

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

ОТЧЕТ
по учебной практике
ТЕМА: АЛГОРИТМ ПРИМА

Студент гр. 2383	_____	Иваницкий И.А.
Студент гр. 2384	_____	Цыганков Р.М.
Студентка гр. 2384	_____	Лавренова Ю.Д.
Руководитель	_____	Шестопалов Р.П.

Санкт-Петербург
2024

ЗАДАНИЕ НА УЧЕБНУЮ ПРАКТИКУ

Студент Иваницкий И.А. группы 2383

Студент Цыганков Р.М. группы 2384

Студентка Лавренова Ю.Д. группы 2384

Тема практики: Алгоритм Прима

Задание на практику:

Командная итеративная разработка визуализатора алгоритма на Java с графическим интерфейсом.

Алгоритм: Прима.

Сроки прохождения практики: 26.06.2024 – 09.07.2024

Дата сдачи отчета: 00.07.2024

Дата защиты отчета: 00.07.2024

Студент гр. 2383	_____	Иваницкий И.А.
Студент гр. 2384	_____	Цыганков Р.М.
Студентка гр. 2384	_____	Лавренова Ю.Д.
Руководитель	_____	Шестопалов Р.П.

АННОТАЦИЯ

Цель практики — создание визуализатора выбранного алгоритма на языке программирования Java с графическим интерфейсом.

Мини-проект создается в команде, результаты работы представляются преподавателю итеративно: согласование спецификации, сдача прототипа и версий, защита финального продукта исполняются в конкретные сроки. Цель команды — изучение нового языка программирования Java, разделение обязанностей и выполнение задач для разработки приложения итеративно, представляя промежуточные этапы и корректировка продукта.

СОДЕРЖАНИЕ

	Введение	5
1.	Требования к программе	6
1.1.	Исходные требования к программе*	6
1.2.	Уточнение требований после сдачи прототипа	8
1.3.	Уточнение требований после сдачи бета-версии	8
2.	План разработки и распределение ролей в бригаде	9
2.1.	План разработки	9
2.2.	Распределение ролей в бригаде	10
3.	Особенности реализации	11
3.1.	Структуры данных	11
3.2.	Основные методы	12
4.	Тестирование	14
4.1	Тестирование графического интерфейса	14
4.2	Тестирование алгоритма Прима	17
	Заключение	20
	Список использованных источников	21
	Приложение А. Исходный код – только в электронном виде	22

ВВЕДЕНИЕ

Цель работы:

Создание визуализатора алгоритма Прима на языке программирования Java с графическим интерфейсом.

Задачи:

1. Изучение языка программирования Java
2. Изучение алгоритма Прима и его реализацию
3. Сдача спецификации и плана разработки
4. Распределение ролей в команде
5. Создание визуализатора итеративно
6. Представление итогового продукта и отчета

Реализованный алгоритм в рамках выполнения задачи практики: алгоритм Прима — алгоритм поиска минимального остовного дерева неориентированного взвешенного связного графа.

Данный алгоритм применяется, если необходимо найти в графе ациклический подграф, который бы покрывал все вершины и включал в себя ребра с минимальным суммарным весом.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Исходные Требования к программе

1.1.1. Требования к вводу исходных данных

В приложении граф может быть введен несколькими способами:

- ввод из активной панели: пользователь сам добавляет вершины и строит ребра, указывая вес
- из файл, в котором храниться матрица весов ребер графа, который загружается в приложение с помощью необходимой кнопки на панели управления(матрица должна быть симметричной и антирефлексивной)

