Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Институт информационных технологий и анализа данных

наименование института

ОТЧЕТ  
к лабораторной работе по дисциплине

Объектно-ориентированное программирование

Разработка классов с использованием механизмов наследования, полиморфизма и инкапсуляции

Наименование темы

Выполнил студент группы ИСТб 19–2 Первых А.А.

шифр Фамилия И.О.

Проверила Маланова Т.В.

Фамилия И.О.

Содержание отчета на 18 стр.

Иркутск 2020 г.

Оглавление

[1 Постановка задачи 3](#_Toc58184130)

[2 Структура меню 3](#_Toc58184131)

[3. Проектирование классов. 4](#_Toc58184132)

[4 Таблица спецификаций 5](#_Toc58184133)

[4.1 Описание класса Staff 5](#_Toc58184134)

[4.2 Описание класса Group 5](#_Toc58184135)

[4.3 Описание класса Administration. 6](#_Toc58184136)

[4.4 Описание класса Engineer. 6](#_Toc58184137)

[4.5 Описание класса Worker. 7](#_Toc58184138)

[5 Разработка тестов 8](#_Toc58184139)

[6 Результаты тестирования 9](#_Toc58184140)

[7. Листинг исходного кода 11](#_Toc58184141)

[Заключение 17](#_Toc58184142)

[Список литературы 18](#_Toc58184143)

1 Постановка задачи

**Цель работы**: Применение объектно-ориентированного подхода в

разработке приложений с графическим интерфейсом.

**Задачи**:

- Получение общего представления о шаблоне программирования

Модель/Вид/Контроллер и его практического применения;

- Получение практического опыта работы с базовыми графическими

библиотеками java.

**Содержание задания**

Для выполнения задания необходимо разработать графический

пользовательский интерфейс средствами java для работы с классами

реализованными при выполнении либо 3-ей, либо 4-ой лабораторной работы.

Графический пользовательский интерфейс должен предоставлять

возможности работы со всеми возможными функциями программы (ввод и

вывод данных в соответствующей форме, изменение условий выполнения

задания, работа с файлами и т.д.). Состав используемых компонентов

определяется самостоятельно, но для них должны быть предусмотрены

соответствующие планировщики раскладки компонентов в основных

контейнерах.

Классы, реализующие графический пользовательский интерфейс,

должны содержать только обращения к методам классов для выполнения

заданий. Для этого может понадобиться модифицировать программный код

из предыдущих лабораторных работ.

Индивидуальное задание: Реализовать следующие классы: Рабочий, Кадры, Инженер, Администратор

Разработать класс-группу. Описать иерархию данных классов.

2 Проектирование графического интерфейса

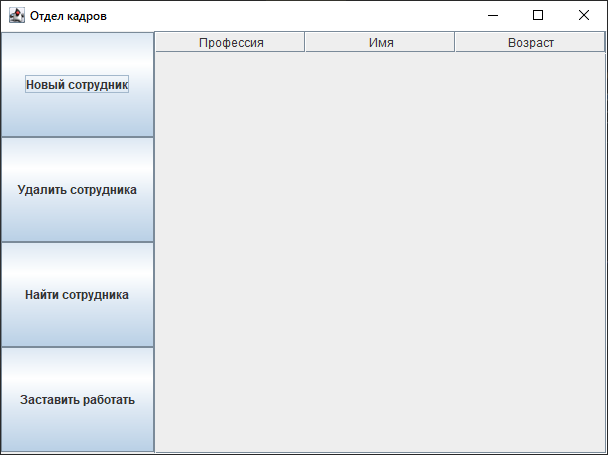
Главное окно программы содержит панель с кнопками и таблицу, отображающую список сотрудников. 

Рисунок 1 – Главное окно программы

Диалоговое окно добавления нового сотрудника, профессия выбирается радиокнопками, а имя и возраст вводится и проверяется вручную.

