Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №4
По дисциплине: «ССП»
Вариант 11

Выполнил:
Студент 3 курса
Группы ПО-8
Замалетдинов Д.А.
Проверил:
Крощенко А.А

Брест 2024

Лабораторная работа №4

Цель работы: приобрести практические навыки в области объектно-ориентированного проектирования.

Задание 1: Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.
2) Создать класс Payment (покупка) с внутренним классом, с помощью объектов которого можно сформировать покупку из нескольких товаров.

Код программы:

Payment.java

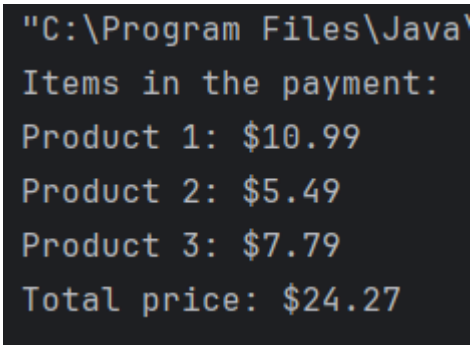
```
import java.util.ArrayList;
import java.util.List;

public class Payment {
    private List<Item> items;

    public Payment() {
        this.items = new ArrayList<>();
    }

    public void addItem(String name, double price) {
        Item item = new Item(name, price);
        items.add(item);
    }

    public double getTotalPrice() {
        double totalPrice = 0;
        for (Item item : items) {
            totalPrice += item.getPrice();
        }
        return totalPrice;
    }

    public void displayItems() {
        System.out.println("Items in the payment:");
        for (Item item : items) {
            System.out.println(item.getName() + ": $" + item.getPrice());
        }
    }


    private class Item {
        private String name;
        private double price;

        public Item(String name, double price) {
            this.name = name;
            this.price = price;
        }

        public String getName() {
            return name;
        }
        public double getPrice() {
            return price;
        }

        public void setName(String name) {
```

```
            this.name = name;
        }

        public void setPrice(double price) {
            this.price = price;
        }
    }

}
```

Main.java

```
public class Main {
    public static void main(String[] args) {
        Payment payment = new Payment();
        payment.addItem("Product 1", 10.99);
        payment.addItem("Product 2", 5.49);
        payment.addItem("Product 3", 7.79);

        payment.displayItems();
        System.out.println("Total price: $" + payment.getTotalPrice());
    }
}
```

Результат работы программы:



Задание 2: Реализовать агрегирование. При создании класса агрегируемый класс объявляется как атрибут (локальная переменная, параметр метода). Включить в каждый класс 2-3 метода на выбор. Продемонстрировать использование разработанных классов. 11) Создать класс Звездная система, используя классы Планета, Звезда.

Код программы:

StarSystem.java

```
import java.util.ArrayList;
import java.util.List;

public class StarSystem {
    private Star star;
    private List<Planet> planets;

    public StarSystem(Star star) {
        this.star = star;
        this.planets = new ArrayList<>();
    }

    public void addPlanet(Planet planet) {
```

```
        planets.add(planet);
    }

    public void displayStarInfo() {
        System.out.println("Star information:");
        star.displayInfo();
    }

    public void displayPlanetsInfo() {
        System.out.println("Planets information:");
        for (Planet planet : planets) {
            planet.displayInfo();
        }
    }
}
```

Planet.java

```
class Planet {
    private String name;
    private double distanceFromStar;

    public Planet(String name, double distanceFromStar) {
        this.name = name;
        this.distanceFromStar = distanceFromStar;
    }

    public void displayInfo() {
        System.out.println("Planet name: " + name);
        System.out.println("Distance from star: " + distanceFromStar + " AU");
    }
}
```

Star.java

```
class Star {
    private String name;
    private String type;

    public Star(String name, String type) {
        this.name = name;
        this.type = type;
    }

    public void displayInfo() {
        System.out.println("Star name: " + name);
        System.out.println("Star type: " + type);
    }
}
```

Main.java

```
public class Main {
    public static void main(String[] args) {
        StarSystem solarSystem = new StarSystem(new Star("Sun", "G-type"));

        Planet mercury = new Planet("Mercury", 0.39);
        Planet venus = new Planet("Venus", 0.72);
        Planet earth = new Planet("Earth", 1);
        Planet mars = new Planet("Mars", 1.52);

        solarSystem.addPlanet(mercury);
        solarSystem.addPlanet(venus);
        solarSystem.addPlanet(earth);
```

```
        solarSystem.addPlanet(mars);

        solarSystem.displayStarInfo();

        solarSystem.displayPlanetsInfo();
    }
}
```

Результаты работы программы:

```
"C:\Program Files\Java\jdk-21
Star information:
Star name: Sun
Star type: G-type
Planets information:
Planet name: Mercury
Distance from star: 0.39 AU
Planet name: Venus
Distance from star: 0.72 AU
Planet name: Earth
Distance from star: 1.0 AU
Planet name: Mars
Distance from star: 1.52 AU
```

Задание 3: Разработать автоматизированную систему на основе некоторой структуры данных, манипулирующей объектами пользовательского класса. Реализовать требуемые функции обработки данных.