1.1.2. Требование к визуализации

Проект приложения

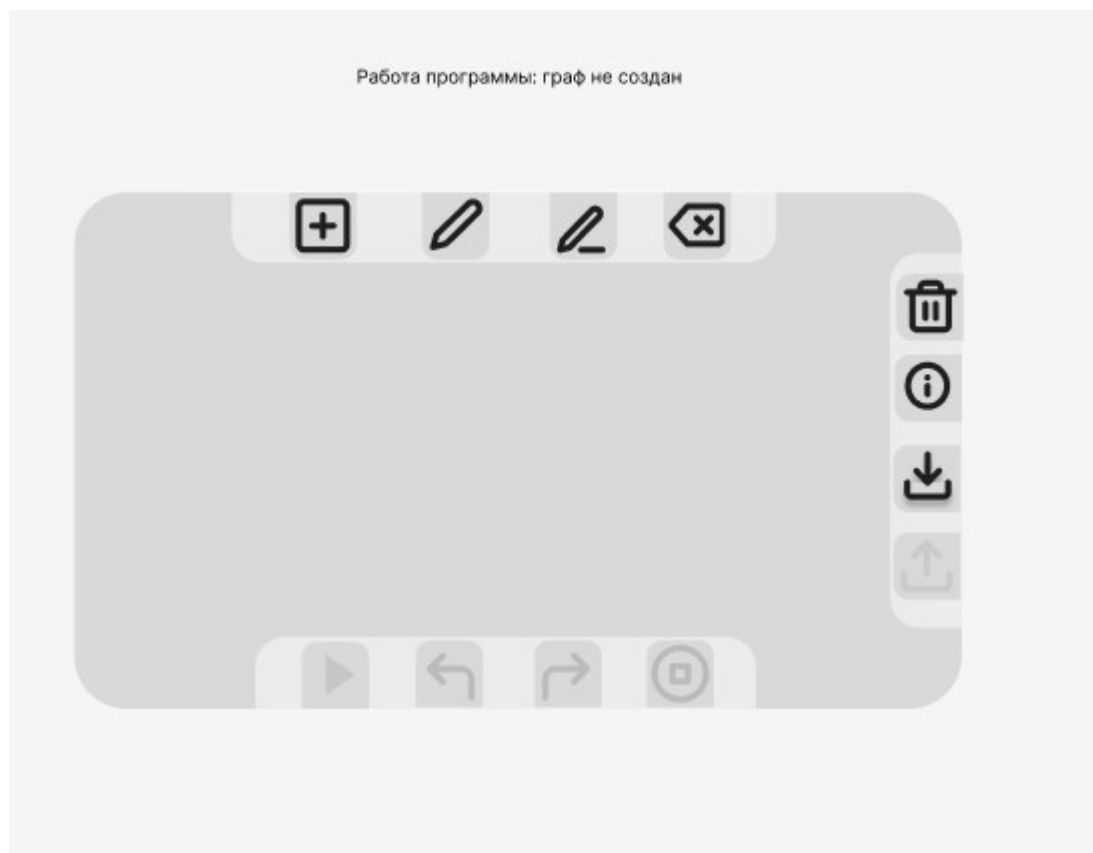


Рисунок 1 — граф не создан

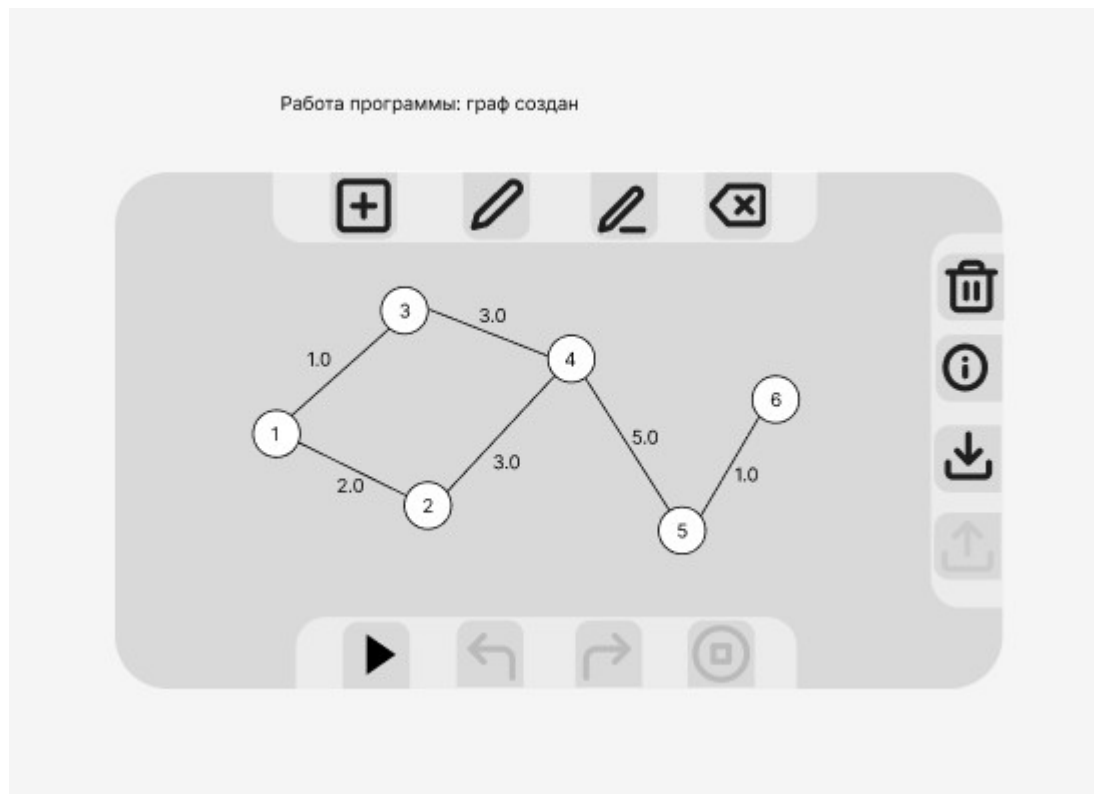


Рисунок 2 — граф создан

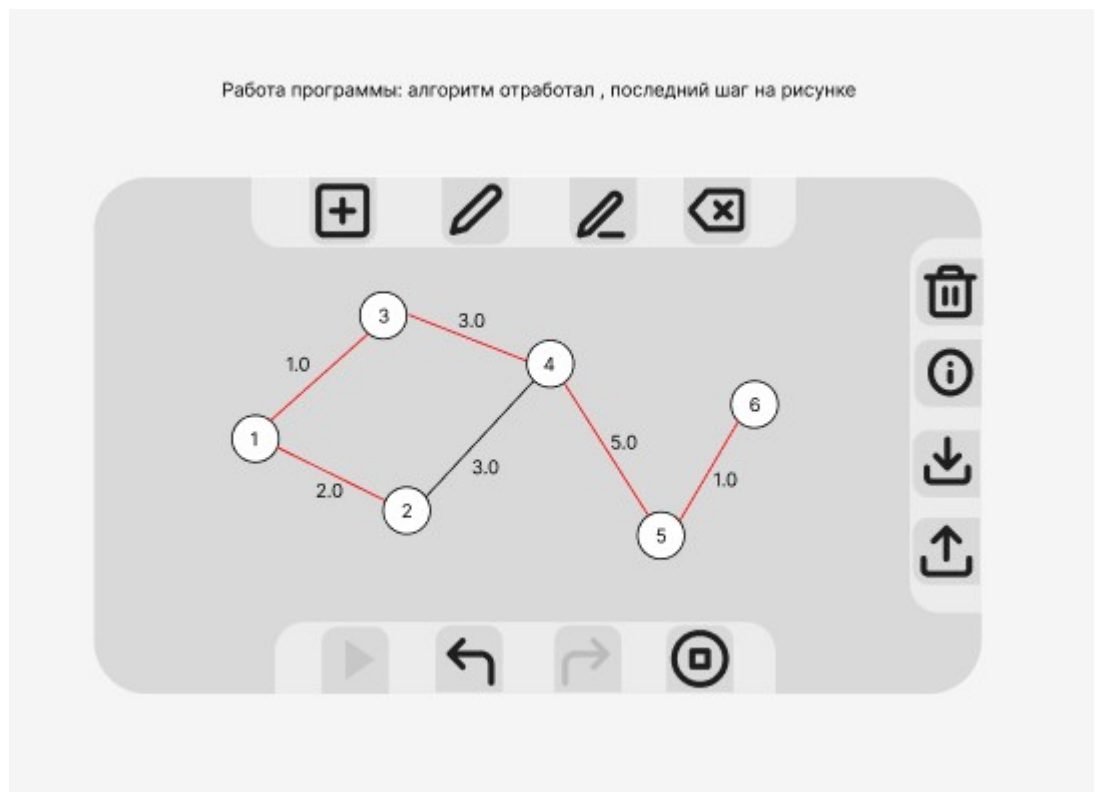


Рисунок 3 — алгоритм отработал

1.2. Уточнение требований после сдачи прототипа

Необходимо:

- добавление кнопки быстрого выполнения алгоритма
- добавление кнопки сохранения графа(матрицы весов)

1.3. Уточнение требований после сдачи бета-версии

Необходимо добавить:

- реализацию прохода алгоритма по шагам
- сохранение графа с активной панели в файл
- информацию об правилах использования программы

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1. План разработки

Дата	Этап проекта	Реализованные возможности	Выполнено
28.06.24	Согласование спецификации и плана разработки	Изображения главного окна приложения; UML-диаграмма без классов визуализации; План разработки	Полностью
01.07.24	Сдача прототипа	Демонстрация интерфейса без основных возможностей; Частично реализованные классы для графа и интерфейса	Полностью
05.07.24	Сдача бета-версии	Демонстрация ввода графа с активной панели и с файла; Выполнение алгоритма Прима; Возможность удаления графа полностью/части графа;	Полностью
08.07.24	Сдача финальной версии	Улучшена бета-версия; Возможность выполнения алгоритма Прима пошагово; Сохранение графа с активной панели в файл; Улучшен внешний вид	

		приложения	
08.07.24	Сдача отчёта		
	Защита отчёта		

2.2. Распределение ролей в бригаде

Иваницкий Илья гр 2383

- Реализация GUI
- Связь интерфейса и алгоритма

Цыганков Роман гр 2384

- Тестирование
- Реализация ввода графа в приложении

Лавренова Юлия гр 2384

- Реализация структур данных и алгоритма Прима
- Оформление отчетности

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

3.1. Структуры данных

Для реализации приложения структуры данных разделены на 2 части: frontend-компоненты и backend-компоненты.

Программа приложения написана с использованием принципов ООП, поэтому реализованы классы, описанные ниже.

