МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
КАФЕДРА ИИТ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

Выполнил:
Студент группы АС-50
ФЭИС, Левкович Р.А.
Проверил:
Крощенко А.А.

Брест 2020

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

Вариант 3

Задание 1: Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы.

Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.

3) Создать класс Account (счет) с внутренним классом, с помощью объектов которого можно

хранить информацию обо всех операциях со счетом (снятие, платежи, поступления).

Код программы:

```
package com.company;

import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;

class Account{

    class Location{
        private int id;
        private String name;
        private String address;

        public Location(int id, String name, String address){
            this.id = id;
            this.name = name;
            this.address = address;
        }

        public void print(){
            System.out.println("id: "+id+"; название: "+name+"; адрес:
"+address);
        }

        public Location clone(){
            return new Location(id, name, address);
        }
    }

    class Operation{
        private int type;
        private LocalDateTime time;
        private double sum;
        private Location location;

        public Operation(int type, double sum, Location location){
            this.type = type;
            time = LocalDateTime.now();
            this.sum=sum;
            this.location = location;
```

```
        }

        public void print(){
            System.out.print("Тип операции: ");
            if(type==0)
                System.out.print("поступление");
            else if(type==1)
                System.out.print("снятие");
            else if(type==2)
                System.out.print("платёж");

            System.out.print("; время:
"+time.format(DateTimeFormatter.ofPattern("dd/MM/YYYY HH:mm:ss"))+"; сумма:
"+sum);

            if(type==0)
                System.out.print("; источник: ");
            else if(type==1)
                System.out.print("; место операции: ");
            else if(type==2)
                System.out.print("; получатель: ");

            System.out.println(location.name);
        }
    }

    String accountId;
    double money = 0;
    private ArrayList<Location> locations;
    private ArrayList<Operation> operations;

    public Account(String accountId){
        this.accountId = accountId;

        locations = new ArrayList<Location>();
        operations = new ArrayList<Operation>();

        locations.add(new Location(0,"ООО Пищевые Продукты", "ул. Советская,
54"));
        locations.add(new Location(1,"Банкомат АТМ Беларусбанк", "ул. Луцкая,
12"));
        locations.add(new Location(2,"Оператор МТС", "ул. Стафеева, 1,
к.1"));
    }

    public void printLocations(){
        for(int i=0; i<locations.size(); i++)
            locations.get(i).print();
    }

    public void printOperations(){
        for(int i=0; i<operations.size(); i++)
            operations.get(i).print();
    }

    public boolean addOperation(int type, int idLocation, double sum){
        if(sum<=0)
            return false;

        if((type==1||type==2)&&sum>money)
            return false;

        Location location = null;
        for(int i=0; i<locations.size(); i++)
```

```
            if(locations.get(i).id==idLocation)
            {
                location = locations.get(i).clone();
                break;
            }

        if(location==null)
            return false;

        operations.add(new Operation(type, sum, location));

        if(type==0)
            money+=sum;
        else
            money-=sum;

        return true;
    }

    public void print(){
        System.out.println("Остаток на счёте "+accountId+": "+money);
    }
}

public class Main {

    public static void main(String[] args) {
        Account account = new Account("000259325023A2");

        account.printLocations();
        System.out.println();

        account.addOperation(0, 0, 25);
        account.addOperation(1, 1, 10.2);
        account.addOperation(2, 2, 5);

        account.printOperations();
        System.out.println();
        account.print();
    }
}
```

Полученный результат:

```
id: 0; название: ООО Пищевые Продукты; адрес: ул. Советская, 54
id: 1; название: Банкомат АТМ Беларусбанк; адрес: ул. Луцкая, 12
id: 2; название: Оператор МТС; адрес: ул. Стафеева, 1, к.1

Тип операции: поступление; время: 02/12/2020 11:50:38; сумма: 25.0; источник: ООО Пищевые Продукты
Тип операции: снятие; время: 02/12/2020 11:50:38; сумма: 10.2; место операции: Банкомат АТМ Беларусбанк
Тип операции: платёж; время: 02/12/2020 11:50:38; сумма: 5.0; получатель: Оператор МТС

Остаток на счёте 000259325023A2: 9.8
```

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.

3) Создать класс Страница, используя класс Абзац.

