

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Отчёт
по лабораторной работе №4

Выполнил:
студент группы ПО-9
Ступак Д.Р

Проверил:
Крощенко А. А.

Брест 2024

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

Вариант 7

Задание 1

Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов. Создать класс City (город) с внутренним классом, с помощью объектов которого можно хранить информацию о проспектах, улицах, площадях.

Код программы (файл City.java)

```
package Lab4_1;

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

public class City {

    private String name;
    private List<Address> addresses;

    public City(String name) {
        this.name = name;
        this.addresses = new ArrayList<>();
    }

    public void addAddress(Address address) {
        addresses.add(address);
    }

    public List<Address> getAddresses() {
        return addresses;
    }

    @Override
    public String toString() {
        return "Город: " + name + "\n" +
            "Адреса:\n" +
            addresses;
    }

    public static class Address {

        private String type;
        private String name;

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

        @Override
        public String toString() {
            return type + " " + name;
        }
    }
}
```

Код программы (файл Lab4_1.java)

```
package Lab4_1;

public class Lab4_1 {

    public static void main(String[] args) {
        City city = new City("Москва");
    }
}
```

```

        city.addAddress(new City.Address("проспект", "Мира"));
        city.addAddress(new City.Address("улица", "Ленина"));
        city.addAddress(new City.Address("площадь", "Красная"));

        System.out.println(city);
    }
}

```

Вывод

```

Город: Москва
Адреса:
[проспект Мира, улица Ленина, площадь Красная]

```

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.

Создать класс Страница, используя классы Строка, Слово.

Код программы (Page.java)

```

package Lab4_2;

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

public class Page {

    private List<Line> lines;

    public Page() {
        this.lines = new ArrayList<>();
    }

    public void addLine(Line line) {
        lines.add(line);
    }

    public List<Line> getLines() {
        return lines;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        for (Line line : lines) {
            sb.append(line).append("\n");
        }
        return sb.toString();
    }

    public static class Line {

        private List<Word> words;

        public Line() {
            this.words = new ArrayList<>();
        }

        public void addWord(Word word) {

```

```

        words.add(word);
    }

    public List<Word> getWords() {
        return words;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        for (Word word : words) {
            sb.append(word).append(" ");
        }
        return sb.toString().trim();
    }
}

public static class Word {

    private String text;

    public Word(String text) {
        this.text = text;
    }

    @Override
    public String toString() {
        return text;
    }
}
}

```

Код программы (Lab4_2.java)

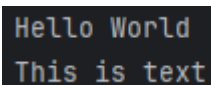
```

public class Lab4_2 {

    public static void main(String[] args) {
        Page page = new Page();
        Page.Line line1 = new Page.Line(), line2 = new Page.Line(), line3 = new Page.Line();
        line1.addWord(new Page.Word("Hello"));
        line1.addWord(new Page.Word("World"));
        line2.addWord(new Page.Word("This"));
        line2.addWord(new Page.Word("is"));
        line2.addWord(new Page.Word("text"));
        page.addLine(line1);
        page.addLine(line2);
        page.addLine(line3);
        System.out.println(page);
    }
}

```

Вывод



```

Hello World
This is text

```

Задание 3

Построить модель программной системы с применением отношений (обобщения, агрегации, ассоциации, реализации) между классами. Задать атрибуты и методы классов. Реализовать (если необходимо) дополнительные классы. Продемонстрировать работу разработанной системы.

Система Автобаза. **Диспетчер** распределяет заявки на **Рейсы** между **Водителями** и назначает для этого **Автомобиль**. **Водитель** может сделать заявку на ремонт. **Диспетчер** может отстранить **Водителя** от работы. **Водитель** делает отметку о выполнении **Рейса** и состоянии **Автомобиля**.
Код программы(Car.java)

```
package Lab4_3;

public class Car {

    private String model;
    private String licensePlate;
    private boolean available;

    public Car(String model, String licensePlate) {
        this.model = model;
        this.licensePlate = licensePlate;
        this.available = true;
    }

    public String getModel() {
        return model;
    }

    public String getLicensePlate() {
        return licensePlate;
    }

    public boolean isAvailable() {
        return available;
    }

    public void setAvailable(boolean available) {
        this.available = available;
    }

    @Override
    public String toString() {
        return "Автомобиль: " + model + " (" + licensePlate + ")";
    }
}
```

