МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по учебной практике

Тема: Визуализация алгоритма Косарайю-Шарира

Студент гр. 9381 _____ Судаков Е.В.

Студент гр. 9381 _____ Фоминенко А.Д.

Студент гр. 9381 _____ Авдеев И.В.

Руководитель _____ Жангиров Т.Р.

Санкт-Петербург

2021

ЗАДАНИЕ

НА УЧЕБНУЮ ПРАКТИКУ

Студент Судаков Е.В. группы 9381

Студент Фоминенко А.Д. группы 9381

Студент Авдеев И.В. группы 9381
Тема практики: Визуализация алгоритма Косарайю-шарира

Задание на практику:

Командная итеративная разработка визуализатора алгоритма(ов) на Java с графическим интерфейсом.

Алгоритм: Косарайю-Шарира

Сроки прохождения практики: 01.06.2021 – 14.06.2021

Дата сдачи отчета: 12.06.2021

Дата защиты отчета: 12.06.2021

Студент	_____	Судаков Е.В.
Студент	_____	Фоминенко А.Д.
Студент	_____	Авдеев И.В.
Руководитель	_____	Жангиров Т.Р.

АННОТАЦИЯ

Целью данной учебной практики является итеративная разработка GUI приложения для визуализации работы алгоритма Косарайю-Шарира, предназначенного для поиска всех компонент сильной связности в ориентированном графе. Программа разрабатывается на языке Java с использованием библиотеки JavaFX для реализации GUI. Разработка ведется командой из трех человек, за которыми закреплены определенные роли.

SUMMARY

The purpose of this training practice is the iterative development of a GUI application for visualizing the work of the Kosaraju & Sharira algorithm, designed to find all strong connectivity components of the oriented graph. The program is developed in Java using the JavaFX library for GUI implementation. The development is carried out by a team of three people, who are assigned certain roles.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью данной учебной практики является итеративная разработка GUI приложения для визуализации работы алгоритма Косарайю-Шарира, предназначенного для поиска всех компонент сильной связности в ориентированном графе.

Программа должна предоставить пользователю удобный интерфейс, дающий возможность изучить работу алгоритма на каждом шагу: визуализация графа, на данном шаге, управление работой алгоритма (а именно переход к следующему или предыдущему шагу, или прогон алгоритма до завершения). Пользователю должна быть предоставлена возможность задать граф через GUI или загрузить из файла.

Разработка ведется командой из трех человек, за каждым из которых закреплены определенные роли: разработка GUI приложения, визуализация алгоритма, реализация алгоритма Косарайю-Шарира, сборка и тестирование.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Исходные требования к программе

Программа должна предоставлять интерфейс для пошаговой визуализации алгоритма Косарайю-Шарира.

1.1.1 Требования к вводу исходных данных

Пользователь должен иметь возможность задавать граф через:

- Интерактивное добавление, изменение или удаление взвешенных ребер при помощи диалогового ввода в GUI;
- Загрузку файла с графом.

1.1.2 Требования к визуализации

Пользователю должно быть доступно графическое представление графа и статуса выполнения алгоритма на каждом шаге.

Пользователь должен иметь возможность применить алгоритм Косарайю-Шарира, а также включить или выключить вывод промежуточных данных алгоритма. Должна быть возможность выполнять алгоритм пошагово (то есть сделать одну итерацию алгоритма вперед или вернуться на одну итерацию назад) или выполнять алгоритм сразу до завершения его работы.

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1. План разработки

- 1.06: Распределение ролей в бригаде;
- 5.06: Создание репозитория для проекта и настройка системы автоматической сборки, подготовка отчета, разработка архитектуры и выполнение других заданий, необходимых для сдачи первой итерации;
- 8.06: Реализация алгоритма Косарайю-Шарира, реализация частичного функционала GUI и создание архитектуры для будущего тестирования;
- 11.06: Реализация взаимодействия с алгоритмом, полностью рабочий GUI и CLI;
- 13.06: Реализация дополнительного функционала.

