**Лабораторная работа №5**

**Бинарные файлы**

Написать приложение, позволяющее создавать целочисленный бинарный файл, выводить его в многострочный редактор текстов или таблицу, а также производить указанные действия (создание, вывод и операции совсем не обязательно должны выполняться последовательно: пользователь может выбрать ранее созданный файл). Количество чисел в создаваемом файле задается случайным образом из диапазона **1...n**, где **n** вводится. Имя файла для создания и обработки запрашивается с помощью диалоговых окон. После создания или выбора файла его элементы выводятся. Программа должна выдавать ответ при любых введенных пользователем данных. Интерфейс программы (используется фреймворк Vaadin), кроме оговоренных моментов произвольный, максимально удобный для пользователя и по возможности эстетичный.

**Вариант 9:** вывести в порядке возрастания точные квадраты, которые не встречаются в данном файле. Последнее из выведенных чисел не должно превосходить наибольшего числа файла. Считывать элементы файла в массив или другую структуру нельзя, но для других целей ее создать можно. Компоненты файла задаются случайно из диапазона **a…b**. Считать, что число элементов файла не ограничено конкретным числом. При обработке файл можно прочитать только один раз.

**Теоретическая часть**

Бинарный (двоичный) файл – это файл, который содержит данные в двоичном формате (последовательность байтов), а не в текстовом. В отличие от текстовых файлов, которые хранят символы в кодировке (UTF-8, ASCII и т. д.), бинарные файлы могут содержать любые данные: изображения, аудио, видео, исполняемые программы (exe), архивы (zip), сериализованные объекты и др.

**Текст программы**

package com.northpole.snow;  
  
import com.vaadin.flow.component.button.Button;  
import com.vaadin.flow.component.dialog.Dialog;  
import com.vaadin.flow.component.html.Label;  
import com.vaadin.flow.component.notification.Notification;  
import com.vaadin.flow.component.orderedlayout.VerticalLayout;  
import com.vaadin.flow.component.textfield.IntegerField;  
import com.vaadin.flow.component.textfield.TextArea;  
import com.vaadin.flow.component.textfield.TextField;  
import com.vaadin.flow.router.Route;  
  
import java.io.DataInputStream;  
import java.io.DataOutputStream;  
import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.io.IOException;  
import java.nio.file.Files;  
import java.nio.file.Paths;  
import java.util.TreeSet;  
import java.util.Random;  
  
*// Аннотация Route указывает, что этот класс является корневым представлением*@Route("")  
public class MainView extends VerticalLayout {  
  
 *// Поля ввода и отображения данных* private final TextArea outputArea = new TextArea("Содержимое файла");  
 private final TextArea squaresArea = new TextArea("Отсутствующие точные квадраты");  
 private final IntegerField nField = new IntegerField("Макс количество чисел (n)");  
 private final IntegerField aField = new IntegerField("Диапазон a");  
 private final IntegerField bField = new IntegerField("Диапазон b");  
  
 *// Конструктор главного представления* public MainView() {  
 setSizeFull(); *// Занимать все доступное пространство* setPadding(true); *// Внутренние отступы* setSpacing(true); *// Расстояние между компонентами  
  
 // Настройка размеров текстовых областей* outputArea.setWidthFull();  
 outputArea.setHeight("200px");  
 squaresArea.setWidthFull();  
 squaresArea.setHeight("150px");  
  
 *// Установка значений по умолчанию для полей ввода* nField.setValue(100);  
 aField.setValue(0);  
 bField.setValue(1000);  
  
 *// Создание кнопок* Button createFileBtn = new Button("Создать файл");  
 Button openFileBtn = new Button("Открыть файл и показать");  
 Button showSquaresBtn = new Button("Показать отсутствующие точные квадраты");  
  
 *// Назначение обработчиков событий для кнопок* createFileBtn.addClickListener(e -> showFileNameDialog("Создать файл", this::createFile));  
 openFileBtn.addClickListener(e -> showFileNameDialog("Открыть файл", this::openFile));  
 showSquaresBtn.addClickListener(e -> showFileNameDialog("Показать квадраты", this::showMissingSquares));  
  
 *// Добавление компонентов на форму в определенном порядке* add(nField, aField, bField,  
 createFileBtn, openFileBtn, showSquaresBtn,  
 new Label("Содержимое файла:"), outputArea,  
 new Label("Отсутствующие точные квадраты:"), squaresArea);  
 }  
  
