Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и кибербезопасности
Высшая школа программной инженерии

К У Р С О В А Я Р А Б О Т А

по дисциплине «Объектно-ориентированное программирование»

Выполнил
студент гр. в3530904/20322 Н.В. Акуленков

Руководитель
старший преподаватель А.П. Маслаков

«___» _____ 2024 г.

Санкт-Петербург

2024

Содержание

Постановка задачи 3

Выбор технологии 3

Структура проекта 4

Результаты выполнения программы 6

Лабораторная работа 1 6

Лабораторная работа 2 7

Лабораторная работа 3 .....**Error! Bookmark not defined.**

Лабораторная работа 4 11

Выводы .....**Error! Bookmark not defined.**

Код проекта.....**Error! Bookmark not defined.**

Постановка задачи

Разработать приложение с графическим интерфейсом для заданий 1–4. Для этого приложения должна быть реализована возможность выбора из списка любого приложения, ввод входных данных и его выполнение. Модифицировать задания 1–4 так, чтобы весь вывод происходил в текстовых областях, защищённых от редактирования. Предусмотреть для заданий:

• 3 - выбор файлов словаря и текста для перевода, возможность ручного ввода текста;

• 4 - ввод входных данных для методов.

Блок практических заданий 1.1-1.4 призван сформировать у студента понимание особенностей хранения, умение настраивать и поддерживать данные.

Выбор технологии

В качестве технологии для создания графического приложения курсовой работы, был выбран изученный в ходе лекций фреймворк JavaFX. Разработка велась в IntelliJ IDEA.

Структура проекта

На рисунке 1 изображена структура проекта:

1. В папках l1-4 содержатся файлы реализующие код для лабораторных работ 1-4 соответственно. Также в них содержатся файлы l1-4Controller, реализующие функционал для вкладок пользовательского интерфейса.
2. Файл HelloApplication содержит основной класс программы HelloApplication, реализующий проект курсовой работы.
3. Папка resources содержит fxml-файлы, использующиеся для создания графического интерфейса программы.

