1412\jre\bin\javaw.exe (17-Nov-2025, 3:33:42 pm elapsed: 0:00:23) [pid: 28956]

Enter the flight number to be searched:
1 Flight found
Flight no| Flight name | Source | Destination | Departure Time | Arrival Time |
1 Kingfisher | Chennai | Madurai | 10pm | 12pm
1.Add Flights
2.View all Flights
3.Search Flight by Flightnumber
4.Update Flight
5.Delete Flight
6.Exit

eclipse-workspace - PostAssessment/src/MainMethod.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

AirLineJava MainMethod.java

```
43 }
44 public static void searchbyno() {
45   boolean Found=true;
46   if(Flights.isEmpty()) {
47     System.out.println("Yet Flights are not added to databases");
48     add();
49   }
50   else {
51     System.out.println("Enter the flight number to be searched:");
52     int fno=sc.nextInt();
53   for(Airline f:Flights) {
54     if(f.fno==fno) {
55       System.out.println("Flight found");
56       System.out.println("Flight no | Flight name | Source | Destination | Departure Time | Arrival Time |");
57       System.out.println( f.fno+"|"+f.fname+"|"+f.source+"|"+f.destination+"|"+f.departuretime+"|"+f.arrivaltime);
58       found=false;
59     }
60   }
61 }
62 if(found) []
63 System.out.println("Flight not found");
64 return;
65 }
66 }
```

Problems Javadoc Declaration Console × ① Install Java 25 Support ② Eclipse IDE for Java Developers 2025-12 M2
MainMethod [Java Application] D:\eclipse\java-2025-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jst\openjdk\hotspot\jre\full\win32\x86_64_21.0.8.v20250724-1412\jre\bin\javaw.exe (17-Nov-2025, 3:33:42 pm elapsed: 0:00:37) [pid: 28956]

Enter the flight number to be searched:
1 Flight found
Flight no| Flight name | Source | Destination | Departure Time | Arrival Time |
1 Kingfisher | Chennai | Madurai | 10pm | 12pm
1.Add Flights
2.View all Flights
3.Search Flight by Flightnumber
4.Update Flight
5.Delete Flight
6.Exit

PRACTICAL ASSESSMENT

The screenshot shows two instances of the Eclipse IDE interface. Both instances have the same title bar: "eclipse-workspace - PostAssessment/src/MainMethod.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.

The left instance displays the `MainMethod.java` code, which contains methods for updating and deleting flights. The right instance shows the execution output in the Console tab, where the user enters flight details and the program outputs the updated flight information or deletion confirmation.

MainMethod.java Code (Left):

```
public static void update() {
    boolean found=true;
    if(FLights.isEmpty())
        System.out.println("Yet database are empty Flights are to be added ");
    else {
        System.out.println("Enter the flight number to be searched:");
        int fno=sc.nextInt();
        for(AirLine f:FLights) {
            if(f.fno==fno) {
                System.out.println("Enter fname");
                int fno=sc.nextInt();
                f.fno=fno;
                System.out.println("Enter fname");
                String fname=sc.next();
                f.fname=fname;
                System.out.println("Enter source");
                String source=sc.next();
                f.source=source;
                System.out.println("Enter destination");
                String destination=sc.next();
                f.destination=destination;
                System.out.println("Enter departure time");
                String dt=sc.next();
                f.departuretime=dt;
                System.out.println("Enter arrival time");
            }
        }
        System.out.println("Flight updated successfully!!! ");
    }
}
```

Execution Output (Right):

```
Enter fno
AirIndia
Enter source
Madurai
Enter destination
Chennai
Enter departure time
12pm
Enter arrival time
5pm
Flight updated successfully!!!
```

MainMethod.java Code (Second Left):

```
public static void delete() {
    boolean found=true;
    if(FLights.isEmpty())
        System.out.println("Yet database are empty Flights are to be added ");
    else {
        System.out.println("Enter the flight number to be searched:");
        int fno=sc.nextInt();
        for(AirLine f:FLights) {
            if(f.fno==fno) {
                FLights.remove(f);
                System.out.println("Flight deleted successfully");
                found=false;
                view();
            }
        }
    }
    if(found)
        System.out.println("Invalid flight no");
    return;
}
}
```

Execution Output (Second Right):

```
Enter the flight number to be searched:
5
Flight deleted successfully
Yet Flights are not added to databases
1.Add Flights
2.View All Flights
3.Search Flight by Flightnumber
4.Update Flight
5.Delete Flight
6.Exit
```

PRACTICAL ASSESSMENT

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - PostAssessment/src/MainMethod.java - Eclipse IDE
- Toolbar:** Standard Eclipse toolbar with icons for file operations, search, and project management.
- Left Sidebar:** Project Explorer showing two files: AirLine.java and MainMethod.java.
- Central Area:** Code editor containing Java code for an airline management system. The code includes a main method that prints a welcome message, handles user input for flight operations (add, view, search by ID, update, delete), and exits the program.
- Console Tab:** Shows the output of the Java application's execution. It displays the welcome message, lists available operations (Add Flight, View all Flights, Search Flight by Flightnumber, Update Flight, Delete Flight, Exit), and logs that flights are not added to the database. It also shows the exit message "Exiting form the process.....GoodBye!!!! Have a Nice day!!!!".
- Status Bar:** Provides information about the Java environment and the current session.