|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3** | |
| **по дисциплине** | |
| **«**Объектно-ориентированное программирование**»**  **Тема: «Графический пользовательский интерфейс с использованием**  **библиотеки swing»** | |
|  | |
| Выполнил студент группы ИКБО-17-18 | Самородов А.А. |
| Принял преподаватель | Баранова И.А. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_201\_\_ г. | *(подпись руководителя)* |

Москва 2019

Задание 1. Разработать приложение со структурой классов

варианта

1. Условие задачи

Разработка проекта пользовательского интерфейса и его представление в форме дерева связей между элементами (окнами);

Реализация интерфейса средствами swing, используя оба подхода: программирование интерфейса путем создания кода классов форм вручную и

средствами визуального GUI редактора.

2. Постановка задачи

1. Для каждого класса лабораторной 3 разработать проект формы заполнения, модификации, отображения свойств экземпляра класса и удаления экземпляра.
2. Разработать проекты форм для демонстрации выполнения функционала программы, лабораторной работы 3
3. Разработать проект дерева интерфейса приложения.
4. Реализовать формы для ввода и вывода свойств одного объекта, предусмотрев:

* использование списковых элементов управления для заполнения отдельных свойств объекта класса;
* контроль достоверности вводимых данных.

1. Разработать форму для заполнения экземпляров класса из файла. Используйте класс, который был выбран в лабораторной работе 4 Предусмотрите сохранение данных объекта в файл и отображение объектов из файла в элементе JTable. При выполнении открытия и сохранения файлов используйте компонент JFileChooser.
2. Реализовать форму управления приложением – тестирование функционала задания лабораторной работы 3

Требования выполнению

1. Две формы должны быть разработаны без использования редактора GUI, остальные либо JFrame.
2. Основной фон форм – светло серый.
3. Текст элементов управления должен быть оформлен: шрифт – Times New Rоman, размер14, обычный, синий, выравнивание меток – по левому краю, в текстовом поле: числовые значения по правому краю, строковые полевому. Надписи на кнопках по центру.
4. Способ размещения элементов на форме выбрать самостоятельно.
5. Приложение должно иметь главную форму – первый уровень дерева интерфейса.
6. Вход пользователей в приложение по авторизации. Данные пользователей должны храниться в файле, лучше в форме объектов.
7. Предусмотреть обработку исключительных ситуаций через аппарат исключений.

8. Для представления сообщений использовать диалоговые окна сообщений.