Для backend части реализованы:

- Класс вершин Vertice графа, в котором хранится информация о вершине: ее имя, лейбл — включена ли вершина в алгоритм Прима;
- Класс ребер Edge графа, в котором храниться информация о ребре: вершины: от которой оно исходит, в какую приходит; вес ребра;
- Класс графа Graph — в нем хранятся массивы ребер и вершин, а также два конструктора для различной инициализации;
- Класс FileInput, позволяющий считывать с файла матрицу, которую пользователь загружает в приложение;
- Класс AlgPrima, в котором реализован основной алгоритм — алгоритм Прима для нахождения минимального остовного дерева;

Для frontend части реализованы:

- Класс CustomJButton — класс, переопределяющий класс JButton, находящийся в библиотеке swing, для более удобного управления интерфейсом кнопки, используемых в приложении;
- Класс CustomJPanel — класс, переопределяющий класс JPanel, находящийся в библиотеке swing, для удобства управления панелями управления в приложении и активной областью;
- Класс DeleteModeButton — класс, наследуемый от уже переопределенного класса CustomJButton, предназначенный для удаления компонентов графа: ребер и вершин;

- Класс `GraphEdgePanel` — класс, наследуемый от `JPanel`, определяющий ребро в интерфейсе приложения, аналогично с определением в `backend` части;
- Класс `GraphNodePanel` — класс, наследуемый от `JPanel`, определяющий вершины графа в интерфейсе приложения, аналогично с определением в `backend` части;
- Класс `mainWind` — класс, главного окна приложения, в котором определяются все кнопки и панели приложения, а также определение взаимодействия в нем.

3.2. Основные методы

1. ***public void algorithmPrima()*** - метод, реализующий алгоритм Прима.

Данный метод проходится по всем вершинам, пока есть не помеченные вершины — вершины не включенные в остовное дерево, далее анализирует ребра используя методы класса ребер и вершин, для получения вершин, их названия и значение метки, затем находит минимальное на данном этапе и преобразовывают вершины для дальнейшего прохода алгоритма. Все ребра записываются в `ArrayList<Edge> result_edges`, который является полем класса `AlgPrima`.

2. ***public void enableComponent()*** - метод, реализующий поэлементное удаление. Метод принимает на вход панель, на которой будет происходить удаление. Метод просматривает какой тип у компоненты, которую он удаляет, и если эта компонента — ребро, происходит обычное удаление компоненты, однако, если компонента — вершина, то алгоритм проходится по всем ребрам, которые выходят из вершины и также удаляет их.

3. ***addActionListener***

Прослушиватели событий — позволяют пользователю взаимодействовать с программой. Одни из самых интересных:

а) Кнопка запуска алгоритма прима:

Кнопка отключает пользователю возможность редактировать граф , что можно интерпретировать как режим работы программы.

Граф заполняется с панели, используя классы интерфейса для представления ребер и вершин, конвертируя их в ребра backend части. Далее создается объект AlgPrima, выполняется алгоритм Прима и формируется итоговый массив ребер, включенных в остовное дерево.

б) Кнопка шагов

Заводится массив шагов. При нажатии кнопки в массив записывается новый шаг и красится соответствующая часть ответа, либо же при нажатие шага назад соответствующая часть ответа красится обратно а шаг откатывается назад.

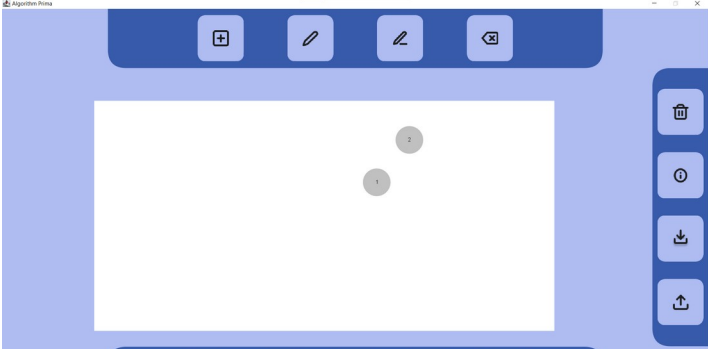
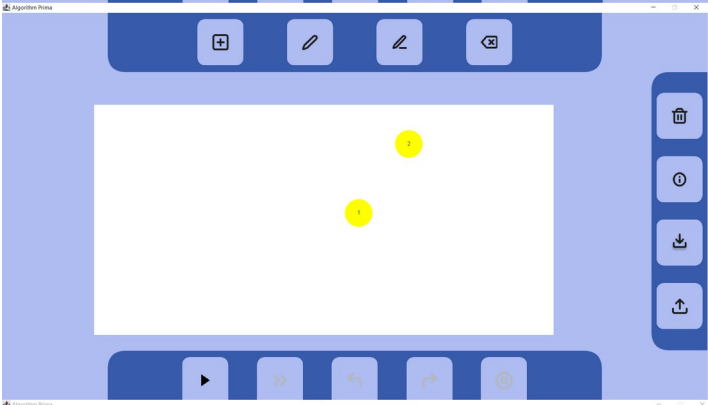
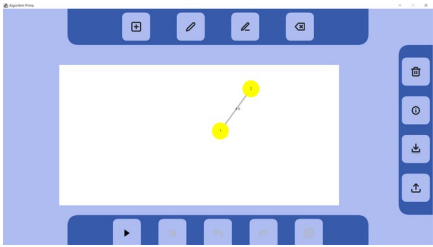
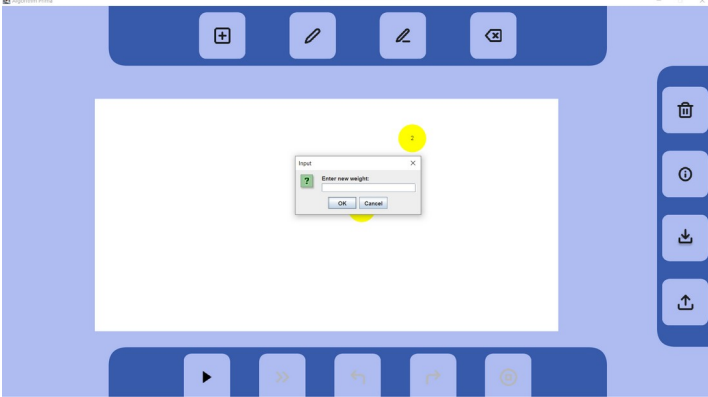
в) Кнопка загрузки графа

