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

ОТЧЕТ

по учебной практике

Тема: Кратчайшие пути в графе. Алгоритм Дейкстры

Студент гр. 0304 _____ Никитин Д.Э.

Студент гр. 0304 _____ Жиглов Д.С.

Студент гр. 0304 _____ Нагибин И.С.

Руководитель _____ Фирсов М.А.

Санкт-Петербург
2022

ЗАДАНИЕ

НА УЧЕБНУЮ ПРАКТИКУ

Студент Никитин Д.Э. группы 0304

Студент Жиглов Д.С. группы 0304

Студент Нагибин И.С. группы 0304

Тема практики: Кратчайшие пути в графе. Алгоритм Дейкстры

Задание на практику:

Командная итеративная разработка визуализатора алгоритма на Kotlin с графическим интерфейсом.

Алгоритм: Дейкстра.

Сроки прохождения практики: 29.06.2022 – 12.07.2022

Дата сдачи отчета: 00.07.2022

Дата защиты отчета: 00.07.2022

Студент	_____	Никитин Д.Э.
Студент	_____	Жиглов Д.С.
Студент	_____	Нагибин И.С.
Руководитель	_____	Фирсов М.А.

АННОТАЦИЯ

Необходимо разработать визуализатор алгоритма на графе, который выполняется пошагово с пояснениями. В качестве алгоритма выбран алгоритм Дейкстры, поскольку он имеет прикладное значение в протоколах маршрутизации OSPF и IS-IS.

Данная практическая работа нацелена на формирование у студентов: умения работать в команде, самодисциплины и навыка оценки трудоёмкости задач.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью практического задания является разработка пошагового визуализатора алгоритма на графе. В качестве алгоритма выбран алгоритм Дейкстры.

Алгоритм Дейкстры применяется для поиска кратчайших путей от исходной вершины графа до любой. Применяется данный алгоритм в протоколах маршрутизации OSPF и IS-IS.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Исходные Требования к программе

Необходимо разработать пошаговый визуализатор алгоритма на графе, позволяющий выполнить следующие функции:

- ввод данных пользователем за счёт графического интерфейса / из файла
- визуализация введённых данных в виде графа.
- пошаговое применение алгоритма Дейкстры для нахождения кратчайшего пути от выбранной начальной вершины до каждой вершины графа с возможностью возврата к предыдущему шагу(машина состояний). Отображение весов путей над вершинами, вывод пояснений в отдельном окне на каждом шаге.

1.1.1. Требования к визуализации

Интерфейс должен быть интуитивно понятным для конечного пользователя. Для этого необходимо выполнить разделение на 3 области: с возможностью выбора действия(назовем её панель инструментов), область с отрисовкой графа(назовем холст), а также с выводом пояснений(информационная панель).

Панель инструментов должна состоять из 8 кнопок со следующими опциями:

- Взаимодействие с файлом(сохранение/загрузка)
- Добавить вершину
- Удалить вершину
- Добавить ребро
- Удалить ребро
- Запуск алгоритма
- Шаг вперёд алгоритма
- Шаг назад алгоритма

Панель инструментов имеет логическую часть и отрисовку. В логической части хранится текущий выбранный инструмент для формирования графа (из 5:

добавление/удаление вершины, добавление/удаление ребра, ничего). В отрисовке находятся: 8 описанных выше кнопок. При нажатии на кнопку, связанную с отрисовкой графа, логическое состояние инструмента меняется. Если нажата кнопка взаимодействия с файлом, то запускается панель с возможностью выбора действия: загрузка или сохранение. Если выбрана загрузка, то вызывается окно с выбором файла. Далее при удачном парсинге происходит отрисовка графа. В ином случае выводится сообщение об ошибке. При выборе сохранения, открывается окно с выбором пути сохранения файла.

Холст — это layout типа Box, который имеет фиксированный максимальный размер и используется для отрисовки вершин и ребер.

Граф состоит из 2-ух множеств: вершин и ребер. Вершины и ребра имеют разделение на логическую часть и отрисовку. Отрисовка представляет собой поверхность определенной формы с возможность нажатия на неё. При нажатии на поверхность и выбранном инструменте происходит определенное действие(например: при нажатии на две различные вершины с выбранным инструментом «добавить ребро» создается ребро в графе и происходит его отрисовка).

Информационная панель используется для вывода пояснений при работе алгоритма, а также подсказок при выборе действия в панели инструментов. Вывод данных осуществляется в виде текста.

1.1.2. Требования к вводу исходных данных

Исходные данные вводятся либо за счет графического интерфейса, либо за счет парсинга данных из файла.