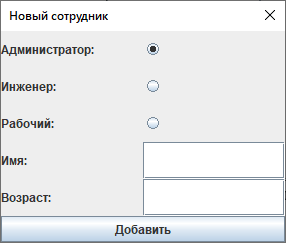


Рисунок 2 –Окно добавления нового сотрудника

Диалоговое окно удаления сотрудника по имени

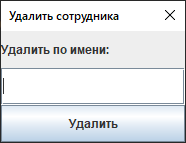


Рисунок 3 –Окно удаления сотрудника

Диалоговое окно поиска сотрудника, при выборе радиокнопки и нажатии на кнопку «Поиск по должности» открывается таблица с результатом поиска, при вводе имени и нажатии на «Поиск по имени», также открывается таблица с результатом.

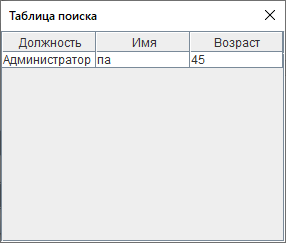
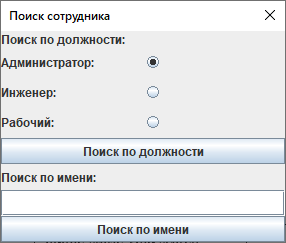


Рисунок 4 – Окно поиска сотрудника и таблица с результатом

Диалоговые сообщения с выводом ошибки при добавлении сотрудника

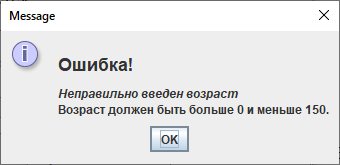
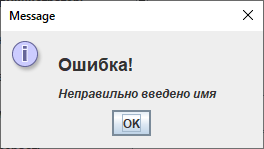


Рисунок 5 – Сообщения обработки корректности ввода добавлении сотрудника

Диалоговые сообщения с выводом ошибки при удалении и поиска сотрудника.

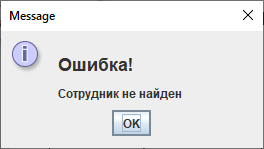


Рисунок 6 – Сообщения обработки исключений удалении и поиска сотрудника

Диалоговые сообщения с выводом ошибки при нажатии на кнопку «Заставить работать» с не выбранной строкой

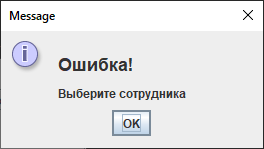


Рисунок 7 – Сообщения обработки исключений при работе сотрудника

3. Проектирование классов.

На рисунке 1 представлена UML диаграмма классов.

Пакет People

Staff

Group

public static ArrayList staff;

public static ArrayList staffSearch;

public static void add;

public static void delete;

public static searchName;

public static searchProff;

public static void getSize;

public static Staff getStaff;

private String name;

private int age;

private String profession;

public Staff;

public abstract void work;

public void getInfo;

public String getName;

public void setName;

public int getAge;

public void setAge;

public String getProfession;

public void setProfession

Engineer

public Engineer;

public void work;

Worker

public Worker;

public void work;

Administration

public Administration;

public void work;

Класс Staff – абстрактный класс кадры служит для выделения общих данных и поведения для других классов.

Класс Group - группирует объекты.

Класс Administration – Класс администратора.

Класс Engineer – Класс инженера.

Класс Worker – Класс рабочего.

Group

Menu\_GUI

private MyTableModel tableModel;

public Menu\_GUI

private class AddButtonEventListener

private class DeleteButtonEventListener

private class SearchButtonEventListener

private class SearchPopupButtonEventListener

public static void infoBox

private void updateTableSearch

Пакет Menu

Menu\_control

Public static staffName;

Public static staffAge

MyTableModel

private Group data

public MyTableModel

public int getRowCount

public int getColumnCount

public String getColumnName

public Object getValueAt

4 Таблица спецификаций

4.1 Описание класса Staff

Таблица 1 – Поля абстрактного класса Staff

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | name | Имя кадра | String | private |
| 2 | age | Возраст кадра | int | private |
| 3 | profession | Профессия кадра | String | private |

Таблица 2 – Конструкторы абстрактного класса Staff

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип доступа | Параметры |
| 1 | Staff | Конструктор задания полей. | public | name |
| age |
| profession |

