Содержание

[Введение 4](#_Toc106830782)

[1 Представление деревьев. Основные операции над деревьями 5](#_Toc106830783)

[2 Описание алгоритмов и технологии решения 7](#_Toc106830784)

[2.1 Основной функционал программы 7](#_Toc106830785)

[2.2 Описание алгоритма решения задач 8](#_Toc106830787)

[2.3 Проектирование макета интерфейса 8](#_Toc106830794)

[3 Выполнение практической части 11](#_Toc106830801)

[3.1 Описание интерфейса 11](#_Toc106830802)

[3.2 Программная реализации каждой подпрограммы 15](#_Toc106830817)

[3.3 Контроль корректности ввода данных 20](#_Toc106830819)

[4 Результат работы 22](#_Toc106830820)

[Заключение 33](#_Toc106830882)

[Список использованных источников 34](#_Toc106830885)

[Приложение А Листинг программы 35](#_Toc106830886)

# ВВЕДЕНИЕ

Целью курсового проекта является создание обучающего приложения по языку программирования «Java» и изучение теоретического материала на тему «Представление деревьев. Основные операции над деревьями» [1].

Для достижения цели курсового проекта необходимо выполнить задачи:

* изучить теоретические данные на тему «Представление деревьев. Основные операции над деревьями»;
* закрепить знания о деревьях;
* организовать работу со структурированным файлом;
* разработать графический пользовательский интерфейс;
* разработать алгоритмы программы;
* осуществить модульное тестирование программы.

В результате выполнения всех действий будет разработано обучающее приложение по языку программирования «Java».

Обучающие приложения — это сервисы, помогающие пользователям разного возраста и с разным уровнем подготовки изучить те или иные учебные дисциплины.

Основной задачей каждого обучающего приложения является предоставление информации в какой-либо области знаний.

Приложение создается и разрабатывается на языке программирования «Java» в средах разработки «IntelliJ IDEA» и «SceneBuilder».

«Java» — это высокоуровневый язык программирования со строгой типизацией. Позволяет создавать кроссплатформенные программы разного уровня сложности, производить тестирование и оптимизацию, работать с базами данных. Язык программирования «Java» исполняет свой код при помощи «JVM» – «Java Virtual Machine».

«IntelliJ IDEA» – специальная IDE для профессиональной разработки на языке программирования «Java».

«SceneBuilder» — инструмент для проектирования и создания графических пользовательских интерфейсов (GUI) из компонентов «JavaFX». Позволяет создать и настроить виджеты или диалоговые окна и проверить их используя разные стили и разрешающие способности.

# 1 ПРЕДСТАВЛЕНИЕ ДЕРЕВЬЕВ. ОСНОВНЫЕ ОПЕРАЦИИ НАД ДЕРЕВЬЯМИ

Дерево — одна из наиболее широко распространённых [структур данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85" \o "Структура данных) в информатике, эмулирующая древовидную структуру в виде набора связанных узлов. Является связным графом, не содержащим циклы [2].

Существует множество различных способов представления деревьев. Наиболее общий способ представления изображает узлы как записи, расположенные в [динамически выделяемой памяти](https://ru.wikipedia.org/w/index.php?title=%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8_%D0%B2%D1%8B%D0%B4%D0%B5%D0%BB%D1%8F%D0%B5%D0%BC%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C&action=edit&redlink=1" \o "Динамически выделяемая память (страница отсутствует)) с указателями на своих потомков, предков (или и тех и других), или как элементы [массива](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%81%D0%B8%D0%B2_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)" \o "Массив (программирование)), связанные между собой отношениями, определёнными их позициями в массиве.

Узел является экземпляром одного из двух типов элементов графа, соответствующим объекту некоторой фиксированной природы. Узел может содержать значение, состояние или представление отдельной информационной структуры или самого дерева.

В [теории графов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2" \o "Теория графов) [дерево](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2)" \o "Дерево (теория графов)) — связный [ациклический граф](https://ru.wikipedia.org/wiki/%D0%90%D1%86%D0%B8%D0%BA%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B3%D1%80%D0%B0%D1%84" \o "Ациклический граф). Корневое дерево — это граф с [вершиной](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D1%80%D1%88%D0%B8%D0%BD%D0%B0_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2)" \o "Вершина (теория графов)), выделенной в качестве корневой. В этом случае любые две вершины, связанные ребром, наследуют отношения «родитель-потомок». Несвязный граф, состоящий исключительно из деревьев, называется лесом.

Основные операции над деревьями:

* вставка нового элемента в определённую позицию;
* вставка поддерева;
* добавление ветви дерева (называется прививкой);
* нахождение корневого элемента для любого узла;
* нахождение [наименьшего общего предка](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D0%B8%D0%BC%D0%B5%D0%BD%D1%8C%D1%88%D0%B8%D0%B9_%D0%BE%D0%B1%D1%89%D0%B8%D0%B9_%D0%BF%D1%80%D0%B5%D0%B4%D0%BE%D0%BA) двух вершин;
* обход дерева;
* поиск [изоморфного](https://ru.wikipedia.org/wiki/%D0%98%D0%B7%D0%BE%D0%BC%D0%BE%D1%80%D1%84%D0%B8%D0%B7%D0%BC) поддерева;
* поиск элемента;
* удаление ветви дерева (называется [обрезкой](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%D1%80%D0%B5%D0%B7%D0%BA%D0%B0_(%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC)&action=edit&redlink=1));
* удаление поддерева;
* удаление элемента.

Наиболее частой операцией над деревом считается обход дерева.

Обход дерева (известный также как поиск по дереву) — вид [обхода графа](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%D1%85%D0%BE%D0%B4_%D0%B3%D1%80%D0%B0%D1%84%D0%B0&action=edit&redlink=1" \o "Обход графа (страница отсутствует)), обусловливающий процесс посещения (проверки и/или обновления) каждого узла [структуры дерева данных](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)" \o "Дерево (структура данных)) ровно один раз. Такие обходы классифицируются по порядку, в котором узлы посещаются. Алгоритмы в статье относятся к [двоичным деревьям](https://ru.wikipedia.org/wiki/%D0%94%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE" \o "Двоичное дерево), но могут быть обобщены и для других деревьев.

Обход дерева итеративно проходит по всем узлам согласно некоторому алгоритму. Поскольку из данного узла имеется более одного следующего узла (это не линейная структура данных), то, в предположении последовательных вычислений (а не параллельных), некоторые узлы должны быть отложены, то есть запомнены некоторым способом для дальнейшего посещения.

Часто это делается с помощью [стека](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%BA" \o "Стек) (LIFO = последний вошёл — первый вышел) или [очереди](https://ru.wikipedia.org/wiki/%D0%9E%D1%87%D0%B5%D1%80%D0%B5%D0%B4%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)" \o "Очередь (программирование)) (FIFO = первый вошёл — первым вышел).

Так как дерево самореферентная (ссылающаяся на себя, определённая рекурсивно) структура данных, обход может быть определён [рекурсией](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BA%D1%83%D1%80%D1%81%D0%B8%D1%8F" \o "Рекурсия) или, более тонко, [корекурсией](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%80%D0%B5%D0%BA%D1%83%D1%80%D1%81%D0%B8%D1%8F" \o "Корекурсия) естественным и ясным образом. В этих случаях отложенные узлы запоминаются либо явно в обычном [стеке](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%BA" \o "Стек), либо неявно в [стеке вызовов](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%BA_%D0%B2%D1%8B%D0%B7%D0%BE%D0%B2%D0%BE%D0%B2" \o "Стек вызовов), либо явно в [очереди](https://ru.wikipedia.org/wiki/%D0%9E%D1%87%D0%B5%D1%80%D0%B5%D0%B4%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)" \o "Очередь (программирование)).

Поиск в глубину легко имплементируется через стек, включая имплементацию через рекурсию (стек вызовов), в то время как поиск в ширину легко имплементируется через очередь, включая имплементацию через корекурсию.

Эти поиски называются поиском в глубину ввиду того, что дерево поиска проходится вниз насколько это можно на каждом потомке прежде чем переходить к следующей родственной ветке. Для двоичного дерева они определяются как операции обработки вершины рекурсивно в каждом узле, начиная скорня.

Основной рекурсивный подход для обхода (непустого) бинарного дерева: Начиная с узла N делаем следующее:

(L) Рекурсивно обходим левое поддерево. Этот шаг завершается при попадании опять в узел N.

(R) Рекурсивно обходим правое поддерево. Этот шаг завершается при попадании опять в узел N.

(N) Обрабатываем сам узел N.

Эти шаги могут быть проделаны в любом порядке. Если (L) осуществляется перед (R), процесс называется обходом слева направо, в противном случае — обходом справа налево.

Ниже, на рисунке 1.1, приведен пример процедуры для обхода дерева:

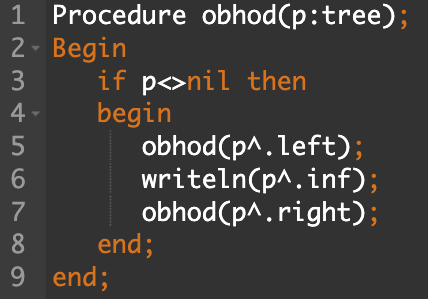


Рисунок 1.1 — Процедура обхода дерева

# 2 ОПИСАНИЕ АЛГОРИТМОВ И ТЕХНОЛОГИИ РЕШЕНИЯ

## **2.1 Основной функционал программы**

Данное обучающее приложение по языку программирования «Java» будет разработано на языке программирования «Java», в средах разработки «JavaFX» и «IntelliJ IDEA».

Главной целью данного приложения будет является обучение пользователя такому языку программирования, как «Java» [3].

Все данные будут храниться в структурированном файле формата «json».

На рисунке 2.1 показан пример структуры файла.

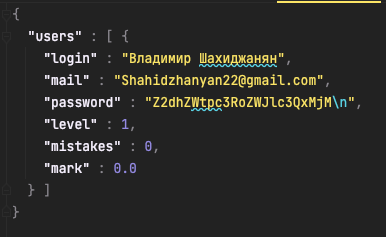


Рисунок 2.1 — Структура файла

Данное обучающее приложение будет выполнять следующие функции:

* автоматическое создание файлов данных пользователя и их статистики;
* регистрация нового пользователя;
* авторизация пользователя;
* проверка корректности вводимых данных авторизации;
* запись и чтение данных пользователей в файл формата «json»;
* предоставление различных тем для изучения [4];
* предоставления различных заданий по изученным темам [5];
* запись результатов пройденного раздела в файл;
* разблокировка новых разделов после прохождения предыдущего.

### Для работы со структурированным файлом будет использоваться класс библиотеки «Jackson.Core» «ObjectMapper».

На рисунке 2.2 показан пример работы с классом «ObjectMapper».



Рисунок 2.2 — Пример работы с классом «ObjectMapper»

## **2.2 Описание алгоритма решения задач**

Для начала работы необходимо запустить приложение. После чего запуститься основная форма авторизации и регистрации.

Для регистрации нового аккаунта необходимо ввести все данные согласно шаблону. Если такового аккаунта не существует, то в таком случае можно будет зарегистрировать нового пользователя.

Для авторизации и входа в аккаунт необходимо ввести требуемые данные согласно шаблону, после чего выполнить вход. Если данный аккаунт существует, произойдет переход на основную форму. В случае если данный аккаунт не найден, авторизация не пройдет.

### Для начала изучения разделов, необходимо выбрать доступный для изучения раздел. После чего начать его изучение.

### Все разделы становятся доступными в зависимости от результата изучения предыдущего раздела.

### В конце каждого раздела присутствует тест по изученному материалу.

### В приложении предусмотрен просмотр уровня аккаунта.

### Для выхода из обучающего приложения, необходимо нажать на крестик закрытия окна.

### Выйти из аккаунта можно нажав на кнопку «Выход» в разделе «Профиль».

## **2.3 Проектирование макета интерфейса**

Интерфейс данного обучающего приложения будет содержать виджеты на которых будут расположены компоненты для регистрации и авторизации, ввода и вывода данных.

Для создания и открытия файла с информацией об аккаунте будет предусмотрено окно регистрации и авторизации с помощью компонента «TextField».

Для вывода информации будет использован компонент «AlertWindow».

Для подтверждения каких-либо действий будет использован компонент «Button».

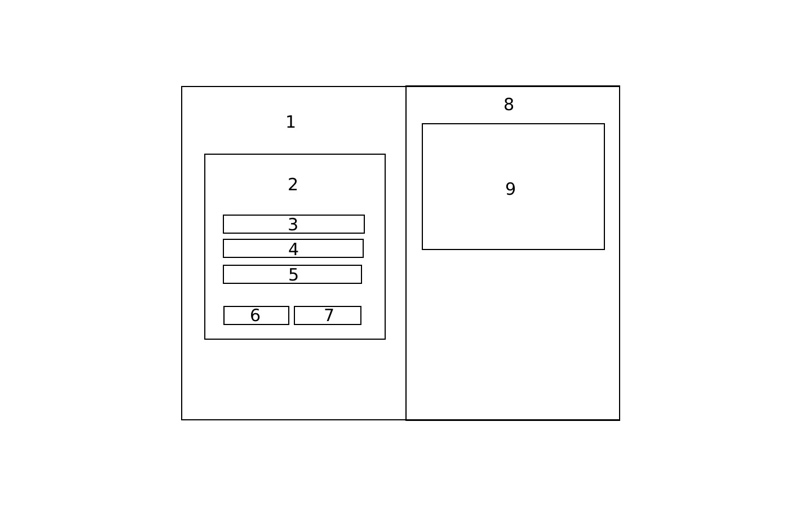
Для расположения виджетов используются компоненты «AnchorPane» и «ScrollPane».

Для хранения текстовой информации используется компонент «Label».

Для хранения многострочной текстовой информации используется компонент «TextArea».

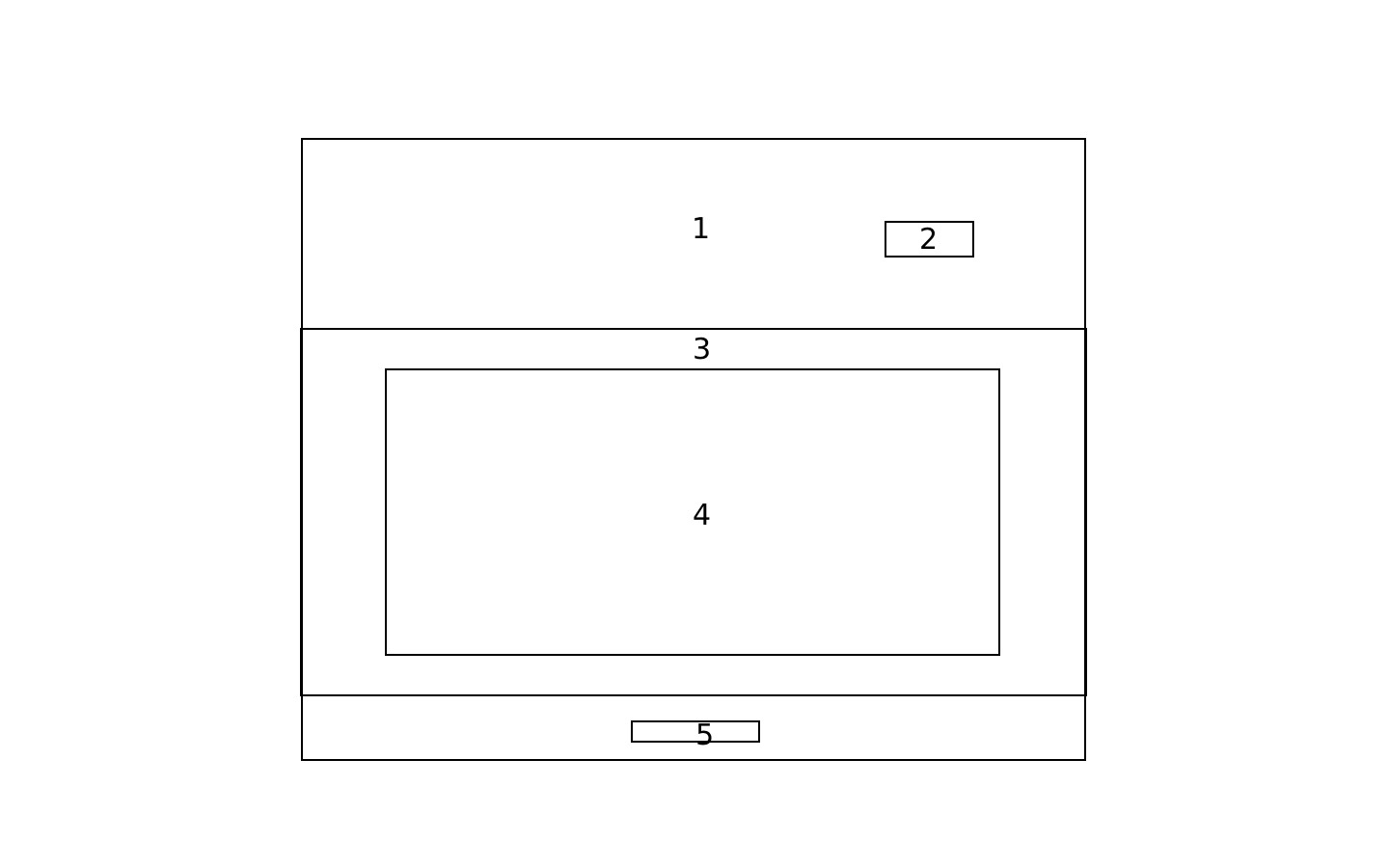
Для хранения изображения используется компонент «ImageView».