С помощью класса FileInput программа конвертирует матрицу смежности графа в подходящую структуру. Затем производится создание графа в виде правильного n -угольника, где n — количество вершин. Решение просто рисовать граф на панели сопровождается тем, что производится избавление от большого пласта кода, при запуске алгоритма Прима, что упрощает читабельность кода и позволяет другому разработчику смело вносить свои изменения, не боясь пропустить что-либо.

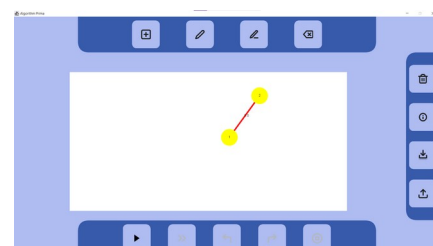
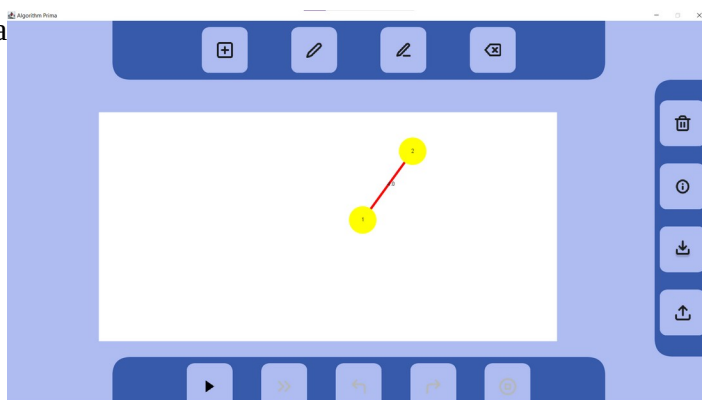
4. ТЕСТИРОВАНИЕ

4.1. Тестирование графического интерфейса

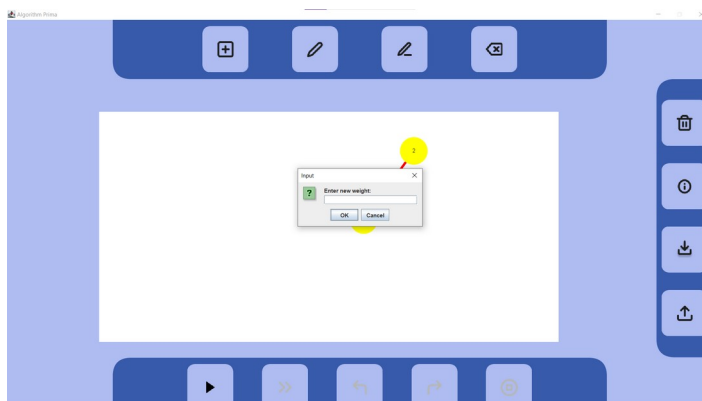
Таблица 1 — тестирование кнопок

Название кнопки	Данные	Итог
Добавление вершин		Выполнено
Добавление ребра	 	Выполнено
		