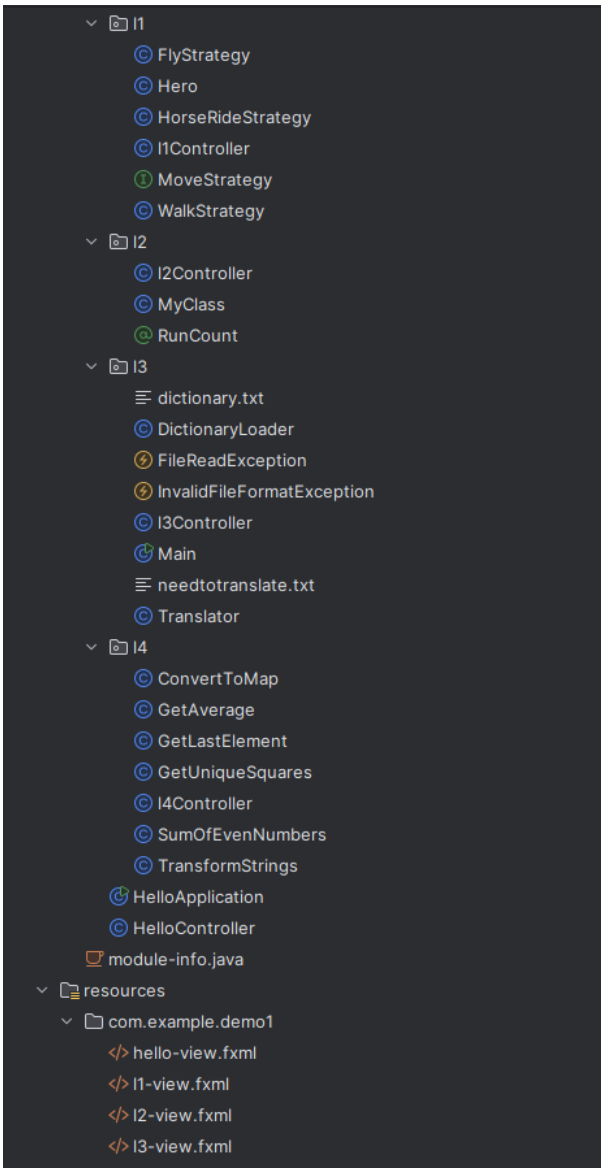

Рисунок 1. Структура проекта

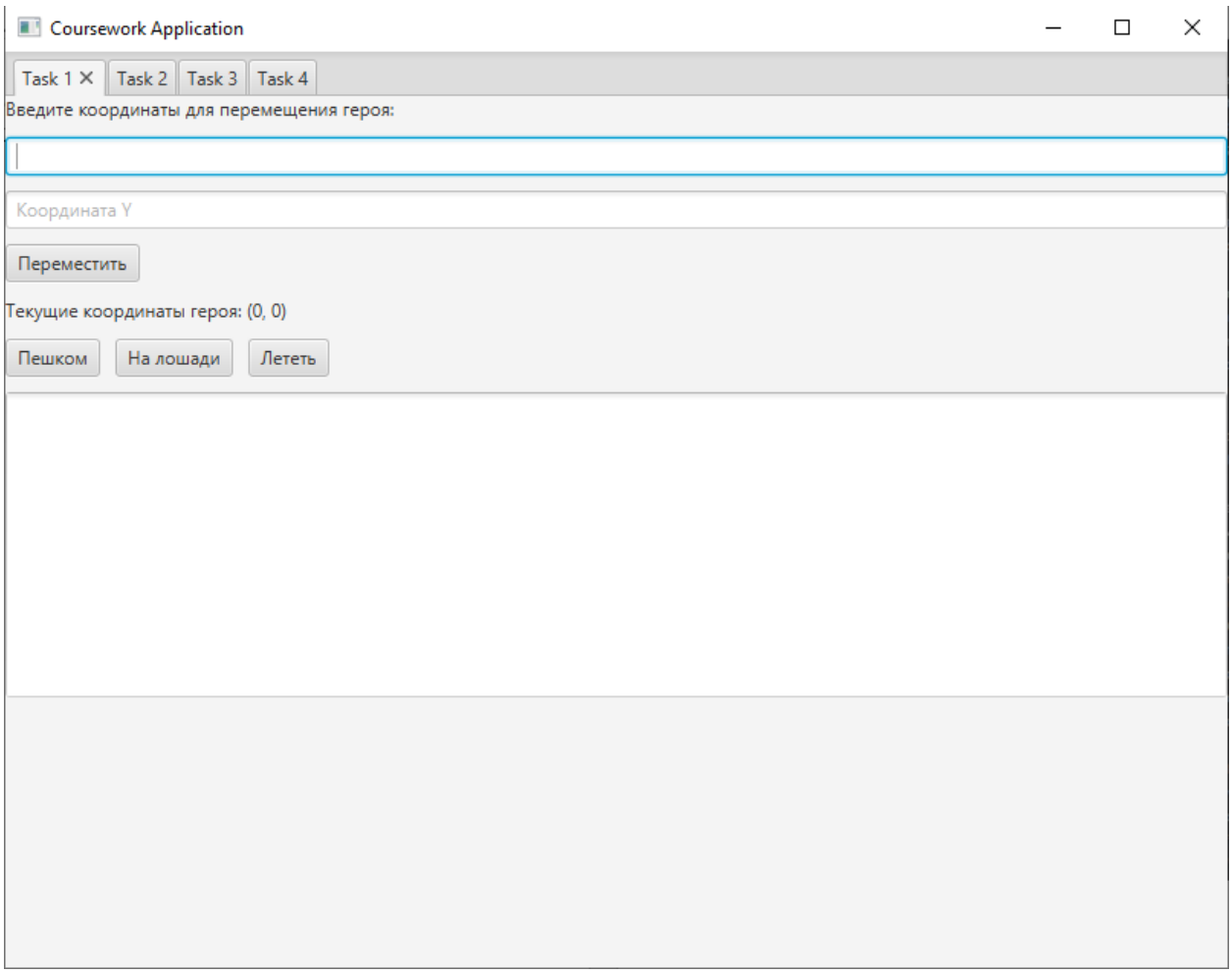

Рисунок 2. Графический интерфейс программы

Результаты выполнения программы

Лабораторная работа 1

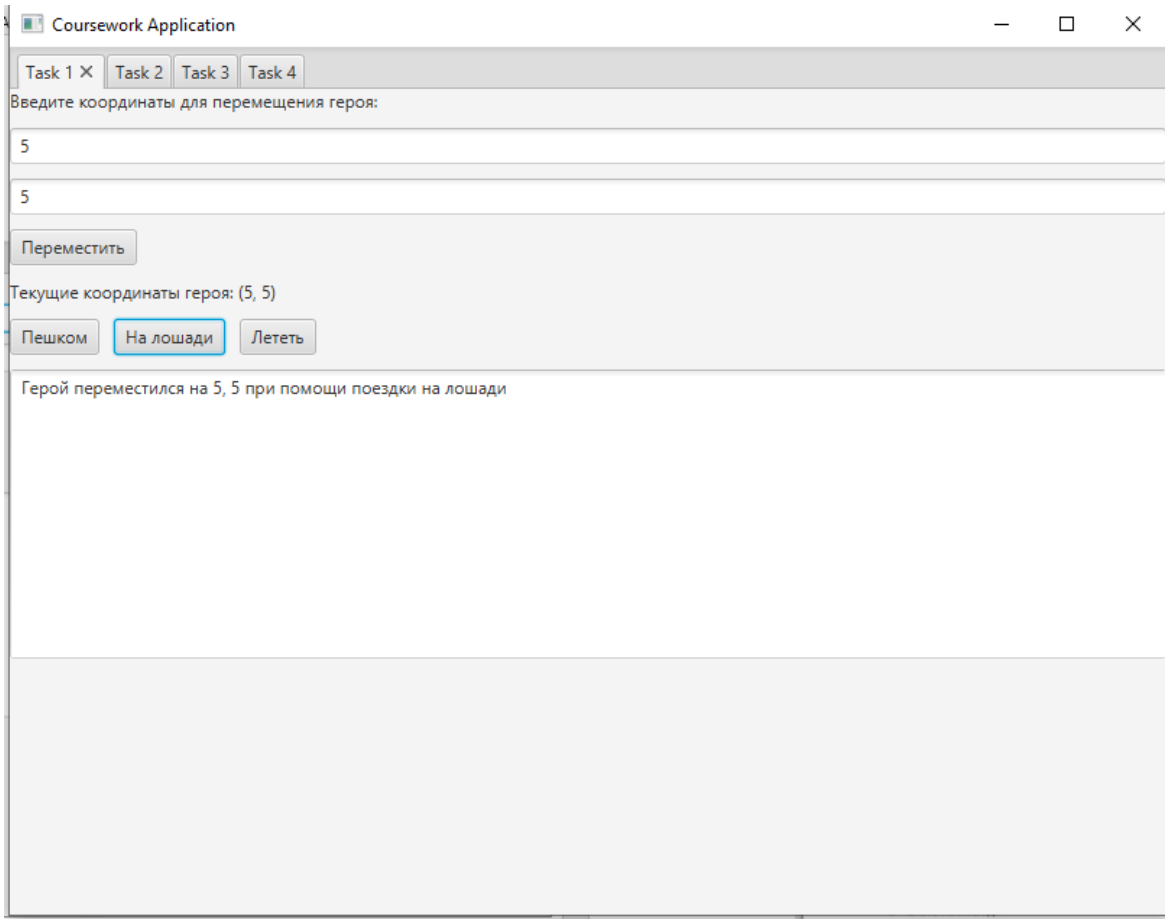

Рисунок 3. Выполнение 1 лабораторной работы. Выбор стратегии и результат

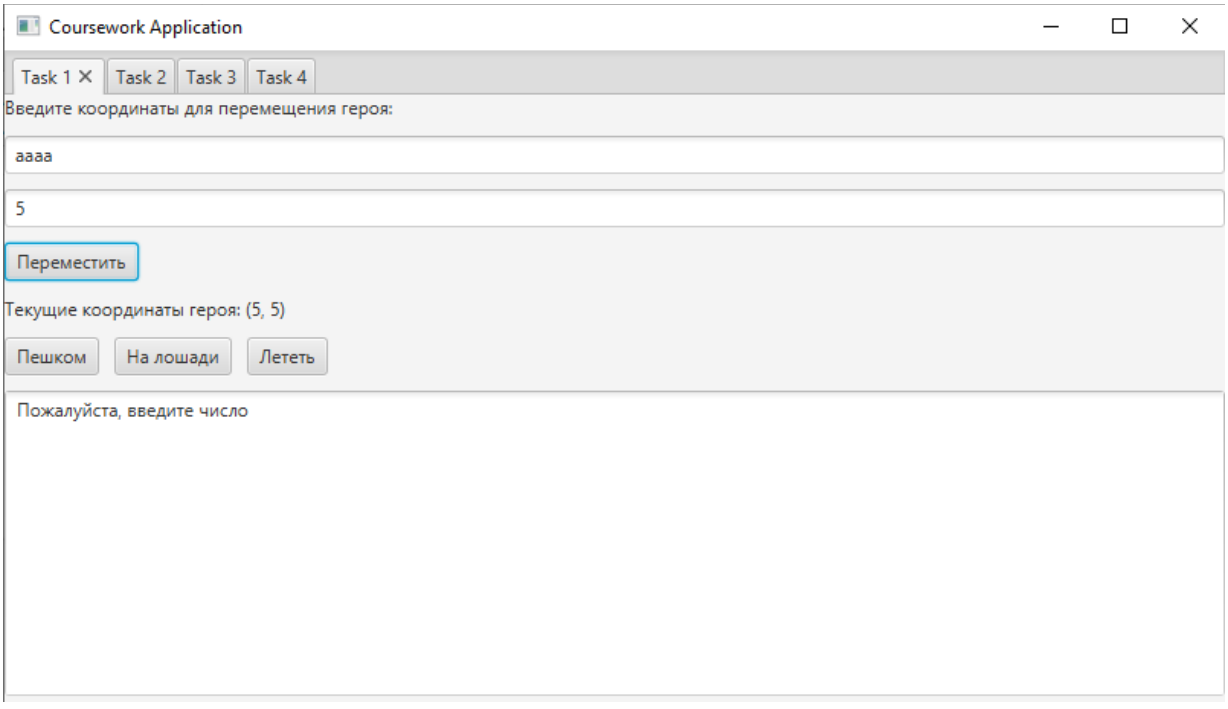

Рисунок 4. Выполнение 1 лабораторной работы. Обработка некорректного ввода

Лабораторная работа 2

Рисунок 5. Выполнение 2 лабораторной работы