# **3. UML-диаграмма вариантов использования**

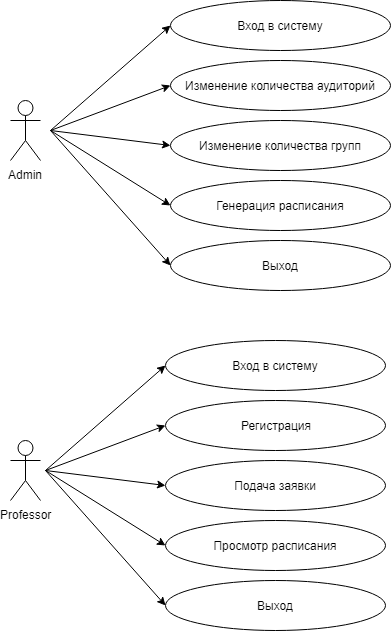


Рис. 1 Диаграмма вариантов использования

# **4. UML-диаграмма последовательности**

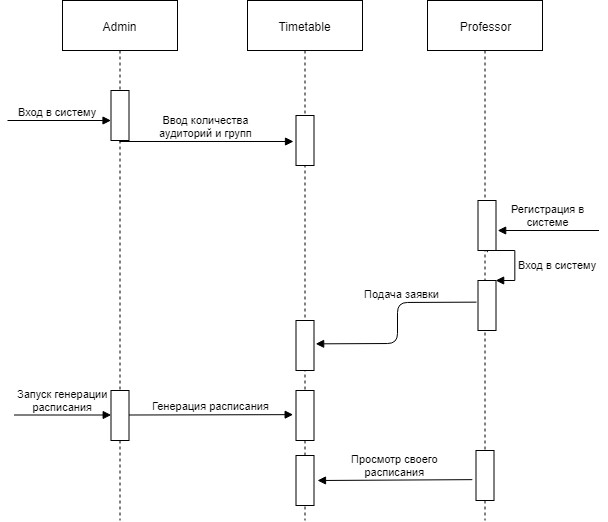


Рис. 2 Диаграмма последовательности действий

**5. UML-диаграмма классов**

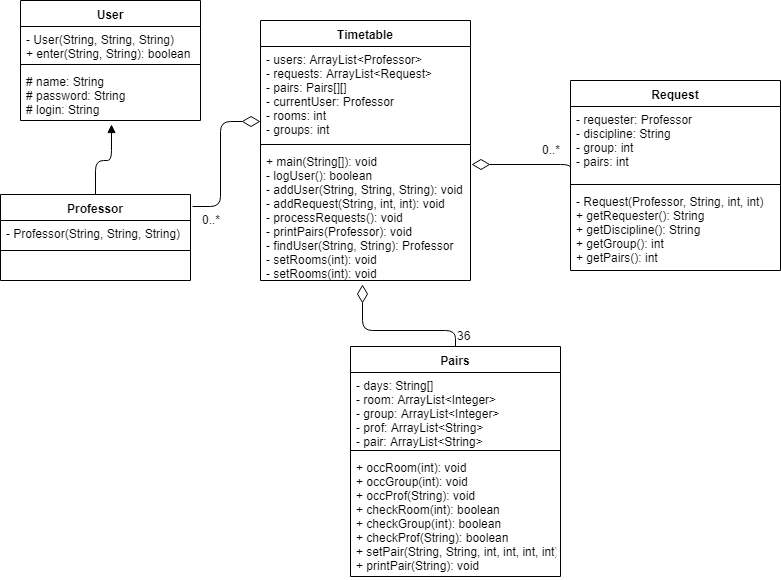


Рис. 3 Диаграмма классов

6. Разработка тестов

1. Регистрация преподавателя

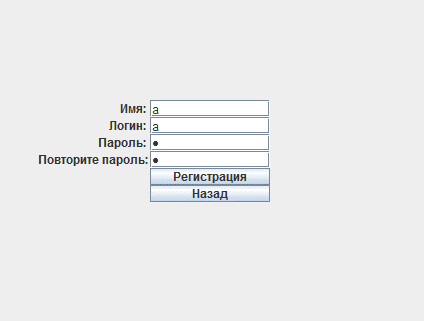


Рис. 4 Разработка тестов. Форма регистрации

2. Вход преподавателя в аккаунт

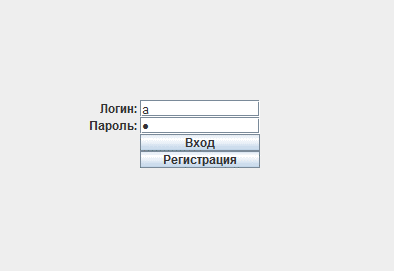


Рис. 5 Разработка тестов. Форма входа

3. Вход администратора в аккаунт

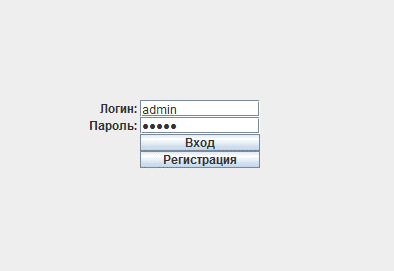


Рис. 6 Разработка тестов. Форма входа

4. Установка количества групп и аудиторий.

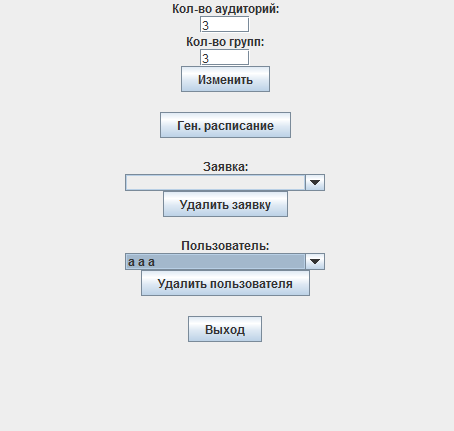


Рис. 7 Разработка тестов. Окно администратора

5. Удаление пользователя.

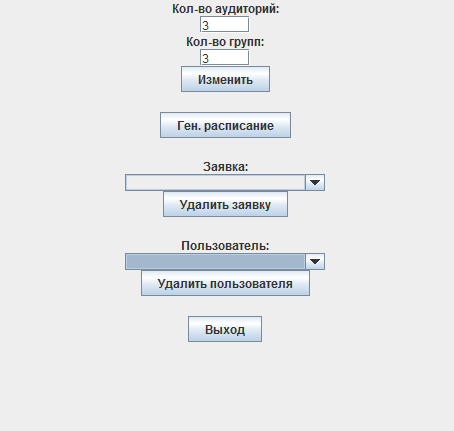


Рис. 8 Разработка тестов. Окно администратора

6. Подача заявки преподавателем.

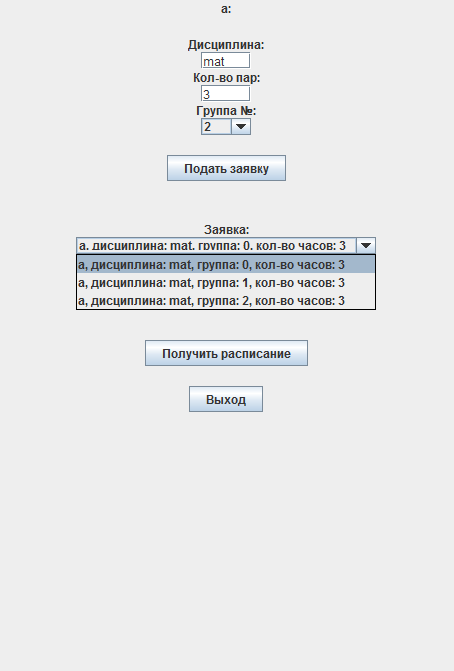


Рис. 9 Разработка тестов. Окно преподавателя

7. Генерация расписания администратором.

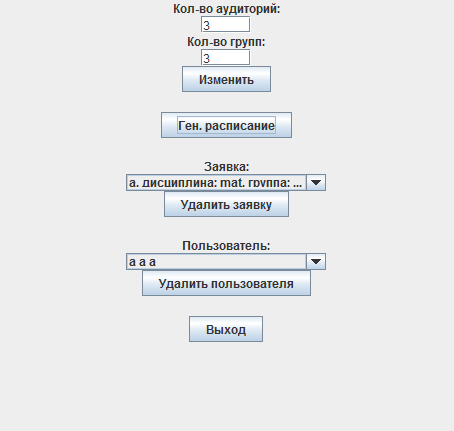


Рис. 10 Разработка тестов. Окно администратора

8. Получение расписания.