2.2. Распределение ролей в бригаде

Судаков Е.В.. – сборка, разработка архитектуры, разработка GUI программы и визуализация алгоритма;

Фоминенко А.Д. – реализация алгоритма;

Авдеев И.В. – Тестирование приложения и помощь при подготовке документации и архитектуры.

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

3.1. Архитектура программы

В основе архитектуры приложения взята модель MVC, обеспечивающая высокий уровень абстрагирования, а следовательно надежности, удобочитаемости и простоты кода и разработки. Ниже представлены некоторые uml-диаграммы, характеризующие логику и архитектуру построения приложения.

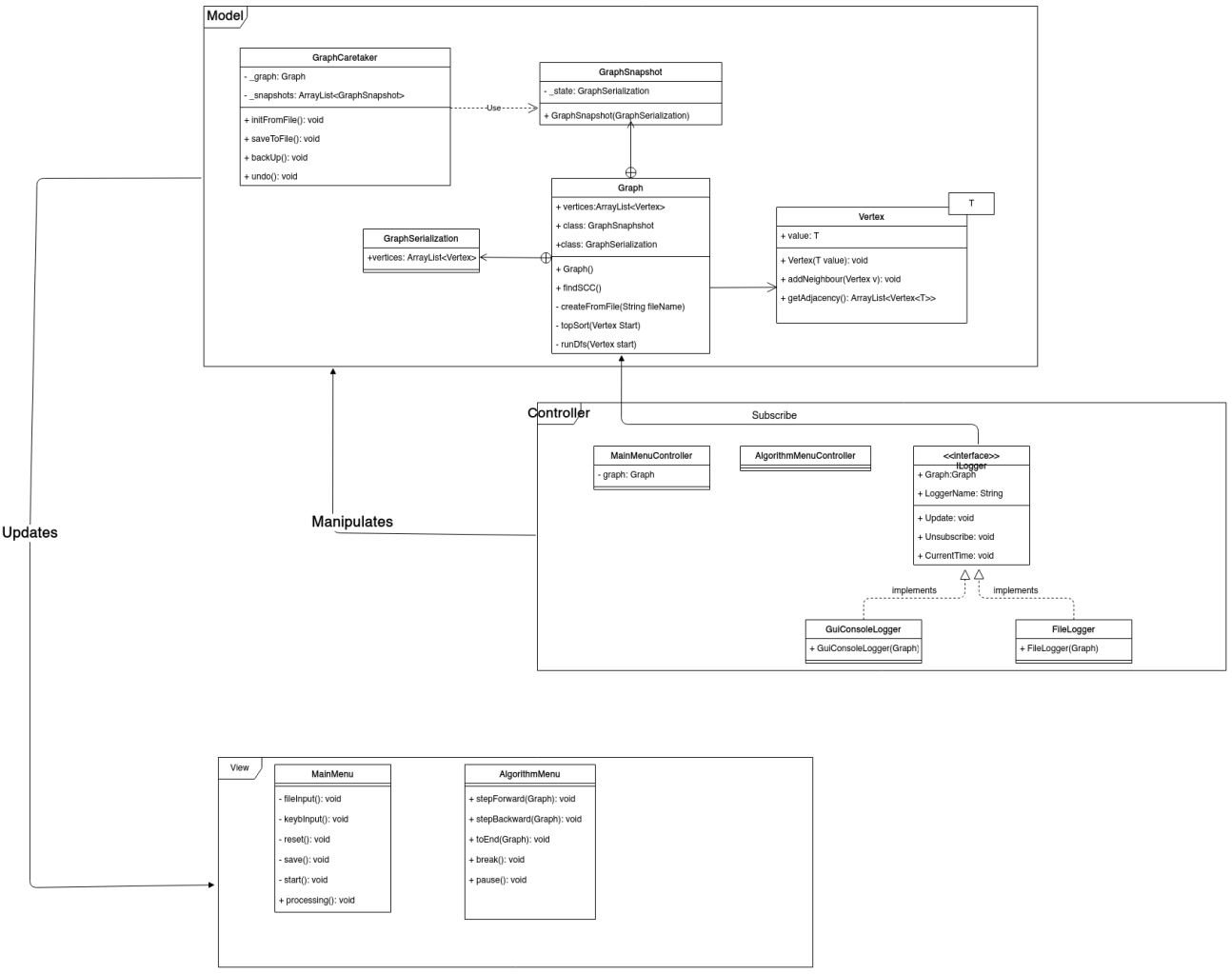


Рис 1. Uml - диаграмма классов

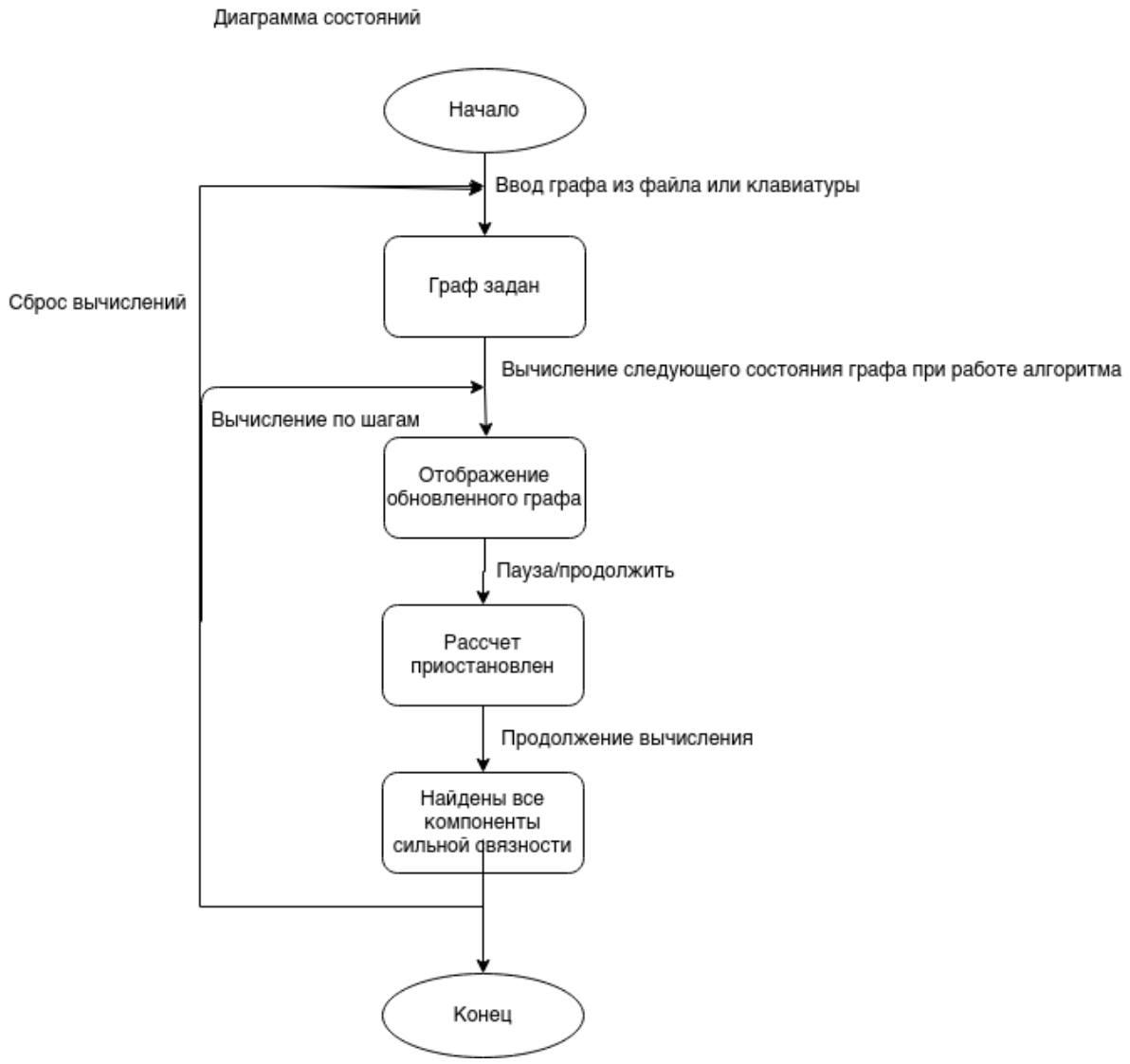