Таблица 3 – Методы класса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/ неабстрактный | Параметры |
| 1 | work | Метод работы | String | public | Абстрактный |  |
| 2 | getInfo | getter информации о кадре | String | public | Абстрактный |  |
| 3 | getName | getter имени | String | public | неабстрактный |  |
| 4 | setName | setter имени | String | public | неабстрактный | name |
| 5 | getAge | getter возраста | int | public | неабстрактный |  |
| 6 | setAge | setter возраста | int | public | неабстрактный | age |
| 7 | getProfession | getter профессии | String | public | неабстрактный |  |
| 8 | setProfession | setter профессии | String | public | неабстрактный | profession |

4.2 Описание класса Group

Таблица 4 – Поля абстрактного класса Group

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | ArrayList<Staff> staff | Лист объектов | List | Public static |
| 1 | ArrayList<Staff> staffSearch | Лист объектов | List | Public static |

Таблица 5 – Методы класса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/ неабстрактный | Параметры |
| 1 | add | Добавление объекта | String | public static | неабстрактный | String[] list |
| 2 | delete | Удаление объекта | String | public static | неабстрактный | String name |
| 3 | searchName | Поиск сотрудника по имени | ArrayList<Staff> | public static | неабстрактный | String name |
| 4 | searchProff | Поиск сотрудника по профессии | ArrayList<Staff> | public static | неабстрактный | String proff |
| 5 | getStaff | Получение сотрудника по индексу | Staff | public static | неабстрактный | String name |
| 6 | getSize | Получение размера staff | Staff | public | неабстрактный |  |

4.3 Описание класса Administration.

Таблица 6 – Поля абстрактного класса Administration

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | Random rnd | Создания рандома |  | public |

Таблица 7 – Конструкторы абстрактного класса Administration

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип доступа | Параметры |
| 1 | Administration | Конструктор объекта Administration | public | name |
| age |
|  |

Таблица 8 – Методы класса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/ неабстрактный | Параметры |
| 1 | work | Метод работы | String | public | неабстрактный |  |

4.4 Описание класса Engineer.

Таблица 9 – Поля абстрактного класса Engineer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | Random rnd | Создания рандома |  | public |

Таблица 10 – Конструкторы абстрактного класса Engineer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип доступа | Параметры |
| 1 | Engineer | Конструктор объекта Engineer | public | name |
| age |
|  |

Таблица 11 – Методы класса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/ неабстрактный | Параметры |
| 1 | work | Метод работы | String | public | неабстрактный |  |

4.5 Описание класса Worker.

Таблица 12 – Поля абстрактного класса Worker

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа |
| 1 | Random rnd | Создания рандома |  | public |

Таблица 13 – Конструкторы абстрактного класса Worker

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип доступа | Параметры |
| 1 | Worker | Конструктор объекта Worker | public | name |
| age |
|  |

Таблица 14 – Методы класса

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Название | Назначение | Тип данных | Тип доступа | Абстрактный/ неабстрактный | Параметры |
| 1 | work | Метод работы | String | public | неабстрактный |  |

5 Разработка тестов

Таблица 15 – Таблица тестов

|  |  |  |
| --- | --- | --- |
| № | Действие | Результат действия |
| 1 | Первоначальный запуск программы | При старте программы появляется главная  Форма «Отдел кадров» |
| 2 | Создание нового кадра | Создания окна (кнопка «Новый сотрудник»), ввод имени, возвраста, и выбор профессии. Добавление сотрудника кнопкой «Добавить». |
| 3 | Удаление кадра | Создания окна (кнопка «Удалить сотрудника»), Удаление кадра в списке по имени. Удаление сотрудника кнопкой «Добавить». |
| 4 | Поиск сотрудника | Создания окна (кнопка «Найти сотрудника»), Поиск кадра в списке по имени. Поиск по должности или имени, кнопкой «Поиск». |
| 5 | Действие кадра | Вывод сообщения о действии кадра при выбранном в таблице сотрудника. Команда «Заставить работать». |
| 6 | Предупреждения при добавлении сотрудника | Сообщения обработки корректности ввода добавлении сотрудника |
| 7 | Предупреждения при отсутсвие кадров | Вывод предупреждения |
| 8 | Отсутстве выбора сотрудника | Вывод предупреждения |