Лабораторная работа 3

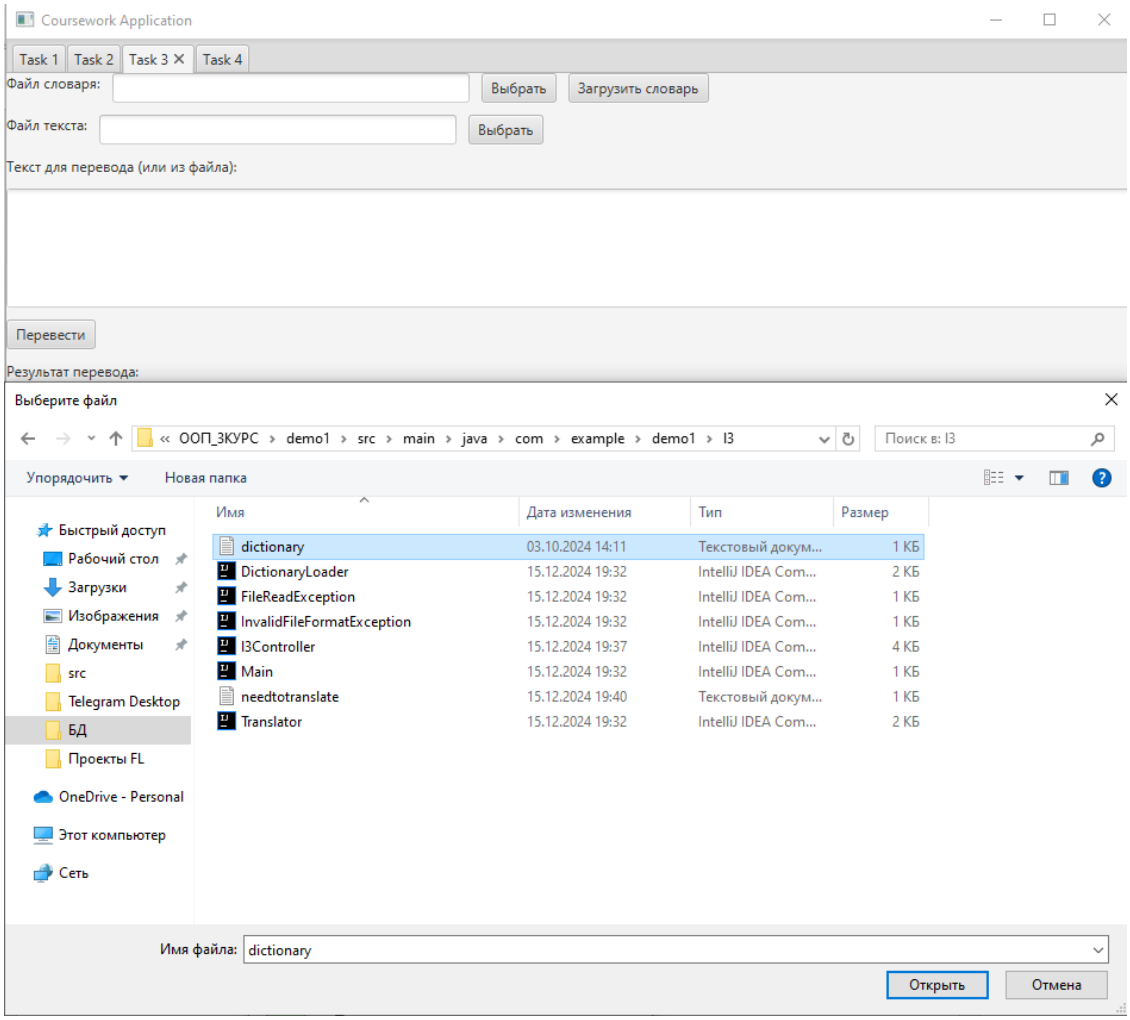

Рисунок 6. Выполнение 3 лабораторной работы. Выбор файла словаря

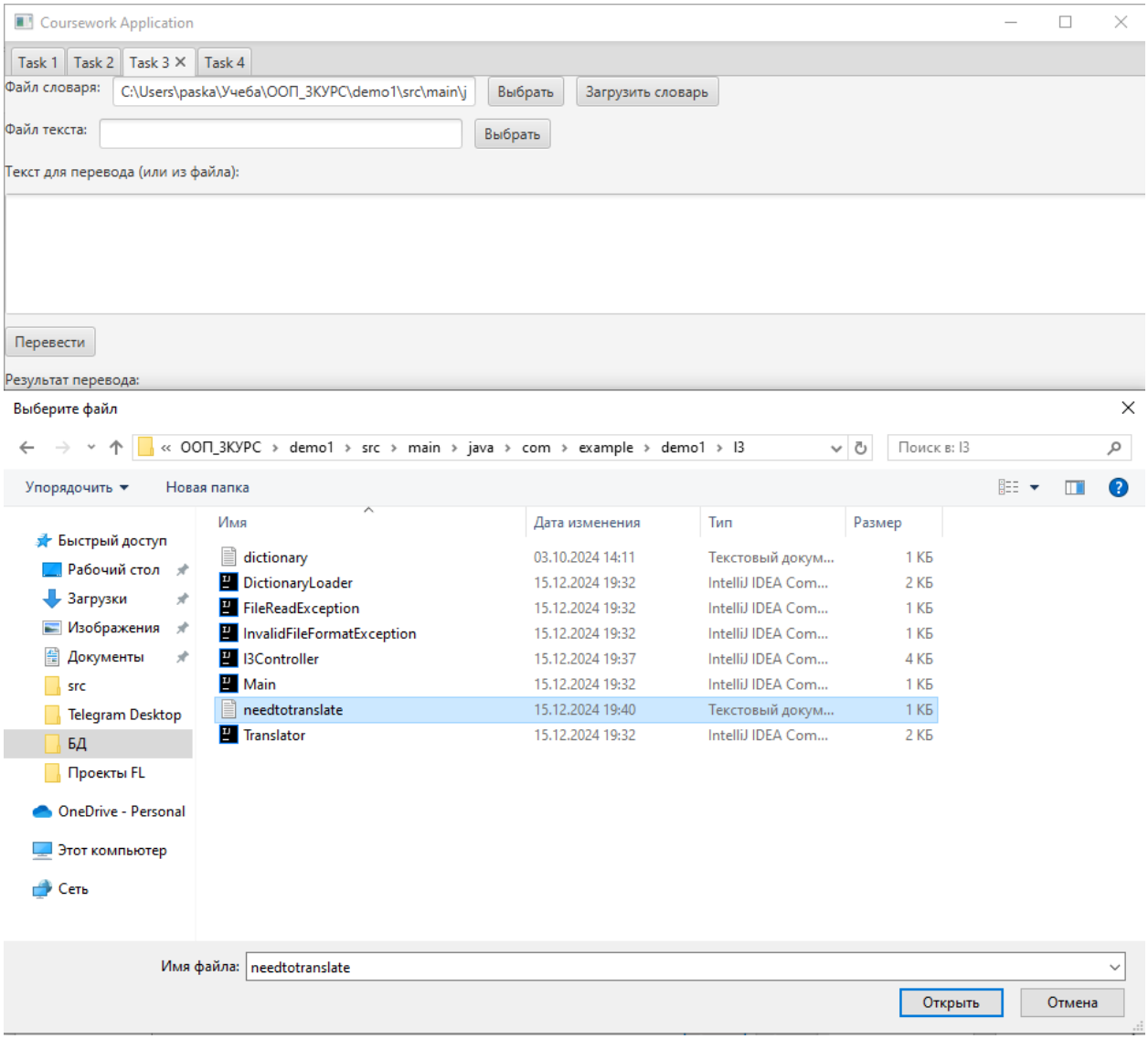

Рисунок 7. Выполнение 3 лабораторной работы. Выбор файла с текстом для перевода

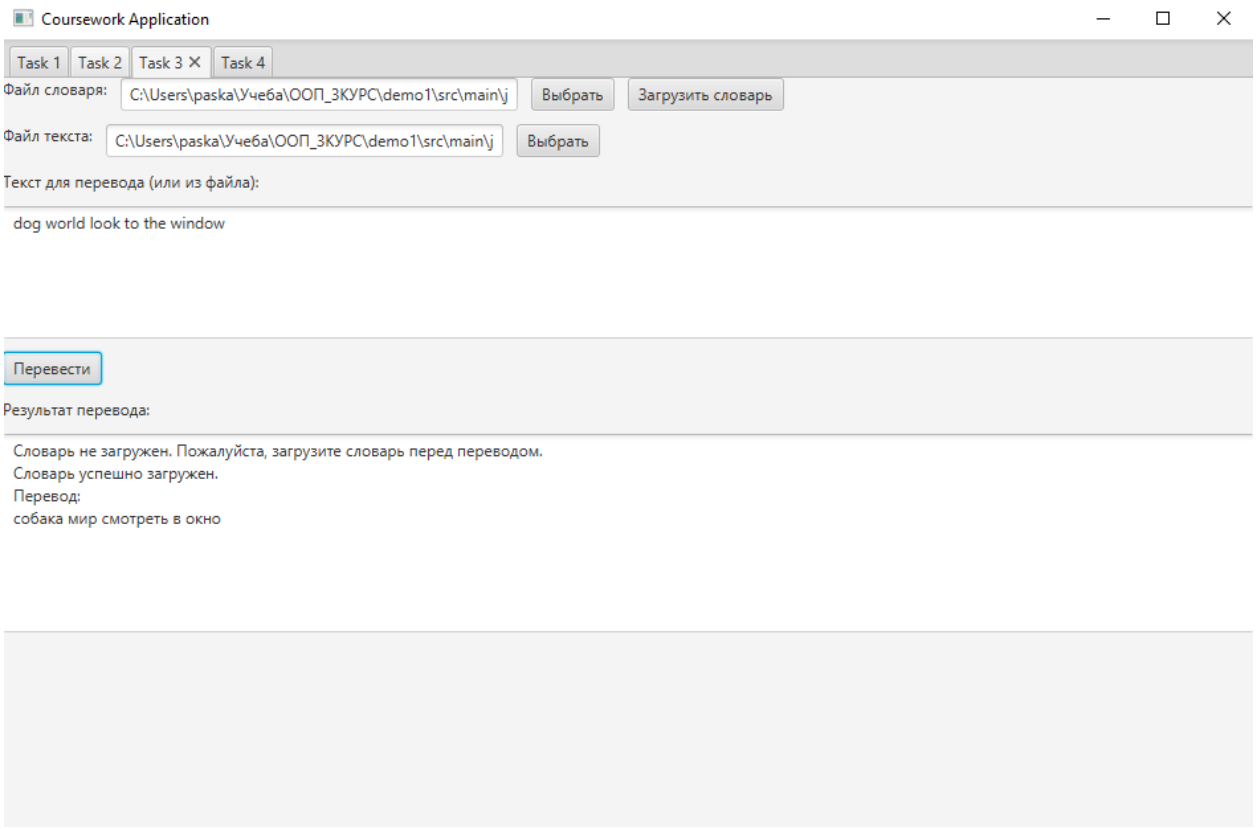

Рисунок 8. Выполнение 3 лабораторной работы. Вывод перевода

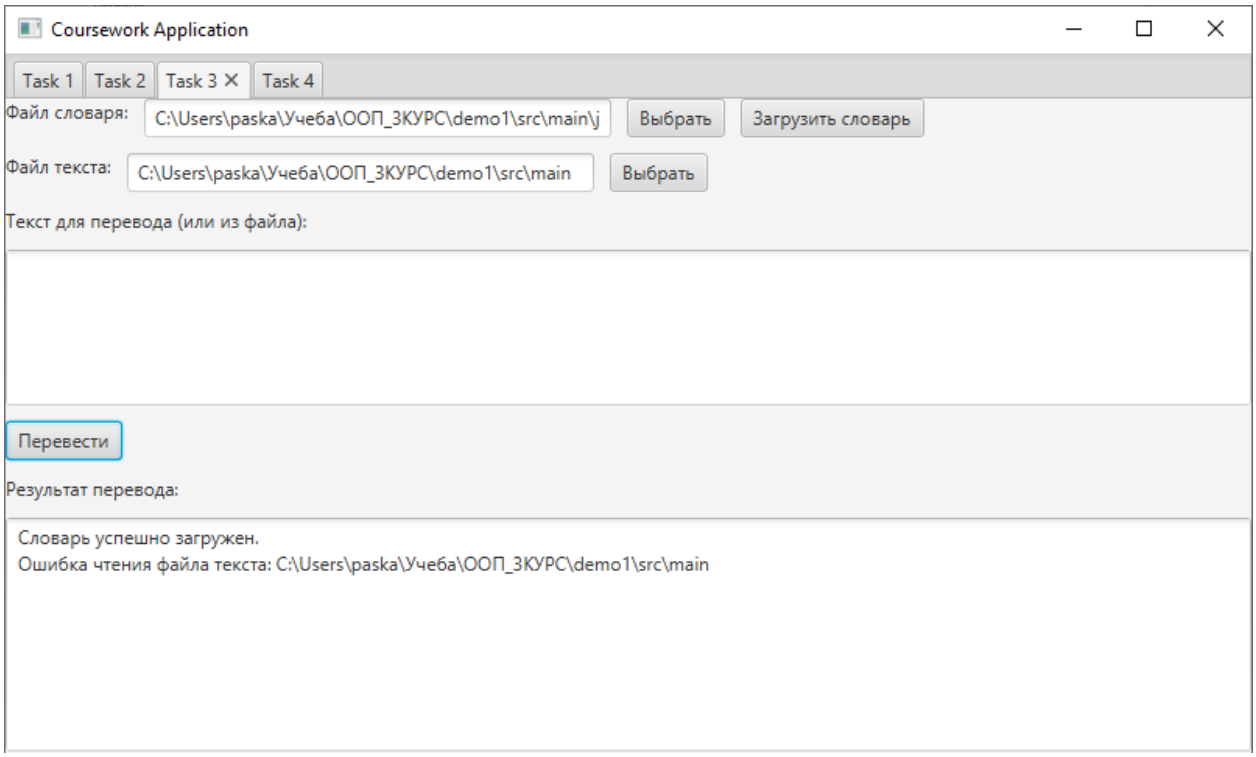

Рисунок 9. Выполнение 3 лабораторной работы. Обработка загрузки некорректного пути файла слов для перевода