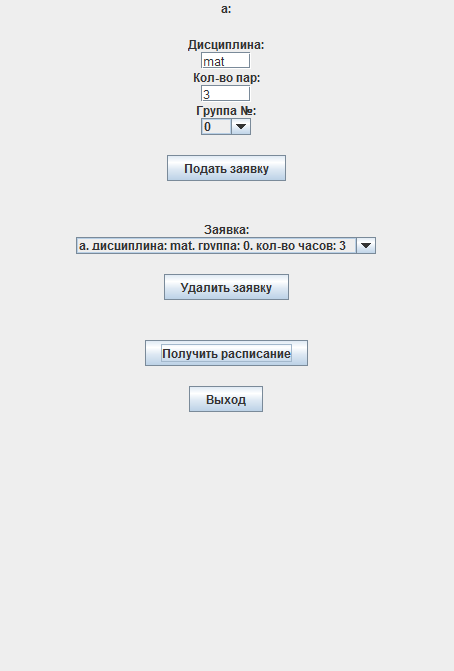


Рис. 11 Разработка тестов. Окно преподавателя

9. Полученное расписание в файле pairs.txt.

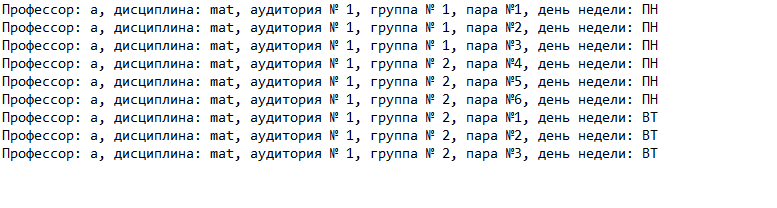


Рис. 12 Разработка тестов. Окно вывода результатов

**Работа с Git**

1. Просматриваем статус всех файлов в рабочей папке

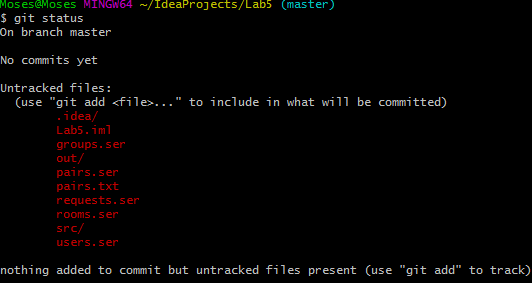


Рис. 13 Измененные и добавленные файлы

1. Затем индексируем файлы



Рис. 14 Индексация всех файлов в папке

1. Затем производим коммит изменений



Рис. 15 Коммит измененных файлов

1. Потом загружаем коммит в репозиторий

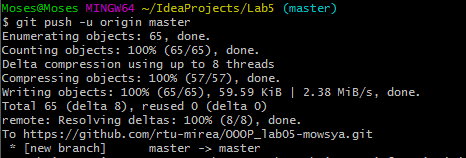


Рис. 16 Загрузка коммита в репозиторий

1. Проект находится в репозитории OOOP\_lab05-mowsya на GitHub

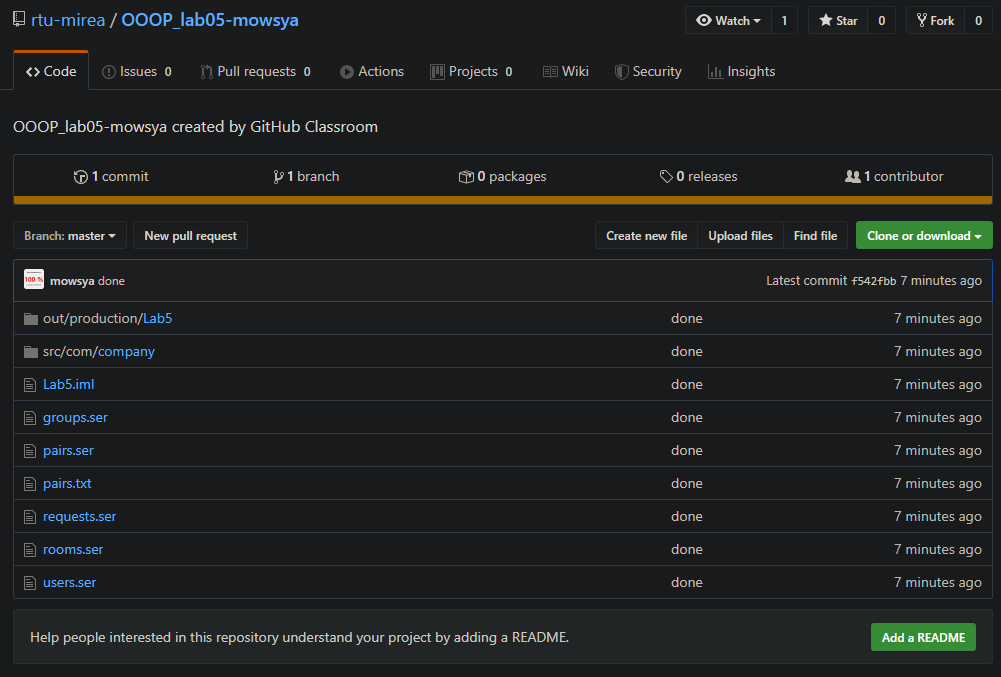


Рис. 17 Проект на GitHub

Готовый проект загружен и размещен на GitHub, доступен по ссылке: <https://github.com/rtu-mirea/OOOP_lab05-mowsya>