Рис 2. Uml - диаграмма состояний

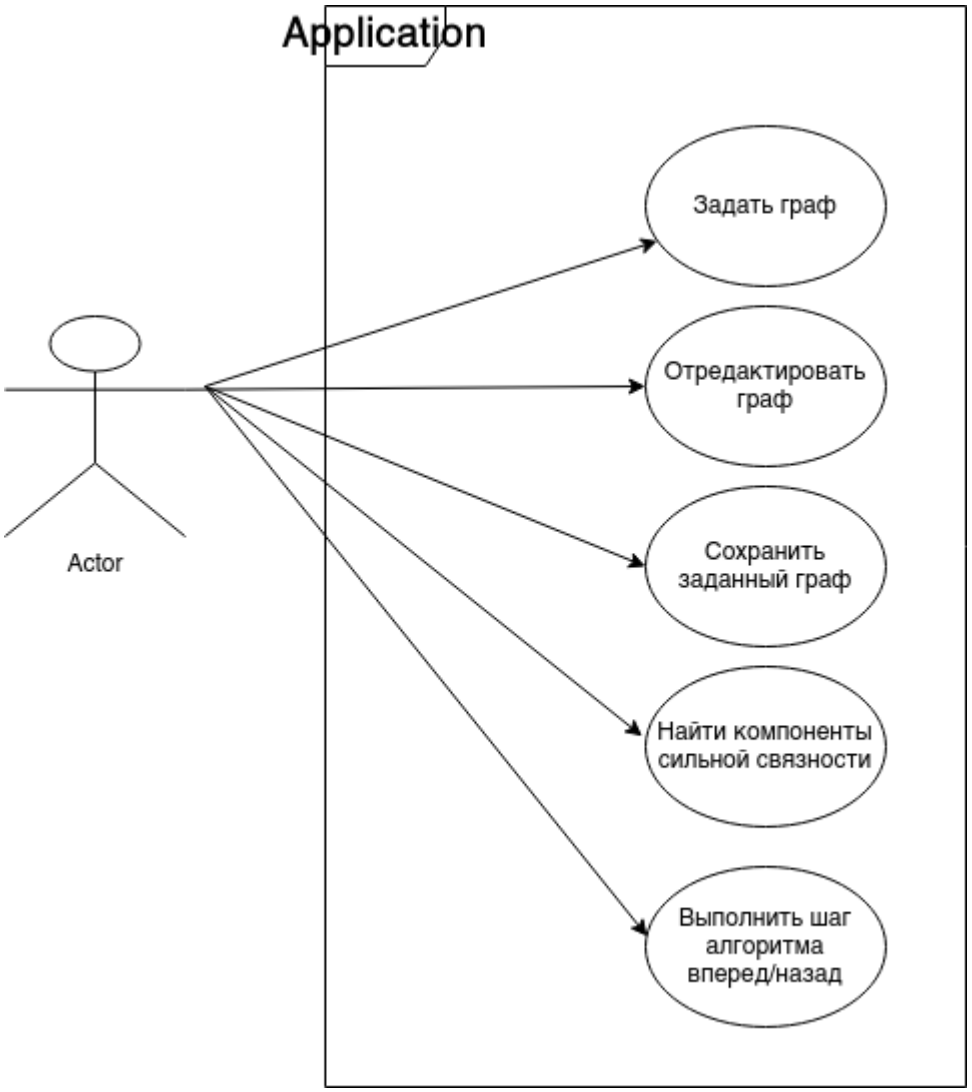


Рис 3. Uml - диаграмма возможностей

Диаграмма последовательности

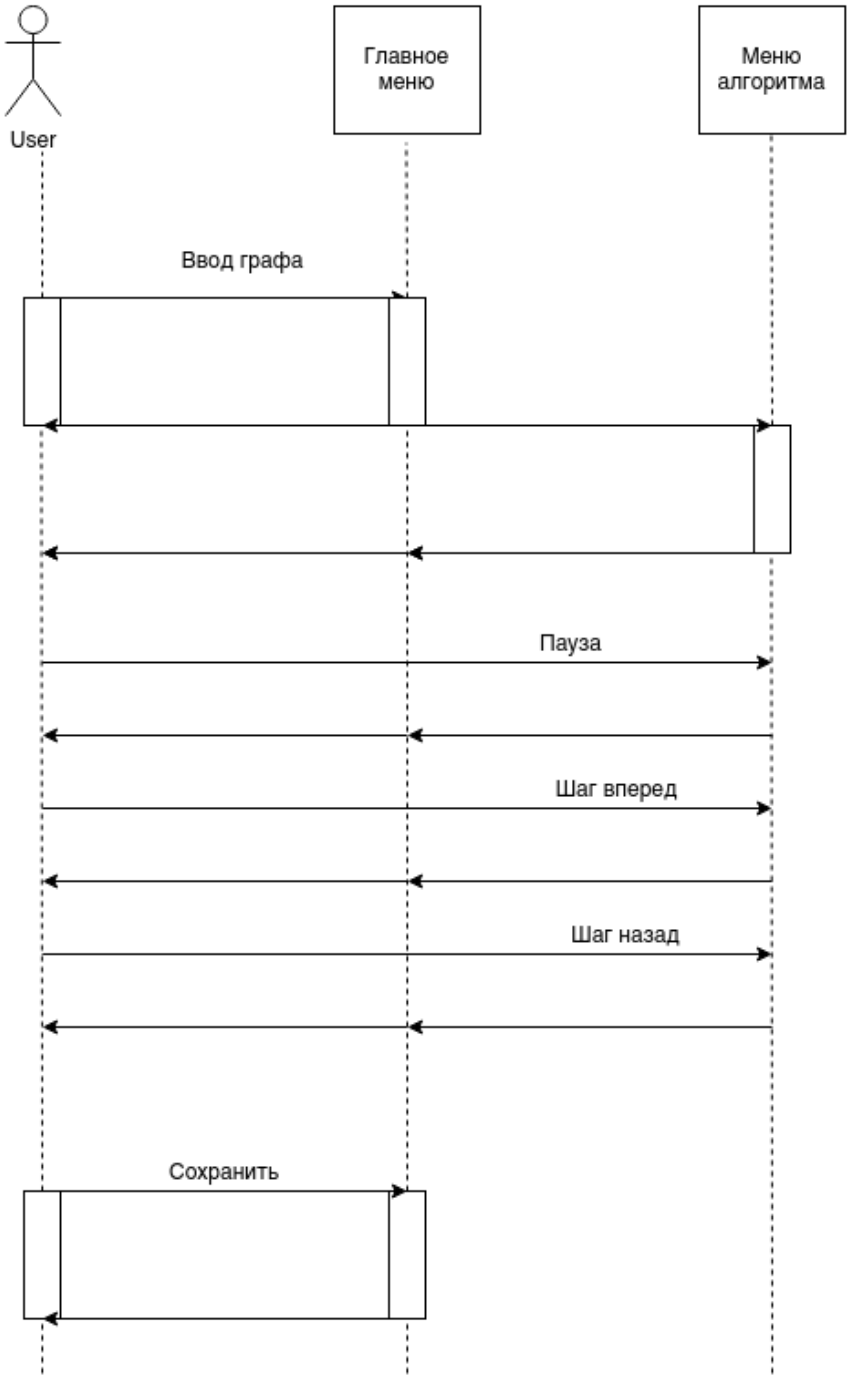


Рис 4. Uml - диаграмма последовательности

3.2. Описание алгоритма

3.3. Структуры данных

3.4. Графический интерфейс программы

3.4.1 Описание графического интерфейса программы

На вход программе подается ориентированный граф, который задается в текстовом виде через диалоговое окно или из файла. После завершения алгоритма полученные компоненты сильной связности отображаются на главном окне, также присутствует возможность записи полученного результата в текстовый файл. В ходе выполнения алгоритма пользователь может видеть предыдущее и текущее состояние программы.