6 Результаты тестирования

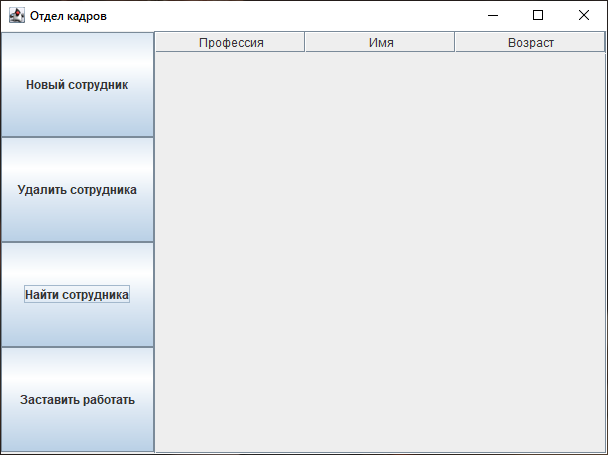


Рисунок 8 – Главное меню программы.

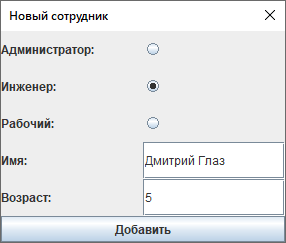


Рисунок 9 – Создание нового кадра.

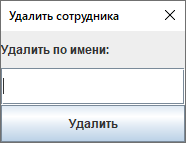


Рисунок 10 – Удаление кадра.

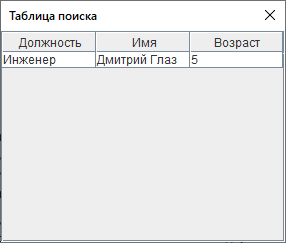
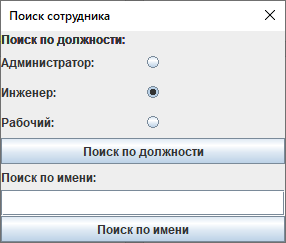


Рисунок 11 – Поиск заданного кадра.

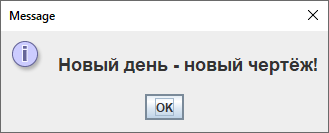


Рисунок 12 – Действие кадра.

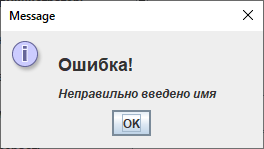
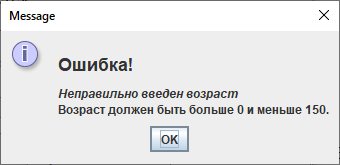
 

Рисунок 13 – Предупреждения при добавлении сотрудника.

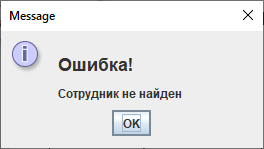


Рисунок 14 – Предупреждения при отсутсвие кадров.

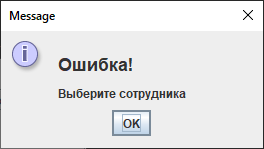


Рисунок 15 – Отсутстве выбора сотрудника.

7. Листинг исходного кода

**Класс Main**

Menu\_GUI GUI = new Menu\_GUI();

GUI.setVisible(true);

**Класс Menu\_control**

public static boolean staffName (String name) {

if (name.matches("^[\\p{L} ]+")) { //Содержание букв и пробел

return true;

}

else {

Menu\_GUI.infoBox("<i>Неправильно введено имя</i>");

}

return false;

}

public static boolean staffAge (String age) {

if (age.matches("-?\\d+(\\d+)?") ) {

if (Integer.parseInt(age) > 0 && Integer.parseInt(age) < 150 ){return true;}

else {Menu\_GUI.infoBox("<i>Неправильно введен возраст</i><p>Возраст должен быть больше 0 и меньше 150.</p>");}

}

else {Menu\_GUI.infoBox("<i>Неправильно введен возраст</i><p>Возраст должен быть больше 0 и меньше 150.</p>");}

return false;

}

**Класс Menu\_control**

**Класс Staff**

private String name;

private int age;

private String profession;

public Staff(String name, int age, String profession) {

this.name = name;

this.age = age;

this.profession = profession;

}

public abstract void work ();

public void getInfo () {

System.out.printf("Имя: %s%n", name);

System.out.printf("Возраст: %d%n", age);

System.out.printf("Профессия: %s%n", profession);