 *// Метод для отображения диалогового окна с вводом имени файла* private void showFileNameDialog(String title, java.util.function.Consumer<String> action) {  
 Dialog dialog = new Dialog();  
 dialog.setWidth("300px");  
 dialog.setCloseOnEsc(true); *// Закрывать по ESC* dialog.setCloseOnOutsideClick(false); *// Не закрывать по клику вне диалога  
  
 // Поле для ввода имени файла* TextField fileNameInput = new TextField("Имя файла");  
 fileNameInput.setWidthFull();  
  
 *// Кнопка OK - выполняет действие с переданным именем файла* Button okButton = new Button("OK", event -> {  
 String fileName = fileNameInput.getValue();  
 if (fileName == null || fileName.trim().isEmpty()) {  
 Notification.*show*("Введите имя файла");  
 return;  
 }  
 dialog.close();  
 action.accept(fileName.trim()); *// Вызов переданного действия* });  
  
 *// Кнопка отмены - просто закрывает диалог* Button cancelButton = new Button("Отмена", event -> dialog.close());  
  
 *// Компоновка компонентов в диалоге* VerticalLayout layout = new VerticalLayout(new Label(title), fileNameInput, okButton, cancelButton);  
 layout.setPadding(false);  
 layout.setSpacing(true);  
 dialog.add(layout);  
 dialog.open(); *// Открытие диалога* }  
  
 *// Метод для создания файла с случайными числами* private void createFile(String fileName) {  
 Integer n = nField.getValue();  
 Integer a = aField.getValue();  
 Integer b = bField.getValue();  
  
 *// Валидация введенных данных* if (n == null || n <= 0) {  
 Notification.*show*("Введите корректное n (>0)");  
 return;  
 }  
 if (a == null || b == null || a > b) {  
 Notification.*show*("Введите корректный диапазон a <= b");  
 return;  
 }  
  
 *// Запись данных в файл* try (DataOutputStream dos = new DataOutputStream(new FileOutputStream(fileName))) {  
 *// Записываем диапазон в начало файла* dos.writeInt(a);  
 dos.writeInt(b);  
  
 *// Генерация случайных чисел в заданном диапазоне* Random rand = new Random();  
 int count = 1 + rand.nextInt(n); *// Случайное количество чисел* for (int i = 0; i < count; i++) {  
 int val = a + rand.nextInt(b - a + 1);  
 dos.writeInt(val);  
 }  
 Notification.*show*("Файл создан с " + count + " элементами");  
 outputArea.clear();  
 squaresArea.clear();  
 } catch (IOException ex) {  
 Notification.*show*("Ошибка записи файла: " + ex.getMessage());  
 }  
 }  
  
 *// Метод для открытия файла и отображения его содержимого* private void openFile(String fileName) {  
 if (!Files.*exists*(Paths.*get*(fileName))) {  
 Notification.*show*("Файл не найден");  
 return;  
 }  
 try (DataInputStream dis = new DataInputStream(new FileInputStream(fileName))) {  
 *// Читаем диапазон (но не выводим его)* dis.readInt(); *// a* dis.readInt(); *// b  
  
 // Чтение и отображение содержимого файла* StringBuilder sb = new StringBuilder();  
 while (dis.available() >= 4) {  
 int val = dis.readInt();  
 sb.append(val).append("\n");  
 }  
 outputArea.setValue(sb.toString());  
 squaresArea.clear();  
 Notification.*show*("Файл успешно открыт");  
 } catch (IOException ex) {  
 Notification.*show*("Ошибка чтения файла: " + ex.getMessage());  
 }  
 }  
  
 *// Метод для поиска отсутствующих точных квадратов в файле* private void showMissingSquares(String fileName) {  
 if (!Files.*exists*(Paths.*get*(fileName))) {  
 Notification.*show*("Файл не найден");  
 return;  
 }  
  
 try (DataInputStream dis = new DataInputStream(new FileInputStream(fileName))) {  
 *// Читаем диапазон из файла* int a = dis.readInt();  
 int b = dis.readInt();  
  
 *// Создаем множество всех возможных квадратов в диапазоне* TreeSet<Integer> allSquares = new TreeSet<>();  
 int i = 0;  
 while (true) {  
 int square = i \* i;  
 if (square > b) break; *// Прекращаем, если квадрат превышает верхнюю границу* if (square >= a) {  
 allSquares.add(square); *// Добавляем квадрат, если он в диапазоне* }  
 i++;  
 }  
  