Главное окно программы (рисунок 1) представляет собой окно, содержащее 2 кнопки для ввода графа: ввод из файла (открывается диалоговое окно), ввод с клавиатуры (открывается окно с полем для ввода текста). При неправильном формате ввода программа показывает сообщение с ошибкой, пользователь может ввести граф еще раз. Также окно содержит кнопку сохранения графа в файл и кнопку запуска алгоритма, по нажатию на которую открывается окно, отображающее состояние программы, главное окно скрывается. Также окно поле, на котором визуализируется введенный граф.

Рис 5. Главное меню программы

Окно, отображающее состояние программы (рисунок 2), содержит два поля, в которых отображается состояние алгоритма до и после выполнения текущего шага. В нижней части поля вывода предыдущего состояния расположено поле для вывода комментариев лога работы алгоритма. В правой части окна содержатся кнопки “Шаг вперед” и “Шаг назад”, по нажатию на которые программа переходит к следующему шагу алгоритма или возвращается на шаг назад. Также есть кнопка “Прервать алгоритм”, по нажатию на которую закрывается окно с состоянием программы и отображается главное окно, при этом поле с результатом работы остается пустым. В силу специфики алгоритма добавлен лог топологической сортировки вершин по времени выхода.

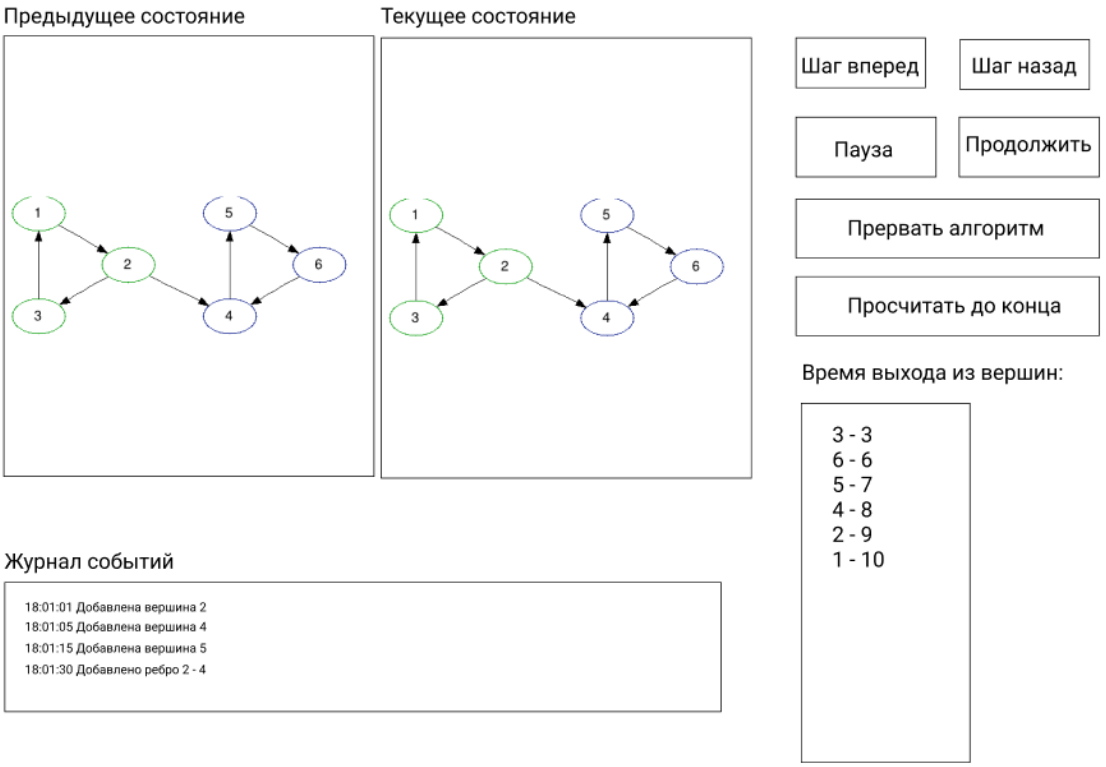


Рис 6. Окно исполнения алгоритма

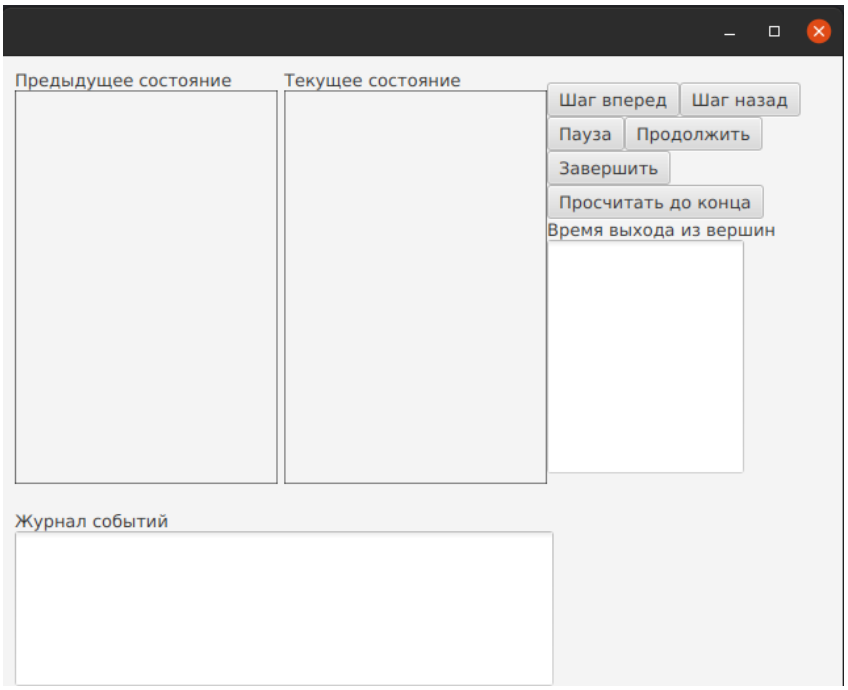


Рис 7. Реализованное окно исполнения алгоритма

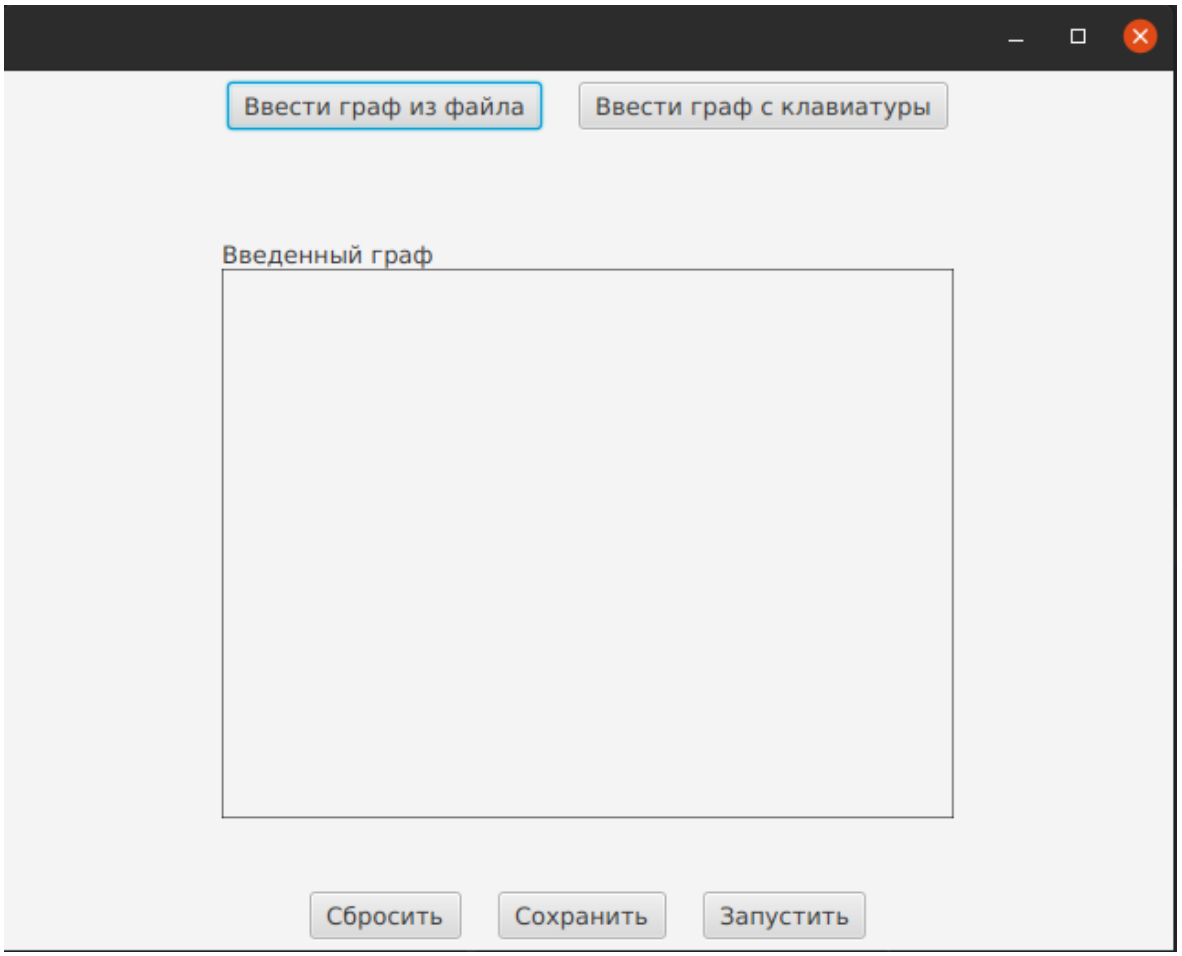


Рис 8. Реализованное окно главного меню

4. ТЕСТИРОВАНИЕ

4.1. План тестирования программы.

4.1.1. Объект тестирования

Объектом тестирования является программа для визуализации работы алгоритма Косарайю-Шарира.

4.1.2. Тестируемый функционал.

Необходимо протестировать метод mainAlgo*()* класса Graph, так как в этом методе происходит полное выполнение алгоритма.

4.1.3. Подход к тестированию.

Тестирование будет проводиться на модульном уровне при помощи фреймворка автоматического тестирования JUnit.

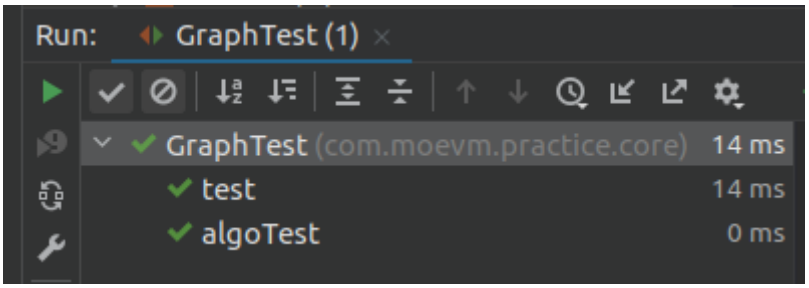
4.1.4. Критерии прекращения тестирования.

Тестирование считается успешно завершенным, если все тесты выполнены без ошибок. В противном случае программа возвращается на доработку.

4.2. Тестовые случаи.

4.3. Результаты тестирования.

4.3.1. Скриншот запуска unit-теста:



Таким образом, тестирование можно считать пройденным, так как фактические и ожидаемые результаты всех запланированных тестов совпали.

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Java Platform, Standard Edition 8 API Specification // Oracle Help Center. URL: https://docs.oracle.com/javase/8/docs/api/overview-summary.html (дата обращения: 07.07.2021).

2. Java. Базовый курс // Stepik. URL: https://stepik.org/course/187/info (дата обращения: 05.07.2021).

3. JavaFX Reference Documentation // JavaFX. URL: https://openjfx.io/ (дата обращения: 07.07.2021).

4. Учебник по JavaFX (Русский) // code.makery. URL: https://code.makery.ch/ru/library/javafx-tutorial/ (дата обращения: 07.07.2021).

5. Руководства JavaFX // betacode. URL: https://betacode.net/11009/javafx (дата обращения: 07.07.2021).

6. Поиск компонент сильной связности, построение конденсации графа // emaxx. URL: https://e-maxx.ru/algo/strong_connected_components (дата обращения: 08.07.2021).

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