При задании графа через файл, данные должны быть помещены в файл с расширением .txt и записаны в следующем виде:

<Вершина старта>

<Вершина X><пробел><Вершина Y><пробел><Вес ребра от X к Y>

<Вершина X_1><пробел><Вершина Y_1><пробел><Вес ребра от X_1 к Y_1>

…

<Вершина X_i><пробел><Вершина Y_i><пробел><Вес ребра от X_i к Y_i>

И так далее, пока не будут заданы все ребра графа. Вершины при этом задаются какой-то строкой, состоящей из различных символов и цифр. Для различных вершины строки не должны повторяться. Ожидается, что веса рёбер будут положительные, так как алгоритм Дейкстры работает некорректно с ребрами отрицательной длины.

В случае, если граф задается графически, вышеописанная панель инструментов с добавлением и удалением элементов графа позволит корректно создать граф.

1.1.3. Требования к структуре программы

Явное разделение отрисовки и бизнес-логики. Создание отдельных пакетных модулей под логику и отрисовку.

Для очевидного разделения отрисовки и бизнес логики в проекте были созданы классы, представляющие собой элементы графа и сам граф, причём для их отрисовки используются другие, соответствующие им классы UI.

Так, класс Graph хранит в себе все вершины(Vertex) и ребра(Edge) следующим образом: в ассоциативном массиве хранятся вершины и соответствующие им ребра, ведущие в смежные вершины. Таким образом можно отслеживать в какие вершины можно попасть из какой-либо вершины графа. Для удобного отображения графа используется отдельный словарь, хранящий в себе пары вида: ребро в Void. За счёт этого обеспечивается доступ к нужному ребру за O(1). Класс Graph реализует функционал добавления и удаление ребра/вершины, путем изменения вышеописанных структур данных.

Класс Edge представляет собой структуру, хранящую пару смежных вершин и величину веса между ними, выраженную целочисленной переменной. Также класс хранит в себе поле edgeUI - экземпляр класса, отрисовывающего ребро.

В классе Vertex также хранится поле vertexUI, отрисовывающее вершину.

Далее класс Tools, который является некоторым среднем уровнем между пользовательским интерфейсом и логикой приложения, вызывая соответствующие отрисовки и добавление элемента графа, при нажатии на кнопку инструмента. Все кнопки были описаны выше в пункте 1.1.1. Таким образом, методы класса проверяют корректность полученных координат на холсте, добавляя или убирая соответствующий выбранный элемент.

Обобщая всё вышеописанное, логика устроена так что после выбора инструмента и нажатии на холст произойдет вызовы методов добавления/удаления класса Tools, которые в свою очередь совершат два ключевых действия: сообщат UI и классу Graph об их изменении, вызывая соответствующие выбранным действиям пользователя методы. Таким образом реализовано добавления/удаление рёбер и вершин графа.

Отрисовка.

Отрисовка обеспечивается модулем jetbrains.compose. В Compose всё завязано на композиции функций. Для рисуемых функций создана отдельная аннотация @Composable.

Основные компоненты UI:

VertexUI — класс, реализующий отрисовку вершины. Вершина представляет собой холст(Canvas) круглой формы с фиксированным размером. Для отрисовки создан метод drawVertex. На весь размер холста отрисован круг. Холст — кликабельный. При нажатии на него и наличии в tools выбранного инструмента:

- ADD_EDGE, если не была выбрана вершина: она будет выбрана в качестве начальной в ребре, а если была выбрана — то если эта вершина не совпадает с предыдущей выбранной, то у tools вызывается метод создания ребра.
- REMOVE_VERTEX, происходит вызов метода removeVertex у tools.

Каждая вершина подписана на холст, где рисуется граф.

EdgeUI — класс, реализующий отрисовку ребра. Представляет собой холст, на котором рисуется линия. Для отрисовки создан метод drawEdge.

Холст — кликабельный. Если в tools выбрана опция REMOVE_EDGE, то при нажатии на ребро происходит его удаление за счет метода tools.removeEdge. Каждое ребро подписано на холст, где рисуется граф.

Canvas — холст. Представляет собой класс, в котором содержится layout типа Box. В него помещаются @composable функции(отрисовка вершин и ребер). Имеет фиксированный максимальный размер. При нажатии на внутреннюю область и выбранной опции в tools ADD_VERTEX вызывается метод tools.addVertex и рисуется вершина.

Footer — область, куда выводится вспомогательная информация. Вывод осуществляется в виде текста на экран на каждом шаге алгоритма.