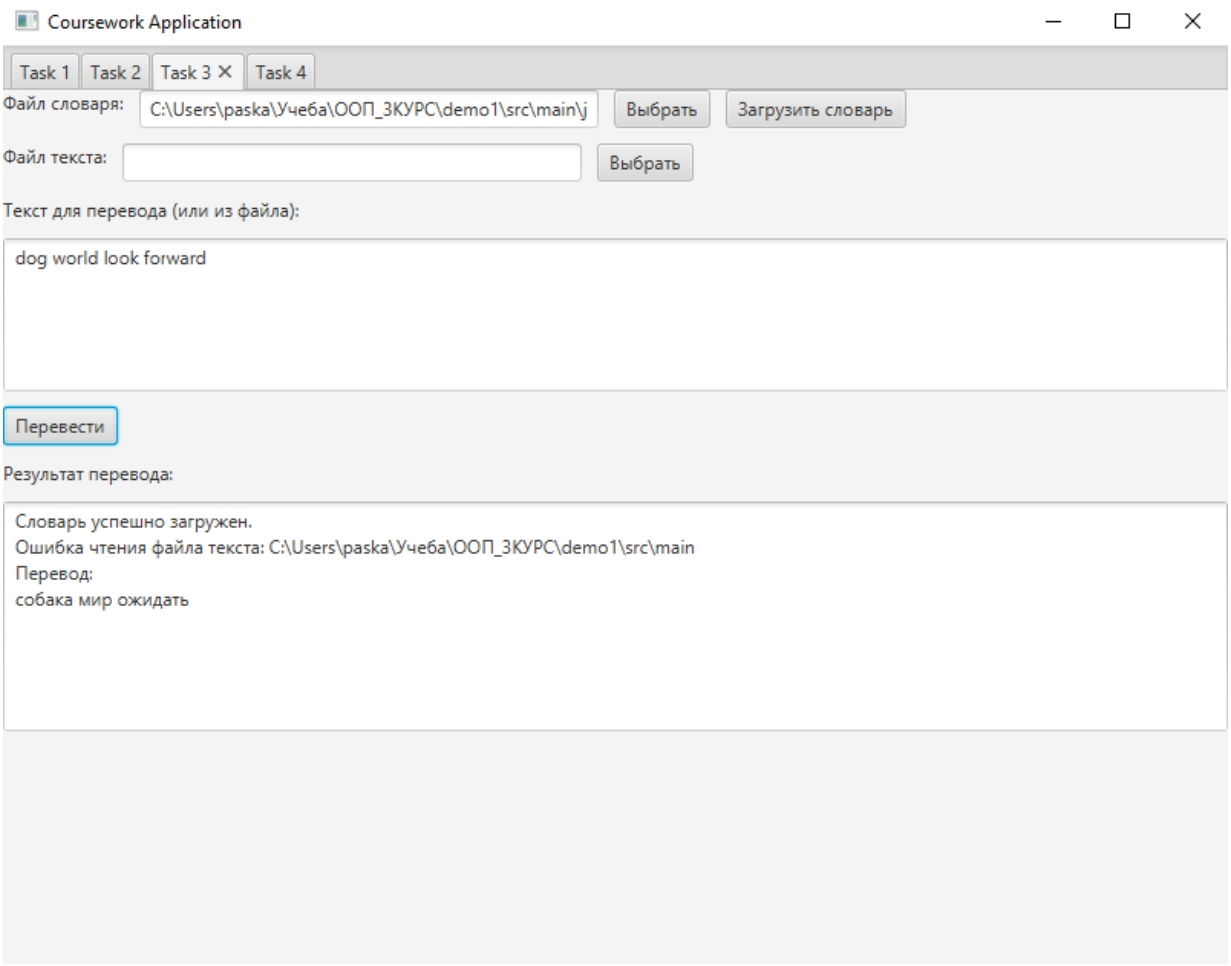

Рисунок 10. Выполнение 3 лабораторной работы. Вывод перевода слов, введенных вручную

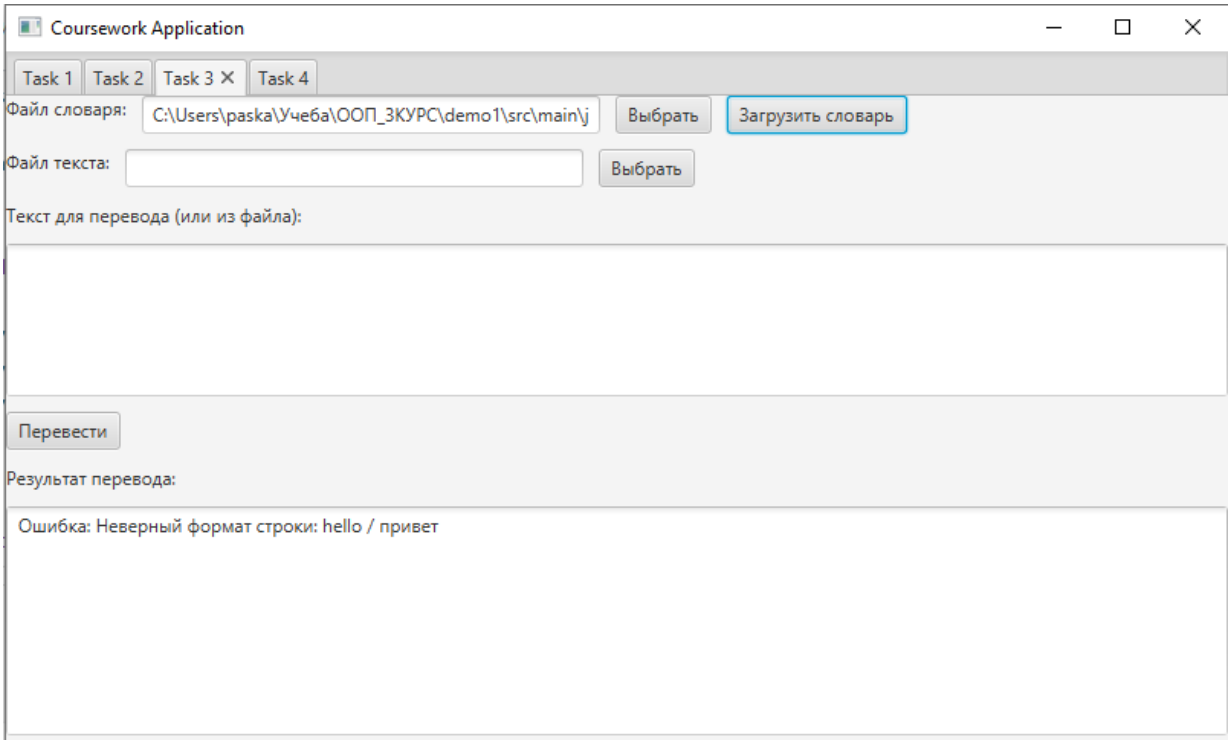

Рисунок 11. Выполнение 3 лабораторной работы. Обработка загрузки словаря с некорректным форматом данных

Лабораторная работа 4

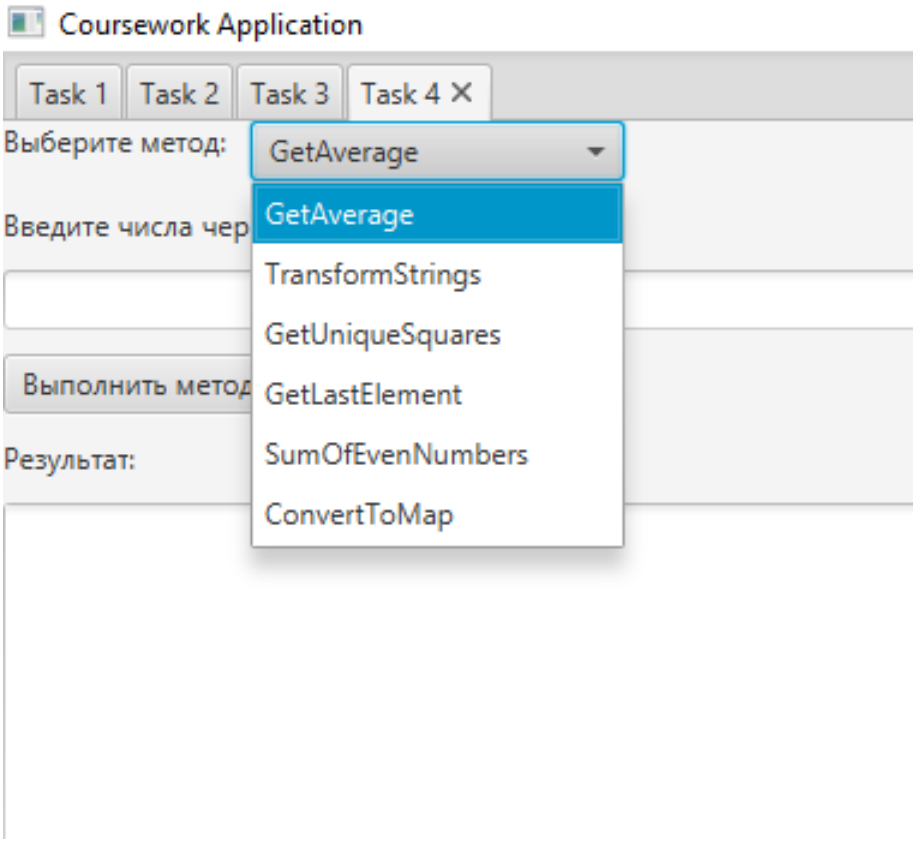

Рисунок 12. Выполнение 4 лабораторной работы. Выбор метода

Рисунок 13. Выполнение 4 лабораторной работы. Выполнение программы с выбранным методом GetAverage

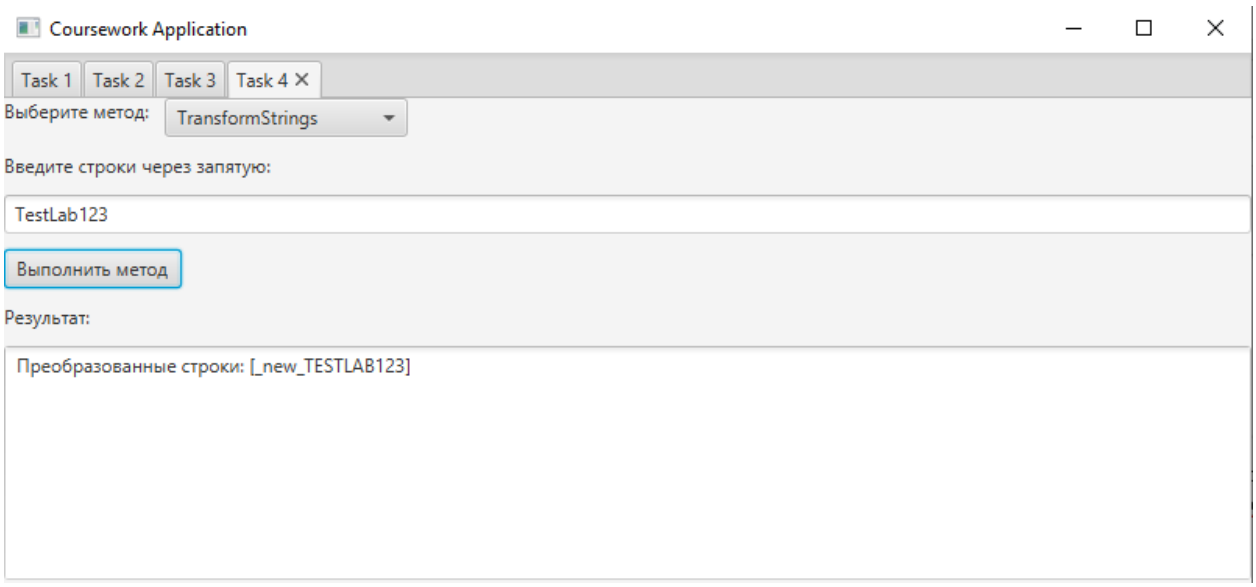

Рисунок 14. Выполнение 4 лабораторной работы. Выполнение программы с выбранным методом TransformString