**Исходный код программы**

**Main.java**

package com.company;

import javax.swing.\*;

import java.io.\*;

import java.nio.charset.Charset;

import java.util.\*;

public class Main {

static ArrayList<Professor> users = new ArrayList<>();

static ArrayList<Request> requests = new ArrayList<>();

static Pairs[][] pairs = new Pairs[6][6];

static Professor currentUser;

static int rooms, groups;

public static Login log;

public static Register reg;

public static MainWindow mw;

public static Edit edit;

public static void main(String[] args) {

for (int i = 0; i < 6; i++) {

for (int j = 0; j < 6; j++) {

pairs[i][j] = new Pairs();

}

}

try

{

// Reading the object from a file

ObjectInputStream in\_requests = new ObjectInputStream(new FileInputStream("requests.ser"));

ObjectInputStream in\_users = new ObjectInputStream(new FileInputStream("users.ser"));

ObjectInputStream in\_pairs = new ObjectInputStream(new FileInputStream("pairs.ser"));

ObjectInputStream in\_rooms = new ObjectInputStream(new FileInputStream("rooms.ser"));

ObjectInputStream in\_groups = new ObjectInputStream(new FileInputStream("groups.ser"));

// Method for deserialization of object

requests = (ArrayList<Request>)in\_requests.readObject();

users = (ArrayList<Professor>)in\_users.readObject();

rooms = (int)in\_rooms.readObject();

groups = (int)in\_groups.readObject();

pairs = (Pairs[][])in\_pairs.readObject();

System.out.println(users);

try {

for (int i = 0; i < Main.groups; i++) {

mw.groups\_box.addItem(i+1);

}

}catch(NullPointerException eeeee){

}

in\_requests.close();

in\_users.close();

in\_rooms.close();

in\_groups.close();

} catch(IOException e) { }

catch(ClassNotFoundException e) {}

char[] pwd = {'a','d','m','i','n'};

users.add(new Professor("admin","admin",pwd,1));

log = new Login();

reg = new Register();

mw = new MainWindow();

edit = new Edit();

log.setVisible(true);

}

public static void generateSchedule() {

for (Request r : requests) {

int number = r.getPairs();

int room;

for (int i = 0; i < 6; i++) {

for (int j = 0; j < 6; j++) {

room = 1;

if (pairs[i][j].getRooms() < rooms && number > 0) {

while (pairs[i][j].checkRoom(room)) {

room++;

}

if (!pairs[i][j].checkGroup(r.getGroup()) && !pairs[i][j].checkProf(r.getRequester())) {

pairs[i][j].setPair(r.getRequester(), r.getDiscipline(), room, r.getGroup(), j, i);

pairs[i][j].occGroup(r.getGroup());

pairs[i][j].occProf(r.getRequester());

pairs[i][j].occRoom(room);

number--;

}

}

}

}

}

}

public static void getResults() {

FileWriter out\_pairs;

try {

out\_pairs = new FileWriter("pairs.txt");

for (int i = 0; i < 6; i++) {

for (int j = 0; j < 6; j++) {

out\_pairs.write(pairs[i][j].printPair(currentUser.getName()));

out\_pairs.write(System.lineSeparator());

}

}

out\_pairs.close();

}catch (IOException ee){}

}

public static void setUsers(ArrayList<Professor> users) {

Main.users = users;

}

public static int getRooms(){

return rooms;

}

public static int getGroups(){

return groups;

}

public static void setCurrentUser(Professor currentUser) {

Main.currentUser = currentUser;

}

public static void setRooms(int rooms){

Main.rooms = rooms;

}

public static void setGroups(int groups){

Main.groups = groups;

}

public static void setRequests(ArrayList<Request> requests) {

Main.requests = requests;

}

public static ArrayList<Professor> getUsers() {

return users;

}

public static User getCurrentUser() {

if (currentUser != null) {

return currentUser;

} else

return new User("", "", new char[0]);

}

public static ArrayList<Request> getRequests() {

return requests;

}

}

**User.java**

package com.company;

import java.util.Arrays;

public class User implements java.io.Serializable {

protected String name;

protected String login;

protected char[] password;

public int access = 0;

public User() {

}

public User(String name, String login, char[] password) {

setName(name);

setLogin(login);

setPassword(password);

}

public User(String name, String login, char[] password, int access) {

setName(name);

setLogin(login);

setPassword(password);

this.access = access;

}

public void setLogin(String login) {

this.login = login;

}

public void setName(String name) {

this.name = name;

}

public void setPassword(char[] password) {

this.password = password;

}

public String getLogin() {

return login;

}

public String getName() {

return name;

}

public char[] getPassword() {

return password;

}

public boolean enter(String login, char[] password) {

if (this.login == login && Arrays.equals(this.password, password)) {

return true;

} else {

return false;

}

}

}

**Elector.java**

package com.company;

public class Professor extends User implements java.io.Serializable {

public Professor(String name, String login, char[] password) {

super(name, login, password);

access = 0;

}

public Professor(String name, String login, char[] password, int access) {

super(name, login, password, access);

}

@Override

public String toString() {

String pass = "";

for (char i: getPassword()){

pass = pass + i;

}

return getName() + " " + getLogin() + " " + pass;

}

}

**Request.java**

package com.company;

public class Request implements java.io.Serializable {

private Professor requester;

private String discipline;

private int group, pairs;

Request(Professor requester, String disc, int group, int pairs) {

this.requester = requester;

this.discipline = disc;

this.group = group;

this.pairs = pairs;

}

String getRequester() {

return this.requester.getName();