Toolbar — отрисовка для панели инструментов. Описание тулбара в 1.1.1.

Функция main реализует singleWindowApplication — приложение с единственным окном. В нём создаётся граф, tools и рисуются: toolbar, canvas, footer, как показано ниже

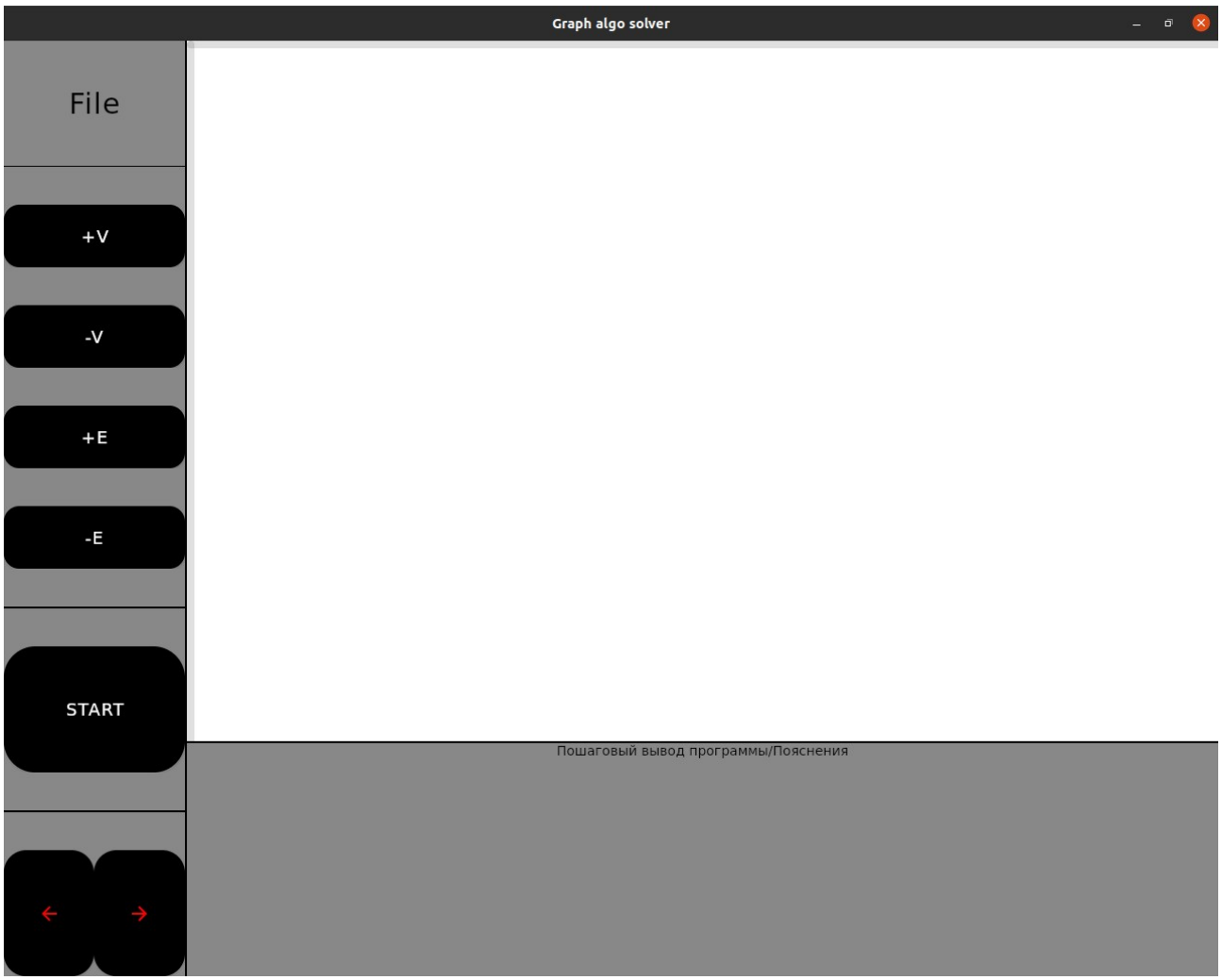

Рисунок 1 — Внешний вид приложения

1.1.4. Требования к коду

Отсутствие повтора кода, спагетти-кода, а также использование ООП парадигмы.

1.1.5. Требования к языку

Написание программы на ЯП Kotlin с графической отрисовкой Compose Multiplatform от JetBrains и логгированием SLF4J.

1.2. Уточнение требований после...

