

## **ОБ'ЄКТНО-ОРІЄНТОВАНА ДЕКОМПОЗИЦІЯ**

**Мета:** Використання об'єктно-орієнтованого підходу для розробки об'єкта предметної (прикладної) галузі.

### **ВИМОГИ**

#### **Розробник:**

- Чугунов Вадим Юрійович;
- КІТ-119а;
- Варіант №24.

#### **Загальне завдання:**

- 1) Використовуючи об'єктно-орієнтований аналіз, реалізувати класи для представлення сутностей відповідно прикладної задачі - domain-об'єктів.
- 2) Забезпечити та продемонструвати коректне введення та відображення кирилиці.
- 3) Продемонструвати можливість управління масивом domain-об'єктів.

#### **Індивідуальне завдання:**

Прикладна галузь - **Автостанція**. Запис в розкладі: номер рейсу; час відправлення; дні тижня; кількість вільних місць; маршрут - необмежений набір значень у вигляді "назва станції, час прибуття".

## ОПИС ПРОГРАМИ

<code>BusStation[] ScheduleEntry;</code>	<code>// масив записів у розкладі</code>
<code>int flightNumber;</code>	<code>//номер рейсу</code>
<code>String departureTime;</code>	<code>// час відправлення</code>
<code>String dayOfTheWeek;</code>	<code>//день тижня</code>
<code>int numberOfFreeSeats;</code>	<code>//кількість вільних місць</code>
<code>String[] route;</code>	<code>// маршрут (станції)</code>

### Ієрархія та структура класів:

**class Main** – точка входу в програму.

**class BusStation** – клас, який реалізує автостанцію, члени класу якого є записи у розкладі.

## ТЕКСТ ПРОГРАМИ

### Текст файлу **Main**:

```
package ua.oop.khpi.chugunov07;

import java.io.IOException;
import java.text.ParseException;

/**
 * Main class contains array of domain objects.
```

```

* Cyrillic input/output is demonstrated by setters and getters.
*
* @author chugunov-vadim
* @version 1.0
*/

public class Main {

    /**
     * Main method - entry point of a program.
     *
     * @param args - command line parameters
     * @throws IOException - if there is any unresolved input or output
     * @throws ParseException
     */

    public static void main(String[] args) throws IOException {

        /* Array of domain objects */

        int countOfEntries = 2;

        BusStation[] ScheduleEntry = new BusStation[countOfEntries];

        ScheduleEntry[0] = new BusStation();

        /* Initialization (cyrillic input) */

        ScheduleEntry[0].setFlightNumber(120);

        ScheduleEntry[0].setDepartureTime("12:00");

        ScheduleEntry[0].setDayOfTheWeek("Вторник");

        ScheduleEntry[0].setNumberOfFreeSeats(35);

        ScheduleEntry[0].enterRoute(2);
    }
}

```

```

/* Cyrillic output */

System.out.println("=====");
System.out.print("Запись в расписании #1\n");
System.out.println("=====");
System.out.println("Номер рейса: "
    +ScheduleEntry[0].getFlightNumber());
System.out.println("Время отправления: "
    + ScheduleEntry[0].getDepartureTime());
System.out.println("День недели: "
    + ScheduleEntry[0].getDayOfTheWeek());
System.out.println("Кол-во свободных мест: "
    + ScheduleEntry[0].getNumberOfFreeSeats());

System.out.print("Маршрут: ");
for (String it : ScheduleEntry[0].getRoute()) {

    System.out.print(it);

    System.out.print("; ");

}

}

}

```

Текст файлу **BusStation**:

```
package ua.oop.khpi.chugunov07;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Arrays;

/**
 * Bus Station class.
 * Class defines the entity of a special task.
 * Schedule entries that contains of flightNumber,departureTime etc.
 */
public class BusStation {

    private int flightNumber;      // A flight number in Schedule

    private String departureTime;  // departure time of the bus

    private String dayOfTheWeek;   // day of the week when the bus travels

    private int numberOfFreeSeats; // count of free seats in bus

    private String[] route;       // The bus route (name of station, arrival
time)

    int size = 0;

    /**
     * The setters of our information variables
     */

    public void setFlightNumber(int flightNumber) {
```

```

        this.flightNumber = flightNumber;
    }

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

    public void setDayOfTheWeek(String dayOfTheWeek) {
        this.dayOfTheWeek = dayOfTheWeek;
    }

    public void setNumberOfFreeSeats(int numberOfFreeSeats) {
        this.numberOfFreeSeats = numberOfFreeSeats;
    }

    public void setRoute(String[] route) {
        this.route = route;
    }

    /**
     * Adding stations in the route.
     * @param stationNum - the number of stations
     * @throws IOException - if there is any unresolved input/output
     */
    public void enterRoute(int stationNum) throws IOException {
        BufferedReader reader = new BufferedReader(
            new InputStreamReader(System.in));
    }