На рисунке 2.3 показан макет окна регистрации и авторизации.



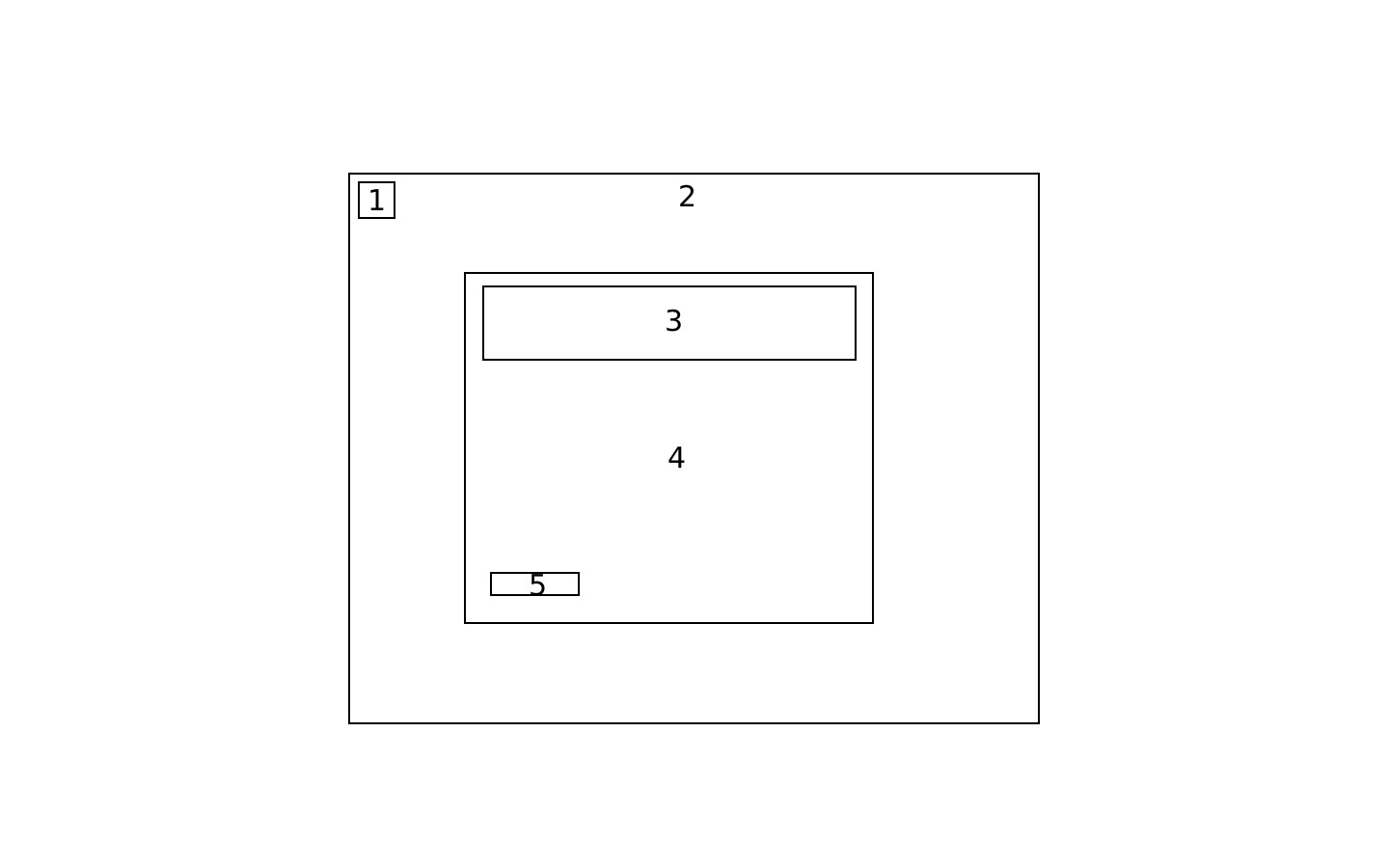
### Рисунок 2.3 — Макет 1: 1– блок главной панели; 2 – блок панели регистрации; 3 – блок строки ввода почты; 4 – блок строки ввода логина; 5 – блок строки ввода пароля; 6 – блок кнопки регистрации; 7 – блок кнопки логина; 8 – блок панели с паттерном; 9 – блок логотипа

На рисунке 2.4 показан макет главного окна разделов.



### Рисунок 2.4 — Макет 2: 1 – блок главной панели; 2 – блок отображения уровня; 3 – блок декоративного виджета для хранения разделов; 4 – блок списка разделов для изучения; 5 – блок для декорации

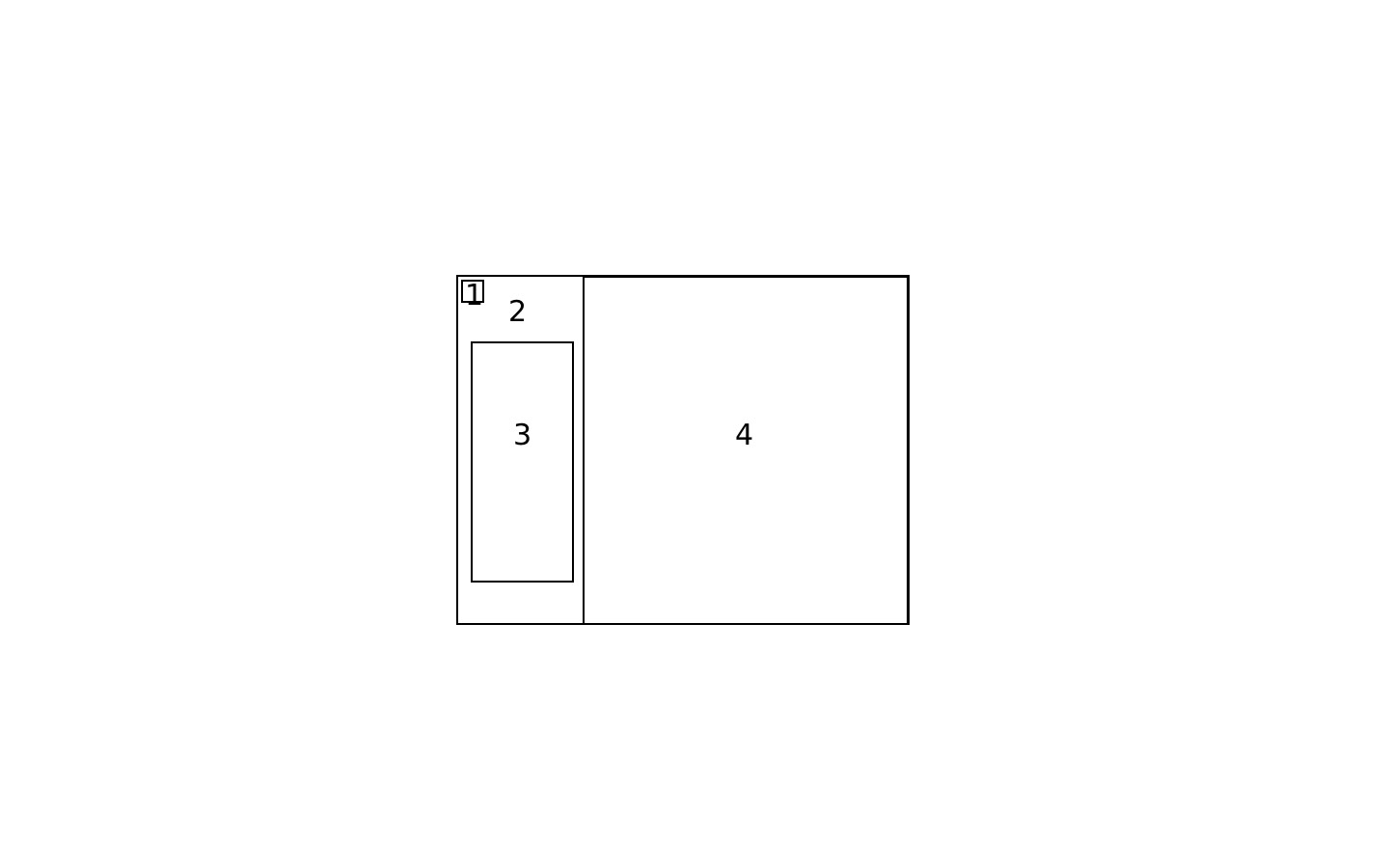
На рисунке 2.5 показан макет окна профиля.



### Рисунок 2.5 — Макет 3: 1 – блок кнопки возврата; 2 – блок главной панели;

### 3 – блок информации об уровне;4 – блок для информации; 5 – блок кнопки выхода

На рисунке 2.6 показан макет окна раздела.



### Рисунок 2.6 — Макет 4: 1 – блок кнопки возврата;

### 2 – блок главной панели; 3 – блок списка тем; 4 – блок панели виджетов

# 3 ВЫПОЛНЕНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ

## **3.1 Описание интерфейса**

### Окно регистрации и авторизации содержит компонент «Label» с помощью которого указано название и изображение приложения. Для ввода данных представлен компонент «TextEdit». Авторизация, регистрации и переход на основное окно осуществляется с помощью компонента «Button».

### Окно авторизации показано на рисунке 3.1.

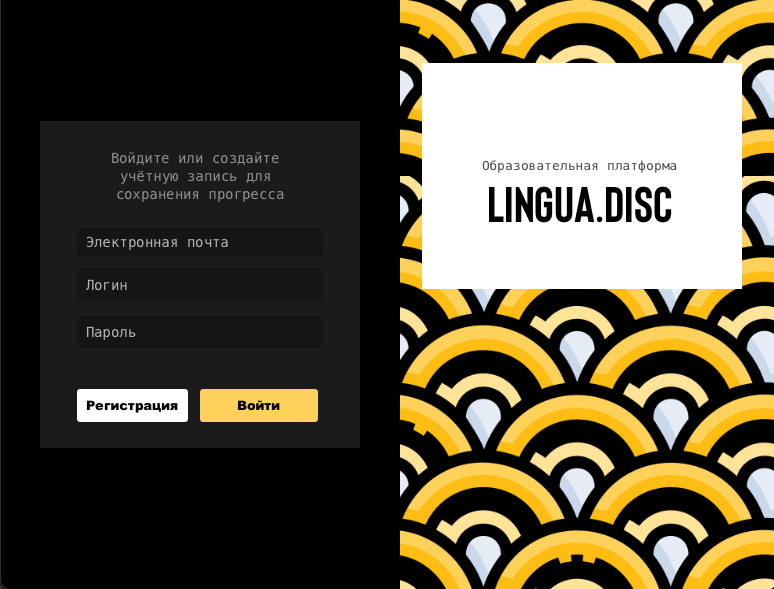
****

Рисунок 3.1 —Окно авторизации

Основное окно разделов с темами содержит компоненты «AnchorPane»,

«ImageView», «Label» и «Button». Компонент «Button» осуществляет переход на окно соответствующего раздела или окна.

Основное окно показана на рисунке 3.2.

****

Рисунок 3.2 —Основная форма

### Окно «Профиль» содержит компоненты «AnchorPane», «Button», «Label», «ImageView». Для возврата на основное окно или окно авторизации используется компонент «Button».

### В разделе «Профиль» отображается следующая информация:

* почта;
* логин;
* ошибки;
* лучшая оценка;
* уровень.

Окно «Профиль» показано на рисунке 3.3.

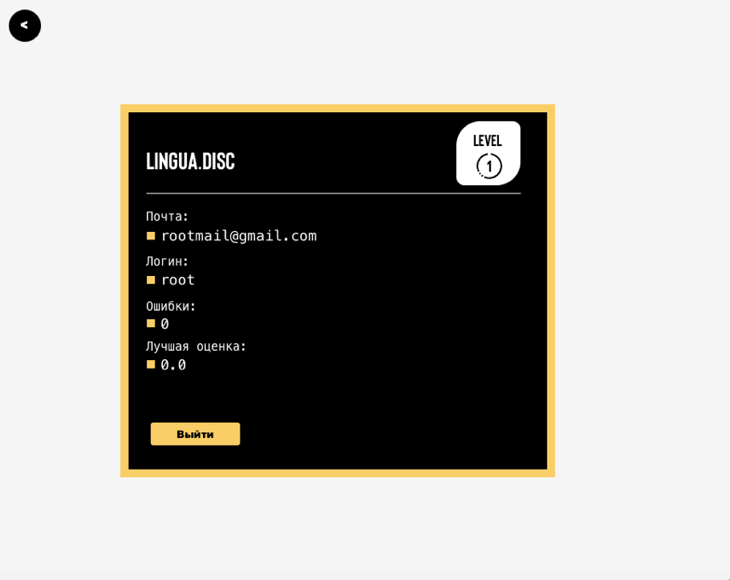


Рисунок 3.3 — Окно «Профиль»

Окно «Список литературы» содержит компоненты «AnchorPane», «Button», «Label». Для возврата на основное окно используется компонент «Button». Вся информация о списке рекомендуемой литературы содержится в компоненте «Label».

Окно «Список литературы» показано на рисунке 3.4.

