МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
КАФЕДРА ИИТ

ОТЧЁТ

по лабораторной работе №4

Выполнил:

Студент 3 курса
группы ПО-9
Харитонович Захар Сергеевич

Проверил:

Крощенко А. А.

Брест 2024

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

Вариант 10

Задание 1. Создать класс Notepad (записная книжка) с внутренним классом или классами, с помощью объектов которого могут храниться несколько записей на одну дату.

Код программы.

Класс записной книжки.

```
public class Notepad {
    private static final String VALUES_DELIMITER = "\n-------------------\n";

    private final List<Note> notes = new ArrayList<>();

    public boolean add(LocalDate date, String value) {
        return notes.add(new Note(date, value));
    }
    public String getValuesByDate(LocalDate dateFilter) {
        return "Записи на " + dateFilter + "\n" + notes.stream().filter(obj -
>
obj.getDate().equals(dateFilter)).map(Note::getValue).collect(Collectors.join
ing(VALUES_DELIMITER)) + "\n" + dateFilter;
    }

    private class Note {
        private LocalDate date;
        private String value;

        public Note(LocalDate date, String value) {
            this.date = date;
            this.value = value;
        }

        public LocalDate getDate() {
            return date;
        }

        public void setDate(LocalDate date) {
            this.date = date;
        }

        public String getValue() {
            return value;
        }

        public void setValue(String value) {
            this.value = value;
        }
    }

}
```

Примеры использования.

```
public class Task1 {
    public static void main(String[] args) {
        Notepad notepad = new Notepad();
        notepad.add(LocalDate.now(), "Запись в записной книжке номер 1");
        notepad.add(LocalDate.now(), "Запись в записной книжке номер 2");
        notepad.add(LocalDate.now(), "Запись в записной книжке номер 3");
        notepad.add(LocalDate.of(2024, 3, 10), "Запись в записной книжке
номер 1");
```

```
        notepad.add(LocalDate.of(2024, 3, 10), "Запись в записной книжке
номер 2");

        System.out.println(notepad.getValuesByDate(LocalDate.now()));
        System.out.println();
        System.out.println(notepad.getValuesByDate(LocalDate.of(2024,       3,
10)));
    }
}
```

Результат работы.

```
Записи на 2024-04-10
Запись в записной книжке номер 1
-------------------
Запись в записной книжке номер 2
-------------------
Запись в записной книжке номер 3
2024-04-10

Записи на 2024-03-10
Запись в записной книжке номер 1
-------------------
Запись в записной книжке номер 2
2024-03-10
```

Задание 2. Создать класс Планета, используя класс Материк.
Код программы.
Класс планеты.
```
public class Planet {

    private final List<Continent> continents = new ArrayList<>();

    public boolean addContinent(Continent continent) {
        return continents.add(continent);
    }

    public List<Continent> getContinents() {
        return continents;
    }

    public double getTotalSize() {
        return continents.stream().mapToDouble(Continent::getSize).sum();
    }

    public Continent getContinentByName(String nameFilter) {
        return                continents.stream().filter(cont                ->
cont.getName().equalsIgnoreCase(nameFilter))
                .findFirst().orElse(null);
    }
}
```
Класс материка.
```
public class Continent {

    private String name;
    private double size;

    public Continent(String name, double size) {
        this.name = name;
        this.size = size;
    }

    public String getName() {
        return name;
    }

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

    public double getSize() {
        return size;
    }

    public void setSize(double size) {
        this.size = size;
    }

    @Override
    public String toString() {
        return "Continent{" +
                "name='" + name + '\'' +
                ", size=" + size +
                '}';
    }
}
```

Примеры использования.

```
public class Task2 {
    public static void main(String[] args) {
        Planet planet = new Planet();

        planet.addContinent(new Continent("Евразия", 54.6));
        planet.addContinent(new Continent("Африка", 30.3));
        planet.addContinent(new Continent("Северная Америка", 24.4));
        planet.addContinent(new Continent("Южная Америка", 17.8));
        planet.addContinent(new Continent("Антарктида", 14.1));
        planet.addContinent(new Continent("Австралия", 7.7));

        System.out.println("Общая площадь: " + planet.getTotalSize());
        System.out.println("Континент по запросу 'евразия': " +
planet.getContinentByName("евразия"));
    }
}
```

Результат работы.

```
Общая площадь: 148.9
Континент по запросу 'евразия': Continent{name='Евразия', size=54.6}
```

Задание 3. Система Городской транспорт. На Маршрут назначаются Автобус или Троллейбус. Транспортные средства должны двигаться с определенным для каждого Маршрута интервалом.
При поломке на Маршрут должен выходить резервный транспорт или увеличиваться интервал движения.

Код программы.