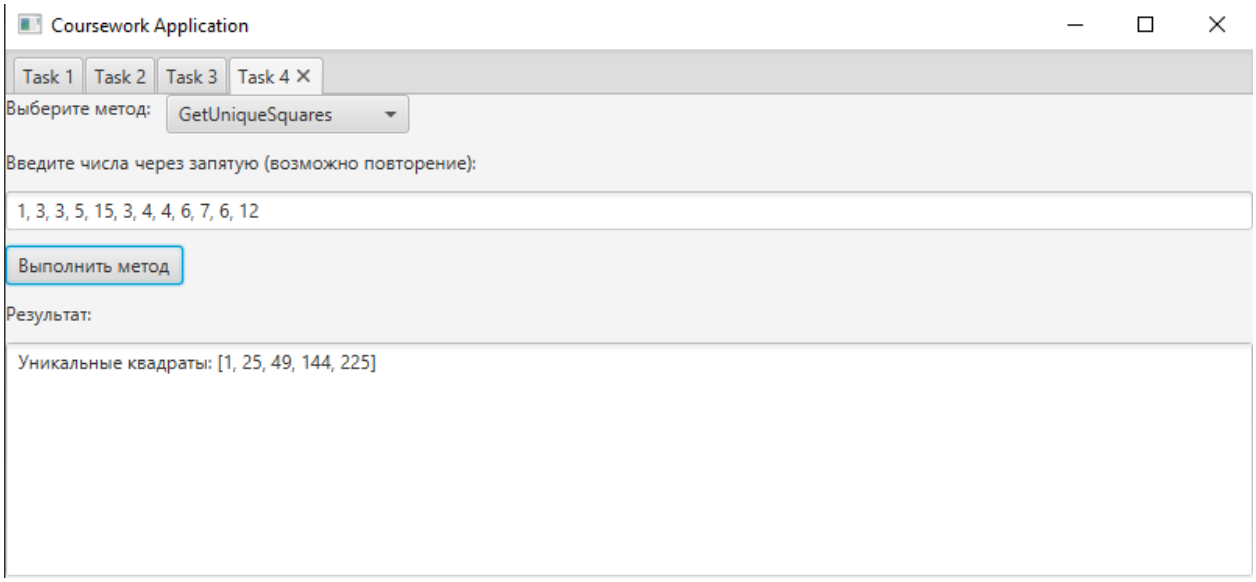

Рисунок 15. Выполнение 4 лабораторной работы. Выполнение программы с выбранным методом GetUniqueSquares

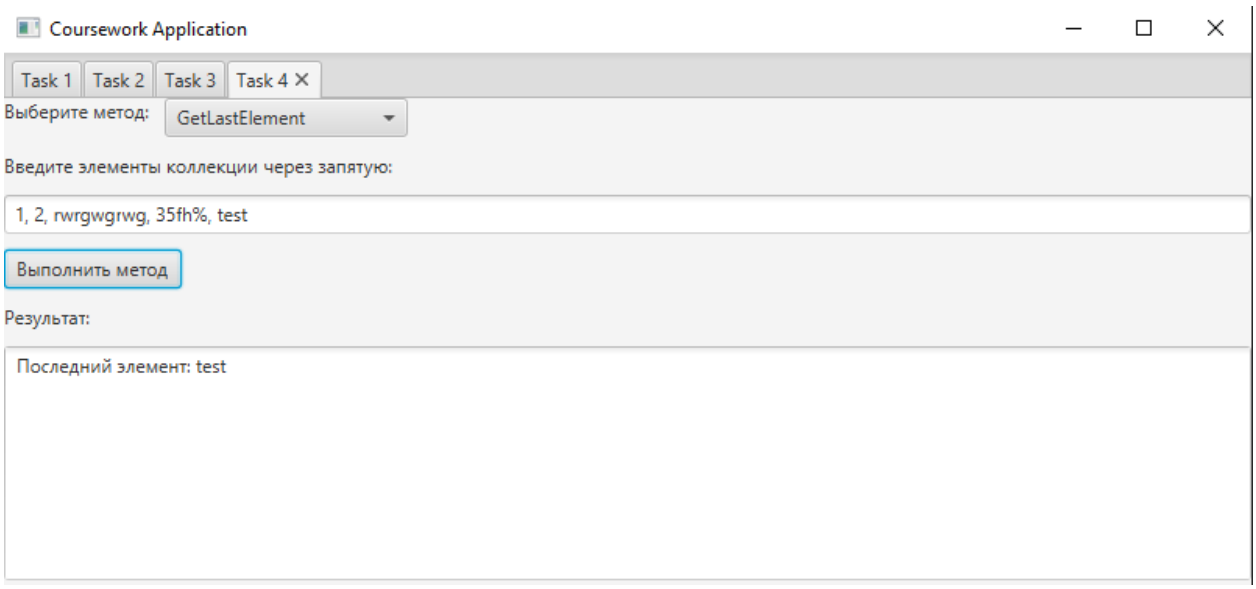

Рисунок 16. Выполнение 4 лабораторной работы. Выполнение программы с выбранным методом GetLastElement

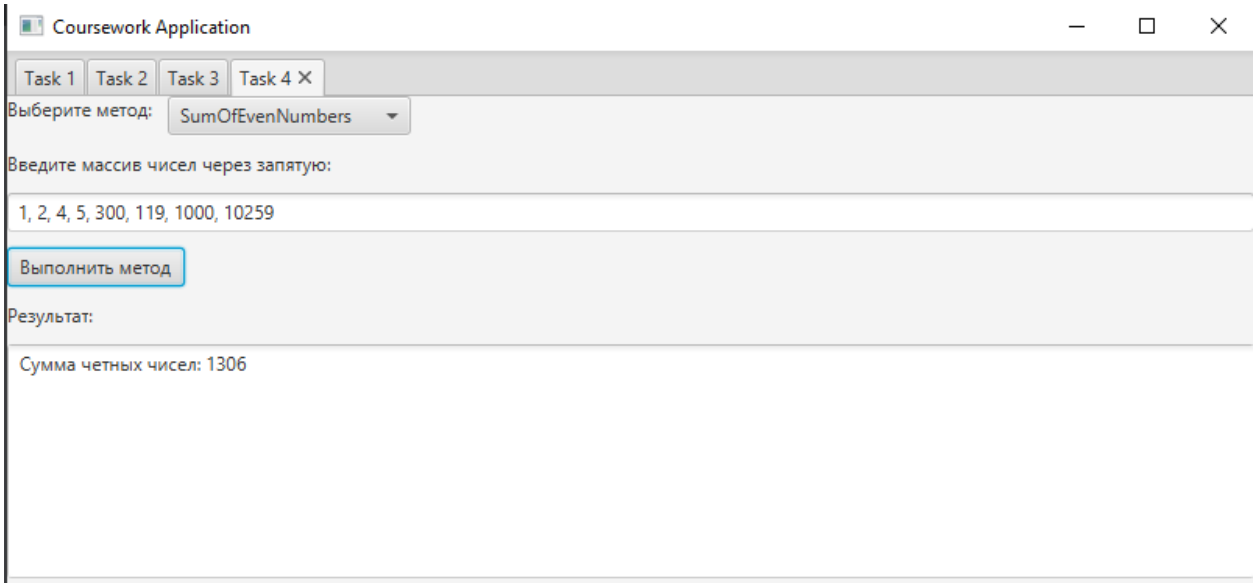

Рисунок 17. Выполнение 4 лабораторной работы. Выполнение программы с выбранным методом SumOfEvenNumbers

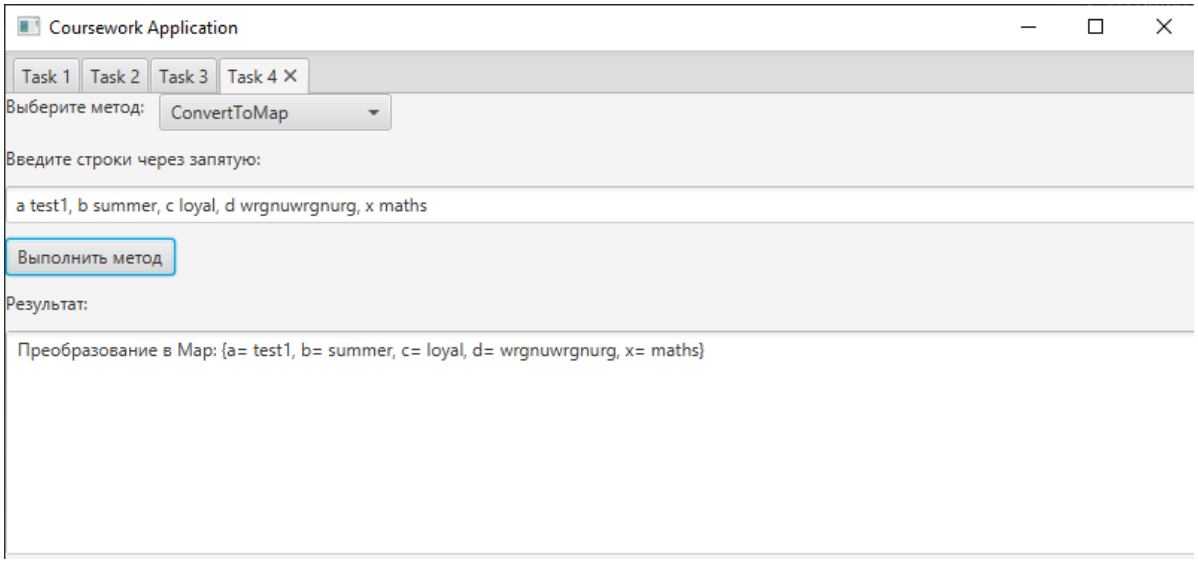

Рисунок 18. Выполнение 4 лабораторной работы. Выполнение программы с выбранным методом ConvertToMap