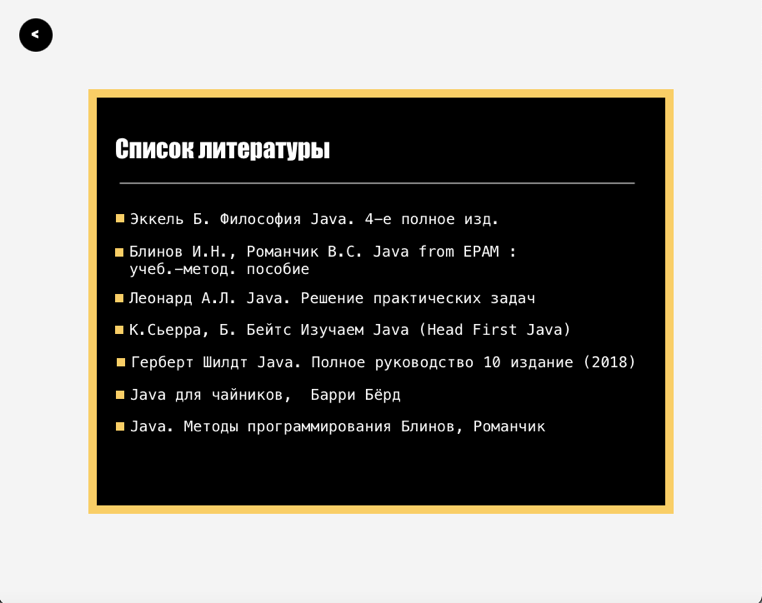
****

Рисунок 3.4 — Окно «Список литературы»

Окно каждого теоретического раздела содержит компоненты «AnchorPane», «Button», «Label», «ImageView». Для возврата на основное окно, а также для переключения между разделами с теоретическими данными используется компонент «Button».

Окно теоретического раздела показано на рисунке 3.5.

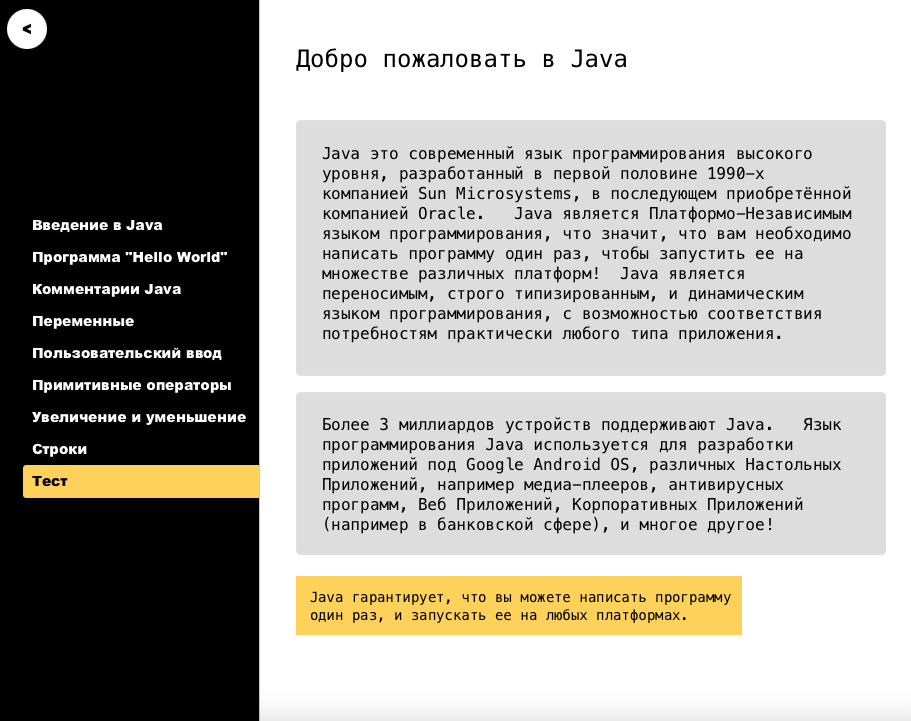
****

Рисунок 3.5 — Окно теоретического раздела

### Теоретические данные находятся в компонентах «ScrollPane», «AnchorPane», «TextArea» и «Label».

### Пример теоретических данных показан на рисунке 3.6.

### 

### Рисунок 3.6 — Пример теоретических данных

### Для ввода данных при выполнении практических заданий в конце каждого раздела используются компоненты «TextEdit» и «CheckBox».

### Пример теоретических заданий показан на рисунке 3.7.

### 

### Рисунок 3.7 — Пример теоретических заданий

### Окно «Контрольный тест» состоит из компонентов «AnchorPane», «Button», «RadioButton».

### Пример окна «Контрольный тест» показан на рисунке 3.8.

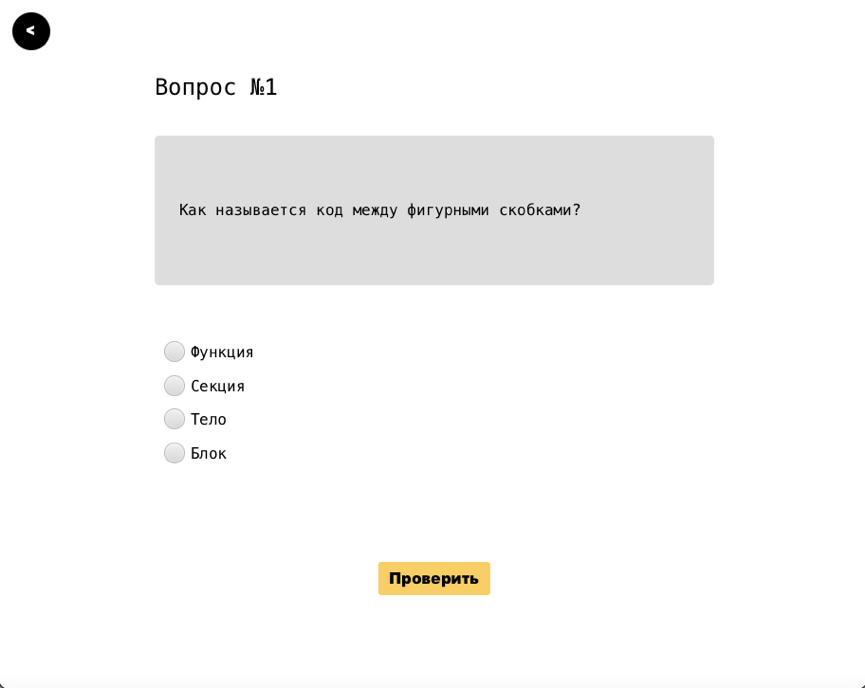


Рисунок 3.8 — Окно «Контрольный тест»

## **3.2 Программная реализации каждой подпрограммы**

В программе реализованы функции: регистрации и авторизации аккаунта, проверка наличия существующего аккаунта, чтение, запись и перезапись файла, проверка наличия файла и его создания, переключение между виджетами, проверка вводимых данных, очистка введенных данных, проверка корректности тестов.

Регистрация аккаунта происходит при нажатии кнопки «Регистрации», после введения логина и пароля. Происходит чтение из файла и его перезапись с дополнением.

На рисунке 3.9 показан код кнопки «Регистрации».

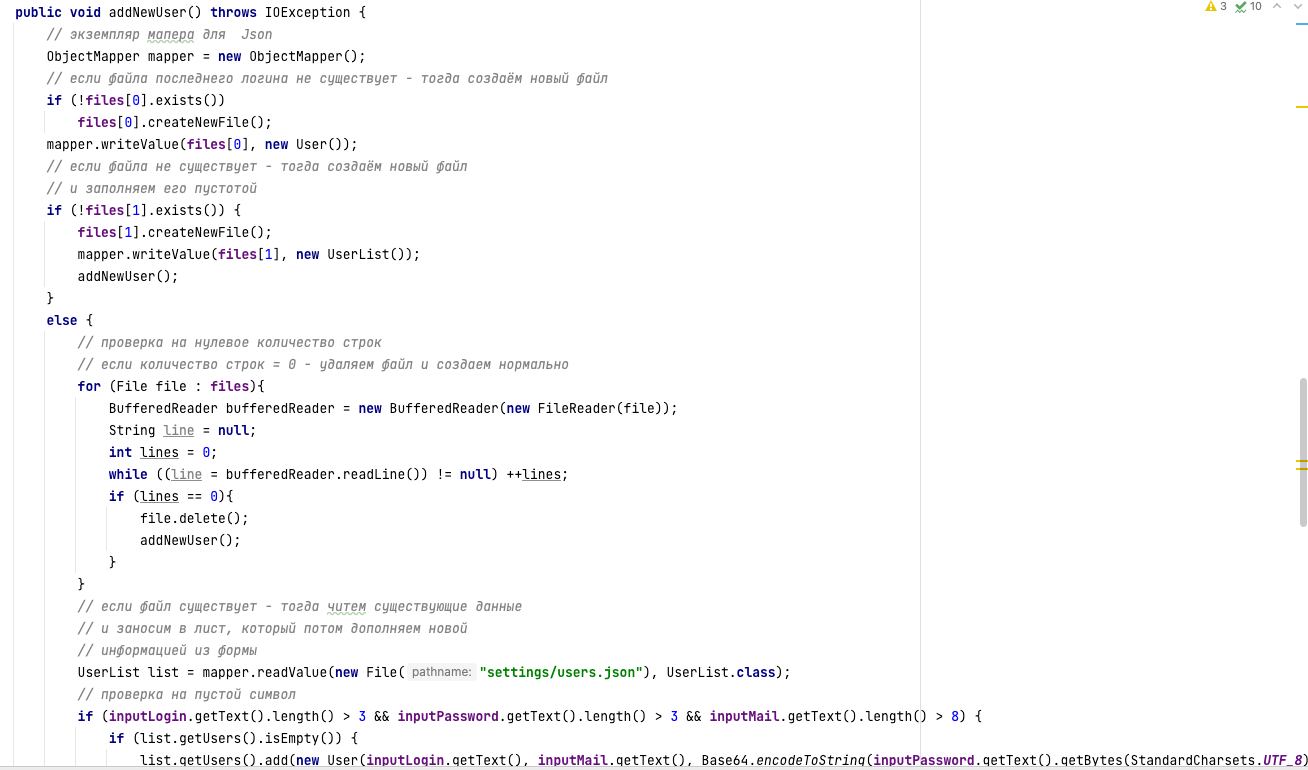


Рисунок 3.9 — Код кнопки «Регистрация»

Авторизация аккаунта происходит при нажатии кнопки «Войти», после введения логина и пароля. Происходит чтение из файла и проверка совпадения данных.

На рисунке 3.10 показан код кнопки «Войти».

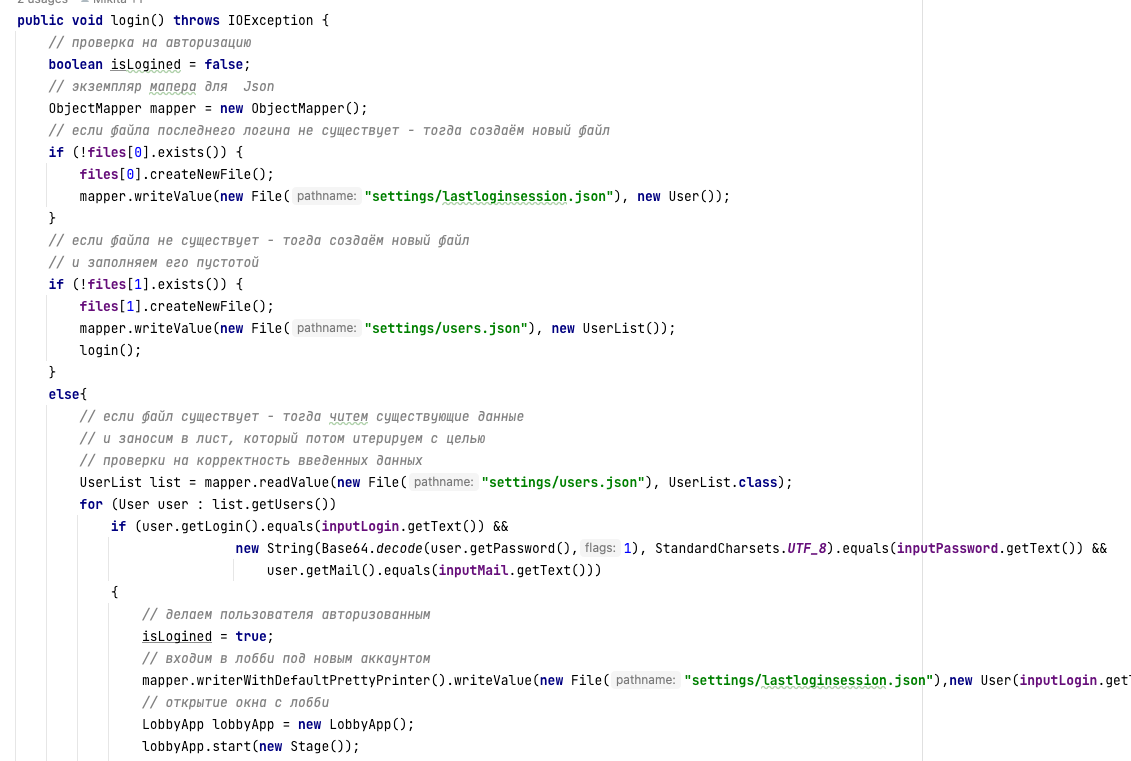


Рисунок 3.10 — Код кнопки «Войти»

Переключение виджетов происходит путем изменения видимости для выбранного виджета. Изначально видимость всех виджетов кроме первого отключена, при переключении на некоторый виджет, видимость данного виджета становиться активной.

Код переключения виджетов показан на рисунке 3.11.

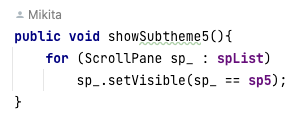


Рисунок 3.11 — Код переключения виджетов

При прохождении теста теоретического раздела выполняется проверка. В случае выявления неправильного ответа всплывает окно с ошибкой.

Код проверки тестов показан на рисунке 3.12.

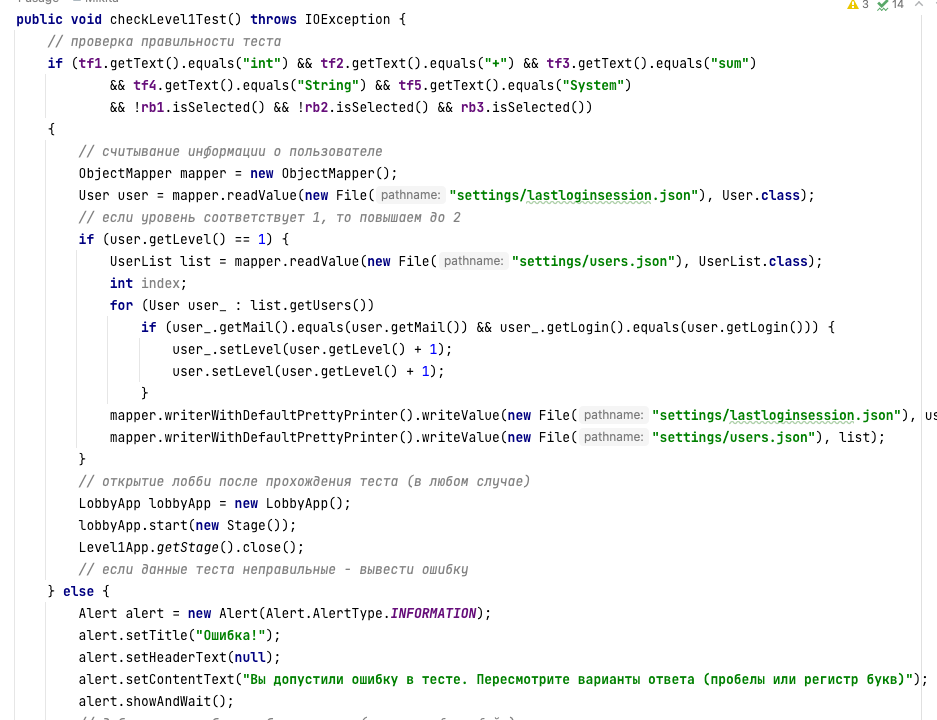


Рисунок 3.12 — Код проверки тестов

Для контроля прохождения разделов в соответствующем порядке используются условия для проверки уровня пользователя. Сперва, с файла считываются данные, которые формируют объект пользователя.