 *// Читаем числа из файла и удаляем встречающиеся квадраты из множества* while (dis.available() >= 4) {  
 int val = dis.readInt();  
 allSquares.remove(val);  
 }  
  
 *// Формируем результат - только квадраты, не превышающие b* StringBuilder sb = new StringBuilder();  
 for (Integer square : allSquares) {  
 sb.append(square).append("\n");  
 }  
  
 *// Отображаем результат* squaresArea.setValue(sb.length() == 0 ? "Все квадраты присутствуют" : sb.toString());  
  
 } catch (IOException ex) {  
 Notification.*show*("Ошибка обработки файла: " + ex.getMessage());  
 }  
 }  
}

**Описание программы**

Класс **MainView**:

* **TextArea outputArea** – текстовая область для отображения содержимого файла
* **TextArea squaresArea** – текстовая область для отображения отсутствующих точных квадратов
* **IntegerField nField** – поле для ввода максимального количества чисел (**n**)
* **IntegerField aField** – поле для ввода нижней границы диапазона генерации чисел (**a**)
* **IntegerField bField** – поле для ввода верхней границы диапазона генерации чисел (**b**)
* **MainView()** – конструктор:

1. Настраивает размеры и отступы компонента. Устанавливает размеры для **outputArea** и **squaresArea**.
2. Инициализирует поля ввода значениями по умолчанию: **n = 100**, **a = 0**, **b = 1000**.
3. Создаёт кнопки и связывает их с действиями: «Создать файл» → открывает диалог и вызывает **createFile**, «Открыть файл и показать» → диалог и вызов **openFile**, «Показать отсутствующие точные квадраты» → диалог и вызов **showMissingSquares**.
4. Добавляет все компоненты на экран.

* **showFileNameDialog()** – показывает диалог с полем для ввода имени файла, а также кнопками «Ок» и «Отмена». При нажатии кнопки «Ок» проверяет, что имя файла не пустое, закрывает диалог и вызывает действие, передавая введенное имя файла.
* **createFile()**:

1. Считывает значения из полей **nField**, **aField**, **bField**.
2. Проверяет корректность: **n > 0**, **a ≤ b**. Если ошибки – выводит уведомление и прекращает.
3. Создаёт бинарный файл с именем **fileName**.
4. Записывает в начало файла значения **a** и **b**.
5. Генерирует случайное число **count** в диапазоне **[1, n]**.
6. Генерирует **count** случайных чисел в диапазоне **[a, b]**.
7. Записывает каждое число в файл как 4-байтовое целое.
8. Очищает поля **outputArea** и **squaresArea**.
9. Уведомляет пользователя о создании файла и количестве элементов.

* **openFile()**:

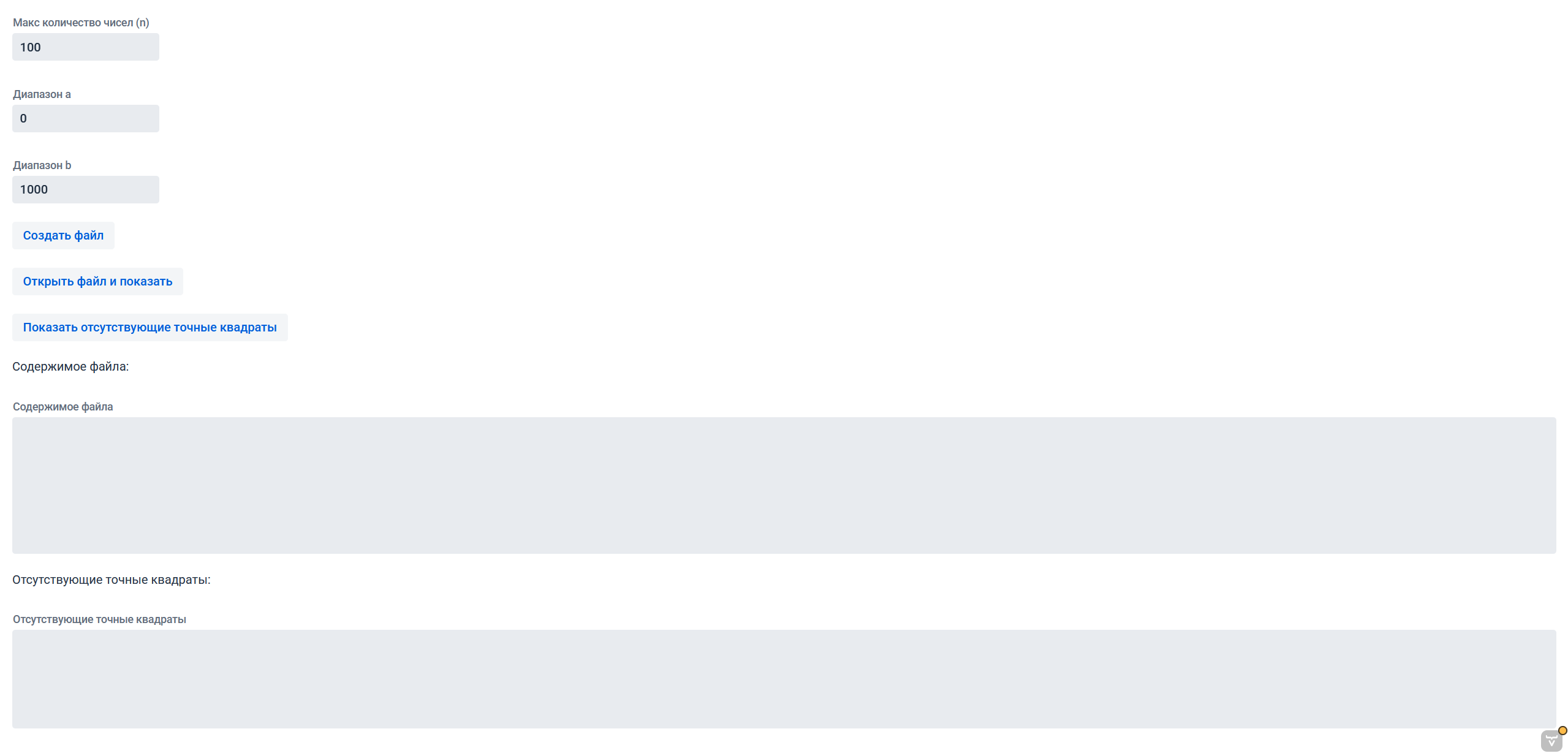
1. Проверяет, существует ли файл **fileName**. Если нет – уведомляет пользователя. Если да – открывает бинарный файл для чтения.
2. Открывает бинарный файл и считывает первые 2 числа (**a** и **b**), но не выводит их, а затем все остальные числа.
3. Формирует строку со всеми числами и выводит в **outputArea**.
4. Очищает **squaresArea**.
5. Уведомляет пользователя об успешном или неуспешном открытии файла.

* **showMissingSquares()**:

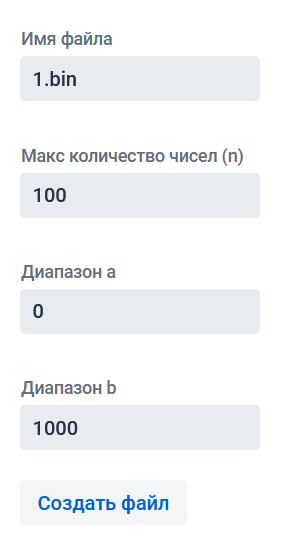
1. Проверяет, существует ли файл. Если нет – уведомляет. Если есть – открывает для чтения.
2. Считывает из файла **a**, **b**.
3. Генерирует множество всех точных квадратов в диапазоне **[a, b]**: для каждого **i** вычисляет **i2**, пока квадрат не превысит **b**, а затем добавляет квадраты в **TreeSet**.
4. Считывает остальные числа из файла и удаляет из множества квадраты, которые встречаются в этом файле.
5. Формирует результат. Если множество квадратов пусто – выводит *«Все квадраты присутствуют»*. Иначе – выводит оставшиеся квадраты в **squaresArea**.

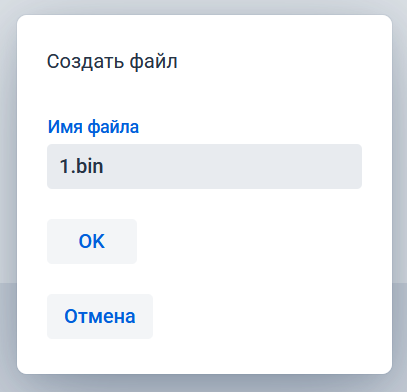
**Результаты тестов**

Программа перед началом работы:

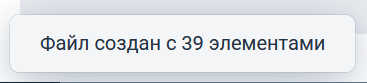


Создание файла:

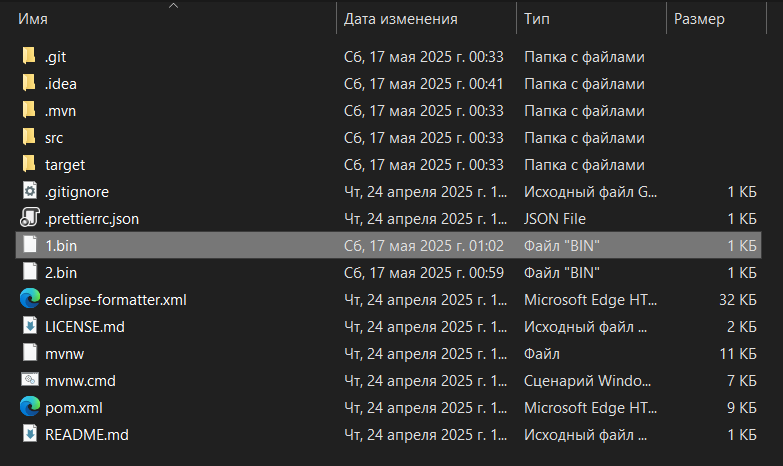




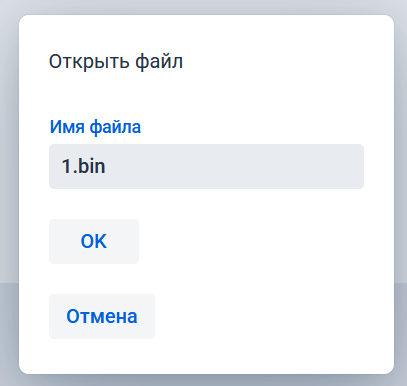
Сообщение о создании файла:

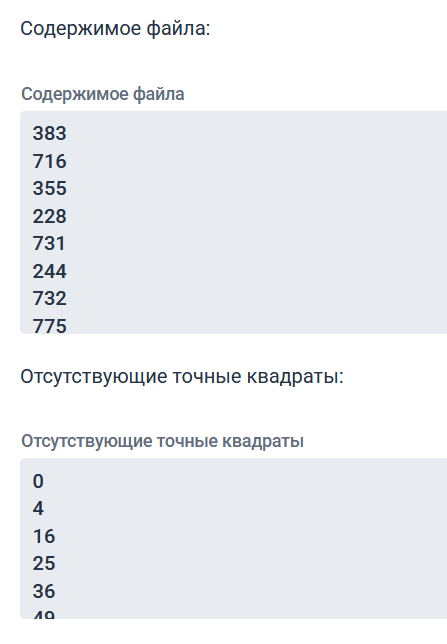


Созданный файл:

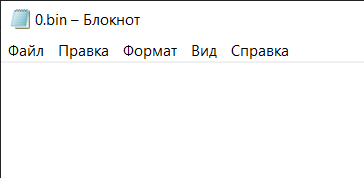


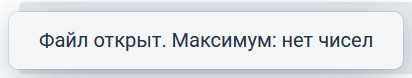
Открытие данного файла через программу:



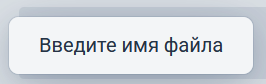


Открытие пустого файла:

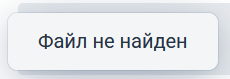




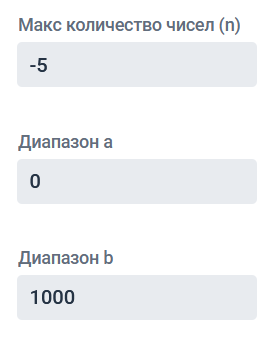
Сообщение при незаполненном поле «Имя файла»:

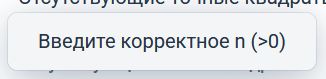


Сообщение при вводе названия несуществующего файла:

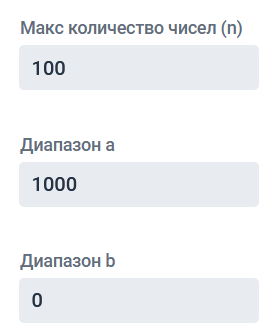


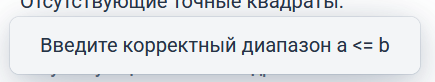
Ввод **n < 0**:





Ввод **b > a**:





**Вывод**

В ходе выполнения лабораторной работы была создана программа, которая создает и читает бинарные файлы, находит максимальное число и выводит список квадратов, не превышающих это число и не входящих в файл. Интерфейс программы был разработан с помощью Vaadin.