}

String getDiscipline() {

return this.discipline;

}

int getGroup() {

return this.group;

}

int getPairs() {

return this.pairs;

}

@Override

public String toString(){

return getRequester() + ", дисциплина: " + getDiscipline() + ", группа: " + getGroup() + ", кол-во часов: " + getPairs();

}

}

**Pairs.java**

package com.company;

import java.util.ArrayList;

public class Pairs implements java.io.Serializable{

private final String[] days = {"ПН", "ВТ", "СР", "ЧТ", "ПТ", "СБ"};

private ArrayList<Integer> room;

private ArrayList<Integer> group;

private ArrayList<String> prof;

private ArrayList<String> pair;

Pairs() {

room = new ArrayList<>();

group = new ArrayList<>();

prof = new ArrayList<>();

pair = new ArrayList<>();

}

public void occRoom(int room) {

this.room.add(room);

}

public void occGroup(int group) {

this.group.add(group);

}

public void occProf(String prof) {

this.prof.add(prof);

}

public boolean checkRoom(int room) {

return this.room.contains(room);

}

public boolean checkGroup(int group) {

return this.group.contains(group);

}

public boolean checkProf(String prof) {

return this.prof.contains(prof);

}

public int getRooms() {

return this.room.size();

}

public void setPair(String prof, String disc, int room, int group, int hour, int day) {

this.pair.add("Профессор: " + prof + ", дисциплина: " + disc + ", аудитория № " + room + ", группа № " + group + ", пара №" + (hour + 1) + ", день недели: " + days[day]);

}

public String printPair(String prof) {

for (String i : pair) {

if (i.contains(prof)) {

return i;

}

}

return "";

}

}

**MainWindow.java**

package com.company;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectOutputStream;

public class MainWindow extends JFrame {

public JLabel user\_label = new JLabel(":");

public JLabel disc\_label = new JLabel("Дисциплина:");

public JLabel group\_label = new JLabel("Группа №:");

public JLabel pairs\_label = new JLabel("Кол-во пар:");

public JTextField disc\_textField= new JTextField();

public JComboBox groups\_box = new JComboBox();

public JTextField pairs\_textField = new JTextField();

public JLabel request\_label = new JLabel("Заявка:");

public JComboBox request\_box = new JComboBox();

public JButton schedule\_button = new JButton("Получить расписание");

public JButton request\_button = new JButton("Подать заявку");

public JButton edit\_button = new JButton("Удалить заявку");

public JButton exit\_button = new JButton("Выход");

public JPanel p = new JPanel();

MainWindow() {

super("Система составления расписания");

setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

setResizable(false);

p.setLayout(new BoxLayout(p, BoxLayout.Y\_AXIS));

p.add(user\_label);

p.add(Box.createVerticalStrut(20));

p.add(disc\_label);

p.add(disc\_textField);

p.add(pairs\_label);

p.add(pairs\_textField);

p.add(group\_label);

p.add(groups\_box);

p.add(Box.createVerticalStrut(20));

p.add(request\_button);

p.add(Box.createVerticalStrut(40));

p.add(request\_label);

p.add(request\_box);

p.add(Box.createVerticalStrut(20));

p.add(edit\_button);

p.add(Box.createVerticalStrut(40));

p.add(schedule\_button);

p.add(Box.createVerticalStrut(20));

p.add(exit\_button);

disc\_textField.setMaximumSize(new Dimension(50,17));

pairs\_textField.setMaximumSize(new Dimension(50,17));

groups\_box.setMaximumSize(new Dimension(50,17));

request\_box.setMaximumSize(new Dimension(300,17));

user\_label.setAlignmentX((float)0.5);

disc\_label.setAlignmentX((float)0.5);

disc\_textField.setAlignmentX((float)0.5);

pairs\_label.setAlignmentX((float)0.5);

pairs\_textField.setAlignmentX((float)0.5);

group\_label.setAlignmentX((float)0.5);

groups\_box.setAlignmentX((float)0.5);

request\_label.setAlignmentX((float)0.5);

request\_button.setAlignmentX((float)0.5);

request\_box.setAlignmentX((float)0.5);

schedule\_button.setAlignmentX((float)0.5);

exit\_button.setAlignmentX((float)0.5);

edit\_button.setAlignmentX((float)0.5);

ActionListener scheduleListener = new ScheduleListener();

schedule\_button.addActionListener(scheduleListener);

ActionListener requestListener = new RequestListener();

request\_button.addActionListener(requestListener);

ActionListener editListener = new EditListener();

edit\_button.addActionListener(editListener);

ActionListener exitListener = new ExitListener();

exit\_button.addActionListener(exitListener);

add(p);

this.setBounds(500, 300, 460, 500);

}

public class ScheduleListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

Main.getResults();

}

}

public class RequestListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

Main.requests.add(new Request(Main.currentUser, disc\_textField.getText(), groups\_box.getSelectedIndex(), Integer.parseInt(pairs\_textField.getText())));

try {

ObjectOutputStream out\_requests = new ObjectOutputStream(new FileOutputStream("requests.ser"));

out\_requests.writeObject(Main.getRequests());

out\_requests.close();

} catch (IOException e1) { }

request\_box.removeAllItems();

for(Request i : Main.requests) {

request\_box.addItem(i);

}

}

}

public class EditListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

for(int i = 0; i < Main.requests.size(); i++){

if (Main.requests.get(i) == request\_box.getSelectedItem()){

Main.requests.remove(i);

break;