После этого уровень пользователя сравнивается с требуемым. Если уровень совпадает с требуемым – разблокировать соответствующий раздел.

Код контроля прохождения разделов показан на рисунке 3.13.

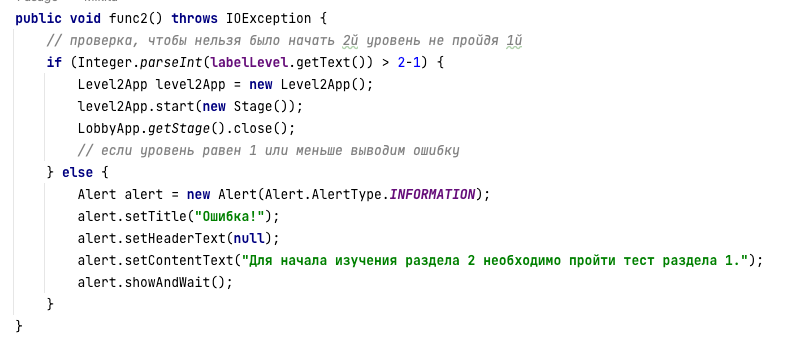


Рисунок 3.13 — Код контроля прохождения разделов

При прохождении контрольного теста происходит проверка на то, выбран ли правильный ответ. Если выбран правильный ответ, то повышается число правильных ответов.

Код проверки ответа контрольного теста показан на рисунке 3.14.

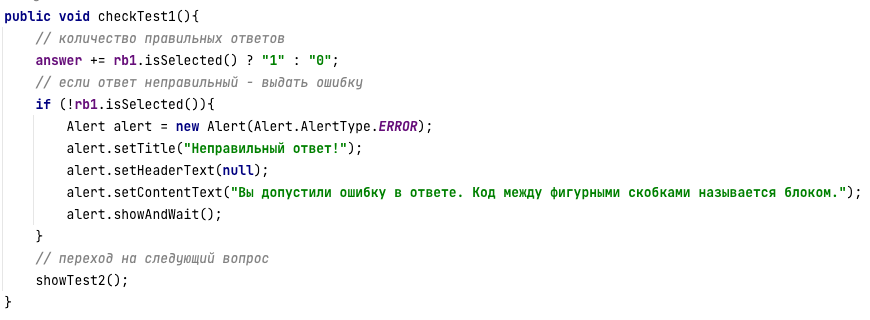


Рисунок 3.14 — Код проверки ответа контрольного теста

В конце контрольного теста производится подсчёт общего числа правильных и неправильных ответов.

Код подсчёта общего числа правильных и неправильных ответов показан на рисунке 3.15.



Рисунок 3.15 — Код подсчёта общего числа правильных и неправильных ответов

Перед сохранением пароля пользователя в файл производится его кодирование. Перед чтением пароль декодируется. Кодирование и декодирование сделаны на базе шифрования «Base64».

Base64  – это группа схожих [binary-to-text encoding](https://en.wikipedia.org/wiki/Binary-to-text_encoding) схем, которые представляют двоичные данные в ASCII-формате методом перевода в radix-64 представление [6].

Код кодирования пароля показан на рисунке 3.16.

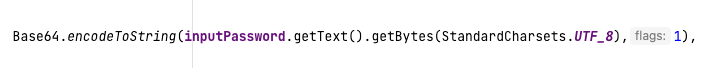


Рисунок 3.16 — Код кодирования пароля

### Полный код программы представлен в приложении А.

## **3.3 Контроль корректности ввода данных**

В программе реализованы функции: регистрации и авторизации аккаунта, проверка наличия существующего аккаунта, чтение и запись в файл, проверка наличия файла и его создание, переключение между виджетами, проверка вводимых данных, шифрование пароля.

Во время авторизации аккаунта происходят следующие проверки:

* проверка на совпадение логина, пароля и почты;
* проверка на наличие файлов с данными;
* проверка на статус авторизации пользователя.

Во время проверки на совпадение логина, пароля и почты происходит обход по списку со всеми существующими аккаунтами в файле и сравнение всех значений с введёнными.

Код проверки представлен на рисунке 3.17.

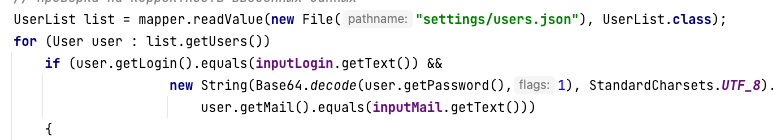


Рисунок 3.17 — Код проверки на совпадение логина, пароля и почты

Во время проверки на наличие файлов с данными проверяется наличие файлов в специальной папке. В случае их отсутствия происходит создание новых.

Код проверки представлен на рисунке 3.18.



Рисунок 3.18 — Код проверки на наличие файлов с данными

Во время проверки на статус авторизации пользователя происходит проверка статуса авторизации. В случае, если пользователь не авторизован, то выдать ошибку [7].

Код проверки представлен на рисунке 3.19.

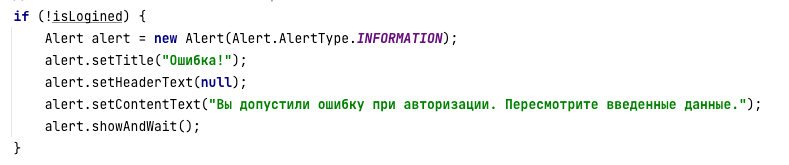


Рисунок 3.19 — Код проверки на наличие файлов с данными

# 4 РЕЗУЛЬТАТ РАБОТЫ

### При открытии исполняемого файла программы появляется окно авторизации. Окно авторизации содержит информацию о приложении, компоненты ввода почты, логина и пароля, кнопки входа и регистрации.

### Пример окна авторизации показан на рисунке 4.1.

### 

### Рисунок 4.1 — Пример окна регистрации и авторизации

### Для регистрации аккаунта необходимо ввести почту, логин и пароль, после чего нажать кнопку регистрации.

### Пример ввода данных показан на рисунке 4.2.

### 

### Рисунок 4.2 — Пример ввода данных

### Если регистрация прошла успешно, то в файлах с записями о пользователях появится соответствующая созданному аккаунту запись с зашифрованным паролем.

### Результат регистрации показан на рисунке 4.3.

### 

### Рисунок 4.3 — Результат регистрации

### В случае если аккаунт с таким именем или почтой уже существует, регистрация не выполнится. Будет выдано соответствующее сообщение с ошибкой.

### Пример сообщения показан на рисунке 4.4.

### 

### Рисунок 4.4 — Пример сообщения

### В случае отсутствия какого-либо файла с аккаунтами, при регистрации файл будет создан автоматически.

### Пример папки программы до регистрации какого-либо аккаунта показан на рисунке 4.5.

### 

### Рисунок 4.5 — Папка до регистрации какого-либо аккаунта

### Пример папки после регистрации аккаунта показан на рисунке 4.6.

### 

### Рисунок 4.6 — Папка после регистрации аккаунта

### Результат регистрации аккаунта в файл показан на рисунке 4.7.

### 

### Рисунок 4.7 — Результат регистрации аккаунта в файл

### В случае, если пользователь допускает какую-либо ошибку при авторизации, то будет выведено сообщение об ошибке.

### Пример сообщения об ошибке при авторизации показан на рисунке 4.8.

### 

### Рисунок 4.8 — Пример сообщения об ошибке при авторизации

### При авторизации существующего аккаунта, будет выполнен переход на основное окно. На основном окне представлены теоретические разделы, раздел профиля, список литературы и контрольный тест.

### Результат авторизации показан на рисунке 4.9.

### 

### Рисунок 4.9 — Результат авторизации.

### При нажатии на область кнопки «Профиль», будет открыто окно с информацией об аккаунте, с которого авторизовался пользователь.

### Результат нажатия кнопки «Профиль показан на рисунке 4.10.

### 

### Рисунок 4.10 — Результат нажатия кнопки «Профиль»

Также в окне профиля можно выйти из данного аккаунта и авторизоваться под новым. После нажатия пользователь будет перенаправлен на окно авторизации.

Результат нажатия кнопки «Выйти» показан на рисунке 4.11.

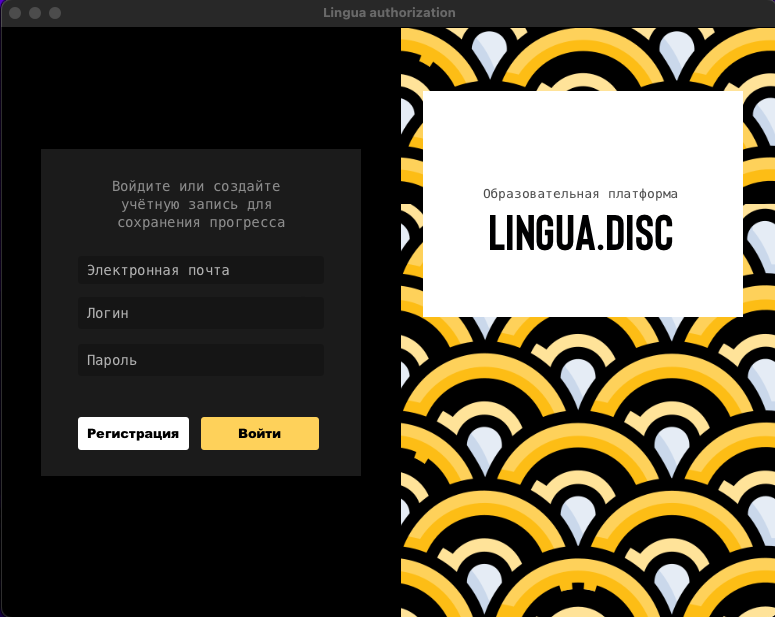


Рисунок 4.11 — Результат нажатия кнопки «Выйти»

### При нажатии кнопки «Список литературы», будет открыто окно со списком литературы, которая поможет пользователю в изучении языка программирования «Java».

### Результат нажатия кнопки «Список литературы» показан ниже, на рисунке 4.12.

### 

### Рисунок 4.12 — Результат нажатия кнопки «Список литературы»

### При нажатии на доступный для изучения раздел, будет открыто окно со списком тем данного раздела и информацией о данном разделе.

### Пример открытия доступного раздела показан на рисунке 4.13.

### 

### Рисунок 4.13 — Пример открытия доступного раздела

В случае, если уровень пользователя ниже необходимого для изучения раздела, то будет открыто окно с ошибкой.

Пример ошибки при попытке изучения недоступного раздела показан на рисунке 4.14.



Рисунок 4.14 — Пример ошибки при попытке изучения недоступного раздела

В каждом разделе есть кнопка «Тест», при нажатии на которую открывается раздел с тестом по изученному материалу.

Пример нажатия кнопки практических заданий показан ниже, на рисунке 4.15.



Рисунок 4.15 — Пример нажатия кнопки «Тест»

### Для выполнения теста необходимо ввести данные, которые требуются по условию задания. После ввода данных необходимо нажать кнопку «Проверить».

### Пример ввода данных в разделе «Тест» показан на рисунке 4.16.

### 

### Рисунок 4.16 — Пример ввода данных в разделе «Тест»

### При нажатии кнопки «Проверить», выполняется проверка введенных данных, если данные были введены не верно, будет открыто окно с информацией о том, что в тесте была допущена ошибка.

### Результат неправильно введенных данных показан на рисунке 4.17.

### 

### Рисунок 4.17 — Результат неправильного ввода данных

### В случае, если пользователь правильно ответил на все вопросы, у него повысится уровень и он будет перенаправлен на главное окно.

### Результат правильного ввода данных показан на рисунке 4.18.

### 

### Рисунок 4.18 — Результат правильного ввода данных

### Пример отображаемой информации о прохождении тестов в разделе профиля показан на рисунке 4.19.

### 

### Рисунок 4.19 — Пример добавления информации о пройденном разделе

В любой момент пользователь может пройти контрольный тест. Контрольный тест состоит из некоторых вопросов по языку программирования «Java».

Пример окна с контрольным тестом показан на рисунке 4.20.



Рисунок 4.20 — Пример окна с контрольным тестом

Для начала прохождения контрольного теста пользователь должен нажать кнопку «Пройти тест».

Пример прохождения контрольного теста показан на рисунке 4.21.

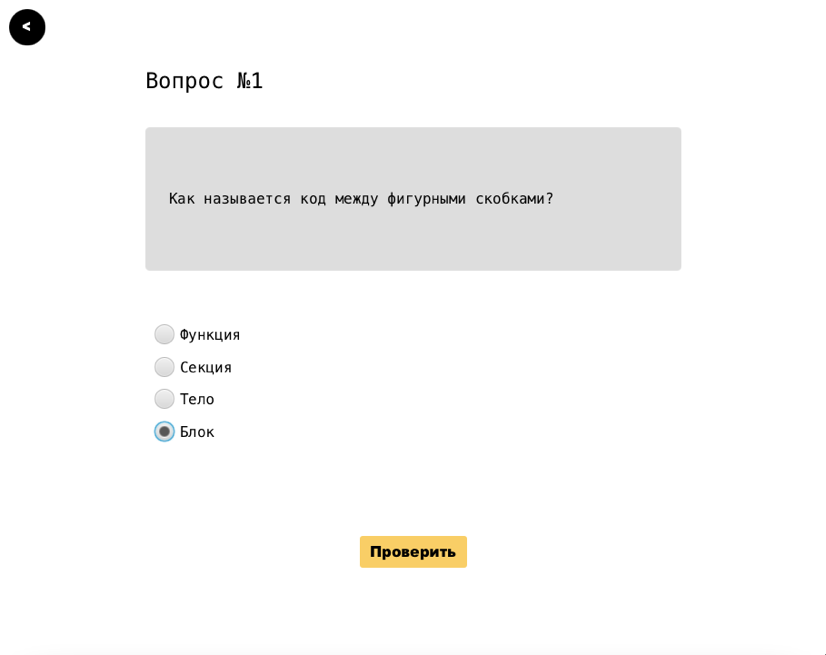


Рисунок 4.21 — Пример прохождения контрольного теста

Для ответа на вопрос необходимо выбрать один из предложенных вариантов ответа. После этого, для проверки, нужно нажать на кнопку «Проверить». Когда пользователь проверит ответ, его перенаправит на следующий вопрос.

В случае, если ответ оказался ложным, пользователю будет выведено окно с предупреждением, в котором написан правильный ответ и пояснение.

Пример окна ошибки при неправильном ответе показан ниже, на рисунке 4.22.

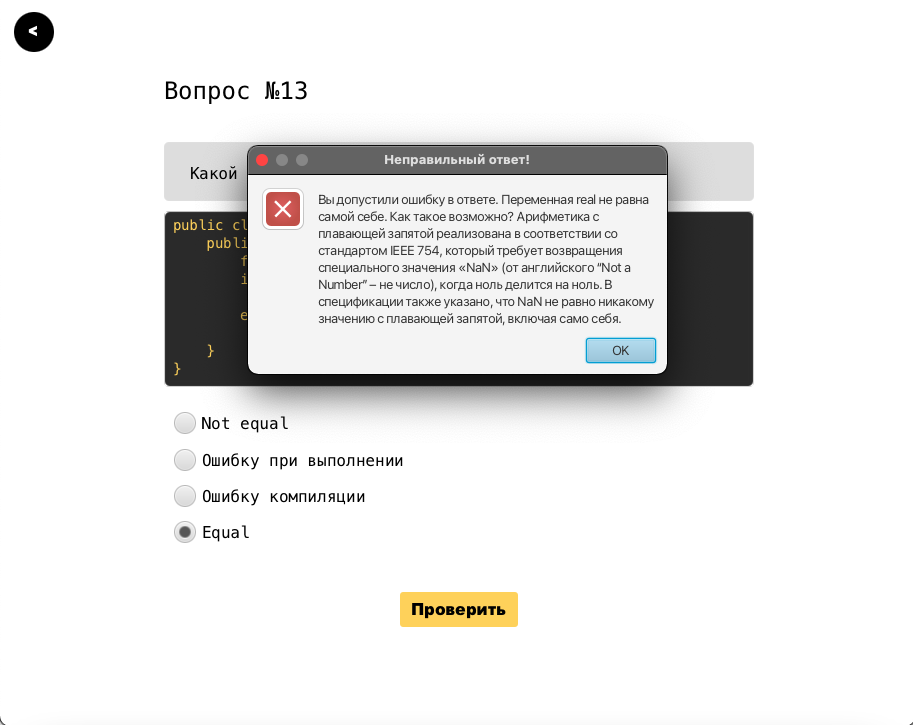


Рисунок 4.22 — Пример окна ошибки при неправильном ответе

После прохождения контрольного теста пользователь сможет увидеть статистику ответов на вопросы и средний балл.