11) Система Аэрофлот. Администратор формирует летную Бригаду (пилоты, штурман, радист, стюардессы) на Рейс. Каждый Рейс выполняется Самолетом с определенной вместимостью и дальностью полета. Рейс может быть отменен из-за погодных условий в Аэропорту отлета или назначения. Аэропорт назначения может быть изменен в полете из-за технических неисправностей, о которых сообщил командир.

Код программы:

CrewMember.java

```
import java.lang.String;
class CrewMember {
    private String name;

    public CrewMember(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```

Pilot.java

```
class Pilot extends CrewMember {
    public Pilot(String name) {
        super(name);
    }
}
```

Navigator.java

```
class Navigator extends CrewMember {
    public Navigator(String name) {
        super(name);
    }
}
```

RadioOperator.java

```
class RadioOperator extends CrewMember {
    public RadioOperator(String name) {
        super(name);
    }
}
```

FlightAttendant.java

```
class FlightAttendant extends CrewMember {
    public FlightAttendant(String name) {
        super(name);
    }
}
```

Aircraft.java

```
class Aircraft {

    private String name;
    private int capacity;
    private double range;

    public Aircraft(String name, int capacity, double range) {
        this.name = name;
        this.capacity = capacity;
        this.range = range;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getCapacity() {
        return capacity;
    }

    public double getRange() {
        return range;
    }
```

```
    public void setCapacity(int capacity) {
        this.capacity = capacity;
    }

    public void setRange(double range) {
        this.range = range;
    }

    public void showAircraftInfo(){
        System.out.println(name + ":\nДальность полёта: " + capacity +
"\nВместительность: " + range);
    }

}
```

Flight.java

```
import java.util.ArrayList;
import java.util.List;

class Flight {

    private List<CrewMember> crew;
    private Aircraft aircraft;
    private String departureAirport;
    private String destinationAirport;
    private boolean canceled;

    public Flight(Aircraft aircraft, String departureAirport, String
destinationAirport) {
        this.aircraft = aircraft;
        this.departureAirport = departureAirport;
        this.destinationAirport = destinationAirport;
        this.crew = new ArrayList<>();
        this.canceled = false;
    }

    public String getDepartureAirport() {
        return departureAirport;
    }

    public String getDestinationAirport() {
        return destinationAirport;
    }

    public boolean isCanceled() {
        return canceled;
    }

    public List<CrewMember> getCrew() {
        return crew;
    }

    public void addCrewMember(CrewMember crewMember) {
        crew.add(crewMember);
    }

    public void cancelFlight() {
        this.canceled = true;
        System.out.println("\nРейс отменён!");
    }

    public void changeDestination(String newDestinationAirport) {
        this.destinationAirport = newDestinationAirport;
    }

    public void changeDeparture(String newDepartureAirport) {
```

```
        this.departureAirport = newDepartureAirport;
    }

    public void displayFlightInfo() {
        System.out.println("Экипаж:");
        for (CrewMember member : crew) {
            System.out.println(member.getClass().getName() + ": " + member.getName());
        }

        System.out.println("");

        aircraft.showAircraftInfo();

        System.out.println("\n" + departureAirport + " -> " + destinationAirport);
    }
}
```

Main.java

```
public class Main {

    public static void main(String[] args) {
        Pilot pilot = new Pilot("Фролов Михаил Дмитриевич");
        Navigator navigator = new Navigator("Дроздов Тимофей Андреевич");
        RadioOperator radioOperator = new RadioOperator("Беляев Александр Фёдорович");
        FlightAttendant flightAttendant1 = new FlightAttendant("Попова Яна Юрьевна");
        FlightAttendant flightAttendant2 = new FlightAttendant("Шилова Полина
Львовна");

        Aircraft aircraft = new Aircraft("Boeing-737",150, 5000);

        Flight flight = new Flight(aircraft, "Аэропорт Минск", "Аэропорт Шереметьево");

        flight.addCrewMember(pilot);
        flight.addCrewMember(navigator);
        flight.addCrewMember(radioOperator);
        flight.addCrewMember(flightAttendant1);
        flight.addCrewMember(flightAttendant2);

        flight.displayFlightInfo();

        flight.cancelFlight();

        flight.changeDestination("Аэропорт Калуга");

        flight.displayFlightInfo();
    }
}
```

Результаты работы программы:

```
"C:\Program Files\Java\jdk-21\bin\java.exe
Экипаж:
Pilot: Фролов Михаил Дмитриевич
Navigator: Дроздов Тимофей Андреевич
RadioOperator: Беляев Александр Фёдорович
FlightAttendant: Попова Яна Юрьевна
FlightAttendant: Шилова Полина Львовна

Boeing-737:
Дальность полёта: 150
Вместительность: 5000.0

Аэропорт Минск -> Аэропорт Шереметьево

Рейс отменён!
Экипаж:
Pilot: Фролов Михаил Дмитриевич
Navigator: Дроздов Тимофей Андреевич
RadioOperator: Беляев Александр Фёдорович
FlightAttendant: Попова Яна Юрьевна
FlightAttendant: Шилова Полина Львовна

Boeing-737:
Дальность полёта: 150
Вместительность: 5000.0

Аэропорт Минск -> Аэропорт Калуга
```

Вывод: приобрел практические навыки в области объектно-ориентированного проектирования.