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

Лабораторная работа №4

По дисциплине «СПП» за 5 семестр

Выполнил:

Студент группы ПО-3

Ковалёва А. И.

Проверил:

Крощенко А. А.

Брест 2020

Вариант 12

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

Задание 1:

Создать класс Account (счет) с внутренним классом, с помощью объектов которого можно хранить информацию обо всех операциях со счетом (снятие, платежи, поступления).

Текст программы:

```
package com.company;

public class Main {

    public static void main(String[] args) {
        Account account = new Account();
        account.setId("12345");
        Account.info info = account.new info(100, 50, 400);
        info.show();
    }
}

class Account {
    private String id;

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

    class info {
        int takes;
        int payments;
        int income;

        info(int takes, int payments, int income) {
            this.takes = takes;
            this.payments = payments;
            this.income = income;
        }

        public void setTakes(int takes) {
            this.takes = takes;
        }

        public void setPayments(int payments) {
            this.payments = payments;
        }

        public void setIncome(int income) {
            this.income = income;
        }

        public void show() {
            System.out.println("id: " + id);
            System.out.println("Withdrawal: " + takes);
```

```
            System.out.println("Payments: " + payments);
            System.out.println("Income: " + income);
        }
    }
}
```

Результат выполнения:

```
id: 12345
Withdrawal: 100
Payments: 50
Income: 400
```

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. Проде- монстрировать использование разработанных классов.
Создать объект класса Компьютер, используя классы МатеринскаяПлата, Дисковод, ОЗУ.

Текст программы:

```
package com.company;

public class Main {
    public static void main(String[] args) {
        MatherBoard matherBoard1 = new MatherBoard("634", "amd 487");
        RAM ram1 = new RAM("ram2", 123);
        Drive drive = new Drive("toshiba");
        Computer computer1 = new Computer(matherBoard1, ram1);
        computer1.setDrive(drive);
        computer1.show();

        MatherBoard matherBoard2 = new MatherBoard("246", "aaa 986");
        RAM ram2 = new RAM("ram1", 100);
        Computer computer2 = new Computer(matherBoard1, ram1);
        computer2.show();
    }
}


class Computer {
    private MatherBoard matherboard;
    private RAM ram;
    private Drive drive;
    Computer(MatherBoard matherboard, RAM ram){
        this.matherboard = matherboard;
        this.ram = ram;
    }

    public void setDrive(Drive rive) {
        this.drive = drive;
    }
```

```
    public void show(){
        System.out.println("MatherBoard info: ");
        matherboard.show();
        System.out.println("RAM info: ");
        ram.show();
        if(drive != null){
            System.out.println("Drive info: ");
            drive.show();
        } else {
            System.out.println("No drive");
        }
        System.out.println();
    }
}

class MatherBoard {
    String id;
    String socket;
    MatherBoard(String id, String socket){
        this.id = id;
        this.socket = socket;
    }

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

    public void setSocket(String socket) {
        this.socket = socket;
    }

    public void show() {
        System.out.println("Id: " + id);
        System.out.println("Socket: " + socket);
    }
}

class RAM {
    String name;
    int power;
    RAM(String name, int power){
        this.name = name;
        this.power = power;
    }

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

    public void setPower(int volume) {
        this.power = volume;
    }
    public void show(){
        System.out.println("Name: " + name);
        System.out.println("Power: " + power);
    }
```

```
}

class Drive {
    String name;
    Drive(String name){
        this.name = name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public void show(){
        System.out.println("Name: " + name);
    }
}
```

Результат выполнения:

```
MatherBoard info:
Id: 634
Socket: amd 487
RAM info:
Name: ram2
Power: 123
No drive

MatherBoard info:
Id: 634
Socket: amd 487
RAM info:
Name: ram2
Power: 123
No drive
```

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 Факультатив. Преподаватель объявляет запись на Курс. Студент записывается на Курс, обучается и по окончании Преподаватель выставляет Оценку, которая сохраняется в Архиве. Студентов, Преподавателей и Курсов при обучении может быть несколько.

Текст программы:

```
package com.company; import java.util.ArrayList;

class Archive {
    Optional.Student student;
    int mark;
    Archive(Optional.Student student,int mark)  {
        this.student = student;
        this.mark = mark;
    }
    void show() {
```

```
        student.show();
        System.out.println("Mark: " + mark);
        System.out.println();
    }
}

class Optional {
    private ArrayList<Archive> arh;
    private ArrayList<Course> arr;

    Optional() {
        arh = new ArrayList<Archive>();
        arr = new ArrayList<Course>();
    }

    void add(Course obj) {
        arr.add(obj);
    }

    void showCourse() {
        for (Course course: arr) {
            course.show();
        }
    }

    void showArchive() {
        for (Archive archive: arh) {
            archive.show();
        }
    }

    class Teacher extends Person {
        Teacher(String name) {
            super(name);
        }
        void addCourse(Course course) {
            work(course);
        }
        void work(Course obj) {
            add(obj);
        }
        void setMark(Student student, int mark) {
            if (student.isLearning()) {
                student.setMark(mark);
                Archive archive = new Archive(student, mark);
                arh.add(archive);
            }
        }
    }

    class Student extends Person {
        int mark;
        Course studCourse;
        boolean learn;

        Student(String name) {
```

```
            super(name);
            learn = false;
        }
        void work(Course obj) {
            if (arr.contains(obj)) {
                studCourse = obj;
                System.out.println("Added course");
                learn = true;
            } else {
                System.out.println("No such course");
            }
        }

        boolean isLearning() {
            return learn;
        }

        void setMark(int mark) {
            this.mark = mark;
        }

        void show() {
            System.out.print("Student name:");
            super.show();
            studCourse.show();
        }
    }
}

class Person {
    String name;
    Person() {
        name = "";
    }
    Person(String name) {
        this.name=name;
    }
    void show() {
        System.out.println(name);
    }
}

class Course {
    int num;
    String title;
    Course() {
        num = 0;
        title = "";
    }
    Course(String title, int num) {
        this.title=title;
        this.num=num;
    }
    void show() {
        System.out.println("Course num: " + num);
        System.out.println("Course name: " + title);
```

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

public class Main {
    public static void main(String[] args) {
        Optional class1 = new Optional();

        Optional.Teacher teacher1= class1.new Teacher("Alex");
        Optional.Teacher teacher2= class1.new Teacher("Kate");

        Course mathCourse = new Course("Math",1);
        Course historyCourse = new Course("History",2);
        Course englishCourse = new Course("English",3);

        teacher1.addCourse(mathCourse);
        teacher2.addCourse(historyCourse);
        teacher2.addCourse(englishCourse);

        Optional.Student student1 = class1.new Student("Liz");
        Optional.Student student2 = class1.new Student("Nastya");

        student2.work(englishCourse);
        student1.work(historyCourse);

        teacher2.setMark(student1,9);
        teacher1.setMark(student2,7);

        class1.showCourse();
        System.out.println("-----------------");
        class1.showArchive();
    }
}
```

Вывод: В ходе выполненной работы приобрела практические навыки в области объектно-ориентированного проектирования.