Пример статистики в контрольном тесте показан на рисунке 4.23.

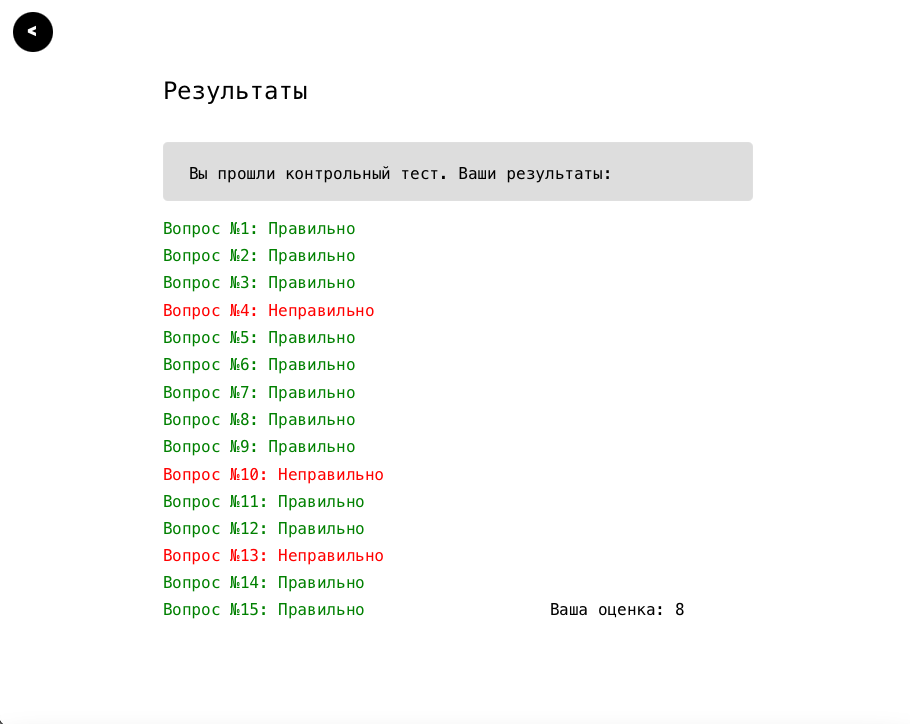


Рисунок 4.23 — Пример статистики в контрольном тесте

# ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы было разработано обучающее приложение по языку программирования «Java».

Были выполнены следующие задачи:

* изучены теоретические данные на тему «Представление деревьев. Основные операции над деревьями»;
* закреплены знания о деревьях;
* организована работа со структурированным файлом;
* разработан графический пользовательский интерфейс;
* разработаны алгоритмы программы;
* осуществлено модульное тестирование программы.

Программа предназначена для людей, что хотят изучить теоретические сведения по предложенному материалу.

Помимо теоретических сведений, в данном обучающем приложении присутствуют разные тесты для закрепления пройденного материала.

Функционал программы позволяет автоматически создавать файл, создавать и авторизовать аккаунт, изучать теоретические сведенья, проходить разного рода тесты для проверки своих знаний.

### При проектировании программы были изучены теоретические сведенья на тему «Представление деревьев. Основные операции над деревьями».

Учащимся продемонстрированы прекрасные навыки работы с языком программирования «Java» и его окружением, в том числе с библиотеками «JavaFX» и «Jackson».

# Cписок использованных источников

1. Представление деревьев. Основные операции над деревьями [Электронный ресурс]. – Электронные данные. – Режим доступа: http://www.pascaler.ru/pascal/dynamics/derevo/2/. – Дата доступа: 17.05.2022;
2. Дерево (Структура данных) [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://ru.wikipedia.org/wiki/Дерево_(Структура> данных). – Дата доступа: 17.05.2022;
3. SoloLearn курсы Java [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://www.sololearn.com/learning/1068>. – Дата доступа: 18.05.2022;
4. Эккель Б. Философия Java. 4-е полное изд. – СПб.: Питер, 2015. – 1168 с.: ил. – (Серия «Классика computer science»);
5. Леонард А. Java. Решение практических задач : пер. с англ. – СПб.: БХВ-Петербург, 2021. – 720 с.: ил.
6. Блинов, И. Н., Романчик, В. С. Java from EPAM : учеб.-метод. пособие / И. Н. Блинов, В. С. Романчик. – Минск : Четыре четверти, 2020. – 560 с.
7. Сиерра К., Бейтс Б. Head First Java / Изучаем Java. 2-е полное изд.: учеб.-метод. пособие / Сиерра К., Бейтс Б. – СПб.: Питер, 2012. – 708 с.: ил.
8. Шилдт, Герберт. Java. Полное руководство, 10-е изд. : Пер. с англ. -СПб. ООО "Альфа­ книга'; 2018. - 1488 с. : ил. - Парал. тит. англ.

# ПРИЛОЖЕНИЕ А

### (обязательное)

### Листинг программы

Текст модуля AuthorizationApp.java

package com.lvl80.lingua.Authorization;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
import java.io.IOException;  
  
  
public class AuthorizationApp extends Application {  
  
 private static Stage stage;  
  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(com.lvl80.lingua.Authorization.AuthorizationApp.class.getResource("AuthorizationApp.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua authorization");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля AuthorizationAppController.java

package com.lvl80.lingua.Authorization;

import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import com.lvl80.lingua.Types.UserList;  
import com.lvl80.lingua.Utils.Base64;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.PasswordField;  
import javafx.scene.control.TextField;  
import javafx.stage.Stage;  
  
import java.io.\*;  
import java.nio.charset.StandardCharsets;  
  
public class AuthorizationAppController {  
 // массив файлов  
 private File[] files = {new File("settings/lastloginsession.json"), new File("settings/users.json")};  
 @FXML  
 private TextField inputLogin;  
 @FXML  
 private TextField inputMail;  
 @FXML  
 private PasswordField inputPassword;  
  
 // логин в систему  
 public void login() throws IOException {  
 // проверка на авторизацию  
 boolean isLogined = false;  
 // экземпляр мапера для Json  
 ObjectMapper mapper = new ObjectMapper();  
 // если файла последнего логина не существует - тогда создаём новый файл  
 if (!files[0].exists()) {  
 files[0].createNewFile();  
 mapper.writeValue(new File("settings/lastloginsession.json"), new User());  
 }  
 // если файла не существует - тогда создаём новый файл  
 // и заполняем его пустотой  
 if (!files[1].exists()) {  
 files[1].createNewFile();  
 mapper.writeValue(new File("settings/users.json"), new UserList());  
 login();  
 }  
 else{  
 // если файл существует - тогда читем существующие данные  
 // и заносим в лист, который потом итерируем с целью  
 // проверки на корректность введенных данных  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 for (User user : list.getUsers())  
 if (user.getLogin().equals(inputLogin.getText()) &&  
 new String(Base64.decode(user.getPassword(),1), StandardCharsets.UTF\_8).equals(inputPassword.getText()) &&  
 user.getMail().equals(inputMail.getText()))  
 {  
 // делаем пользователя авторизованным  
 isLogined = true;  
 // входим в лобби под новым аккаунтом  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"),new User(inputLogin.getText(),inputMail.getText(),Base64.encodeToString(inputPassword.getText().getBytes(StandardCharsets.UTF\_8),1),user.getLevel(),user.getMistakes(), user.getMark()));  
 // открытие окна с лобби  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 AuthorizationApp.getStage().close();  
 }  
 // если пользователь не смог авторизоваться  
 if (!isLogined) {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку при авторизации. Пересмотрите введенные данные.");  
 alert.showAndWait();  
 }  
 }  
 }  
  
 // регистрация новых пользователей  
 public void addNewUser() throws IOException {  
 // экземпляр мапера для Json  
 ObjectMapper mapper = new ObjectMapper();  
 // если файла последнего логина не существует - тогда создаём новый файл  
 if (!files[0].exists())  
 files[0].createNewFile();  
 mapper.writeValue(files[0], new User());  
 // если файла не существует - тогда создаём новый файл  
 // и заполняем его пустотой  
 if (!files[1].exists()) {  
 files[1].createNewFile();  
 mapper.writeValue(files[1], new UserList());  
 addNewUser();  
 }  
 else {  
 // проверка на нулевое количество строк  
 // если количество строк = 0 - удаляем файл и создаем нормально  
 for (File file : files){  
 BufferedReader bufferedReader = new BufferedReader(new FileReader(file));  
 String line = null;  
 int lines = 0;  
 while ((line = bufferedReader.readLine()) != null) ++lines;  
 if (lines == 0){  
 file.delete();  
 addNewUser();  
 }  
 }  
 // если файл существует - тогда читем существующие данные  
 // и заносим в лист, который потом дополняем новой  
 // информацией из формы  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 // проверка на пустой символ  
 if (inputLogin.getText().length() > 3 && inputPassword.getText().length() > 3 && inputMail.getText().length() > 8) {  
 if (list.getUsers().isEmpty()) {  
 list.getUsers().add(new User(inputLogin.getText(), inputMail.getText(), Base64.encodeToString(inputPassword.getText().getBytes(StandardCharsets.UTF\_8),1), 1,0,0f));  
 // запись в файл со всеми пользователями  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"),list);  
 // входим в лобби под новым аккаунтом  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"),new User(inputLogin.getText(),inputMail.getText(), Base64.encodeToString(inputPassword.getText().getBytes(StandardCharsets.UTF\_8),1),1,0,0f));  
 // очищаем поля ввода  
 inputLogin.clear();  
 inputMail.clear();  
 inputPassword.clear();  
 // открытие окна с лобби  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 AuthorizationApp.getStage().close();  
 }  
 else {  
 // статус регистрации  
 boolean registrationStatus = true;  
 for (User user : list.getUsers())  
 // если данные заняты - вывести ошибку  
 if (user.getLogin().equals(inputLogin.getText()) || user.getMail().equals(inputMail.getText())) {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Данный логин или почта уже заняты.");  
 alert.showAndWait();  
 registrationStatus = false;  
 break;  
 }  
 // если логин и почта не заняты - то добавляем нового пользователя в список  
 if (registrationStatus) {  
 list.getUsers().add(new User(inputLogin.getText(), inputMail.getText(), Base64.encodeToString(inputPassword.getText().getBytes(StandardCharsets.UTF\_8),1), 1, 0,0f));  
 // запись в файл со всеми пользователями  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 // входим в лобби под новым аккаунтом  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), new User(inputLogin.getText(), inputMail.getText(), Base64.encodeToString(inputPassword.getText().getBytes(StandardCharsets.UTF\_8),1), 1, 0,0f));  
 // очищаем поля ввода  
 inputLogin.clear();  
 inputMail.clear();  
 inputPassword.clear();  
 // открытие окна с лобби  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 AuthorizationApp.getStage().close();  
 }  
 }  
 }  
 }  
 }  
}

Текст модуля Level1App.java

package com.lvl80.lingua.Level1;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
import java.io.IOException;  
  
  
public class Level1App extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(com.lvl80.lingua.Level1.Level1App.class.getResource("Level1App.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля Level1AppController.java

package com.lvl80.lingua.Level1;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import com.lvl80.lingua.Types.UserList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.ScrollPane;  
import javafx.scene.control.TextField;  
import javafx.scene.layout.AnchorPane;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Level1AppController {  
 @FXML  
 private ScrollPane sp1;  
 @FXML  
 private ScrollPane sp2;  
 @FXML  
 private ScrollPane sp3;  
 @FXML  
 private ScrollPane sp4;  
 @FXML  
 private ScrollPane sp5;  
 @FXML  
 private ScrollPane sp6;  
 @FXML  
 private ScrollPane sp7;  
 @FXML  
 private ScrollPane sp8;  
 @FXML  
 private ScrollPane sp9;  
 @FXML  
 private TextField tf1;  
 @FXML  
 private TextField tf2;  
 @FXML  
 private TextField tf3;  
 @FXML  
 private TextField tf4;  
 @FXML  
 private TextField tf5;  
 @FXML  
 private RadioButton rb1;  
 @FXML  
 private RadioButton rb2;  
 @FXML  
 private RadioButton rb3;  
  
  
 private List<ScrollPane> spList = new ArrayList<>();  
  
 @FXML  
 public void initialize(){  
 spList.add(sp1);  
 spList.add(sp2);  
 spList.add(sp3);  
 spList.add(sp4);  
 spList.add(sp5);  
 spList.add(sp6);  
 spList.add(sp7);  
 spList.add(sp8);  
 spList.add(sp9);  
 }  
  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level1App.getStage().close();  
 }  
  
 // функции работы с темами (переключение тем)  
 public void showSubtheme1(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp1);  
 }  
  
 public void showSubtheme2(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp2);  
 }  
  
 public void showSubtheme3(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp3);  
 }  
  
 public void showSubtheme4(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp4);  
 }  
  
 public void showSubtheme5(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp5);  
 }  
  
 public void showSubtheme6(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp6);  
 }  
  
 public void showSubtheme7(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp7);  
 }  
  
 public void showSubtheme8(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp8);  
 }  
  
 public void showSubtheme9(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp9);  
 }  
  
 public void checkLevel1Test() throws IOException {  
 // проверка правильности теста  
 if (tf1.getText().equals("int") && tf2.getText().equals("+") && tf3.getText().equals("sum")  
 && tf4.getText().equals("String") && tf5.getText().equals("System")  
 && !rb1.isSelected() && !rb2.isSelected() && rb3.isSelected())  
 {  
 // считывание информации о пользователе  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 // если уровень соответствует 1, то повышаем до 2  
 if (user.getLevel() == 1) {  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 int index;  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setLevel(user.getLevel() + 1);  
 user.setLevel(user.getLevel() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 // открытие лобби после прохождения теста (в любом случае)  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level1App.getStage().close();  
 // если данные теста неправильные - вывести ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в тесте. Пересмотрите варианты ответа (пробелы или регистр букв)");  
 alert.showAndWait();  
 // добавление ошибки к общему числу (запись инфы в файл)  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setMistakes(user.getMistakes() + 1);  
 user.setMistakes(user.getMistakes() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 }  
}

Текст модуля Level2App.java

package com.lvl80.lingua.Level2;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
import java.io.IOException;  
  
  
public class Level2App extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(com.lvl80.lingua.Level2.Level2App.class.getResource("Level2App.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля Level2AppController.java

package com.lvl80.lingua.Level2;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import com.lvl80.lingua.Types.UserList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.ScrollPane;  
import javafx.scene.control.TextField;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Level2AppController {  
 @FXML  
 private ScrollPane sp1;  
 @FXML  
 private ScrollPane sp2;  
 @FXML  
 private ScrollPane sp3;  
 @FXML  
 private ScrollPane sp4;  
 @FXML  
 private ScrollPane sp5;  
 @FXML  
 private ScrollPane sp6;  
 @FXML  
 private ScrollPane sp7;  
 @FXML  
 private ScrollPane sp8;  
 @FXML  
 private ScrollPane sp9;  
 @FXML  
 private TextField tf1;  
 @FXML  
 private TextField tf2;  
 @FXML  
 private TextField tf3;  
 @FXML  
 private TextField tf4;  
 @FXML  
 private TextField tf5;  
 @FXML  
 private TextField tf6;  
 @FXML  
 private RadioButton rb1;  
 @FXML  
 private RadioButton rb2;  
 @FXML  
 private RadioButton rb3;  
 @FXML  
 private RadioButton rb4;  
  
  
 private List<ScrollPane> spList = new ArrayList<>();  
  
