Задача

Реализовать на языке программирования Java программу, представляющую собой визуализатор алгоритма нахождения компонент сильной связности орграфа.

Описание алгоритма

Используется алгоритм Kosaraju (Косарайю).

Алгоритм:

1) обходом в глубину рассчитывается порядок выхода вершин исходного графа.

2) исходный граф транспонируется.

3) выполняется обход в глубину в транспонированном графе в обратном порядке выхода вершин.

4) в течение обхода в глубину формируются компоненты сильной связности.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Исходные требования к программе

1.1.1 Требования к визуализации

Библиотека: JavaFX

Элементы интерфейса: Холст для графа, панель управления, лог с текстовым выводом шагов алгоритма.

Граф: Граф строго ориентирован, веса не используются, кратные ребра не создаются, максимальное количество вершин 15.

Хранение графа в файле JSON: в файле формата JSON граф будет храниться как список вершин и список рёбер; каждая вершина будет характеризоваться такими полям, как координата на холсте (по x и по y) и номер (id); каждое ребро будет характеризоваться номером вершины, из которой оно исходит (source), номером вершины, в которую оно входит (target), и цветом (color).

Элементы управления холстом:

Кнопки:

- «Загрузить граф» — загружает граф из файла формата JSON.
- «Сохранить граф» — скачивает граф в файл формата JSON.
- «Добавить вершину» — добавляет вершину на холст нажатием левой кнопки мыши; вершины автоматически нумеруются при создании.
- «Добавить ребро» — добавляет ориентированное ребро на холст нажатием левой кнопки мыши — сначала по начальной вершины, потом по конечной.

- «Очистить граф» — удаляет полностью граф с холста.
- «Удалить элемент» — удаляет с холста элемент, выбранный нажатием левой кнопки мыши.

Элементы управления алгоритмом:

Кнопки:

- «СТАРТ / ПАУЗА» — если алгоритм еще не запущен, запускает визуализацию работы алгоритма и вывода поясняющего текста в отдельной области, в ином случае приостанавливает визуализацию алгоритма, повторное нажатие возобновляет работу.
- «Получить результат» — сразу показывает результат полной работы алгоритма (без пошаговой визуализации алгоритма).
- «Выполнить по шагам» — алгоритм выполняется шаг за шагом и для выполнения следующего шага ожидается нажатие пользователем этой же кнопки, которая после начала выполнения алгоритма изменит свое название на «Следующий шаг».

Визуализация алгоритма:

Древесные ребра отмечаются зеленым цветом, обратные — голубым, направленные вперёд — синим и поперечные — фиолетовым. Выход из вершины помечается номером в стеке, во время работы алгоритма на вершинах выводятся номера в порядке обхода.

В конце работы вершины компоненты сильной связности будут выделены соответствующим номером. Примерный вид интерфейса представлен на рис.1:

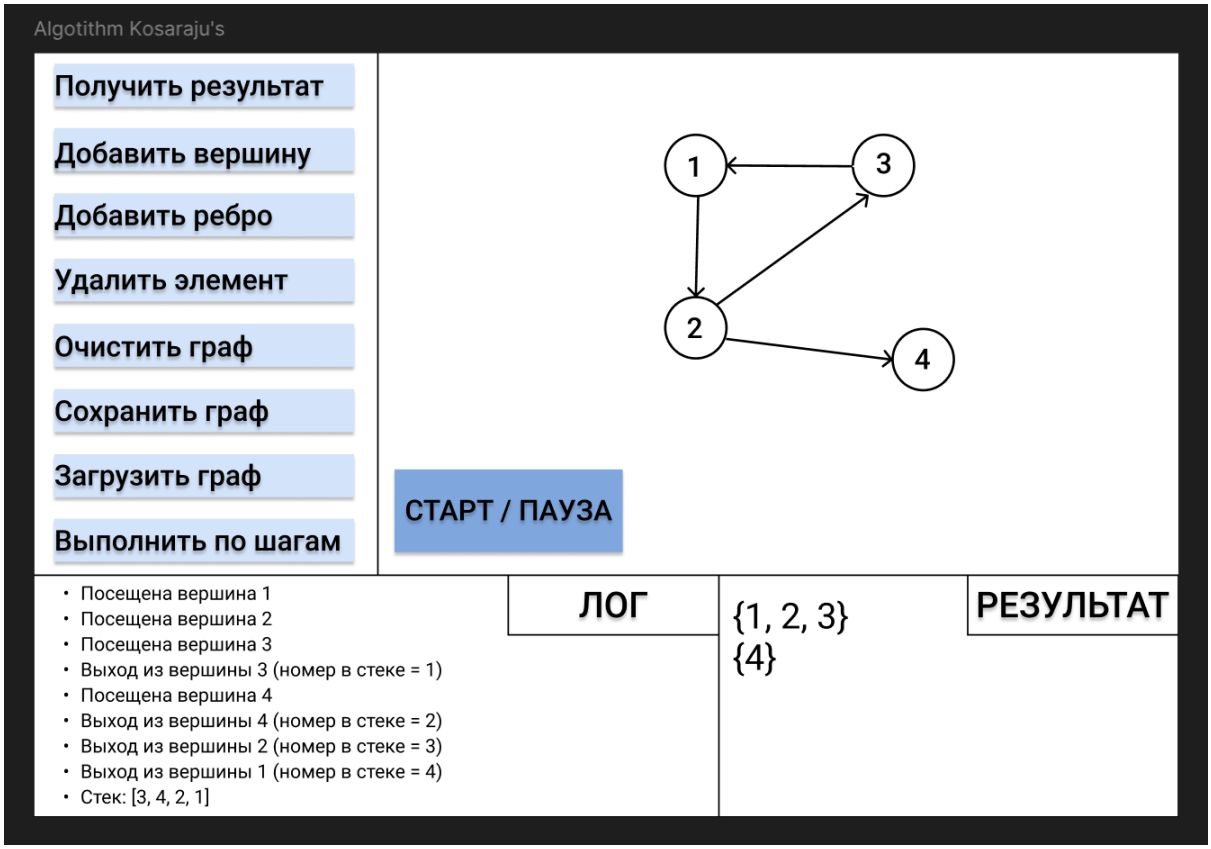

Рисунок 1: Примерный дизайн интрфейса

1.1.2 План тестирования

Обработка ошибок:

- Работа с файлом: некорректный файл, файл не формата JSON.
- Работа алгоритма: пустой граф.
- Работа интерфейса: добавление вершин больше допустимого значения, создание нескольких одинаковых ребер из одной вершины в другую

Проверка корректности работы интерфейса: Нажатие кнопок, расширение экрана, работа холста, работа текстового поля.

Проверка работы алгоритма на различных графах: граф из одной вершины, граф с несколькими компонентами сильной связности, несвязный граф, полный граф, граф с петлями.

1.2 Уточнение требований после сдачи прототипа

Окрасить кнопки в разные цвета в зависимости от их принадлежности к тому или иному смысловому блоку.

Первый смысловой блок (выполнение алгоритма):

- «Получить результат»
- «Выполнить по шагам»

Второй смысловой блок (создание графа на холсте):

- «Добавить вершину»
- «Добавить ребро»
- «Удалить элемент»
- «Очистить граф»

Третий смысловой блок (работа с файлом):

- «Загрузить граф»
- «Сохранить граф»

1.3 Уточнения требований после сдачи версии 1

1. В логе выделить крупные этапы: 1-ый и 2-ой обходы начинать с красной строки.
2. Размер вершин чуть-чуть уменьшить.
3. Выводить диалог сохранения файла (с подтверждением в случае перезаписи существующего) при сохранении графа.
4. Если во время обхода уже некуда идти, но при этом остались непройденные вершины, то "прыжок" отмечать в логе.

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1 План разработки

План разработки представлен в табл.1.

Таблица 1: План разработки

Дата	Этап проекта	Реализованные возможности	Выполнено
30.06.25	Согласование плана и спецификации	Написание требований и плана	
01.06.25	Сдача прототипа	Разработка прототипа интерфейса и написание алгоритма	
02.06.25	Сдача версии 1	Добавить функционал, обеспечивающий работоспособность кнопок, встроить алгоритм без поэтапной визуализации на исходном графе, то есть кнопки «Выполнить по шагам» и «СТАРТ / ПАУЗА» не будут работать, но будет работать кнопка «Перейти к результату», после нажатия которой на исходном графе будет визуализирован результат и в ячейке «ЛОГ» будут прописаны текстовые пояснения к тому, какие шаги выполнялись для получения результата	
04.06.25	Сдача версии 2	Добавить поэтапную визуализацию	

		алгоритма на исходном графе, обеспечивающую работоспособность кнопок «Выполнить по шагам» и «СТАРТ / ПАУЗА»	
06.06.25	Сдача версии 3		
06.06.25	Сдача отчёта		

2.2 Распределение ролей

Разработка алгоритма и его внедрение в интерфейс: Кривошеина Дарья

Разработка интерфейса: Матвеев Никита