Класс планеты.

```
public class Planet {

    private final List<Continent> continents = new ArrayList<>();

    public boolean addContinent(Continent continent) {
        return continents.add(continent);
    }

    public List<Continent> getContinents() {
        return continents;
    }

    public double getTotalSize() {
        return continents.stream().mapToDouble(Continent::getSize).sum();
    }

    public Continent getContinentByName(String nameFilter) {
        return                continents.stream().filter(cont                ->
cont.getName().equalsIgnoreCase(nameFilter))
                .findFirst().orElse(null);
    }
}
```

Класс материка.

```
public class Continent {

    private String name;
    private double size;

    public Continent(String name, double size) {
        this.name = name;
        this.size = size;
    }

    public String getName() {
        return name;
    }

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

    public double getSize() {
        return size;
    }

    public void setSize(double size) {
        this.size = size;
    }

    @Override
    public String toString() {
        return "Continent{" +
```

```
                "name='" + name + '\'' +
                ", size=" + size +
                '}';
    }
}
```

Примеры использования.

```
public class Task2 {
    public static void main(String[] args) {
        Planet planet = new Planet();

        planet.addContinent(new Continent("Евразия", 54.6));
        planet.addContinent(new Continent("Африка", 30.3));
        planet.addContinent(new Continent("Северная Америка", 24.4));
        planet.addContinent(new Continent("Южная Америка", 17.8));
        planet.addContinent(new Continent("Антарктида", 14.1));
        planet.addContinent(new Continent("Австралия", 7.7));

        System.out.println("Общая площадь: " + planet.getTotalSize());
        System.out.println("Континент по запросу 'евразия': " +
planet.getContinentByName("евразия"));
    }
}
```

Результат работы.

```
Общая площадь: 148.9
Континент по запросу 'евразия': Continent{name='Евразия', size=54.6}
```

Задание 3. Система Городской транспорт. На Маршрут назначаются Автобус или Троллейбус. Транспортные средства должны двигаться с определенным для каждого Маршрута интервалом. При поломке на Маршрут должен выходить резервный транспорт или увеличиваться интервал движения.

Класс «Городской транспорт»

```
public class CityTransport {

    private final List<Route> routes = new ArrayList<>();
    private final List<Vehicle> freeVehicles = new ArrayList<>();

    public boolean addRoute(Route route) {
        routes.add(route);
        return (route.assignVehicle(freeVehicles));
    }

    public boolean addVehicle(Vehicle vehicle) {
        return freeVehicles.add(vehicle);
    }

    public void update() {
        for (Route route : routes) {
            if (!route.update()) {
                route.assignVehicle(freeVehicles);
            }
        }
    }

}
```

Класс «Маршрут»

```
public class Route {

    private final String name;

    private final VehicleTypeEnum vehicleType;
    private int interval;
    private int currentInterval;
    private Vehicle vehicle;

    public Route(String name, VehicleTypeEnum vehicleType, int interval) {
        this.name = name;
        this.vehicleType = vehicleType;
        this.interval = interval;
        currentInterval = 0;
    }

    public boolean update() {
        currentInterval--;

        if (vehicle == null) {
            System.out.println("[" + name + " / " + interval + "]: нет
транспорта.");
            return false;
        }

        vehicle.move();

        if (vehicle.isBroken()) {
            System.out.println("[" + name + " / " + interval + "]: " +
vehicle.getName() + " сломан.");
            return false;
        }

        if (currentInterval <= 0) {
            currentInterval = interval;
            System.out.println("[" + name + " / " + interval + "]: " +
vehicle.getName() + " начал движение по маршруту.");
        }
        return true;
    }

    public boolean assignVehicle(List<Vehicle> freeVehicles) {
        Vehicle vehicle = freeVehicles.stream()
                .filter(v -> v.getVehicleType().equals(vehicleType))
                .findFirst().orElse(null);
        if (vehicle != null) {
            freeVehicles.remove(vehicle);
            this.vehicle = vehicle;
            System.out.println("[" + name + " / " + interval + "]: " +
vehicle.getName() + " назначен на маршрут.");
            return true;
        } else {
            this.vehicle = null;
            interval++;
            currentInterval++;
            return false;
        }
    }
}
```

Класс «Транспортное средство»

```
public class Vehicle {

    private final VehicleTypeEnum vehicleType;

    private final String name;
    private final double breakProbability;
    private boolean isBroken = false;

    public Vehicle(VehicleTypeEnum vehicleType, String name, double
breakProbability) {
        this.vehicleType = vehicleType;
        this.name = name;
        this.breakProbability = breakProbability;
    }

    public void move() {
        isBroken = getRandomBoolean(breakProbability);
    }

    public VehicleTypeEnum getVehicleType() {
        return vehicleType;
    }

    public String getName() {
        return name;
    }

    public boolean isBroken() {
        return isBroken;
    }

    public static boolean getRandomBoolean(double probability) {
        double randomValue = Math.random();
        return randomValue <= probability;
    }
}
```

Перечисление «Тип транспорта»

```
public enum VehicleTypeEnum {
    BUS,
    TROLLEYBUS
}
```

