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

Лабораторная работа №5
По дисциплине: «СПП»
Вариант 11

Выполнил:
Студент 3 курса
Группы ПО-9
Лебедович В.А.
Проверил:
Крощенко А.А

Брест 2024

Лабораторная работа №5

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

Задание 1: Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов: interface Устройство Печати ← class Принтер ← class Лазерный Принтер.

Код программы: PrintingDevice.java

```
public interface PrintingDevice {
public void printData();
    public void setData(String data); }
```

Printer.java

```
public class Printer implements PrintingDevice{

    protected String data;
    @Override
    public void printData() {
        System.out.println("Принтер печатает текст: \n" + data);
}

    @Override
    public void setData(String data) {
this.data = data;
    }
}
```

LaserPrinter.java

```
public class LaserPrinter extends Printer{
    @Override
    public void printData() {
        System.out.println("Лазерный принтер печатает текст: \n" + data);
    }
}
```

Main.java

```
public class Main {
    public static void main(String[] args) {
        Printer printer = new Printer();
        Printer laserPrinter = new LaserPrinter();

        printer.setData("Что вершит судьбу человечества?");
laserPrinter.setData("Где моя медаль и грамота?");

        printer.printData();
laserPrinter.printData();
    }
}
```

Результат работы программы:

```
Принтер печатает текст:
Что вершит судьбу человечества?
Лазерный принтер печатает текст:
Где моя медаль и грамота?
```

Задание 2: В следующих заданиях требуется создать суперкласс (абстрактный класс, интерфейс) и определить общие методы для данного класса. Создать подклассы, в которых добавить специфические свойства и методы. Часть методов переопределить. Создать массив объектов суперкласса и заполнить объектами подклассов. Объекты подклассов идентифицировать конструктором по имени или идентификационному номеру. Использовать объекты подклассов для моделирования реальных ситуаций и объектов.

Создать суперкласс Музыкальный инструмент и классы Ударный, Струнный, Духовой. Создать массив объектов Оркестр. Осуществить вывод состава оркестра.

Код программы:

MusicalInstrument.java

```
abstract class MusicalInstrument {
protected String name;
    public MusicalInstrument(String name) {
this.name = name;
    }
    public abstract void play();
}
```

Percussion.java

```
class Percussion extends MusicalInstrument {
private String drumType;

    public Percussion(String name, String drumType) {
super(name);
        this.drumType = drumType;
    }

    @Override
    public void play() {
        System.out.println(name + " (type: " + drumType + ") is being played.");
    }
}
```

Stringed.java

```
class Stringed extends MusicalInstrument {
private int numberOfStrings;
    public Stringed(String name, int numberOfStrings) {
super(name);
        this.numberOfStrings = numberOfStrings;
}     @Override
```

```
    public void play() {
        System.out.println(name + " (number of strings: " + numberOfStrings + ") is
being played.");
    }
```
} **Wind.java**

```
class Wind extends MusicalInstrument {
private String material;

    public Wind(String name, String material) {
super(name);
        this.material = material;
    }

    @Override
    public void play() {
        System.out.println(name + " (made of " + material + ") is being played.");
    }
}
```

Orchestra.java

```
class Orchestra {
    private MusicalInstrument[] instruments;

    public Orchestra(int size) {
        instruments = new MusicalInstrument[size];
    }
    public void addInstrument(int index, MusicalInstrument instrument) {
instruments[index] = instrument;
    }
    public void displayOrchestra() {
        System.out.println("Orchestra composition:");
for (MusicalInstrument instrument : instruments) {
if (instrument != null) {
System.out.print("- ");                 instrument.play();
            }
        }
    }
```
} **Main.java**

```
public class Main {
    public static void main(String[] args) {
Orchestra orchestra = new Orchestra(5);
         orchestra.addInstrument(0, new Percussion("Drums", "Bass"));
orchestra.addInstrument(1, new Stringed("Guitar", 6));
orchestra.addInstrument(2, new Wind("Flute", "Wood"));
orchestra.addInstrument(3, new Percussion("Cymbals", "Crash"));
orchestra.addInstrument(4, new Stringed("Violin", 4));

orchestra.displayOrchestra();
    }
}
```

Результаты работы программы:

```
Orchestra composition:
- Drums (type: Bass) is being played.
- Guitar (number of strings: 6) is being played.
- Flute (made of Wood) is being played.
- Cymbals (type: Crash) is being played.
- Violin (number of strings: 4) is being played.
```

Задание 3: В задании 3 ЛР №4, где возможно, заменить объявления суперклассов объявлениями абстрактных классов или интерфейсов.

Код программы:

CrewMember.java

```
import java.lang.String; abstract
class CrewMember {

    protected String name;

    public CrewMember(String name) {
this.name = name;
    }
    public String getName() {
return name;
    }
    public void setName(String name) {
this.name = name;
    }
    public abstract String getInfo();
}
```

Pilot.java

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

    @Override
    public String getInfo() {
return "Пилот: " + this.name;
    }
}
```

Navigator.java

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

    @Override
    public String getInfo() {
return "Штурман: " + this.name;     }
```

```
}
```

RadioOperator.java

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

    @Override
    public String getInfo() {
return "Радист: " + this.name;
    }
}
```

FlightAttendant.java

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

    @Override
    public String getInfo() {
        return "Стюардесса: " + this.name;
    }
}
```

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.getInfo());
        }

        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:

```
Экипаж:
Пилот: Фролов Михаил Дмитриевич
Штурман: Дроздов Тимофей Андреевич
Радист: Беляев Александр Фёдорович
Стюардесса: Попова Яна Юрьевна
Стюардесса: Шилова Полина Львовна

Boeing-737:
Дальность полёта: 150
Вместительность: 5000.0

Аэропорт Минск -> Аэропорт Шереметьево

Рейс отменён!
Экипаж:
Пилот: Фролов Михаил Дмитриевич
Штурман: Дроздов Тимофей Андреевич
Радист: Беляев Александр Фёдорович
Стюардесса: Попова Яна Юрьевна
Стюардесса: Шилова Полина Львовна

Boeing-737:
Дальность полёта: 150
Вместительность: 5000.0

Аэропорт Минск -> Аэропорт Калуга
```

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