}

}

try {

ObjectOutputStream out\_requests = new ObjectOutputStream(new FileOutputStream("requests.ser"));

out\_requests.writeObject(Main.getRequests());

out\_requests.close();

} catch (IOException e1) { }

request\_box.removeAllItems();

for(Request i : Main.requests) {

request\_box.addItem(i);

}

}

}

public class ExitListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

setVisible(false);

Main.log.login\_textField.setText("");

Main.log.password\_textField.setText("");

Main.log.setVisible(true);

}

}

}

**Login.java**

package com.company;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Arrays;

public class Login extends JFrame {

public JTextField login\_textField = new JTextField(40);

public JPasswordField password\_textField = new JPasswordField(40);

public JLabel login\_label = new JLabel("Логин:");

public JLabel empty\_login\_label = new JLabel("Введите логин");

public JLabel wrong\_login\_label = new JLabel("Неверный логин");

public JLabel password\_label = new JLabel("Пароль:");

public JLabel empty\_password\_label = new JLabel("Введите пароль");

public JLabel wrong\_password\_label = new JLabel("Неверный пароль");

public JButton login\_button = new JButton("Вход");

public JButton register\_button = new JButton("Регистрация");

public JPanel p = new JPanel();

Login() {

super("Вход");

setResizable(false);

setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

setSize(400, 300);

p.setLayout(null);

login\_textField.setText("");

password\_textField.setText("");

p.add(login\_label);

p.add(empty\_login\_label);

p.add(wrong\_login\_label);

p.add(password\_label);

p.add(empty\_password\_label);

p.add(wrong\_password\_label);

p.add(login\_textField);

p.add(password\_textField);

p.add(login\_button);

p.add(register\_button);

add(p);

login\_label.setBounds(100, 100, 60,17);

empty\_login\_label.setBounds(260, 100, 150,17);

wrong\_login\_label.setBounds(260, 100, 150,17);

password\_label.setBounds(89, 117, 60,17);

empty\_password\_label.setBounds(260, 117, 150,17);

wrong\_password\_label.setBounds(260, 117, 150,17);

login\_textField.setBounds(140, 100, 120, 17);

password\_textField.setBounds(140, 117, 120, 17);

login\_button.setBounds(140, 134, 120, 17);

register\_button.setBounds(140, 151, 120, 17);

ActionListener loginListener = new LoginListener();

login\_button.addActionListener(loginListener);

ActionListener registrationListener = new RegistrationListener();

register\_button.addActionListener(registrationListener);

empty\_login\_label.setVisible(false);

wrong\_login\_label.setVisible(false);

empty\_password\_label.setVisible(false);

wrong\_password\_label.setVisible(false);

Insets insets = this.getInsets();

this.setBounds(500, 300, 400 + insets.left + insets.right, 300 + insets.top + insets.bottom);

}

public class RegistrationListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

setVisible(false);

Main.reg.name\_textField.setText("");

Main.reg.login\_textField.setText("");

Main.reg.password\_textField.setText("");

Main.reg.repeat\_password\_textField.setText("");

Main.reg.setVisible(true);

}

}

public class LoginListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

findUserProcess();

}

}

private Professor findUser(String login, char[] password) throws Exception {

empty\_login\_label.setVisible(false);

wrong\_login\_label.setVisible(false);

empty\_password\_label.setVisible(false);

wrong\_password\_label.setVisible(false);

if (!login.isEmpty()) {

if (!Main.getUsers().isEmpty()) {

for (Professor user : Main.getUsers()) {

if (user.login.equals(login)) {

if (Arrays.equals(user.password, password)) {

return user;

} else {

if (password.length <= 0){

empty\_password\_label.setVisible(true);

}else {

wrong\_password\_label.setVisible(true);

}

}

}

}

}

if (!wrong\_password\_label.isVisible() && !empty\_password\_label.isVisible()){

wrong\_login\_label.setVisible(true);

}

} else {

empty\_login\_label.setVisible(true);

}

return new Professor("", "", new char[0]);

}

private void findUserProcess() {

String login = login\_textField.getText();

char[] password1 = password\_textField.getPassword();

try {

Professor user = findUser(login, password1);

if (!user.getName().equals("")) {

Main.setCurrentUser(user);

Main.log.setVisible(false);

if (Main.getCurrentUser().login.equals("admin")) {

Main.mw.user\_label.setText(Main.getCurrentUser().getName()+ ":");

for(Request i : Main.requests){

Main.edit.request\_box.addItem(i);

}

for(Professor i : Main.users){

if(!i.getName().equals("admin")){

Main.edit.user\_box.addItem(i);}

}

Main.edit.setVisible(true);

} else {

Main.mw.user\_label.setText(Main.getCurrentUser().getName() + ":");

Main.mw.groups\_box.removeAllItems();

for(int i = 0; i < Main.groups; i++){

Main.mw.groups\_box.addItem(i);

}

Main.mw.request\_box.removeAllItems();

for(Request i : Main.requests){

if(Main.currentUser.getName().equals(i.getRequester()))

Main.mw.request\_box.addItem(i);

}

Main.mw.setVisible(true);

}

} else {

Main.log.setVisible(true);