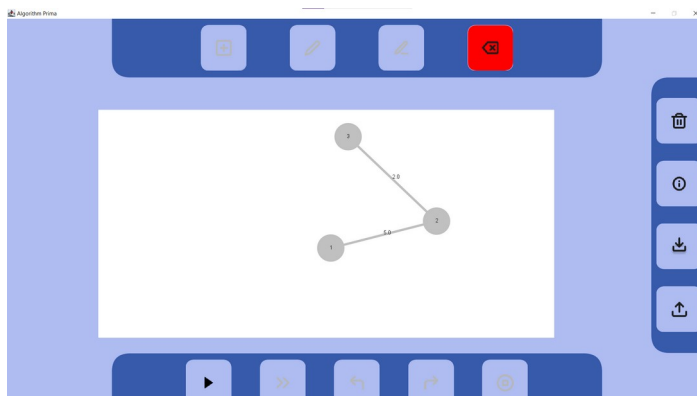
Изменение
веса
ребра



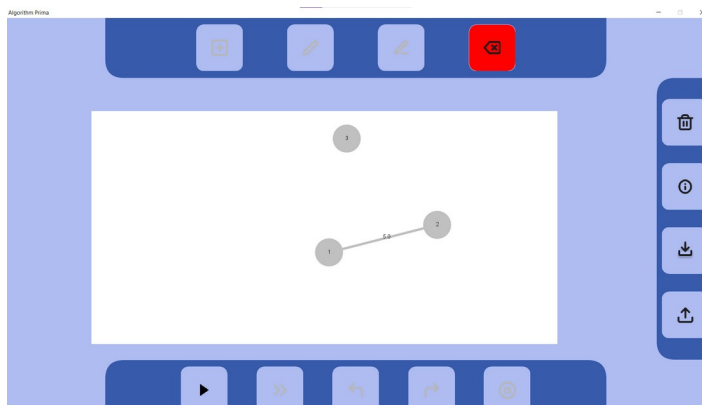
Выполнено



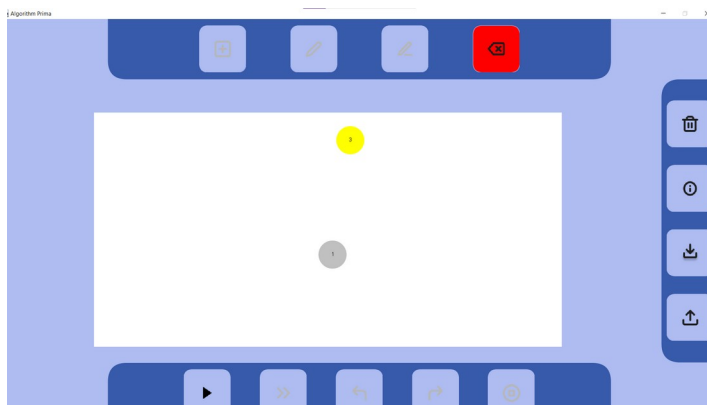
Удаление
компонент графа



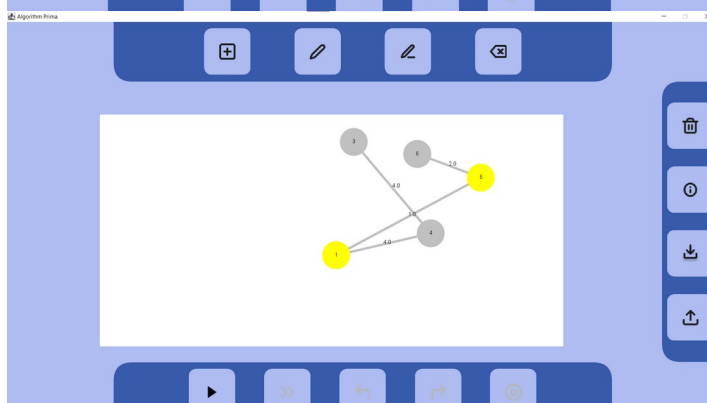
Удаление ребра корректно



Удаление вершины корректно

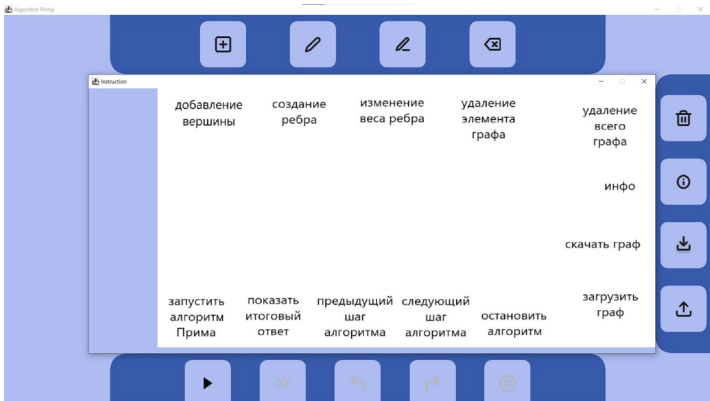