Код программы:

```
package com.company;

import java.util.ArrayList;

class Indent{
    int otstyp;
    String line;

    public Indent(int otstyp, String line){
        this.otstyp = otstyp;
        this.line = line;
    }

    public void setLine(String line){
        this.line = line;
    }

    public void setOtstyp(int otstyp){
        this.otstyp = otstyp;
    }

    public void show(){
        for(int i=0; i<otstyp; i++)
            System.out.print(" ");

        System.out.println(line);
    }

}


class Page{
    Indent indent = null;
    ArrayList<String> lines;

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

    public void setIndent(int otstyp, String line){
        indent = new Indent(otstyp, line);
    }

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

    public void show(){
        if(indent==null)
            System.out.println("Ошибка - у страницы нет абзаца.");
```

```
        else{
            indent.show();
            for(int i=0; i<lines.size(); i++)
                System.out.println(lines.get(i));
        }
    }
}

public class Main {

    public static void main(String[] args) {
        Page page = new Page();

        page.show();

        page.setIndent(3, "Это первая строка этого");
        page.addLine("текста,  а это продолжение");
        page.addLine("этого текста.");


        System.out.println();
        page.show();
    }
}
```

Полученный результат:

```
   Это первая строка этого
текста,  а это продолжение
этого текста.
```

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.

3) Система Больница. Пациенту назначается лечащий Врач. Врач может сделать назначение Пациенту (процедуры, лекарства, операции). Медсестра или другой Врач выполняют назначение. Пациент может быть выписан из Больницы по окончании лечения, при нарушении режима или иных обстоятельствах.

Код программы:

Main.class:

```
package com.company;

import com.company.entities.Doctor;
import com.company.entities.MainDoctor;
import com.company.entities.Patient;
import com.company.entities.Purpose;

import java.util.ArrayList;

public class Main {

    public static void main(String[] args) {
        MainDoctor mainDoctor = new MainDoctor(23, "Абдулов", "Абдул");

        Doctor doctor = new Doctor(35, "Дмитрий", "Серов", 1);
        Doctor nurse = new Doctor(37, "Анастасия", "Румянцева", 0);

        Patient patient = new Patient(50, "Евгений", "Больной");

        Purpose purpose1 = new Purpose("Бисептол", 1);
        purpose1.setCountOfDay(3);
        Purpose purpose2 = new Purpose("Массаж спины", 0);
        purpose2.setCountOfDay(1);
        Purpose purpose3 = new Purpose("Удаление аденоидов", 2);

        ArrayList<Purpose> purposes = new ArrayList<Purpose>();
        purposes.add(purpose1);
        purposes.add(purpose2);
        purposes.add(purpose3);

        mainDoctor.makePurposes(patient, purposes);

        patient.print();

        for(int i=0; i<5; i++)
            System.out.println("");

        nurse.executePurpose(patient, purpose1);
        doctor.executePurpose(patient, purpose3);
        nurse.executePurpose(patient, purpose1);
        nurse.executePurpose(patient, purpose2);
        nurse.executePurpose(patient, purpose1);

        mainDoctor.dischargePatient(patient, 1);

        patient.print();
    }
}
```

Doctor.class:

```
package com.company.entities;

import com.company.interfaces.IDoctor;

public class Doctor extends Person implements IDoctor {

    private int type; //0 - медсестра, 1 - доктор

    public Doctor(int id, String name, String surname, int type) {
```

```
        super(id, name, surname);
        this.type = type;
    }

    public int getType() {
        return type;
    }

    public void setType(int type) {
        this.type = type;
    }

    public void executePurpose(Patient patient, Purpose purpose){
        Purpose donePurpose = purpose.clone();
        donePurpose.setDoctor(this);
        donePurpose.updateTime();
        donePurpose.setDone(true);

        patient.addTohistory(donePurpose);
    }
}
```

MainDoctor.class:

```
package com.company.entities;

import com.company.interfaces.IMainDoctor;

import java.util.ArrayList;

public class MainDoctor extends Doctor implements IMainDoctor {

    public MainDoctor(int id, String name, String surname) {
        super(id, name, surname, 1);
    }

    public void makePurposes(Patient patient, ArrayList<Purpose> purposes) {
        for(int i=0; i<purposes.size(); i++) {
            Purpose purpose1 = purposes.get(i).clone();
            purpose1.updateTime();
            purpose1.setDoctor(this);
            patient.addTohistory(purpose1);
        }
    }
    public void dischargePatient(Patient patient, int reason){
        patient.setStatus(reason);
    }
}
```

Patient.class:

```
package com.company.entities;

import java.util.ArrayList;

public class Patient extends Person {

    private int status; // 0 - лечится, 1 - выписан по окончанию лечения, 2 -
выписан из-за нарушения режима, 3 - выписан при иных обстоятельствах
    private ArrayList<Purpose> history;
```

```
    public Patient(int id, String name, String surname) {
        super(id, name, surname);
        status=0;
        history = new ArrayList<Purpose>();
    }

    public int getStatus() {
        return status;
    }

    public void setStatus(int status) {
        this.status = status;
    }

    public void addTohistory(Purpose purpose){
        history.add(purpose);
    }

    public void print(){
        System.out.print(super.toString());
        System.out.print(", статус: ");
        if(status==0)
            System.out.println("лечится");
        else if(status==1)
            System.out.println("выписан по окончанию лечения");
        else if(status==2)
            System.out.println("выписан из-за нарушения режима");
        else if(status==3)
            System.out.println("выписан при иных обстоятельствах");

        System.out.println("История действий:");
        for(int i=0; i<history.size(); i++)
            System.out.println(history.get(i).toString());
    }

}
```