 @FXML  
 public void initialize(){  
 spList.add(sp1);  
 spList.add(sp2);  
 spList.add(sp3);  
 spList.add(sp4);  
 spList.add(sp5);  
 spList.add(sp6);  
 spList.add(sp7);  
 spList.add(sp8);  
 spList.add(sp9);  
 }  
  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level2App.getStage().close();  
 }  
  
 // функции работы с темами (переключение тем)  
 public void showSubtheme1(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp1);  
 }  
  
 public void showSubtheme2(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp2);  
 }  
  
 public void showSubtheme3(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp3);  
 }  
  
 public void showSubtheme4(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp4);  
 }  
  
 public void showSubtheme5(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp5);  
 }  
  
 public void showSubtheme6(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp6);  
 }  
  
 public void showSubtheme7(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp7);  
 }  
  
 public void showSubtheme8(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp8);  
 }  
  
 public void showSubtheme9(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp9);  
 }  
  
 public void checkLevel2Test() throws IOException {  
 // проверка правильности теста  
 if (tf1.getText().equals("7") && tf2.getText().equals("x") && tf3.getText().equals("&&")  
 && tf4.getText().equals("72") && tf5.getText().equals("for") && tf6.getText().equals("x")  
 && rb1.isSelected() && !rb2.isSelected() && rb3.isSelected() && !rb4.isSelected())  
 {  
 // считывание информации о пользователе  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 // если уровень соответствует 2, то повышаем до 3  
 if (user.getLevel() == 2) {  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 int index;  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setLevel(user.getLevel() + 1);  
 user.setLevel(user.getLevel() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 // открытие лобби после прохождения теста (в любом случае)  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level2App.getStage().close();  
 // если данные теста неправильные - вывести ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в тесте. Пересмотрите варианты ответа (пробелы или регистр букв)");  
 alert.showAndWait();  
 // добавление ошибки к общему числу (запись инфы в файл)  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setMistakes(user.getMistakes() + 1);  
 user.setMistakes(user.getMistakes() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 }  
}

Текст модуля Level3App.java

package com.lvl80.lingua.Level3;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
import java.io.IOException;  
  
  
public class Level3App extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(com.lvl80.lingua.Level3.Level3App.class.getResource("Level3App.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля Level3AppController.java

package com.lvl80.lingua.Level3;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import com.lvl80.lingua.Types.UserList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.ScrollPane;  
import javafx.scene.control.TextField;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Level3AppController {  
 @FXML  
 private ScrollPane sp1;  
 @FXML  
 private ScrollPane sp2;  
 @FXML  
 private ScrollPane sp3;  
 @FXML  
 private ScrollPane sp4;  
 @FXML  
 private ScrollPane sp5;  
 @FXML  
 private TextField tf1;  
 @FXML  
 private TextField tf2;  
 @FXML  
 private TextField tf3;  
 @FXML  
 private TextField tf4;  
 @FXML  
 private TextField tf5;  
  
  
  
  
 private List<ScrollPane> spList = new ArrayList<>();  
  
 @FXML  
 public void initialize(){  
 spList.add(sp1);  
 spList.add(sp2);  
 spList.add(sp3);  
 spList.add(sp4);  
 spList.add(sp5);  
 }  
  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level3App.getStage().close();  
 }  
  
 // функции работы с темами (переключение тем)  
 public void showSubtheme1(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp1);  
 }  
  
 public void showSubtheme2(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp2);  
 }  
  
 public void showSubtheme3(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp3);  
 }  
  
 public void showSubtheme4(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp4);  
 }  
  
 public void showSubtheme5(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp5);  
 }  
  
 public void checkLevel3Test() throws IOException {  
 // проверка правильности теста  
 if (tf1.getText().equals("2") && tf2.getText().equals("17") && tf3.getText().equals("for")  
 && tf4.getText().equals(":") && tf5.getText().equals("el"))  
 {  
 // считывание информации о пользователе  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 // если уровень соответствует 3, то повышаем до 4  
 if (user.getLevel() == 3) {  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 int index;  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setLevel(user.getLevel() + 1);  
 user.setLevel(user.getLevel() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 // открытие лобби после прохождения теста (в любом случае)  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level3App.getStage().close();  
 // если данные теста неправильные - вывести ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в тесте. Пересмотрите варианты ответа (пробелы или регистр букв)");  
 alert.showAndWait();  
 // добавление ошибки к общему числу (запись инфы в файл)  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setMistakes(user.getMistakes() + 1);  
 user.setMistakes(user.getMistakes() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
  
 }  
}

Текст модуля Level4App.java

package com.lvl80.lingua.Level4;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
import java.io.IOException;  
  
  
public class Level4App extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(com.lvl80.lingua.Level4.Level4App.class.getResource("Level4App.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля Level4AppController.java

package com.lvl80.lingua.Level4;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import com.lvl80.lingua.Types.UserList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.ScrollPane;  
import javafx.scene.control.TextField;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Level4AppController {  
 @FXML  
 private ScrollPane sp1;  
 @FXML  
 private ScrollPane sp2;  
 @FXML  
 private ScrollPane sp3;  
 @FXML  
 private ScrollPane sp4;  
 @FXML  
 private ScrollPane sp5;  
 @FXML  
 private ScrollPane sp6;  
 @FXML  
 private ScrollPane sp7;  
 @FXML  
 private ScrollPane sp8;  
 @FXML  
 private ScrollPane sp9;  
 @FXML  
 private ScrollPane sp10;  
 @FXML  
 private ScrollPane sp11;  
 @FXML  
 private ScrollPane sp12;  
 @FXML  
 private ScrollPane sp13;  
 @FXML  
 private ScrollPane sp14;  
 @FXML  
 private TextField tf1;  
 @FXML  
 private TextField tf2;  
 @FXML  
 private TextField tf3;  
 @FXML  
 private TextField tf4;  
 @FXML  
 private TextField tf5;  
 @FXML  
 private TextField tf6;  
 @FXML  
 private TextField tf7;  
 @FXML  
 private TextField tf8;  
 @FXML  
 private TextField tf9;  
 @FXML  
 private TextField tf10;  
 @FXML  
 private TextField tf11;  
 @FXML  
 private TextField tf12;  
 @FXML  
 private RadioButton rb1;  
 @FXML  
 private RadioButton rb2;  
 @FXML  
 private RadioButton rb3;  
 @FXML  
 private RadioButton rb4;  
  
  
 private List<ScrollPane> spList = new ArrayList<>();  
  
 @FXML  
 public void initialize(){  
 spList.add(sp1);  
 spList.add(sp2);  
 spList.add(sp3);  
 spList.add(sp4);  
 spList.add(sp5);  
 spList.add(sp6);  
 spList.add(sp7);  
 spList.add(sp8);  
 spList.add(sp9);  
 spList.add(sp10);  
 spList.add(sp11);  
 spList.add(sp12);  
 spList.add(sp13);  
 spList.add(sp14);  
 }  
  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level4App.getStage().close();  
 }  
  
 // функции работы с темами (переключение тем)  
 public void showSubtheme1(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp1);  
 }  
  
 public void showSubtheme2(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp2);  
 }  
  
 public void showSubtheme3(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp3);  
 }  
  
 public void showSubtheme4(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp4);  
 }  
  
 public void showSubtheme5(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp5);  
 }  
  
 public void showSubtheme6(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp6);  
 }  
  
 public void showSubtheme7(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp7);  
 }  
  
 public void showSubtheme8(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp8);  
 }  
  
 public void showSubtheme9(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp9);  
 }  
  
 public void showSubtheme10(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp10);  
 }  
  
 public void showSubtheme11(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp11);  
 }  
  
 public void showSubtheme12(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp12);  
 }  
  
 public void showSubtheme13(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp13);  
 }  
  
 public void showSubtheme14(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp14);  
 }  
 public void checkLevel4Test() throws IOException {  
 // проверка правильности теста  
 if (tf1.getText().equals("void") && tf2.getText().equals("int") && tf3.getText().equals("{")  
 && tf4.getText().equals("}") && tf5.getText().equals(")") && tf6.getText().equals("n2")  
 && tf7.getText().equals("else") && tf8.getText().equals("return") && tf9.getText().equals("class")  
 && tf10.getText().equals("{") && tf11.getText().equals("void") && tf12.getText().equals("()")  
 && !rb1.isSelected() && rb2.isSelected() && !rb3.isSelected() && !rb4.isSelected())  
 {  
 // считывание информации о пользователе  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 // если уровень соответствует 4, то повышаем до 5  
 if (user.getLevel() == 4) {  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 int index;  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setLevel(user.getLevel() + 1);  
 user.setLevel(user.getLevel() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 // открытие лобби после прохождения теста (в любом случае)  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level4App.getStage().close();  
 // если данные теста неправильные - вывести ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в тесте. Пересмотрите варианты ответа (пробелы или регистр букв)");  
 alert.showAndWait();  
 // добавление ошибки к общему числу (запись инфы в файл)  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setMistakes(user.getMistakes() + 1);  
 user.setMistakes(user.getMistakes() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 }  
}

Текст модуля Level5App.java

package com.lvl80.lingua.Level5;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
import java.io.IOException;  
  
  
public class Level5App extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(com.lvl80.lingua.Level5.Level5App.class.getResource("Level5App.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля Level5AppController.java

package com.lvl80.lingua.Level5;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import com.lvl80.lingua.Types.UserList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.ScrollPane;  
import javafx.scene.control.TextField;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Level5AppController {  
 @FXML  
 private ScrollPane sp1;  
 @FXML  
 private ScrollPane sp2;  
 @FXML  
 private ScrollPane sp3;  
 @FXML  
 private ScrollPane sp4;  
 @FXML  
 private ScrollPane sp5;  
 @FXML  
 private ScrollPane sp6;  
 @FXML  
 private ScrollPane sp7;  
 @FXML  
 private ScrollPane sp8;  
 @FXML  
 private ScrollPane sp9;  
 @FXML  
 private ScrollPane sp10;  
 @FXML  
 private ScrollPane sp11;  
 @FXML  
 private ScrollPane sp12;  
 @FXML  
 private ScrollPane sp13;  
 @FXML  
 private ScrollPane sp14;  
 @FXML  
 private TextField tf1;  
 @FXML  
 private TextField tf2;  
 @FXML  
 private TextField tf3;  
 @FXML  
 private TextField tf4;  
 @FXML  
 private TextField tf5;  
 @FXML  
 private TextField tf6;  
 @FXML  
 private TextField tf7;  
 @FXML  
 private TextField tf8;  
 @FXML  
 private TextField tf9;  
 @FXML  
 private TextField tf10;  
 @FXML  
 private RadioButton rb1;  
 @FXML  
 private RadioButton rb2;  
 @FXML  
 private RadioButton rb3;  
 @FXML  
 private RadioButton rb4;  
  
  
 private List<ScrollPane> spList = new ArrayList<>();  
  
 @FXML  
 public void initialize(){  
 spList.add(sp1);  
 spList.add(sp2);  
 spList.add(sp3);  
 spList.add(sp4);  
 spList.add(sp5);  
 spList.add(sp6);  
 spList.add(sp7);  
 spList.add(sp8);  
 spList.add(sp9);  
 spList.add(sp10);  
 spList.add(sp11);  
 spList.add(sp12);  
 spList.add(sp13);  
 spList.add(sp14);  
 }  
  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level5App.getStage().close();  
 }  
  
 // функции работы с темами (переключение тем)  
 public void showSubtheme1(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp1);  
 }  
  
 public void showSubtheme2(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp2);  
 }  
  
 public void showSubtheme3(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp3);  
 }  
  
 public void showSubtheme4(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp4);  
 }  
  
 public void showSubtheme5(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp5);  
 }  
  
 public void showSubtheme6(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp6);  
 }  
  
 public void showSubtheme7(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp7);  
 }  
  
 public void showSubtheme8(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp8);  
 }  
  
 public void showSubtheme9(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp9);  
 }  
  
 public void showSubtheme10(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp10);  
 }  
  
 public void showSubtheme11(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp11);  
 }  
  
 public void showSubtheme12(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp12);  
 }  
  
 public void showSubtheme13(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp13);  
 }  
  
 public void showSubtheme14(){  
 for (ScrollPane sp\_ : spList)  
 sp\_.setVisible(sp\_ == sp14);  
 }  
 public void checkLevel5Test() throws IOException {  
 // проверка правильности теста  
 if (tf1.getText().equals("int") && tf2.getText().equals("try") && tf3.getText().equals("/")  
 && tf4.getText().equals("throws") && tf5.getText().equals("if") && tf6.getText().equals("0")  
 && tf7.getText().equals("new") && tf8.getText().equals("2") && tf9.getText().equals("15")  
 && tf10.getText().equals("12")  
 && !rb1.isSelected() && !rb2.isSelected() && rb3.isSelected() && rb4.isSelected())  
 {  
 // считывание информации о пользователе  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 // если уровень соответствует 5, то повышаем до 6  
 if (user.getLevel() == 5) {  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 int index;  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setLevel(user.getLevel() + 1);  
 user.setLevel(user.getLevel() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 // открытие лобби после прохождения теста (в любом случае)  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 Level5App.getStage().close();  
 // если данные теста неправильные - вывести ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в тесте. Пересмотрите варианты ответа (пробелы или регистр букв)");  
 alert.showAndWait();  
 // добавление ошибки к общему числу (запись инфы в файл)  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin())) {  
 user\_.setMistakes(user.getMistakes() + 1);  
 user.setMistakes(user.getMistakes() + 1);  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 }  
 }  
}

Текст модуля LiteratureApp.java

package com.lvl80.lingua.Literature;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class LiteratureApp extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(LiteratureApp.class.getResource("LiteratureApp.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля LiteratureAppController.java

package com.lvl80.lingua.Literature;  
  
import com.lvl80.lingua.Lobby.LobbyApp;  
import javafx.stage.Stage;  
import java.io.IOException;  
  
public class LiteratureAppController {  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 LiteratureApp.getStage().close();  
 }  
}

Текст модуля LobbyApp.java

package com.lvl80.lingua.Lobby;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class LobbyApp extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(LobbyApp.class.getResource("LobbyApp.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua Profile");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля LobbyAppController.java

package com.lvl80.lingua.Lobby;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Level1.Level1App;  
import com.lvl80.lingua.Level2.Level2App;  
import com.lvl80.lingua.Level3.Level3App;  
import com.lvl80.lingua.Level4.Level4App;  
import com.lvl80.lingua.Level5.Level5App;  
import com.lvl80.lingua.Literature.LiteratureApp;  
import com.lvl80.lingua.Profile.ProfileApp;  
import com.lvl80.lingua.Test.TestApp;  
import com.lvl80.lingua.Types.User;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.Label;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
  
public class LobbyAppController {  
 @FXML  
 private Label labelLevel;  
  
 @FXML  
 public void initialize() throws IOException {  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 labelLevel.setText(String.valueOf(user.getLevel()));  
 }  
  