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public String getProfession() {

return profession;

}

public void setProfession(String profession) {

this.profession = profession;

}

**Класс Group**

public static ArrayList<Staff> staff = new ArrayList<Staff>();

public static ArrayList<Staff> staffSearch = new ArrayList<Staff>();

// Метод добавления сотрудника

public static void add (String[] list){

switch (list[0]) {

case "Рабочий":

staff.add(new Worker(list[1], Integer.parseInt(list[2])));

break;

case "Инженер":

staff.add(new Engineer(list[1], Integer.parseInt(list[2])));

break;

case "Администратор":

staff.add(new Administration(list[1], Integer.parseInt(list[2])));

break;

}

}

// Метод удаление сотрудника

public static void delete(String name) {

int i = -1;

if (staff.removeIf(nextStaff -> nextStaff.getName().equals(name))){

i++;

}

if (i == -1) {

Menu\_GUI.infoBox("Сотрудник не найден");

}

}

// Метод поиска сотрудника

public static ArrayList<Staff> searchName (String name) {

int check = 0;

staffSearch.clear();

for (Staff stf : staff) {

if (name.equals(stf.getName())) {

staffSearch.add(stf);

check++;

}

}

if (check == 0) {Menu\_GUI.infoBox("Сотрудник не найден"); return null;}

else {

return staffSearch;

}

}

public static ArrayList<Staff> searchProff (String proff) {

int check = 0;

staffSearch.clear();

for (Staff stf : staff) {

if (proff.equals(stf.getProfession())) {

staffSearch.add(stf);

check++;

}

}

if (check == 0) {Menu\_GUI.infoBox("Сотрудник не найден"); return null;}

else {

return staffSearch;

}

}

public int getSize(){

return staff.size();

}

public static Staff getStaff(int index){

return staff.get(index);

}

**Класс Administration**

Random random = new Random();

public Administration () {

super("Костик",18,"Администратор");

}

public Administration (String name, int age) {

super( name, age, "Администратор");

}

@Override

public String work () {

int workType = random.nextInt(3);

switch (workType) {

case 0:

return (getProfession() + " " + getName() +" устраивает планёрку");

case 1:

return (getProfession() + " " + getName() +" раздаёт задания");

default:

return (getProfession() + " " + getName() +" придумывает стратегию развития");

}

}

**Класс Engineer**

public Engineer () {

super("Костик",18,"Инженер");

}

public Engineer (String name, int age) {

super(name,age,"Инженер");

}

@Override

public String work () {

return "Новый день - новый чертёж!";

}

}

**Класс Worker**

Random random = new Random();

public Worker () {

super("Костик",18,"Рабочий");

}

public Worker (String name, int age) {

super(name,age,"Рабочий");

}

@Override

public String work () {

int salary = random.nextInt(6);

if (salary < 3) return "Неудачный день - маленькие деньги";

else return "Удачный день - большие деньги";

}

Заключение

В процессе выполнения лабораторной работы 4: “Разработка консольного приложения” были разработаны алгоритмы решения индивидуального задания. Было сформировано 9 тестов, покрывающих все ситуации. Составленные на языке Java программы, реализующие разработанные алгоритмы, прошла все тесты успешно.

В ходе лабораторной работы были получены практические навыки использования конструкций языка Java, а также знакомство с платформой.

Все цели практики достигнуты.

Список литературы

1. Java. Экспресс-курс [электронный ресурс] // Сайт Александра Климова [сайт], URL: http://developer.alexanderklimov.ru/android/java/java.php

2. API Specification for the Java 7 SE. [официальный сайт] URL: http://docs.oracle.com/javase/7/docs/api/

3. The Java Tutorials. SE [электронный ресурс], URL: http://docs.oracle.com/javase/tutorial/index.html

4. Гради Буч, Роберт А. Максимчук, Майкл У. Энгл, Бобби Дж. Янг, Джим Коналлен, Келли А. Хьюстон. Объектно-ориентированны анализ и проек-тирование с примерами приложений. Третье издание. М.: "Вильямс", 2010.

5. Хабибуллин И.Ш. Java 7: для программистов / И. Ш. Хабибуллин. – Санкт-Петербург : БХВ–Петербург, 2014.