Person.class:

```
package com.company.entities;

public class Person {
    private int id;
    private String name;
    private String surname;

    public Person(int id, String name, String surname) {
        this.id = id;
        this.name = name;
        this.surname = surname;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
```

```
    }

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

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String toString(){
        return "id: "+id+", name: "+name+", surname: "+surname;
    }
}
```

Purpose.class:

```
package com.company.entities;

import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class Purpose {
    private String name;
    private int type; //0 - процедура, 1 - лекарство, 2 - операция
    private LocalDateTime time;
    private int countOfDay;
    private Doctor doctor;
    private boolean isDone = false;

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

    public String getName() {
        return name;
    }

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

    public int getType() {
        return type;
    }

    public void setType(int type) {
        this.type = type;
    }

    public LocalDateTime getTime() {
        return time;
    }

    public void setTime(LocalDateTime time) {
        this.time = time;
    }
```

```
    public int getCountOfDay() {
        return countOfDay;
    }

    public void setCountOfDay(int countOfDay) {
        this.countOfDay = countOfDay;
    }

    public Doctor getDoctor() {
        return doctor;
    }

    public void setDoctor(Doctor doctor) {
        this.doctor = doctor;
    }

    public boolean isDone() {
        return isDone;
    }

    public void setDone(boolean done) {
        isDone = done;
    }

    public void updateTime(){
        time = LocalDateTime.now();
    }

    public String toString(){
        StringBuilder builder = new StringBuilder();

        builder.append(time.format(DateTimeFormatter.ofPattern("dd/MM/YYYY
HH:mm:ss")));
        builder.append(" пациенту");

        if(!isDone)
            builder.append(" дано");
        else
            builder.append(" проведено");
        builder.append(" назначение, название: "+name+", тип: ");
        if(type==0)
            builder.append("процедура");
        else if(type==1)
            builder.append("лекарство");
        else
            builder.append("операция");

        builder.append(", доктор: "+doctor.getName()+"
"+doctor.getSurname());

        if(type!=2 && !isDone)
            builder.append(", число в день: "+countOfDay);

        return builder.toString();
    }

    public Purpose clone(){
        Purpose purpose = new Purpose(this.name, this.type);
        purpose.doctor = doctor;
        purpose.time = time;
        purpose.countOfDay = countOfDay;
        purpose.isDone = isDone;
        return purpose;
```

```
        }
}
```

Результат программы:

```
id: 50, name: Евгений, surname: Больной, статус: лечится
История действий:
02/12/2020 12:10:24 пациенту дано назначение, название: Бисептол, тип: лекарство, доктор: Абдулов Абдул, число в день: 3
02/12/2020 12:10:24 пациенту дано назначение, название: Массаж спины, тип: процедура, доктор: Абдулов Абдул, число в день: 1
02/12/2020 12:10:24 пациенту дано назначение, название: Удаление аденоидов, тип: операция, доктор: Абдулов Абдул




id: 50, name: Евгений, surname: Больной, статус: выписан по окончанию лечения
История действий:
02/12/2020 12:10:24 пациенту дано назначение, название: Бисептол, тип: лекарство, доктор: Абдулов Абдул, число в день: 3
02/12/2020 12:10:24 пациенту дано назначение, название: Массаж спины, тип: процедура, доктор: Абдулов Абдул, число в день: 1
02/12/2020 12:10:24 пациенту дано назначение, название: Удаление аденоидов, тип: операция, доктор: Абдулов Абдул
02/12/2020 12:10:24 пациенту проведено назначение, название: Бисептол, тип: лекарство, доктор: Анастасия Румянцева
02/12/2020 12:10:24 пациенту проведено назначение, название: Удаление аденоидов, тип: операция, доктор: Дмитрий Серов
02/12/2020 12:10:24 пациенту проведено назначение, название: Бисептол, тип: лекарство, доктор: Анастасия Румянцева
02/12/2020 12:10:24 пациенту проведено назначение, название: Массаж спины, тип: процедура, доктор: Анастасия Румянцева
02/12/2020 12:10:24 пациенту проведено назначение, название: Бисептол, тип: лекарство, доктор: Анастасия Румянцева
```

Вывод: Приобрел практические навыки в области объектно-ориентированного проектирования на языке Java.