Очищение графа



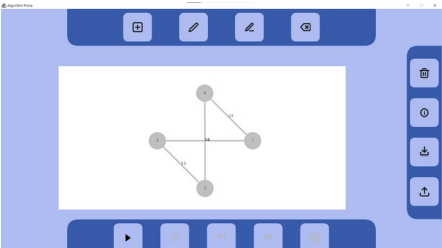
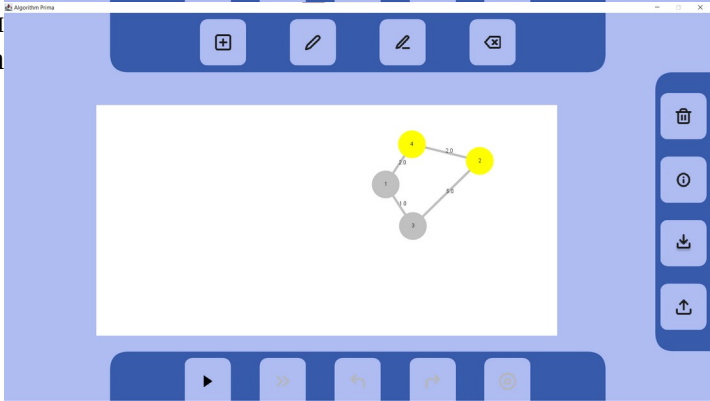
Корректно

Информация

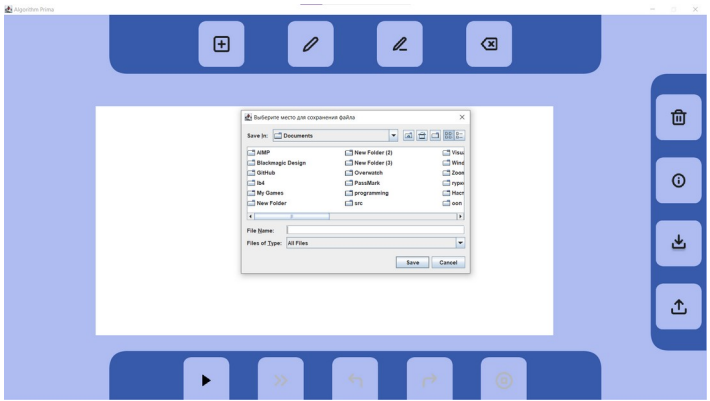


Корректно

Загрузка и
выгрузка
из файла
графа



Корректно



4.2. Тестирование алгоритма Прима

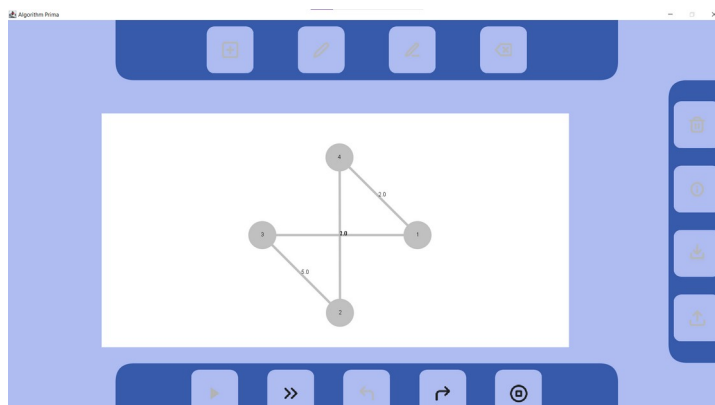
Таблица 2 — тестирование алгоритма Прима