```

```

        this.route = new String[stationNum];

        System.out.println("Введите " + stationNum + " станций");

        String getRouteName;

        for (int i = 0; i < stationNum; i++) {

            System.out.print((i + 1) + ". ");

            getRouteName = reader.readLine();

            this.route[i] = getRouteName;

        }
    }

    /**
     * The getters of our information variables
     */

    public int getFlightNumber() {

        return flightNumber;

    }

    public int getNumberOfFreeSeats() {

        return numberOfFreeSeats;

    }

    public String getDepartureTime() {

        return departureTime;

    }

    public String getDayOfTheWeek() {

        return dayOfTheWeek;

```

```
}
```

```
public String getRouteOnIndex(int index) {  
    return route[index];  
}
```

```
public String[] getRoute () {  
    return route;  
}
```

```
/**
```

```
Default constructor
```

```
*/
```

```
public BusStation() {  
    flightNumber = 0;  
    departureTime = null;  
    dayOfTheWeek = null;  
    numberOfFreeSeats = 0;  
    route = null;  
}
```

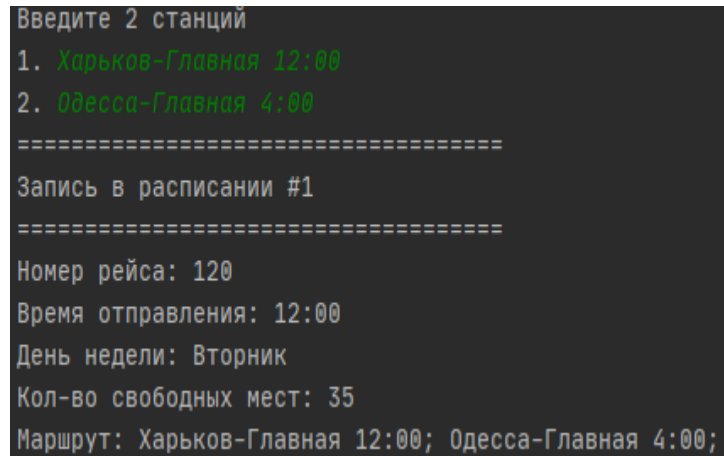
```
@Override
```

```
public String toString() {  
    return "Номер рейса: " + flightNumber + "\n" +  
        "Время отправления: " + departureTime + '\n' +  
        "День недели: " + dayOfTheWeek + '\n' +  
        "Кол-во свободных мест: " + numberOfFreeSeats + '\n' +
```



```
        "Маршрут: " + Arrays.toString(route);  
    }  
  
}
```

## РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ



```
Введите 2 станций  
1. Харьков-Главная 12:00  
2. Одесса-Главная 4:00  
=====  
Запись в расписании #1  
=====  
Номер рейса: 120  
Время отправления: 12:00  
День недели: Вторник  
Кол-во свободных мест: 35  
Маршрут: Харьков-Главная 12:00; Одесса-Главная 4:00;
```

Рисунок 7.1 – Результат виконання завдання

## ВАРІАНТИ ВИКОРИСТАННЯ

Програму можна використовувати для ведення обліку маршрутів у розкладі.

## **ВИСНОВОК**

Під час виконання лабораторної роботи було набуто навичок використання об'єктно орієнтованого підходу для прикладної галузі. Реалізовано класи для представлення сутностей відповідно прикладної задачі (Автостанція) - domain-об'єктів. Продемонстровано коректне введення та виведення кирилиці. Отримано досвід щодо використання об'єктів домену, які показують, як конкретні дані можуть зберігатися разом. Сама вибірка автостанції, що зберігає записи у розкладі з їх особливими характеристиками, представляє систему доменних об'єктів.