Рисунок 19. Выполнение 4 лабораторной работы. Выполнение программы с выбранным методом GetAverage, некорректный ввод

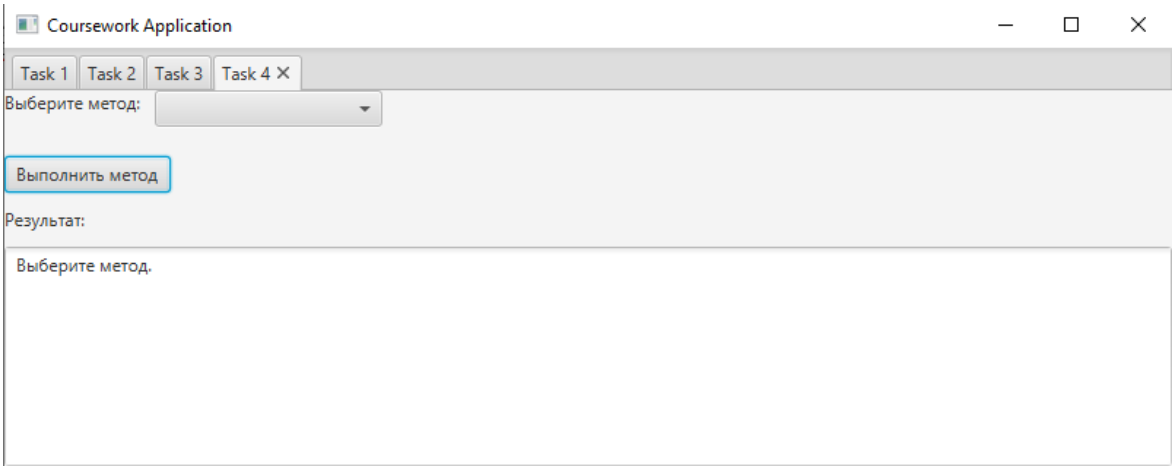

Рисунок 20. Выполнение 4 лабораторной работы. Попытка выполнения без выбранного метода

Выводы

На основе выполненной работы, было разработано приложение с графическим интерфейсом для выполнения лабораторных работ 1-4 на языке программирования Java с использованием фреймворка JavaFX.

Целью данной разработки было предоставление пользователю удобного и понятного средства для выполнения лабораторных работ, включая выбор задания, ввод входных данных и получение результатов, а также обработку некорректного ввода.

Программа демонстрирует правильное исполнение всех лабораторных работ. Практические результаты совпадают с теоретическими ожиданиями.

Таким образом, все цели курсового проекта были достигнуты.

Код проекта

Директория src\main\java\com\example\demo1

HelloApplication.java:

```
package com.example.demo1;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;

import java.io.IOException;

public class HelloApplication extends Application {
    @Override
    public void start(Stage stage) throws IOException {
        FXMLLoader fxmlLoader = new
FXMLLoader(HelloApplication.class.getResource("hello-view.fxml"));
        Scene scene = new Scene(fxmlLoader.load(), 800, 600);
        stage.setTitle("Coursework Application");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] args) {
        launch();
    }
}
```

HelloController.java:

```
package com.example.demo1;

import javafx.fxml.FXML;
import javafx.scene.control.TextArea;

public class HelloController {

    @FXML
    private TextArea outputArea;

    @FXML
    private void handleTask1() {
        outputArea.setText("Результат выполнения задания 1...");
    }

    @FXML
    private void handleTask2() {
        outputArea.setText("Результат выполнения задания 2...");
    }

    @FXML
    private void handleTask3() {
        outputArea.setText("Результат выполнения задания 3...");
    }

    @FXML
    private void handleTask4() {
        outputArea.setText("Результат выполнения задания 4...");
    }
}
```

Директория src\main\java\com\example\demo1\l1:

FlyStrategy.java:

```
package com.example.demo1.l1;

// Реализация стратегии: Лететь
public class FlyStrategy implements MoveStrategy {
    @Override
    public void move(int x, int y, Hero hero) {
        /*System.out.println("Герой летит к координатам (" + x + ", " + y +
").");*/
        hero.addCoordinates(x, y);
    }

    @Override
    public String getName() {
        return "полета";
    }
}
```

Hero.java:

```
package com.example.demo1.l1;

public class Hero {
    private int x; // Координата x
    private int y; // Координата y
    private MoveStrategy moveStrategy; // Стратегия перемещения

    public MoveStrategy getMoveStrategy() {
        return moveStrategy;
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

    // Конструктор, который инициализирует начальные координаты
    public Hero(int x, int y) {
        this.x = x;
        this.y = y;
    }

    // Устанавливаем стратегию перемещения
    public void setMoveStrategy(MoveStrategy moveStrategy) {
        this.moveStrategy = moveStrategy;
    }

    // Метод перемещения героя к новым координатам
    public void move(int newX, int newY) {
        if (moveStrategy != null) {
            moveStrategy.move(newX, newY, this);
            //System.out.println("Герой переместился в новые координаты (" +
x + ", " + y + ").");
        } else {
            throw new RuntimeException("Не выбрана стратегия движения");
            //System.out.println("Способ перемещения не выбран.");
```

```
        }
    }

    public void addCoordinates(int x, int y){
        this.x = x;
        this.y = y;
    }

    // Получение текущих координат
    public void getCoordinates() {
        System.out.println("Текущие координаты героя: (" + x + ", " + y +
").");
    }
}
```

HorseRideStrategy.java:

```
package com.example.demo1.l1;

// Реализация стратегии: На лошади
public class HorseRideStrategy implements MoveStrategy {
    @Override
    public void move(int x, int y, Hero hero) {
        /*System.out.println("Герой едет на лошади к координатам (" + x + ",
" + y + ").");*/
        hero.addCoordinates(x, y);
    }

    @Override
    public String getName() {
        return "поездки на лошади";
    }
}
```

MoveStrategy.java:

```
package com.example.demo1.l1;

interface MoveStrategy {
    void move(int x, int y, Hero hero);

    public String getName();
}
```

WalkStrategy.java:

```
package com.example.demo1.l1;

// Реализация стратегии: Пешком
public class WalkStrategy implements MoveStrategy {
    @Override
    public void move(int x, int y, Hero hero) {
        /*System.out.println("Герой идет пешком к координатам (" + x + ", " +
y + ").");*/
        hero.addCoordinates(x, y);
    }

    @Override
    public String getName() {
        return "шага";
    }
}
```

l1Controller.java:

```
package com.example.demo1.l1;

import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;

public class l1Controller {

    public TextArea logArea;
    @FXML
    private TextField xCoordinateField;

    @FXML
    private TextField yCoordinateField;

    @FXML
    private Label heroCoordinatesLabel;

    @FXML
    private Button walkButton;

    @FXML
    private Button horseRideButton;

    @FXML
    private Button flyButton;

    private Hero hero;

    @FXML
    public void initialize() {
        hero = new Hero(0, 0);
        updateHeroCoordinates();

        walkButton.setOnAction(event -> hero.setMoveStrategy(new
WalkStrategy()));
        horseRideButton.setOnAction(event -> hero.setMoveStrategy(new
HorseRideStrategy()));
        flyButton.setOnAction(event -> hero.setMoveStrategy(new
FlyStrategy()));
    }

    @FXML
    private void moveHero() {
        try {
            int newX = Integer.parseInt(xCoordinateField.getText());
            int newY = Integer.parseInt(yCoordinateField.getText());
            hero.move(newX, newY);
            heroCoordinatesLabel.setText("Текущие координаты героя: (" +
hero.getX() + ", " + hero.getY() + ")");
            logArea.setText("Герой переместился на " + hero.getX() + ", " +
hero.getY() + " при помощи " + hero.getMoveStrategy().getName());
        } catch (NumberFormatException e) {
            logArea.setText("Пожалуйста, введите число");
        } catch (Exception e) {
            logArea.setText(e.getMessage());
        }
    }
```

```
    private void updateHeroCoordinates() {
        heroCoordinatesLabel.setText("Текущие координаты героя: (" +
hero.getX() + ", " + hero.getY() + ")");
    }

}
```

Директория src\main\java\com\example\demo1\l2:

MyClass.java:

```
package com.example.demo1.l2;

public class MyClass {

    // Публичные методы
    @RunCount(3) // Аннотируем метод с параметром 3
    public void publicMethod(String message) {
        System.out.println("Публичный метод: " + message);
    }

    public void anotherPublicMethod(int number) {
        System.out.println("Другой публичный метод, число: " + number);
    }

    // Защищенные методы
    @RunCount(2) // Аннотируем метод с параметром 2
    protected void protectedMethod(int a, int b) {
        System.out.println("Защищенный метод, сумма: " + (a + b));
    }

    protected void anotherProtectedMethod() {
        System.out.println("Другой защищенный метод.");
    }

    // Приватные методы
    @RunCount(5) // Аннотируем метод с параметром 5
    private void privateMethod(String str, int n) {
        System.out.println("Приватный метод, строка: " + str + ", число: " +
n);
    }

    private void anotherPrivateMethod() {
        System.out.println("Другой приватный метод.");
    }
}
```