Название
кнопки

Данные

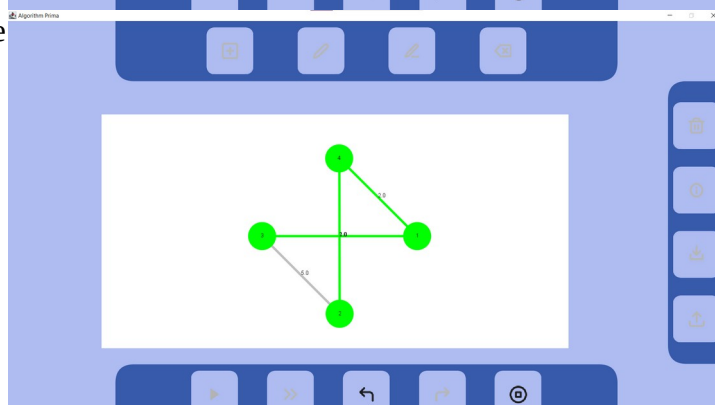
Итог

Начало решения



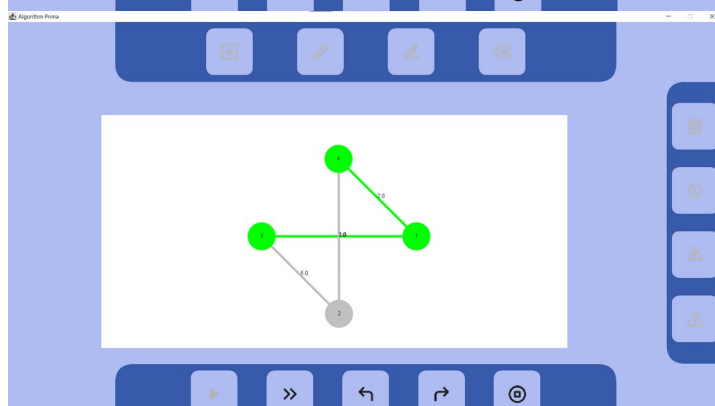
Корректно

Быстрое решение

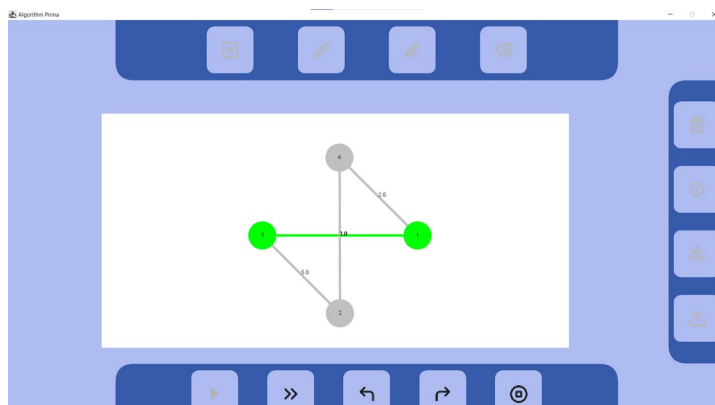


Корректно выполнен алгоритм

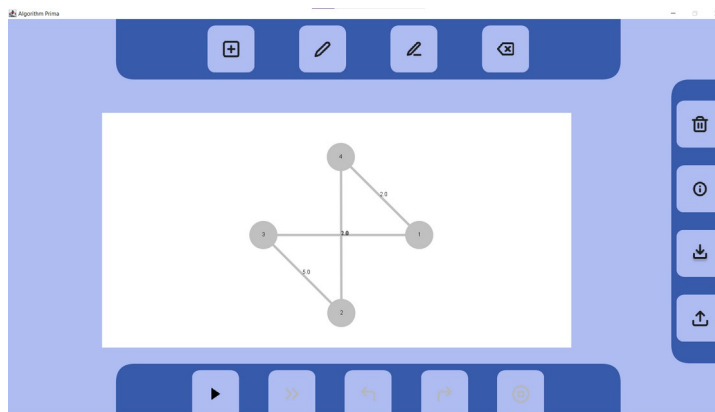
Кнопки шагов



Корректно

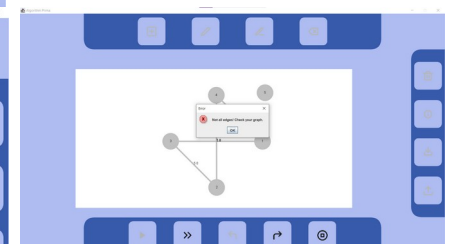
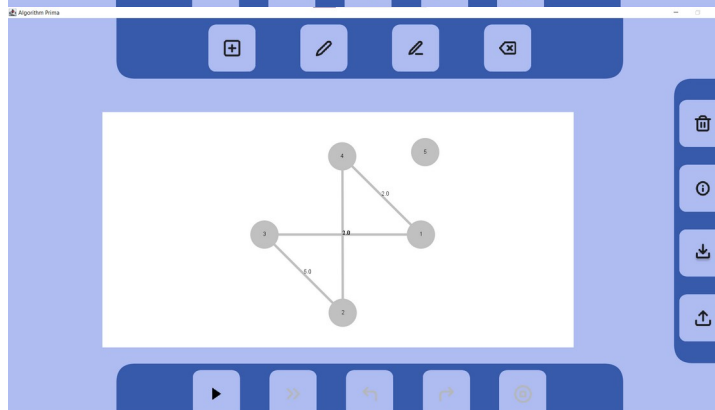


Остановка
решения



Корректно (возвращается в
режим редактирования)

Крайний случай



Корректно — появляется
уведомление об ошибке

ЗАКЛЮЧЕНИЕ

В ходе проделанной работы, был изучен новый язык программирования и написано приложение-визуализатор алгоритма Прима. Все поставленные задачи были выполнены в срок. Программа дает возможность пользователю ввести граф с активного поля и с файла и увидеть пошаговое построение минимального остовного дерева.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Курс на степик: <https://stepik.org/course/187/syllabus>
2. <https://progoschool.ru/java/java-swing/>
3. <https://java-online.ru/libs-swing.shtml>
4. <https://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%9F%D1%80%D0%B8%D0%BC%D0%B0>
5. <https://metanit.com/java/tutorial/6.3.php>
6. <https://ru.hexlet.io/qna/java/questions/kak-schitat-stroki-iz-fayla-java>

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

https://github.com/RieJxX/Algorithm_Prime_java — ссылка на репозиторий, в котором находится проект