Код программы(Driver.java)

```
package Lab4_3;

public class Driver {

    private String name;
    private boolean available;

    private Trip trip;
    public Driver(String name) {
        this.name = name;
        this.available = true;
    }
    public void sendCarToRepair(){
        if (trip !=null){
            trip.getCar().setAvailable(false);
        }
    }

    public void setTrip(Trip trip){
        this.trip=trip;
    }
    public void markTripCompleted() {
        if (trip !=null){
            trip.setStatus(true);
            this.available=true;
        }
    }
}
```

```

        trip.getCar().setAvailable(true);
        return;
    }
    System.out.println("У водителя нет рейса");
}
public String getName() {
    return name;
}

public boolean isAvailable() {
    return available;
}

public void setAvailable(boolean available) {
    this.available = available;
}

@Override
public String toString() {
    return "Водитель: " + name;
}
}

```

Код программы(Trip.java)

```

package Lab4_3;

public class Trip {

    private String destination;
    private Car car;
    private Driver driver;
    private boolean isEnd;

    public Trip(String destination, Car car, Driver driver) {
        this.destination = destination;
        this.car = car;
        this.driver = driver;
        this.isEnd = false;
    }
    public boolean getStatus() {
        return isEnd;
    }

    public void setStatus(boolean isEnd) {
        this.isEnd = isEnd;
    }
    public String getDestination() {
        return destination;
    }

    public Car getCar() {
        return car;
    }

    public Driver getDriver() {
        return driver;
    }

    @Override
    public String toString() {
        return "Рейс: " + destination + " (" + car + ", " + driver + ")";
    }

    public void setCar(Car car) {
        this.car = car;
    }
}

```

```

        public void setDriver(Driver driver) {
            this.driver = driver;
        }
    }
}

```

Код программы(Dispatcher.java)

```

package Lab4_3;

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

public class Dispatcher {

    private List<Car> cars;
    private List<Driver> drivers;
    private List<Trip> trips;

    public Dispatcher() {
        this.cars = new ArrayList<>();
        this.drivers = new ArrayList<>();
        this.trips = new ArrayList<>();
    }

    public void addCar(Car car) {
        cars.add(car);
    }

    public void addDriver(Driver driver) {
        drivers.add(driver);
    }

    public void addTrip(Trip trip) {
        trips.add(trip);
    }

    public void assignTrip(Trip trip) {
        if (!(trip.getStatus())) {
            Car car = findAvailableCar();
            Driver driver = findAvailableDriver();
            driver.setTrip(trip);
            trip.setCar(car);
            trip.setDriver(driver);
            car.setAvailable(false);
            driver.setAvailable(false);
            System.out.println("На рейс назначен: " + driver.toString() + " " + car.toString());
        }
    }

    private Car findAvailableCar() {
        for (Car car : cars) {
            if (car.isAvailable()) {
                return car;
            }
        }
        return null;
    }

    private Driver findAvailableDriver() {
        for (Driver driver : drivers) {
            if (driver.isAvailable()) {
                return driver;
            }
        }
        return null;
    }
}

```

```

    public void removeDriver(Driver driver) {
        drivers.remove(driver);
    }

    public List<Car> getAvailableCars() {
        List<Car> result = new ArrayList<Car>();
        for (Car car : cars){
            if (car.isAvailable())
                result.add(car);
        }
        return result;
    }

    public List<Driver> getAvailableDrivers() {
        List<Driver> result = new ArrayList<Driver>();
        for (Driver driver : drivers){
            if (driver.isAvailable())
                result.add(driver);
        }
        return result;
    }
}

```

Код программы(Lab5_3.java)

```

package Lab4_3;
public class Lab4_3 {

    public static void main(String[] args) {
        Dispatcher dispatcher = new Dispatcher();
        Car car1 = new Car("Lada Granta", "A1234");
        Car car2 = new Car("Toyota Camry", "B5678");
        dispatcher.addCar(car1);
        dispatcher.addCar(car2);

        Driver driver1 = new Driver("Иван");
        Driver driver2 = new Driver("Петр");
        dispatcher.addDriver(driver1);
        dispatcher.addDriver(driver2);

        Trip trip = new Trip("Москва", null, null);

        dispatcher.addTrip(trip);
        dispatcher.assignTrip(trip);
        System.out.println(trip);
        dispatcher.removeDriver(driver2);
        driver1.markTripCompleted();
        for (Car car : dispatcher.getAvailableCars()) {
            System.out.println(car);
        }
        for (Driver driver : dispatcher.getAvailableDrivers()) {
            System.out.println(driver);
        }
    }
}

```

Вывод:

На рейс назначен: Водитель: Иван Автомобиль: Lada Granta (A1234)
Рейс: Москва (Автомобиль: Lada Granta (A1234), Водитель: Иван)
Автомобиль: Lada Granta (A1234)
Автомобиль: Toyota Camry (B5678)
Водитель: Иван