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

ОТЧЁТ
по учебной практике
Тема: Мосты графа

Студент гр. 9303 _____ Ахримов А.М.

Студент гр. 9303 _____ Эйсвальд М.И.

Студент гр. 9303 _____ Максимов Е.А.

Руководитель _____ Фирсов М.А.

Санкт-Петербург
2021

ЗАДАНИЕ НА УЧЕБНУЮ ПРАКТИКУ

Студент Ахримов А.М. группы 9303

Студент Эйсвальд М.И. группы 9303

Студент Максимов Е.А. группы 9303

Тема практики: Мосты графа

Задание на практику: Командная итеративная разработка визуализатора алгоритма на Java с графическим интерфейсом. Алгоритм: нахождение мостов графа модификацией поиска в глубину.

Сроки прохождения практики: 01.07.2021 – 14.07.2021

Дата сдачи отчёта: 12.07.2021

Дата защиты отчёта: 12.07.2021

Студент	_____	Ахримов А.М.
Студент	_____	Эйсвальд М.И.
Студент	_____	Максимов Е.А.
Руководитель	_____	Фирсов М.А.

АННОТАЦИЯ

Задачей учебной практики является получение опыта командной работы над проектом. Практика заключается в разработке приложения — визуализатора указанного в задании алгоритма — на языке Java. Разработка ведётся итеративно: студентами и их руководителем согласовывается спецификация, чётко определяющая предполагаемое поведение программы, после чего создаются несколько версий проекта, каждая из которых расширяет функционал предыдущей. В отчёте приведена информация о ходе выполнения практической работы.

SUMMARY

The aim of the practice is to gain group work experience. The practice work consists of Java application development; the application must visualize an algorithm specified in the given task. Students and their mentor discuss and form project specification, which distinctly defines expected program behavior. Development should be iterative: several consecutive project versions are created; each of these extends the functionality of the previous one. This report contains information on the working process.

Содержание

ВВЕДЕНИЕ

Целью работы является разработка приложения, визуализирующего конкретный алгоритм, на языке программирования Java. Для достижения цели необходимо решить следующие задачи:

1. Сформулировать и согласовать с руководителем спецификацию, описывающую поведение программы;
2. Распределить роли в разработке приложения между членами бригады и составить план разработки;
3. Изучить алгоритм, который требуется визуализировать;
4. Написать и отладить модули проекта;
5. Скомпоновать написанные программные модули в общий проект;
6. Отладить приложение;
7. Представить выполненную работу руководителю.

В данной работе визуализируется алгоритм Тарьяна поиска мостов в графе. Алгоритм Тарьяна имеет линейную временну́ю сложность относительно количества вершин и рёбер графа и основывается на поиске в глубину. Алгоритм Тарьяна, как и другие алгоритмы поиска мостов, используется для выявления уязвимых мест (точек отказа) распределённой системы или сети.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Исходные требования к программе.

1.1.1 Требования ко входным данным.

Граф, мосты которого требуется найти, может быть предоставлен программе тремя способами:

1. Задан пользователем при помощи графического интерфейса: нажатие кнопки `Draw Node` (см. рисунок 1) и последующий клик мыши по рабочей области создают в указанном месте вершину графа; нажатие кнопки `Draw Edge` и последующие клики по двум различным вершинам создают ребро между этими вершинами графа. Попытка создать уже существующее ребро повторно не имеет эффекта. Нажатие кнопки `Erase` и клик мыши по компоненте (ребру или вершине графа) удаляют эту компоненту. Вершина удаляется со всеми инцидентными ей рёбрами.

2. Задан пользователем в виде текстового файла, содержащего список рёбер графа. Каждая строка файла должна иметь вид `<ключ 1> <ключ 2>`, где `<ключ 1>` и `<ключ 2>` — различные символы-ключи вершин графа. Попытка описать уже существующее ребро повторно не имеет эффекта. При задании графа таким способом расположение узлов в рамках рабочей области определяется автоматически.

3. Загружен из файла сохранения. Файл сохранения — файл, содержащий заголовок известного программе формата, после которого следуют: число — количество строк, описывающих вершины графа; указанное количество строчек, каждая из которых содержит ключ вершины (см. предыдущий пункт) и два числа — координаты вершины на рабочей области окна программы; список рёбер в том же формате, что и в предыдущем пункте.

В случае некорректных входных данных — например, нарушении формата файла или попытке описать петлю (ребро, соединяющее вершину графа саму с собой), — программа выдаёт информативное сообщение об ошибке, а не завершает работу.

1.1.2 Требования к визуализации процесса выполнения алгоритма.

Визуализируемый алгоритм — алгоритм Тарьяна поиска мостов в графе. Режим визуализации включается нажатием кнопки `Start` (см. рисунок 1). При этом кнопки добавления вершины и ребра в граф становятся неактивными, что делает невозможным изменение графа во время прогонки алгоритма. Кнопка `Stop` прекращает прогонку алгоритма и возвращает приложение в обычный режим работы. Визуализация алгоритма производится пошагово. Переход к визуализации следующего шага алгоритма осуществляется нажатием кнопки `Next`. Алгоритм содержит несколько этапов, каждый из которых будет визуализирован:

1. Первый этап — поиск в глубину и нумерация вершин. Вершина, из которой на данном шаге производится поиск, покрашена в яркий цвет; уже посещённые вершины покрашены в цвет, отличающийся от цвета ещё не посещённых вершин. Номера посещённых вершин отображаются рядом с вершинами. Пройденные рёбра отображаются как ориентированные. Количество потомков вершины отображается рядом с вершиной, когда у неё не остаётся непосещённых соседей. Текстовое описание текущего шага (какое ребро выбрано для продолжения поиска, полученные на этом шаге численные значения) печатается в поле текстового вывода.