RunCount.java:

```
package com.example.demo1.l2;

import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;

// Аннотация с целочисленным параметром
@Retention(RetentionPolicy.RUNTIME) // Аннотация будет доступна во время
выполнения программы
@interface RunCount {
    int value(); // Параметр аннотации
}
```

l2Controller.java:

```
package com.example.demo1.l2;

import javafx.application.Platform;
import javafx.fxml.FXML;
import javafx.scene.control.TextArea;
import java.io.OutputStream;
import java.io.PrintStream;
import java.io.IOException;
import java.lang.reflect.*;

public class l2Controller {

    @FXML
    private TextArea logArea;

    private PrintStream originalOut;
    private PrintStream originalErr;

    @FXML
    public void initialize() {
        // Сохраняем оригинальные потоки
        originalOut = System.out;
        originalErr = System.err;

        // Перенаправляем потоки вывода
        redirectSystemStreams();
    }

    private void restoreSystemStreams() {
        System.setOut(originalOut);
        System.setErr(originalErr);
    }

    private void redirectSystemStreams() {
        OutputStream out = new OutputStream() {
            @Override
            public void write(byte[] b, int off, int len) throws IOException
{
                String text = new String(b, off, len);
                Platform.runLater(() -> logArea.appendText(text));
            }

            @Override
            public void write(int b) throws IOException {
                // Переводим байт в символ и добавляем в лог
                Platform.runLater(() ->
logArea.appendText(String.valueOf((char) b)));
            }
        };

        System.setOut(new PrintStream(out, true));
        System.setErr(new PrintStream(out, true));
    }

    @FXML
    public void invokeAnnotatedMethods() {
        redirectSystemStreams();
        try {
            MyClass myClassInstance = new MyClass();

            // Получаем все методы класса MyClass
```

```
            Method[] methods = MyClass.class.getDeclaredMethods();

            // Проходим по каждому методу
            for (Method method : methods) {
                // Проверяем, аннотирован ли метод аннотацией @RunCount
                if (method.isAnnotationPresent(RunCount.class)) {
                    // Проверяем, является ли метод защищённым или приватным
                    if (Modifier.isProtected(method.getModifiers()) ||
Modifier.isPrivate(method.getModifiers())) {
                        // Получаем аннотацию
                        RunCount runCount =
method.getAnnotation(RunCount.class);
                        int times = runCount.value(); // Сколько раз нужно
вызвать метод

                        // Делаем метод доступным для вызова, даже если он
приватный
                        method.setAccessible(true);

                        // Вызов метода столько раз, сколько указано в
аннотации
                        for (int i = 0; i < times; i++) {
                            // Получаем параметры метода
                            Class<?>[] parameterTypes =
method.getParameterTypes();
                            Object[] params =
getParamsForMethod(parameterTypes); // Используем отдельный метод для
создания параметров

                            try {
                                // Вызов метода с параметрами
                                Object result =
method.invoke(myClassInstance, params);

                                // Если метод возвращает результат, выводим
его
                                if (result != null) {
                                    System.out.println("Результат: " +
result.toString());
                                }
                            } catch (Exception e) {
                                System.err.println("Ошибка при вызове метода:
" + e.getMessage());
                            }
                        }
                    }
                }
            }
        } catch (Exception e) {
            System.err.println("Ошибка выполнения: " + e.getMessage());
        } finally {
            restoreSystemStreams();
        }
    }

    // Метод для определения параметров для метода
    public static Object[] getParamsForMethod(Class<?>[] parameterTypes) {
        Object[] params = new Object[parameterTypes.length];
        for (int j = 0; j < parameterTypes.length; j++) {
            if (parameterTypes[j] == String.class) {
                params[j] = "Test String"; // Если параметр строка
            } else if (parameterTypes[j] == int.class) {
                params[j] = 42; // Если параметр целое число
```

```
            } else if (parameterTypes[j] == boolean.class) {
                params[j] = true; // Если параметр логический (boolean)
            } else if (parameterTypes[j] == double.class) {
                params[j] = 3.14; // Если параметр типа double
            } else if (parameterTypes[j] == float.class) {
                params[j] = 2.5f; // Если параметр типа float
            } else if (parameterTypes[j] == char.class) {
                params[j] = 'A'; // Если параметр типа char
            } else if (parameterTypes[j] == int[].class) {
                params[j] = new int[]{1, 2, 3}; // Если параметр массив int
            } else if (parameterTypes[j] == String[].class) {
                params[j] = new String[]{"One", "Two", "Three"}; // Если
параметр массив строк
            } else {
                params[j] = null; // Если параметр неизвестен, оставляем null
            }
        }
        return params;
    }
}
```

Директория src\main\java\com\example\demo1\l3:

DictionaryLoader.java:

```
package com.example.demo1.l3;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;

public class DictionaryLoader {
    private Map<String, String> dictionary = new HashMap<>();

    public DictionaryLoader(String filePath) throws IOException,
FileReadException, InvalidFileFormatException {
        loadDictionary(filePath);
    }

    private void loadDictionary(String filePath) throws IOException,
FileReadException, InvalidFileFormatException {
        try (BufferedReader reader = new BufferedReader(new
FileReader(filePath))) {
            String line;
            while ((line = reader.readLine()) != null) {
                String[] parts = line.split("\\|");
                if (parts.length != 2) {
                    throw new InvalidFileFormatException("Неверный формат
строки: " + line);
                }
                String word = parts[0].trim().toLowerCase();
                String translation = parts[1].trim();
                dictionary.put(word, translation);
            }
        } catch (IOException e) {
            throw new FileReadException("Ошибка чтения файла: " + filePath,
e);
        }
    }
```

```
    public Map<String, String> getDictionary() {
        return dictionary;
    }
}
```

Translator.java:

```
package com.example.demo1.13;
import java.util.Map;
import java.util.Arrays;
import java.util.List;

public class Translator {
    private Map<String, String> dictionary;

    public Translator(Map<String, String> dictionary) {
        this.dictionary = dictionary;
    }

    public String translate(String input) {
        String[] words = input.split("\\s+");
        StringBuilder result = new StringBuilder();

        for (int i = 0; i < words.length; i++) {
            String word = words[i].toLowerCase();

            // Найти все подходящие фразы из словаря
            String bestMatch = word;
            for (String dictWord : dictionary.keySet()) {
                List<String> dictWordsList = Arrays.asList(dictWord.split("
"));
                int len = dictWordsList.size();
                if (i + len <= words.length) {
                    List<String> inputWordsList =
Arrays.asList(Arrays.copyOfRange(words, i, i + len));
                    if (inputWordsList.equals(dictWordsList)) {
                        if (dictWord.length() > bestMatch.length()) {
                            bestMatch = dictWord;
                        }
                    }
                }
            }

            // Выполнить перевод
            if (dictionary.containsKey(bestMatch)) {
                result.append(dictionary.get(bestMatch)).append(" ");
                i += bestMatch.split(" ").length - 1; // Пропустить слова,
если фраза из нескольких слов
            } else {
                result.append(words[i]).append(" ");
            }
        }

        return result.toString().trim();
    }
}
```

FileReadException.java:

```
package com.example.demo1.l3;
public class FileReadException extends Exception {
    public FileReadException(String message, Throwable cause) {
        super(message, cause);
    }
}
```

InvalidFileFormatException.java

```
package com.example.demo1.l3;
public class InvalidFileFormatException extends Exception {
    public InvalidFileFormatException(String message) {
        super(message);
    }
}
```

l3Controller.java:

```
package com.example.demo1.l3;

import javafx.application.Platform;
import javafx.fxml.FXML;
import javafx.scene.control.*;
import javafx.stage.FileChooser;

import java.io.File;
import java.io.IOException;

public class l3Controller {

    @FXML
    private TextField dictionaryFilePathField;

    @FXML
    private TextField textFilePathField;

    @FXML
    private TextArea inputTextArea;

    @FXML
    private TextArea outputTextArea;

    @FXML
    private Button loadDictionaryButton;

    @FXML
    private Button chooseDictionaryButton;

    @FXML
    private Button chooseTextButton;

    @FXML
    private Button translateButton;

    private Translator translator;

    @FXML
    public void initialize() {
        chooseDictionaryButton.setOnAction(event ->
```

```
chooseFile(dictionaryFilePathField));
        chooseTextButton.setOnAction(event -> chooseFile(textFilePathField));
        loadDictionaryButton.setOnAction(event -> loadDictionary());
        translateButton.setOnAction(event -> translateText());
    }

    private void chooseFile(TextField targetField) {
        FileChooser fileChooser = new FileChooser();
        fileChooser.setTitle("Выберите файл");
        File file = fileChooser.showOpenDialog(null);
        if (file != null) {
            targetField.setText(file.getAbsolutePath());
        }
    }

    private void loadDictionary() {
        String filePath = dictionaryFilePathField.getText().trim();
        if (filePath.isEmpty()) {
            outputTextArea.appendText("Укажите путь к файлу словаря.\n");
            return;
        }

        try {
            DictionaryLoader loader = new DictionaryLoader(filePath);
            translator = new Translator(loader.getDictionary());
            outputTextArea.appendText("Словарь успешно загружен.\n");
        } catch (FileReadException | InvalidFileFormatException e) {
            outputTextArea.appendText("Ошибка: " + e.getMessage() + "\n");
        } catch (IOException e) {
            outputTextArea.appendText("Ошибка чтения файла: " +
e.getMessage() + "\n");
        }
    }

    private void translateText() {
        if (translator == null) {
            outputTextArea.appendText("Словарь не загружен. Пожалуйста,
загрузите словарь перед переводом.\n");
            return;
        }

        String inputText = inputTextArea.getText().trim();
        if (!textFilePathField.getText().trim().isEmpty()) {
            // Если указан файл для перевода, загружаем текст из него
            try {
                File file = new File(textFilePathField.getText().trim());
                inputText = new
String(java.nio.file.Files.readAllBytes(file.toPath()));
                inputTextArea.setText(inputText); // Заполняем поле текстом
из файла
            } catch (IOException e) {
                outputTextArea.appendText("Ошибка чтения файла текста: " +
e.getMessage() + "\n");
                return;
            }
        }

        if (inputText.isEmpty()) {
            outputTextArea.appendText("Введите текст для перевода.\n");
            return;
        }

        final String finalInputText = inputText;
```

```
        Platform.runLater(() -> {
            String translatedText = translator.translate(finalInputText);
            outputTextArea.appendText("Перевод:\n" + translatedText + "\n");
        });
    }
}
```

Директория src\main\java\com\example\demo1\l4:

GetAverage.java:

```
package com.example.demo1.l4;

import java.util.List;

public class GetAverage {
    public static double getAverage(List<Integer> numbers) {
        return numbers.stream()
                .mapToInt(Integer::intValue)
                .average()
                .orElseThrow(() -> new IllegalArgumentException("Список
пуст"));
    }
}
```