 // открытие разделов (func + номер раздела)  
 public void func1() throws IOException {  
 Level1App level1App = new Level1App();  
 level1App.start(new Stage());  
 LobbyApp.getStage().close();  
 }  
 public void func2() throws IOException {  
 // проверка, чтобы нельзя было начать 2й уровень не пройдя 1й  
 if (Integer.parseInt(labelLevel.getText()) > 2-1) {  
 Level2App level2App = new Level2App();  
 level2App.start(new Stage());  
 LobbyApp.getStage().close();  
 // если уровень равен 1 или меньше выводим ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Для начала изучения раздела 2 необходимо пройти тест раздела 1.");  
 alert.showAndWait();  
 }  
 }  
 public void func3() throws IOException {  
 // проверка, чтобы нельзя было начать 3й уровень не пройдя 2й  
 if (Integer.parseInt(labelLevel.getText()) > 3-1) {  
 Level3App level3App = new Level3App();  
 level3App.start(new Stage());  
 LobbyApp.getStage().close();  
 // если уровень равен 2 или меньше выводим ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Для начала изучения раздела 3 необходимо пройти тест раздела 2.");  
 alert.showAndWait();  
 }  
 }  
 public void func4() throws IOException {  
 // проверка, чтобы нельзя было начать 4й уровень не пройдя 3й  
 if (Integer.parseInt(labelLevel.getText()) > 4-1) {  
 Level4App level4App = new Level4App();  
 level4App.start(new Stage());  
 LobbyApp.getStage().close();  
 // если уровень равен 3 или меньше выводим ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Для начала изучения раздела 4 необходимо пройти тест раздела 3.");  
 alert.showAndWait();  
 }  
 }  
 public void func5() throws IOException {  
 // проверка, чтобы нельзя было начать 5й уровень не пройдя 4й  
 if (Integer.parseInt(labelLevel.getText()) > 5-1) {  
 Level5App level5App = new Level5App();  
 level5App.start(new Stage());  
 LobbyApp.getStage().close();  
 // если уровень равен 4 или меньше выводим ошибку  
 } else {  
 Alert alert = new Alert(Alert.AlertType.INFORMATION);  
 alert.setTitle("Ошибка!");  
 alert.setHeaderText(null);  
 alert.setContentText("Для начала изучения раздела 4 необходимо пройти тест раздела 3.");  
 alert.showAndWait();  
 }  
 }  
  
 public void openTest() throws IOException {  
 TestApp testApp = new TestApp();  
 testApp.start(new Stage());  
 LobbyApp.getStage().close();  
 }  
  
 // особые вкладки  
 public void openLiterature() throws IOException {  
 LiteratureApp literatureApp = new LiteratureApp();  
 literatureApp.start(new Stage());  
 LobbyApp.getStage().close();  
 }  
 public void openProflie() throws IOException {  
 ProfileApp profileApp = new ProfileApp();  
 profileApp.start(new Stage());  
 LobbyApp.getStage().close();  
 }  
}

Текст модуля ProfileApp.java

package com.lvl80.lingua.Profile;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class ProfileApp extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(com.lvl80.lingua.Profile.ProfileApp.class.getResource("ProfileApp.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля ProfileAppController.java

package com.lvl80.lingua.Profile;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Authorization.AuthorizationApp;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import javafx.fxml.FXML;  
import javafx.scene.control.Label;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
  
public class ProfileAppController {  
 @FXML  
 private Label labelMail;  
 @FXML  
 private Label labelLogin;  
 @FXML  
 private Label labelLevel;  
 @FXML  
 private Label labelMistakes;  
 @FXML  
 private Label labelMark;  
  
 // чтение данныъ с файла для профиля  
 @FXML  
 public void initialize() throws IOException {  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 // установка значений  
 labelMail.setText(user.getMail());  
 labelLogin.setText(user.getLogin());  
 labelLevel.setText(String.valueOf(user.getLevel()));  
 labelMistakes.setText(String.valueOf(user.getMistakes()));  
 labelMark.setText(Float.toString(user.getMark()));  
 }  
  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 ProfileApp.getStage().close();  
 }  
  
 // выход из аккаунта  
 public void backToAuthorization() throws IOException {  
 AuthorizationApp authorizationApp = new AuthorizationApp();  
 authorizationApp.start(new Stage());  
 ProfileApp.getStage().close();  
 }  
}

Текст модуля TestApp.java

package com.lvl80.lingua.Test;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class TestApp extends Application {  
 private static Stage stage;  
 @Override  
 public void start(Stage \_stage) throws IOException {  
 stage = \_stage;  
 FXMLLoader fxmlLoader = new FXMLLoader(TestApp.class.getResource("TestApp.fxml"));  
 Scene scene = new Scene(fxmlLoader.load());  
 \_stage.setTitle("Lingua");  
 \_stage.setResizable(false);  
 \_stage.setScene(scene);  
 \_stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch();  
 }  
  
 public static Stage getStage(){  
 return stage;  
 }  
}

Текст модуля TestAppController.java

package com.lvl80.lingua.Test;  
  
import com.fasterxml.jackson.databind.ObjectMapper;  
import com.lvl80.lingua.Lobby.LobbyApp;  
import com.lvl80.lingua.Types.User;  
import com.lvl80.lingua.Types.UserList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.Label;  
import javafx.scene.control.RadioButton;  
import javafx.scene.layout.AnchorPane;  
import javafx.scene.paint.Color;  
import javafx.stage.Stage;  
  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class TestAppController {  
 @FXML  
 private AnchorPane ap1;  
 @FXML  
 private AnchorPane ap2;  
 @FXML  
 private AnchorPane ap3;  
 @FXML  
 private AnchorPane ap4;  
 @FXML  
 private AnchorPane ap5;  
 @FXML  
 private AnchorPane ap6;  
 @FXML  
 private AnchorPane ap7;  
 @FXML  
 private AnchorPane ap8;  
 @FXML  
 private AnchorPane ap9;  
 @FXML  
 private AnchorPane ap10;  
 @FXML  
 private AnchorPane ap11;  
 @FXML  
 private AnchorPane ap12;  
 @FXML  
 private AnchorPane ap13;  
 @FXML  
 private AnchorPane ap14;  
 @FXML  
 private AnchorPane ap15;  
 @FXML  
 private AnchorPane ap16;  
 @FXML  
 private RadioButton rb1;  
 @FXML  
 private RadioButton rb2;  
 @FXML  
 private RadioButton rb3;  
 @FXML  
 private RadioButton rb4;  
 @FXML  
 private RadioButton rb5;  
 @FXML  
 private RadioButton rb6;  
 @FXML  
 private RadioButton rb7;  
 @FXML  
 private RadioButton rb8;  
 @FXML  
 private RadioButton rb9;  
 @FXML  
 private RadioButton rb10;  
 @FXML  
 private RadioButton rb11;  
 @FXML  
 private RadioButton rb12;  
 @FXML  
 private RadioButton rb13;  
 @FXML  
 private RadioButton rb14;  
 @FXML  
 private Label lbl1;  
 @FXML  
 private Label lbl2;  
 @FXML  
 private Label lbl3;  
 @FXML  
 private Label lbl4;  
 @FXML  
 private Label lbl5;  
 @FXML  
 private Label lbl6;  
 @FXML  
 private Label lbl7;  
 @FXML  
 private Label lbl8;  
 @FXML  
 private Label lbl9;  
 @FXML  
 private Label lbl10;  
 @FXML  
 private Label lbl11;  
 @FXML  
 private Label lbl12;  
 @FXML  
 private Label lbl13;  
 @FXML  
 private Label lbl14;  
 @FXML  
 private Label lbl15;  
 @FXML  
 private Label lbl16;  
 // лист слайдов  
 private List<AnchorPane> apList = new ArrayList<>();  
 private List<Label> lblList = new ArrayList<>();  
 // переменная ответов  
 private String answer = "";  
  
 // функция возврата в лобби  
 public void backToLobby() throws IOException {  
 LobbyApp lobbyApp = new LobbyApp();  
 lobbyApp.start(new Stage());  
 TestApp.getStage().close();  
 }  
  
 @FXML  
 public void initialize(){  
 apList.add(ap1);  
 apList.add(ap2);  
 apList.add(ap3);  
 apList.add(ap4);  
 apList.add(ap5);  
 apList.add(ap6);  
 apList.add(ap7);  
 apList.add(ap8);  
 apList.add(ap9);  
 apList.add(ap10);  
 apList.add(ap11);  
 apList.add(ap12);  
 apList.add(ap13);  
 apList.add(ap14);  
 apList.add(ap15);  
 apList.add(ap16);  
  
 lblList.add(lbl1);  
 lblList.add(lbl2);  
 lblList.add(lbl3);  
 lblList.add(lbl4);  
 lblList.add(lbl5);  
 lblList.add(lbl6);  
 lblList.add(lbl7);  
 lblList.add(lbl8);  
 lblList.add(lbl9);  
 lblList.add(lbl10);  
 lblList.add(lbl11);  
 lblList.add(lbl12);  
 lblList.add(lbl13);  
 lblList.add(lbl14);  
 lblList.add(lbl15);  
 }  
  
 // функция начала  
 public void showTest1(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap1);  
 }  
  
 public void showTest2(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap2);  
 }  
  
 public void showTest3(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap3);  
 }  
  
 public void showTest4(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap4);  
 }  
  
 public void showTest5(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap5);  
 }  
  
 public void showTest6(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap6);  
 }  
 public void showTest7(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap7);  
 }  
 public void showTest8(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap8);  
 }  
  
 public void showTest9(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap9);  
 }  
  
 public void showTest10(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap10);  
 }  
  
 public void showTest11(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap11);  
 }  
  
 public void showTest12(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap12);  
 }  
  
 public void showTest13(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap13);  
 }  
  
 public void showTest14(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap14);  
 }  
  
 public void showTest15(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap15);  
 }  
  
 public void showFinal(){  
 for (AnchorPane ap\_ : apList)  
 ap\_.setVisible(ap\_ == ap16);  
 }  
  
 public void checkTest1(){  
 // количество правильных ответов  
 answer += rb1.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb1.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Код между фигурными скобками называется блоком.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest2();  
 }  
  
 public void checkTest2(){  
 // количество правильных ответов  
 answer += rb2.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb2.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. arr[1] обращается ко второй строке двумерного массива, которая содержит 5 элементов.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest3();  
 }  
  
 public void checkTest3(){  
 // количество правильных ответов  
 answer += rb3.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb3.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. char – это примитивный тип, а Character – класс-оболочка. Класс оболочка – это класс, который служит оболочкой для примитивных типов данных. Так как экземпляры этих классов являются объектами, они имеют ряд полезных методов.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest4();  
 }  
  
 public void checkTest4(){  
 // количество правильных ответов  
 answer += rb4.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb4.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Оператор + используется для объединения строк. Например, выражение «Hello» + « world» равно «Hello world».");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest5();  
 }  
  
 public void checkTest5(){  
 // количество правильных ответов  
 answer += rb5.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb5.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Класс Object является суперклассом для всех других классов в Java.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest6();  
 }  
  
 public void checkTest6(){  
 // количество правильных ответов  
 answer += rb6.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb6.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. В отличие от целых чисел, деление значений с плавающей запятой на ноль никаких исключений не вызывает. Арифметика чисел с плавающей запятой реализована в соответствии со стандартом IEEE 754, который требует возвращения специального значения «Infinity» (Бесконечность), когда положительное число делится на ноль.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest7();  
 }  
  
 public void checkTest7(){  
 // количество правильных ответов  
 answer += rb7.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb7.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Массивы в Java реализованы как объекты. В приведенной выше программе arr1 и arr2 – это ссылки, указывающие на разные объекты с одинаковым содержимым. Оператор == не сравнивает содержимое объектов, а сравнивает только ссылки.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest8();  
 }  
  
 public void checkTest8(){  
 // количество правильных ответов  
 answer += rb8.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb8.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Возникнет ошибка компиляции, поскольку 1 используется в условии цикла. В отличие от C++, Java не позволяет неявно преобразовывать целочисленные значения в boolean.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest9();  
 }  
  
 public void checkTest9(){  
 // количество правильных ответов  
 answer += rb9.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb9.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Поскольку строки неизменяемы в Java, для увеличения эффективности производительности и экономии памяти используется интернирование строк (string interning). Интернирование строк – это метод, при котором сохраняется только одна копия каждого различного строкового значения. Следовательно, обе ссылки (str1 и str2) указывают на одну и ту же ячейку памяти.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest10();  
 }  
  
 public void checkTest10(){  
 // количество правильных ответов  
 answer += rb10.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb10.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Вызов нестатических методов из статических функций запрещен, поскольку нестатические методы связаны с экземплярами класса.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest11();  
 }  
  
 public void checkTest11(){  
 // количество правильных ответов  
 answer += rb11.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb11.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Программа выведет 0. В утверждении «num = num++» используется постфиксный оператор инкремента. Поэтому порядок действий следующий: сохранить значение числа во временной переменной, увеличить ее значение на единицу и присвоить переменной num.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest12();  
 }  
  
 public void checkTest12(){  
 // количество правильных ответов  
 answer += rb12.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb12.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Java не запрещает перегрузку функции main. Но имеющая String[] в качестве аргумента функция main всегда служит точкой входа в программу.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest13();  
 }  
  
 public void checkTest13(){  
 // количество правильных ответов  
 answer += rb13.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb13.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Переменная real не равна самой себе. Как такое возможно? Арифметика с плавающей запятой реализована в соответствии со стандартом IEEE 754, который требует возвращения специального значения «NaN» (от английского “Not a Number” – не число), когда ноль делится на ноль. В спецификации также указано, что NaN не равно никакому значению с плавающей запятой, включая само себя.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest14();  
 }  
  
 public void checkTest14(){  
 // количество правильных ответов  
 answer += rb14.isSelected() ? "1" : "0";  
 // если ответ неправильный - выдать ошибку  
 if (!rb14.isSelected()){  
 Alert alert = new Alert(Alert.AlertType.ERROR);  
 alert.setTitle("Неправильный ответ!");  
 alert.setHeaderText(null);  
 alert.setContentText("Вы допустили ошибку в ответе. Размер типа byte составляет 8 бит, а его диапазон [-128, 127]. Все целые числа со знаком в современных компьютерах представлены в виде дополнительного кода (twos-compliment). Когда арифметическая операция дает результат, превышающий максимум, происходит переполнение. В данном случае двоичное представление 127 равно 01111111, при добавлении 1 получаем 10000000, что в дополнительном коде равно -128.");  
 alert.showAndWait();  
 }  
 // переход на следующий вопрос  
 showTest15();  
 }  
  
 public void checkTest15() throws IOException {  
 // количество правильных ответов  
 answer += "1";  
 float mark = 0f;  
 // отобразить результат  
 for (int i = 0; i < 15; i++){  
 // в зависимости от ответа менять цвет текста  
 lblList.get(i).setTextFill(  
 ((answer.charAt(i) == '1') ? Color.GREEN : Color.RED));  
 // установить, на какие вопросы и как ответил пользователь  
 lblList.get(i).setText(  
 lblList.get(i).getText() +  
 ((answer.charAt(i) == '1') ? "Правильно" : "Неправильно"));  
 mark += (answer.charAt(i) == '1') ? 1f : 0f;  
 }  
 // средняя оценка за тест  
 char[] result = Float.toString((mark \* 10) / 15).toCharArray();  
 String result\_ = String.valueOf(result[0]);  
 if (mark == 10f) result\_ = "10";  
 lbl16.setText(  
 lbl16.getText() + result\_);  
 // установить оценку для пользователя  
 ObjectMapper mapper = new ObjectMapper();  
 User user = mapper.readValue(new File("settings/lastloginsession.json"), User.class);  
 UserList list = mapper.readValue(new File("settings/users.json"), UserList.class);  
 for (User user\_ : list.getUsers())  
 if (user\_.getMail().equals(user.getMail()) && user\_.getLogin().equals(user.getLogin()))  
 // если оценка пользователя меньше полученной, то переписываем её  
 if (user\_.getMark() < Float.parseFloat(result\_) && user.getMark() < Float.parseFloat(result\_)){  
 user\_.setMark(Float.parseFloat(result\_));  
 user.setMark(Float.parseFloat(result\_));  
 }  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/lastloginsession.json"), user);  
 mapper.writerWithDefaultPrettyPrinter().writeValue(new File("settings/users.json"), list);  
 // переход на финальную стадию  
 showFinal();  
 }  
}