Пример использования

```
public class Task3 {
    public static void main(String[] args) {
        CityTransport cityTransport = new CityTransport();

        cityTransport.addVehicle(new      Vehicle(VehicleTypeEnum.TROLLEYBUS,
"Троллейбус №1", 0.30));
        cityTransport.addVehicle(new      Vehicle(VehicleTypeEnum.TROLLEYBUS,
"Троллейбус №2", 0.30));
        cityTransport.addVehicle(new      Vehicle(VehicleTypeEnum.TROLLEYBUS,
"Троллейбус №3", 0.00));
        cityTransport.addVehicle(new      Vehicle(VehicleTypeEnum.TROLLEYBUS,
"Троллейбус №4", 0.00));
        cityTransport.addVehicle(new   Vehicle(VehicleTypeEnum.BUS,   "Автобус
№1", 0.30));
```

```
        cityTransport.addVehicle(new   Vehicle(VehicleTypeEnum.BUS,   "Автобус
№2", 0.30));
        cityTransport.addVehicle(new   Vehicle(VehicleTypeEnum.BUS,   "Автобус
№3", 0.00));
        cityTransport.addVehicle(new   Vehicle(VehicleTypeEnum.BUS,   "Автобус
№4", 0.00));

        cityTransport.addRoute(new     Route("Автобусный     маршрут     №1",
VehicleTypeEnum.BUS, 10));
        cityTransport.addRoute(new     Route("Автобусный     маршрут     №2",
VehicleTypeEnum.BUS, 5));
        cityTransport.addRoute(new    Route("Троллейбусный    маршрут    №1",
VehicleTypeEnum.TROLLEYBUS, 4));
        cityTransport.addRoute(new    Route("Троллейбусный    маршрут    №2",
VehicleTypeEnum.TROLLEYBUS, 6));

        for (int i = 1; i <= 15; i++) {
            System.out.println("\n------------------\n" + "Итерация " + i +
"\n------------------\n");
            cityTransport.update();
        }
    }
}
```

Результат работы:

```
[Автобусный маршрут №1 / 10]: Автобус №1 назначен на маршрут.
[Автобусный маршрут №2 / 5]: Автобус №2 назначен на маршрут.
[Троллейбусный маршрут №1 / 4]: Троллейбус №1 назначен на маршрут.
[Троллейбусный маршрут №2 / 6]: Троллейбус №2 назначен на маршрут.
------------------
Итерация 1
------------------
[Автобусный маршрут №1 / 10]: Автобус №1 начал движение по маршруту.
[Автобусный маршрут №2 / 5]: Автобус №2 сломан.
[Автобусный маршрут №2 / 5]: Автобус №3 назначен на маршрут.
[Троллейбусный маршрут №1 / 4]: Троллейбус №1 сломан.
[Троллейбусный маршрут №1 / 4]: Троллейбус №3 назначен на маршрут.
[Троллейбусный маршрут №2 / 6]: Троллейбус №2 сломан.
[Троллейбусный маршрут №2 / 6]: Троллейбус №4 назначен на маршрут.
------------------
Итерация 2
------------------
[Автобусный маршрут №1 / 10]: Автобус №1 сломан.
[Автобусный маршрут №1 / 10]: Автобус №4 назначен на маршрут.
[Автобусный маршрут №2 / 5]: Автобус №3 начал движение по маршруту.
[Троллейбусный маршрут №1 / 4]: Троллейбус №3 начал движение по маршруту.
[Троллейбусный маршрут №2 / 6]: Троллейбус №4 начал движение по маршруту.
------------------
Итерация 3
------------------
------------------
Итерация 4
------------------
------------------
Итерация 5
------------------
------------------
Итерация 6
------------------
[Троллейбусный маршрут №1 / 4]: Троллейбус №3 начал движение по маршруту.
```

```
------------------
Итерация 7
------------------
[Автобусный маршрут №2 / 5]: Автобус №3 начал движение по маршруту.
------------------
Итерация 8
------------------
[Троллейбусный маршрут №2 / 6]: Троллейбус №4 начал движение по маршруту.
------------------
Итерация 9
------------------
------------------
Итерация 10
------------------
[Троллейбусный маршрут №1 / 4]: Троллейбус №3 начал движение по маршруту.
------------------
Итерация 11
------------------
[Автобусный маршрут №1 / 10]: Автобус №4 начал движение по маршруту.
------------------
Итерация 12
------------------
[Автобусный маршрут №2 / 5]: Автобус №3 начал движение по маршруту.
------------------
Итерация 13
------------------
------------------
Итерация 14
------------------
[Троллейбусный маршрут №1 / 4]: Троллейбус №3 начал движение по маршруту.
[Троллейбусный маршрут №2 / 6]: Троллейбус №4 начал движение по маршруту.
------------------
Итерация 15
```

Вывод: в ходе выполнения лабораторной работы приобретены практические навыки в области объектно-ориентированного проектирования.