}

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

}

**Register.java**

package com.company;

import javax.swing.\*;

import java.awt.\*;

import java.awt.Color;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.Scanner;

import javax.swing.border.\*;

public class Register extends JFrame{

public JLabel name\_label = new JLabel("Имя:");

public JLabel empty\_name\_label = new JLabel("Введите имя");

public JLabel login\_label = new JLabel("Логин:");

public JLabel empty\_login\_label = new JLabel("Введите логин");

public JLabel wrong\_login\_label = new JLabel("Логин занят");

public JLabel password\_label = new JLabel("Пароль:");

public JLabel empty\_password\_label = new JLabel("Введите пароль");

public JLabel repeat\_password\_label = new JLabel("Повторите пароль:");

public JLabel empty\_repeat\_password\_label = new JLabel("Повторите пароль:");

public JLabel wrong\_password\_label = new JLabel("Пароли не совпадают");

public JTextField name\_textField = new JTextField(40);

public JTextField login\_textField = new JTextField(40);

public JPasswordField password\_textField = new JPasswordField(40);

public JPasswordField repeat\_password\_textField = new JPasswordField(40);

public JButton register\_button = new JButton("Регистрация");

public JButton login\_button = new JButton("Назад");

public JPanel p = new JPanel();

Register() {

super("Регистрация");

setResizable(false);

setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

setSize(430, 350);

p.setLayout(null);

p.add(name\_label);

p.add(empty\_name\_label);

p.add(login\_label);

p.add(empty\_login\_label);

p.add(wrong\_login\_label);

p.add(password\_label);

p.add(empty\_password\_label);

p.add(repeat\_password\_label);

p.add(empty\_repeat\_password\_label);

p.add(wrong\_password\_label);

p.add(name\_textField);

p.add(login\_textField);

p.add(password\_textField);

p.add(repeat\_password\_textField);

p.add(register\_button);

p.add(login\_button);

add(p);

name\_label.setBounds(120, 100, 36,17);

empty\_name\_label.setBounds(280, 100, 120,17);

login\_label.setBounds(109, 117, 40,17);

empty\_login\_label.setBounds(280, 117, 120,17);

wrong\_login\_label.setBounds(280, 117, 120,17);

password\_label.setBounds(98, 134, 60,17);

empty\_password\_label.setBounds(280, 134, 120,17);

repeat\_password\_label.setBounds(38, 151, 110,17);

empty\_repeat\_password\_label.setBounds(280, 151, 120,17);

wrong\_password\_label.setBounds(280, 151, 150,17);

name\_textField.setBounds(150, 100, 120, 17);

login\_textField.setBounds(150, 117, 120, 17);

password\_textField.setBounds(150, 134, 120, 17);

repeat\_password\_textField.setBounds(150, 151, 120, 17);

register\_button.setBounds(150, 168, 120, 17);

login\_button.setBounds(150, 185, 120, 17);

ActionListener actionListener = new RegistrationListener();

register\_button.addActionListener(actionListener);

ActionListener loginListener = new LoginListener();

login\_button.addActionListener(loginListener);

empty\_name\_label.setVisible(false);

empty\_login\_label.setVisible(false);

wrong\_login\_label.setVisible(false);

empty\_password\_label.setVisible(false);

empty\_repeat\_password\_label.setVisible(false);

wrong\_password\_label.setVisible(false);

Insets insets = this.getInsets();

this.setBounds(500, 300, 430 + insets.left + insets.right, 350 + insets.top + insets.bottom);

}

public class LoginListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

setVisible(false);

Main.log.login\_textField.setText("");

Main.log.password\_textField.setText("");

Main.log.setVisible(true);

}

}

public class RegistrationListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

addUserProcess();

try {

ObjectOutputStream out\_users = new ObjectOutputStream(new FileOutputStream("users.ser"));

out\_users.writeObject(Main.getUsers());

out\_users.close();

} catch (IOException e1) { }

}

}

private User addUser(String name, String login, char[] password) throws Exception {

if (!Main.getUsers().isEmpty()) {

for (User user: Main.getUsers()) {

if (user.login.equals(login)) {

System.out.println("0");

throw new Exception("Такой логин уже существует");

}

}

}

Professor user = new Professor(name, login, password);

ArrayList<Professor> curUsers = Main.getUsers();

curUsers.add(user);

Main.setUsers(curUsers);

return user;

}

private void addUserProcess() {

String name = name\_textField.getText();

String login = login\_textField.getText();

char[] password1 = password\_textField.getPassword();

char[] password2 = repeat\_password\_textField.getPassword();

empty\_name\_label.setVisible(false);

empty\_login\_label.setVisible(false);

wrong\_login\_label.setVisible(false);

empty\_password\_label.setVisible(false);

empty\_repeat\_password\_label.setVisible(false);

wrong\_password\_label.setVisible(false);

int flag = 0;

if (!name.isEmpty()) {

if (!login.isEmpty()) {

if (password1.length > 0) {

if (password2.length > 0) {

for (int i = 0; i < Main.getUsers().size(); i++){

if (Main.getUsers().get(i).getLogin().equals(login)){

flag = 1;

}

}

if (flag == 0) {

if (Arrays.equals(password1, password2)) {

try {

addUser(name, login, password1);

setVisible(false);

Main.log.setVisible(true);

} catch (Exception e) {

System.out.println(e.getMessage());

}

} else {

wrong\_password\_label.setVisible(true);

}

} else {

wrong\_login\_label.setVisible(true);

}

}

else {

empty\_repeat\_password\_label.setVisible(true);

}

} else {

empty\_password\_label.setVisible(true);

}

} else {

empty\_login\_label.setVisible(true);

}

} else {

empty\_name\_label.setVisible(true);