Текст модуля User.java

package com.lvl80.lingua.Types;  
  
import java.io.Serializable;  
  
public class User implements Serializable {  
 private String login;  
 private String mail;  
 private String password;  
 private int level;  
 private int mistakes;  
 private float mark;  
  
 public User() {  
 }  
  
 public User(String login, String mail, String password, int level, int mistakes, float mark) {  
 this.login = login;  
 this.password = password;  
 this.level = level;  
 this.mail = mail;  
 this.mistakes = mistakes;  
 this.mark = mark;  
 }  
  
 public String getLogin() {  
 return login;  
 }  
  
 public void setLogin(String login) {  
 this.login = login;  
 }  
  
 public String getPassword() {  
 return password;  
 }  
  
 public void setPassword(String password) {  
 this.password = password;  
 }  
  
 public int getLevel() {  
 return level;  
 }  
  
 public void setLevel(int level) {  
 this.level = level;  
 }  
  
 public String getMail() {  
 return mail;  
 }  
  
 public void setMail(String mail) {  
 this.mail = mail;  
 }  
  
 public int getMistakes() {  
 return mistakes;  
 }  
  
 public void setMistakes(int mistakes) {  
 this.mistakes = mistakes;  
 }  
  
 public float getMark() {  
 return mark;  
 }  
  
 public void setMark(float mark) {  
 this.mark = mark;  
 }  
}

Текст модуля UserList.java

package com.lvl80.lingua.Types;  
  
import java.io.Serializable;  
import java.util.ArrayList;  
import java.util.List;  
  
public class UserList implements Serializable {  
 private List<User> users = new ArrayList<>();  
  
 public List<User> getUsers() {  
 return users;  
 }  
  
 public void setUsers(List<User> users) {  
 this.users = users;  
 }  
}

Текст модуля Base64.java

package com.lvl80.lingua.Utils;  
import java.io.UnsupportedEncodingException;  
  
public class Base64 {  
 public static final int DEFAULT = 0;  
 public static final int NO\_PADDING = 1;  
 private static final int NO\_WRAP = 2;  
 private static final int CRLF = 4;  
 private static final int URL\_SAFE = 8;  
 private static abstract class Coder {  
 byte[] output;  
 int op;  
  
 }  
 public static byte[] decode(String str, int flags) {  
 return decode(str.getBytes(), flags);  
 }  
 private static byte[] decode(byte[] input, int flags) {  
 return decode(input, 0, input.length, flags);  
 }  
  
 private static byte[] decode(byte[] input, int offset, int len, int flags) {  
 Decoder decoder = new Decoder(flags, new byte[len\*3/4]);  
  
 if (!decoder.process(input, offset, len, true)) {  
 throw new IllegalArgumentException("bad base-64");  
 }  
  
 if (decoder.op == decoder.output.length) {  
 return decoder.output;  
 }  
  
 byte[] temp = new byte[decoder.op];  
 System.arraycopy(decoder.output, 0, temp, 0, decoder.op);  
 return temp;  
 }  
  
 private static class Decoder extends Coder {  
 private static final int DECODE[] = {  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 62, -1, -1, -1, 63,  
 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, -1, -1, -1, -2, -1, -1,  
 -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,  
 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, -1, -1, -1, -1, -1,  
 -1, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,  
 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 };  
  
 private static final int DECODE\_WEBSAFE[] = {  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 62, -1, -1,  
 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, -1, -1, -1, -2, -1, -1,  
 -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,  
 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, -1, -1, -1, -1, 63,  
 -1, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,  
 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  
 };  
  
 private static final int SKIP = -1;  
 private static final int EQUALS = -2;  
  
 private int state;  
 private int value;  
  
 final private int[] alphabet;  
  
 Decoder(int flags, byte[] output) {  
 this.output = output;  
  
 alphabet = ((flags & URL\_SAFE) == 0) ? DECODE : DECODE\_WEBSAFE;  
 state = 0;  
 value = 0;  
 }  
  
 boolean process(byte[] input, int offset, int len, boolean finish) {  
 if (this.state == 6) return false;  
  
 int p = offset;  
 len += offset;  
  
 int state = this.state;  
 int value = this.value;  
 int op = 0;  
 final byte[] output = this.output;  
 final int[] alphabet = this.alphabet;  
  
 while (p < len) {  
 if (state == 0) {  
 while (p+4 <= len &&  
 (value = ((alphabet[input[p] & 0xff] << 18) |  
 (alphabet[input[p+1] & 0xff] << 12) |  
 (alphabet[input[p+2] & 0xff] << 6) |  
 (alphabet[input[p+3] & 0xff]))) >= 0) {  
 output[op+2] = (byte) value;  
 output[op+1] = (byte) (value >> 8);  
 output[op] = (byte) (value >> 16);  
 op += 3;  
 p += 4;  
 }  
 if (p >= len) break;  
 }  
  
 int d = alphabet[input[p++] & 0xff];  
  
 switch (state) {  
 case 0:  
 if (d >= 0) {  
 value = d;  
 ++state;  
 } else if (d != SKIP) {  
 this.state = 6;  
 return false;  
 }  
 break;  
  
 case 1:  
 if (d >= 0) {  
 value = (value << 6) | d;  
 ++state;  
 } else if (d != SKIP) {  
 this.state = 6;  
 return false;  
 }  
 break;  
  
 case 2:  
 if (d >= 0) {  
 value = (value << 6) | d;  
 ++state;  
 } else if (d == EQUALS) {  
 output[op++] = (byte) (value >> 4);  
 state = 4;  
 } else if (d != SKIP) {  
 this.state = 6;  
 return false;  
 }  
 break;  
  
 case 3:  
 if (d >= 0) {  
 value = (value << 6) | d;  
 output[op+2] = (byte) value;  
 output[op+1] = (byte) (value >> 8);  
 output[op] = (byte) (value >> 16);  
 op += 3;  
 state = 0;  
 } else if (d == EQUALS) {  
 output[op+1] = (byte) (value >> 2);  
 output[op] = (byte) (value >> 10);  
 op += 2;  
 state = 5;  
 } else if (d != SKIP) {  
 this.state = 6;  
 return false;  
 }  
 break;  
  
 case 4:  
 if (d == EQUALS) {  
 ++state;  
 } else if (d != SKIP) {  
 this.state = 6;  
 return false;  
 }  
 break;  
  
 case 5:  
 if (d != SKIP) {  
 this.state = 6;  
 return false;  
 }  
 break;  
 }  
 }  
  
 if (!finish) {  
 this.state = state;  
 this.value = value;  
 this.op = op;  
 return true;  
 }  
 switch (state) {  
 case 0:  
 break;  
 case 1:  
 this.state = 6;  
 return false;  
 case 2:  
 output[op++] = (byte) (value >> 4);  
 break;  
 case 3:  
 output[op++] = (byte) (value >> 10);  
 output[op++] = (byte) (value >> 2);  
 break;  
 case 4:  
 this.state = 6;  
 return false;  
 case 5:  
 break;  
 }  
  
 this.state = state;  
 this.op = op;  
 return true;  
 }  
 }  
  
 public static String encodeToString(byte[] input, int flags) {  
 try {  
 return new String(encode(input, flags), "US-ASCII");  
 } catch (UnsupportedEncodingException e) {  
 throw new AssertionError(e);  
 }  
 }  
  
 private static byte[] encode(byte[] input, int flags) {  
 return encode(input, 0, input.length, flags);  
 }  
  
 private static byte[] encode(byte[] input, int offset, int len, int flags) {  
 Encoder encoder = new Encoder(flags, null);  
  
 int output\_len = len / 3 \* 4;  
  
 if (encoder.do\_padding) {  
 if (len % 3 > 0) {  
 output\_len += 4;  
 }  
 } else {  
 switch (len % 3) {  
 case 0: break;  
 case 1: output\_len += 2; break;  
 case 2: output\_len += 3; break;  
 }  
 }  
  
 if (encoder.do\_newline && len > 0) {  
 output\_len += (((len-1) / (3 \* Encoder.LINE\_GROUPS)) + 1) \*  
 (encoder.do\_cr ? 2 : 1);  
 }  
  
 encoder.output = new byte[output\_len];  
 encoder.process(input, offset, len, true);  
  
 assert encoder.op == output\_len;  
  
 return encoder.output;  
 }  
  
 private static class Encoder extends Coder {  
 static final int LINE\_GROUPS = 19;  
  
 private static final byte ENCODE[] = {  
 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',  
 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b', 'c', 'd', 'e', 'f',  
 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v',  
 'w', 'x', 'y', 'z', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '+', '/',  
 };  
  
 private static final byte ENCODE\_WEBSAFE[] = {  
 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',  
 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b', 'c', 'd', 'e', 'f',  
 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v',  
 'w', 'x', 'y', 'z', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '-', '\_',  
 };  
  
 final private byte[] tail;  
 int tailLen;  
 private int count;  
  
 final boolean do\_padding;  
 final boolean do\_newline;  
 final boolean do\_cr;  
 final private byte[] alphabet;  
  
 Encoder(int flags, byte[] output) {  
 this.output = output;  
  
 do\_padding = (flags & NO\_PADDING) == 0;  
 do\_newline = (flags & NO\_WRAP) == 0;  
 do\_cr = (flags & CRLF) != 0;  
 alphabet = ((flags & URL\_SAFE) == 0) ? ENCODE : ENCODE\_WEBSAFE;  
  
 tail = new byte[2];  
 tailLen = 0;  
  
 count = do\_newline ? LINE\_GROUPS : -1;  
 }  
  
 void process(byte[] input, int offset, int len, boolean finish) {  
 final byte[] alphabet = this.alphabet;  
 final byte[] output = this.output;  
 int op = 0;  
 int count = this.count;  
  
 int p = offset;  
 len += offset;  
 int v = -1;  
  
 switch (tailLen) {  
 case 0:  
 break;  
  
 case 1:  
 if (p+2 <= len) {  
 v = ((tail[0] & 0xff) << 16) |  
 ((input[p++] & 0xff) << 8) |  
 (input[p++] & 0xff);  
 tailLen = 0;  
 }  
 break;  
  
 case 2:  
 if (p+1 <= len) {  
 v = ((tail[0] & 0xff) << 16) |  
 ((tail[1] & 0xff) << 8) |  
 (input[p++] & 0xff);  
 tailLen = 0;  
 }  
 break;  
 }  
  
 if (v != -1) {  
 output[op++] = alphabet[(v >> 18) & 0x3f];  
 output[op++] = alphabet[(v >> 12) & 0x3f];  
 output[op++] = alphabet[(v >> 6) & 0x3f];  
 output[op++] = alphabet[v & 0x3f];  
 if (--count == 0) {  
 if (do\_cr) output[op++] = '\r';  
 output[op++] = '\n';  
 count = LINE\_GROUPS;  
 }  
 }  
  
 while (p+3 <= len) {  
 v = ((input[p] & 0xff) << 16) |  
 ((input[p+1] & 0xff) << 8) |  
 (input[p+2] & 0xff);  
 output[op] = alphabet[(v >> 18) & 0x3f];  
 output[op+1] = alphabet[(v >> 12) & 0x3f];  
 output[op+2] = alphabet[(v >> 6) & 0x3f];  
 output[op+3] = alphabet[v & 0x3f];  
 p += 3;  
 op += 4;  
 if (--count == 0) {  
 if (do\_cr) output[op++] = '\r';  
 output[op++] = '\n';  
 count = LINE\_GROUPS;  
 }  
 }  
  
 if (finish) {  
 if (p-tailLen == len-1) {  
 int t = 0;  
 v = ((tailLen > 0 ? tail[t++] : input[p++]) & 0xff) << 4;  
 tailLen -= t;  
 output[op++] = alphabet[(v >> 6) & 0x3f];  
 output[op++] = alphabet[v & 0x3f];  
 if (do\_padding) {  
 output[op++] = '=';  
 output[op++] = '=';  
 }  
 if (do\_newline) {  
 if (do\_cr) output[op++] = '\r';  
 output[op++] = '\n';  
 }  
 } else if (p-tailLen == len-2) {  
 int t = 0;  
 v = (((tailLen > 1 ? tail[t++] : input[p++]) & 0xff) << 10) |  
 (((tailLen > 0 ? tail[t++] : input[p++]) & 0xff) << 2);  
 tailLen -= t;  
 output[op++] = alphabet[(v >> 12) & 0x3f];  
 output[op++] = alphabet[(v >> 6) & 0x3f];  
 output[op++] = alphabet[v & 0x3f];  
 if (do\_padding) {  
 output[op++] = '=';  
 }  
 if (do\_newline) {  
 if (do\_cr) output[op++] = '\r';  
 output[op++] = '\n';  
 }  
 } else if (do\_newline && op > 0 && count != LINE\_GROUPS) {  
 if (do\_cr) output[op++] = '\r';  
 output[op++] = '\n';  
 }  
  
 assert tailLen == 0;  
 assert p == len;  
 } else {  
 if (p == len-1) {  
 tail[tailLen++] = input[p];  
 } else if (p == len-2) {  
 tail[tailLen++] = input[p];  
 tail[tailLen++] = input[p+1];  
 }  
 }  
  
 this.op = op;  
 this.count = count;  
 }  
 }  
  
 private Base64() { }  
  
}

Текст модуля module-info.java

module com.lvl80.lingua {  
 requires javafx.controls;  
 requires javafx.fxml;  
 requires org.kordamp.bootstrapfx.core;  
 requires com.fasterxml.jackson.annotation;  
 requires com.fasterxml.jackson.databind;  
 requires com.fasterxml.jackson.core;  
  
 opens com.lvl80.lingua.Lobby to javafx.fxml;  
 exports com.lvl80.lingua.Lobby;  
  
 opens com.lvl80.lingua.Authorization to javafx.fxml,com.fasterxml.jackson.core,com.fasterxml.jackson.databind,com.fasterxml.jackson.annotation;  
 exports com.lvl80.lingua.Authorization;  
  
 opens com.lvl80.lingua.Types to com.fasterxml.jackson.core,com.fasterxml.jackson.databind,com.fasterxml.jackson.annotation;  
 exports com.lvl80.lingua.Types;  
  
 opens com.lvl80.lingua.Profile to javafx.fxml;  
 exports com.lvl80.lingua.Profile;  
  
 opens com.lvl80.lingua.Level1 to javafx.fxml;  
 exports com.lvl80.lingua.Level1;  
  
 opens com.lvl80.lingua.Level2 to javafx.fxml;  
 exports com.lvl80.lingua.Level2;  
  
 opens com.lvl80.lingua.Literature to javafx.fxml;  
 exports com.lvl80.lingua.Literature;  
  
 opens com.lvl80.lingua.Level3 to javafx.fxml;  
 exports com.lvl80.lingua.Level3;  
  
 opens com.lvl80.lingua.Level4 to javafx.fxml;  
 exports com.lvl80.lingua.Level4;  
  
 opens com.lvl80.lingua.Level5 to javafx.fxml;  
 exports com.lvl80.lingua.Level5;  
  
 opens com.lvl80.lingua.Test to javafx.fxml;  
 exports com.lvl80.lingua.Test;  
}