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1. План разработки

Дата	Содержание задания	Статус(выполнено или нет)
01.07.22	Попытка сдача спецификации программы.	-
02.07.22	Финальная сдача спецификации программы.	-
03.07.22	Разработка каркаса проекта. Добавление панели инструментов и холста. Появление возможности отрисовки компонент графа.	+
04.07.22	Согласование спецификации. Сдача прототипа.	
05.07.22	Добавление возможности ввода веса ребер. Реализация алгоритма Дейкстры.	
06.07.22	Сдача 1-ой версии.	
07.07.22	Отрисовка веса до вершины на текущем шаге. Добавление машины состояний.	
08.07.22	Сдача 2-ой версии	
09.07.22 — 10.07.22	Добавление возможности ввода данных с файла. Сохранение выходного состояния в файл. Тестирование, дебаггинг. Написание отчета.	
11.07.22	Сдача финальной версии.	

2.2. Распределение ролей в бригаде

Никитин Д.Э. - проектировщик, разработчик графического интерфейса.

Нагибин И.С. - разработчик логики, тестировщик.

Жиглов Д.С. - разработчик логики, тестировщик.

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

3.1. Структуры данных

3.2. Основные методы

4. ТЕСТИРОВАНИЕ

4.1. Первый подраздел третьего раздела

4.2. Второй подраздел третьего раздела

ЗАКЛЮЧЕНИЕ

Кратко подвести итоги, проанализировать соответствие поставленной цели и полученного результата.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Ниже представлены примеры библиографического описания, В КАЧЕСТВЕ НАЗВАНИЯ ИСТОЧНИКА в примерах приводится вариант, в котором применяется то или иное библиографическое описание.

1. Иванов И. И. Книга одного-трех авторов. М.: Издательство, 2010. 000 с.

2. Книга четырех авторов / И. И. Иванов, П. П. Петров, С. С. Сидоров, В. В. Васильев. СПб.: Издательство, 2010. 000 с.

3. Книга пяти и более авторов / И. И. Иванов, П. П. Петров, С. С. Сидоров и др.. СПб.: Издательство, 2010. 000 с.

4. Описание книги под редакцией / под ред. И.И. Иванова СПб., Издательство, 2010. 000 с.

5. Иванов И.И. Описание учебного пособия и текста лекций: учеб. пособие. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2010. 000 с.

6. Описание методических указаний / сост.: И.И. Иванов, П.П. Петров. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2010. 000 с.

7. Иванов И.И. Описание статьи с одним-тремя авторами из журнала // Название журнала. 2010, вып. (№) 00. С. 000–000.

8. Описание статьи с четырьмя и более авторами из журнала / И. И. Иванов, П. П. Петров, С. С. Сидоров и др. // Название журнала. 2010, вып. (№) 00. С. 000–000.

9. Иванов И.И. Описание тезисов доклада с одним-тремя авторами / Название конференции: тез. докл. III международной науч.-техн. конф., СПб, 00–00 янв. 2000 г. / СПбГЭТУ «ЛЭТИ», СПБ, 2010, С. 000–000.

10. Описание тезисов доклада с четырьмя и более авторами / И. И. Иванов, П. П. Петров, С. С. Сидоров и др. // Название конференции: тез. докл. III международной науч.-техн. конф., СПб, 00–00 янв. 2000 г. / СПбГЭТУ «ЛЭТИ», СПБ, 2010, С. 000–000.

11. Описание электронного ресурса // Наименование сайта. URL: http://east-front.narod.ru/memo/latchford.htm (дата обращения: 00.00.2010).

12. ГОСТ 0.0–00. Описание стандартов. М.: Изд-во стандартов, 2010.

13. Пат. RU 00000000. Описание патентных документов / И. И. Иванов, П. П. Петров, С. С. Сидоров. Опубл. 00.00.2010. Бюл. № 00.

14. Иванов И.И. Описание авторефератов диссертаций: автореф. дисс. канд. техн. наук / СПбГЭТУ «ЛЭТИ», СПБ, 2010.

15. Описание федерального закона: Федер. закон [принят Гос. Думой 00.00.2010] // Собрание законодательств РФ. 2010. № 00. Ст. 00. С. 000–000.

16. Описание федерального постановления: постановление Правительства Рос. Федерации от 00.00.2010 № 00000 // Опубликовавшее издание. 2010. № 0. С. 000–000.

17. Описание указа: указ Президента РФ от 00.00.2010 № 00 // Опубликовавшее издание. 2010. № 0. С. 000–000.

ПРИЛОЖЕНИЕ А

НАЗВАНИЕ ПРИЛОЖЕНИЯ

полный код программы должен быть в приложении, печатать его не надо