}

}

}

**Edit.java**

package com.company;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectOutputStream;

public class Edit extends JFrame {

public JLabel rooms\_label = new JLabel("Кол-во аудиторий:");

public JLabel groups\_label = new JLabel("Кол-во групп:");

public JTextField rooms\_textField = new JTextField(40);

public JTextField groups\_textField = new JTextField(40);

public JButton rooms\_button = new JButton("Изменить");

public JButton gen\_button = new JButton("Ген. расписание");

public JButton cancel\_button = new JButton("Выход");

public JLabel request\_label = new JLabel("Заявка:");

public JLabel user\_label = new JLabel("Пользователь:");

public JComboBox user\_box = new JComboBox();

public JButton user\_button = new JButton("Удалить пользователя");

public JComboBox request\_box = new JComboBox();

public JButton request\_button = new JButton("Удалить заявку");

public JPanel p = new JPanel();

Edit() {

super("Меню администратора");

setResizable(false);

p.setLayout(new BoxLayout(p,BoxLayout.Y\_AXIS));

p.add(rooms\_label);

p.add(rooms\_textField);

p.add(groups\_label);

p.add(groups\_textField);

p.add(rooms\_button);

p.add(Box.createVerticalStrut(20));

p.add(gen\_button);

p.add(Box.createVerticalStrut(20));

p.add(request\_label);

p.add(request\_box);

p.add(request\_button);

p.add(Box.createVerticalStrut(20));

p.add(user\_label);

p.add(user\_box);

p.add(user\_button);

p.add(Box.createVerticalStrut(20));

p.add(cancel\_button);

add(p);

// rooms\_textField.setSize(20,17);

// groups\_textField.setSize(20,17);

rooms\_label.setAlignmentX((float)0.5);

rooms\_textField.setAlignmentX((float)0.5);

groups\_label.setAlignmentX((float)0.5);

groups\_textField.setAlignmentX((float)0.5);

rooms\_button.setAlignmentX((float)0.5);

gen\_button.setAlignmentX((float)0.5);

cancel\_button.setAlignmentX((float)0.5);

request\_label.setAlignmentX((float)0.5);

request\_button.setAlignmentX((float)0.5);

request\_box.setAlignmentX((float)0.5);

user\_label.setAlignmentX((float)0.5);

user\_button.setAlignmentX((float)0.5);

user\_box.setAlignmentX((float)0.5);

rooms\_textField.setMaximumSize(new Dimension(50,17));

groups\_textField.setMaximumSize(new Dimension(50,17));

request\_box.setMaximumSize(new Dimension(200,17));

user\_box.setMaximumSize(new Dimension(200,17));

// Dimension size\_rooms\_button = rooms\_button.getPreferredSize();

// Dimension size\_gen\_button = gen\_button.getPreferredSize();

// Dimension size\_cancel\_button = cancel\_button.getPreferredSize();

//

// rooms\_textField.setBounds(10, 10, 20, 17);

// groups\_textField.setBounds(110, 10, 20, 17);

// rooms\_button.setBounds(220, 10, size\_rooms\_button.width, 17);

// gen\_button.setBounds(230 + size\_rooms\_button.width, 10, size\_gen\_button.width, 17);

// cancel\_button.setBounds(240 + size\_rooms\_button.width + size\_gen\_button.width, 10, size\_cancel\_button.width, 17);

ActionListener roomsListener = new RoomsListener();

rooms\_button.addActionListener(roomsListener);

ActionListener groupsListener = new GroupsListener();

rooms\_button.addActionListener(groupsListener);

ActionListener genListener = new GenListener();

gen\_button.addActionListener(genListener);

ActionListener requestListener = new RequestListener();

request\_button.addActionListener(requestListener);

ActionListener cancelListener = new CancelListener();

cancel\_button.addActionListener(cancelListener);

ActionListener userListener = new UserListener();

user\_button.addActionListener(userListener);

this.setBounds(500, 300, 460, 460);

}

public class RequestListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

Main.requests.remove(request\_box.getSelectedIndex());

try {

ObjectOutputStream out\_requests = new ObjectOutputStream(new FileOutputStream("requests.ser"));

out\_requests.writeObject(Main.getRooms());

out\_requests.close();

} catch (IOException e1) {

}

request\_box.removeAllItems();

for(Request i : Main.requests){

request\_box.addItem(i);

}

}

}

public class UserListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

Main.users.remove(user\_box.getSelectedIndex()+1);

try {

ObjectOutputStream out\_users = new ObjectOutputStream(new FileOutputStream("users.ser"));

out\_users.writeObject(Main.getRooms());

out\_users.close();

} catch (IOException e1) {

}

user\_box.removeAllItems();

for(Professor i : Main.users){

if(!i.getName().equals("admin")){

user\_box.addItem(i);}

}

}

}

public class RoomsListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

Main.setRooms(Integer.parseInt(rooms\_textField.getText()));

try {

ObjectOutputStream out\_rooms = new ObjectOutputStream(new FileOutputStream("rooms.ser"));

out\_rooms.writeObject(Main.getRooms());

out\_rooms.close();

} catch (IOException e1) {

}

}

}

public class GroupsListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

Main.setGroups(Integer.parseInt(groups\_textField.getText()));

try {

ObjectOutputStream out\_groups = new ObjectOutputStream(new FileOutputStream("groups.ser"));

out\_groups.writeObject(Main.getRooms());

out\_groups.close();

} catch (IOException e1) {

}

}

}

public class GenListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

try {

Main.generateSchedule();

ObjectOutputStream out\_pairs = new ObjectOutputStream(new FileOutputStream("pairs.ser"));

out\_pairs.writeObject(Main.pairs);

} catch (IOException e1) {

}

}

}

public class CancelListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

setVisible(false);

Main.log.setVisible(true);

}

}

}