GetLastElement.java:

```
package com.example.demo1.l4;

import java.util.Collection;

public class GetLastElement {
    public static <T> T getLastElement(Collection<T> collection) {
        return collection.stream()
                .reduce((first, second) -> second)
                .orElseThrow(() -> new IllegalArgumentException("Коллекция
пуста"));
    }
}
```

GetUniqueSqares.java:

```
package com.example.demo1.l4;

import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

public class GetUniqueSquares {
    public static List<Integer> getUniqueSquares(List<Integer> numbers) {
        Map<Integer, Long> frequencyMap = numbers.stream()
                .collect(Collectors.groupingBy(n -> n,
Collectors.counting()));

        return frequencyMap.entrySet().stream()
                .filter(entry -> entry.getValue() == 1)
                .map(entry -> entry.getKey() * entry.getKey())
                .collect(Collectors.toList());
    }
}
```

TransformStrings.java:

```java
package com.example.demo1.14;

import java.util.List;
import java.util.stream.Collectors;

public class TransformStrings {
    public static List<String> transformStrings(List<String> strings) {
        return strings.stream()
                .map(s -> "_new_" + s.toUpperCase())
                .collect(Collectors.toList());
    }
}
```

SumOfEvenNumbers.java:

```java
package com.example.demo1.14;

import java.util.Arrays;

public class SumOfEvenNumbers {
    public static int sumOfEvenNumbers(int[] numbers) {
        return Arrays.stream(numbers)
                .filter(n -> n % 2 == 0)
                .sum();
    }
}
```

ConvertToMap.java:

```java
package com.example.demo1.14;

import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

public class ConvertToMap {
    public static Map<Character, String> convertToMap(List<String> strings) {
        return strings.stream()
                .collect(Collectors.toMap(
                        s -> s.charAt(0), // первый символ - ключ
                        s -> s.substring(1) // остальная часть - значение
                ));
    }
}
```

l4Controller.java:

```
package com.example.demo1.l4;

import javafx.fxml.FXML;
import javafx.scene.control.*;
import javafx.scene.layout.VBox;

import java.util.*;
import java.util.stream.Collectors;

public class l4Controller {

    @FXML
    private ComboBox<String> methodComboBox;

    @FXML
    private VBox inputFieldsBox;

    @FXML
    private TextArea outputTextArea;

    @FXML
    private Button executeButton;

    @FXML
    public void initialize() {
        methodComboBox.getItems().addAll(
                "GetAverage",
                "TransformStrings",
                "GetUniqueSquares",
                "GetLastElement",
                "SumOfEvenNumbers",
                "ConvertToMap"
        );

        methodComboBox.setOnAction(event -> updateInputFields());
        executeButton.setOnAction(event -> executeMethod());
    }

    private void updateInputFields() {
        inputFieldsBox.getChildren().clear();
        String selectedMethod = methodComboBox.getValue();

        if (selectedMethod == null) return;

        switch (selectedMethod) {
            case "GetAverage":
                inputFieldsBox.getChildren().add(new Label("Введите числа
через запятую:"));
                inputFieldsBox.getChildren().add(new TextField());
                break;

            case "TransformStrings":
            case "ConvertToMap":
                inputFieldsBox.getChildren().add(new Label("Введите строки
через запятую:"));
                inputFieldsBox.getChildren().add(new TextField());
                break;

            case "GetUniqueSquares":
                inputFieldsBox.getChildren().add(new Label("Введите числа
через запятую (возможно повторение):"));
```

```
                inputFieldsBox.getChildren().add(new TextField());
                break;

            case "GetLastElement":
                inputFieldsBox.getChildren().add(new Label("Введите элементы
коллекции через запятую:"));
                inputFieldsBox.getChildren().add(new TextField());
                break;

            case "SumOfEvenNumbers":
                inputFieldsBox.getChildren().add(new Label("Введите массив
чисел через запятую:"));
                inputFieldsBox.getChildren().add(new TextField());
                break;

            default:
                break;
        }
    }

    private void executeMethod() {
        String selectedMethod = methodComboBox.getValue();
        if (selectedMethod == null) {
            outputTextArea.appendText("Выберите метод.\n");
            return;
        }

        List<String> inputFields = inputFieldsBox.getChildren().stream()
                .filter(node -> node instanceof TextField)
                .map(node -> ((TextField) node).getText())
                .collect(Collectors.toList());

        if (inputFields.isEmpty() || inputFields.get(0).isEmpty()) {
            outputTextArea.appendText("Введите данные для метода.\n");
            return;
        }

        String result = "";

        try {
            switch (selectedMethod) {
                case "GetAverage":
                    List<Integer> numbers =
Arrays.stream(inputFields.get(0).split(","))
                            .map(String::trim)  // Убираем пробелы
                            .map(Integer::parseInt)  // Преобразуем в целые
числа
                            .collect(Collectors.toList());
                    result = "Среднее значение: " +
GetAverage.getAverage(numbers);
                    break;

                case "GetUniqueSquares":
                    List<Integer> uniqueSquaresNumbers =
Arrays.stream(inputFields.get(0).split(","))
                            .map(String::trim)  // Убираем пробелы
                            .map(Integer::parseInt)  // Преобразуем в целые
числа
                            .collect(Collectors.toList());
                    result = "Уникальные квадраты: " +
GetUniqueSquares.getUniqueSquares(uniqueSquaresNumbers);
                    break;
```

```
                case "SumOfEvenNumbers":
                    int[] array =
Arrays.stream(inputFields.get(0).split(","))
                            .map(String::trim)   // Убираем пробелы
                            .mapToInt(Integer::parseInt)   // Преобразуем в
целые числа
                            .toArray();
                    result = "Сумма четных чисел: " +
SumOfEvenNumbers.sumOfEvenNumbers(array);
                    break;

                case "TransformStrings":
                    List<String> strings =
Arrays.stream(inputFields.get(0).split(","))
                            .map(String::trim)
                            .collect(Collectors.toList());
                    result = "Преобразованные строки: " +
TransformStrings.transformStrings(strings);
                    break;

                case "GetLastElement":
                    List<String> elements =
Arrays.stream(inputFields.get(0).split(","))
                            .map(String::trim)
                            .collect(Collectors.toList());
                    result = "Последний элемент: " +
GetLastElement.getLastElement(elements);
                    break;

                case "ConvertToMap":
                    List<String> mapStrings =
Arrays.stream(inputFields.get(0).split(","))
                            .map(String::trim)
                            .collect(Collectors.toList());
                    Map<Character, String> map =
ConvertToMap.convertToMap(mapStrings);
                    result = "Преобразование в Map: " + map;
                    break;

                default:
                    result = "Метод не поддерживается.";
            }
        } catch (NumberFormatException e) {
            outputTextArea.appendText("Введите числа через запятую.\n");
            return;
        }

        outputTextArea.appendText(result + "\n");
    }

}
```

Директория \src\main\resources\com\example\demo1:

hello-view.fxml:

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<TabPane xmlns:fx="http://javafx.com/fxml"
fx:controller="com.example.demo1.HelloController">
    <Tab text="Task 1">
        <fx:include source="l1-view.fxml" />
    </Tab>
    <Tab text="Task 2">
        <fx:include source="l2-view.fxml" />
    </Tab>
    <Tab text="Task 3">
        <fx:include source="l3-view.fxml" />
    </Tab>
    <Tab text="Task 4">
        <fx:include source="l4-view.fxml" />
    </Tab>
</TabPane>
```

l1-view.fxml:

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox spacing="10" xmlns:fx="http://javafx.com/fxml"
fx:controller="com.example.demo1.l1.l1Controller">
    <Label text="Введите координаты для перемещения героя:"/>
    <TextField fx:id="xCoordinateField" promptText="Координата X"/>
    <TextField fx:id="yCoordinateField" promptText="Координата Y"/>
    <Button text="Переместить" onAction="#moveHero"/>
    <Label fx:id="heroCoordinatesLabel" text="Текущие координаты героя: (0,
0)"/>
    <HBox spacing="10">
        <Button fx:id="walkButton" text="Пешком"/>
        <Button fx:id="horseRideButton" text="На лошади"/>
        <Button fx:id="flyButton" text="Лететь"/>
    </HBox>
    <TextArea fx:id="logArea" editable="false" prefHeight="200"
wrapText="true"/>
</VBox>
```

l2-view.fxml:

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<?import java.lang.*?>
<?import java.util.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox spacing="10" xmlns:fx="http://javafx.com/fxml"
fx:controller="com.example.demo1.l2.l2Controller">
    <Button fx:id="invokeAnnotatedMethod" text="Вызвать методы"
onAction="#invokeAnnotatedMethods"/>
```

```
    <TextArea fx:id="logArea" editable="false" prefHeight="400"
wrapText="true" />
</VBox>
```

l3-view.fxml:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.demo1.l3.l3Controller" spacing="10"
alignment="TOP_LEFT">
    <HBox spacing="10">
        <Label text="Файл словаря:"/>
        <TextField fx:id="dictionaryFilePathField" prefWidth="300"/>
        <Button fx:id="chooseDictionaryButton" text="Выбрать"/>
        <Button fx:id="loadDictionaryButton" text="Загрузить словарь"/>
    </HBox>

    <HBox spacing="10">
        <Label text="Файл текста:"/>
        <TextField fx:id="textFilePathField" prefWidth="300"/>
        <Button fx:id="chooseTextButton" text="Выбрать"/>
    </HBox>

    <Label text="Текст для перевода (или из файла):"/>
    <TextArea fx:id="inputTextArea" prefHeight="100"/>

    <Button fx:id="translateButton" text="Перевести"/>

    <Label text="Результат перевода:"/>
    <TextArea fx:id="outputTextArea" prefHeight="150" editable="false"/>
</VBox>
```

l4-view.fxml:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.demo1.l4.l4Controller" spacing="10"
alignment="TOP_LEFT">
    <HBox spacing="10">
        <Label text="Выберите метод:"/>
        <ComboBox fx:id="methodComboBox"/>
    </HBox>

    <VBox fx:id="inputFieldsBox" spacing="10">
        <!-- Поля ввода динамически добавляются сюда -->
    </VBox>

    <Button fx:id="executeButton" text="Выполнить метод"/>

    <Label text="Результат:"/>
    <TextArea fx:id="outputTextArea" prefHeight="150" editable="false"/>
</VBox>
```