2. Второй этап — вычисление дополнительных коэффициентов (они обозначены в статье как $L(v)$ и $H(v)$). На каждом шаге этого этапа обрабатываемая вершина выделяется цветом, и рядом с ней отображаются вычисленные значения. В поле текстового вывода печатается краткое обоснование того, как был получен очередной коэффициент.

3. Третий этап — проверка рёбер. На каждом шаге данного этапа проверяемое ребро подсвечивается одним из двух цветов в зависимости от того, является ли оно мостом. В поле текстового вывода печатается, на основании чего алгоритмом было принято такое решение.

1.1.3 Требования к пользовательскому интерфейсу.

1. Приложение должно иметь графический оконный интерфейс.

2. Способы взаимодействия пользователя с графом посредством графического интерфейса описаны в разделе «Требования ко входным данным».

3. Загрузка графа из файла сохранения осуществляется путём нажатия кнопки `Open File` (см. рисунок 1) и выбора нужного файла в возникшем диалоге.

4. Сохранение графа в файл осуществляется путём нажатия кнопки `Save File` и ввода имени файла в возникшем диалоге.

Эскиз пользовательского интерфейса представлен на рисунке 1.

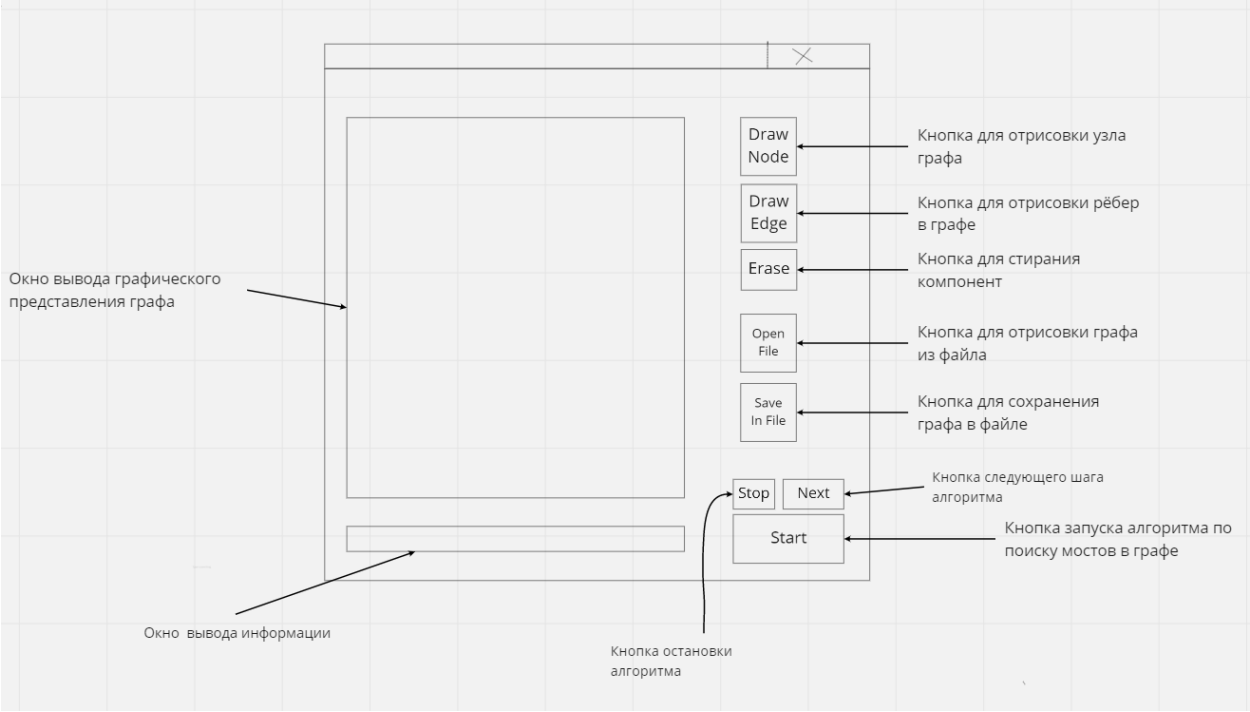

Рисунок 1 – Эскиз графического интерфейса программы

1.2. Уточнения требований после сдачи прототипа.

1. Поле вывода текстовой информации должно быть увеличено, чтобы на нём было видно несколько строк; текст в нём должен быть выделяемым и копируемым.

2. Кнопка `Next` должна быть справа от `Stop`, как в эскизе интерфейса.

3. В заголовок окна должно быть добавлено название алгоритма.

1.3 Уточнения требований после сдачи первой версии.

1. Реализовать функциональность удаления рёбер;
2. Заменить кнопки Draw Node, Draw Edge и Erase на радиокнопки, чтобы не нужно было нажимать кнопку, пока выполняется одно и то же действие редактирования графа;
3. Имена вершин выводить внутри вершин более крупным шрифтом.

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1. План разработки.

1. К 5 июля — прототип приложения, демонстрирующий внешний вид интерфейса, а также графическое отображение вершин и рёбер графа.
2. К 7 июля — первая версия. Ожидаемые изменения по сравнению с прототипом: считывание графа из файла, возможность запуска алгоритма и получения результата в текстовом (не графическом) виде.
3. К 9 июля — вторая версия. Ожидаемые изменения по сравнению с первой версией: визуализация результатов работы (подсветка мостов), подсветка компоненты графа, с которой происходит работа, вывод текстовых пояснений.
4. К 12 июля — отчёт по результатам практики и третья версия приложения. Ожидаемые изменения по сравнению со второй версией: печать числовых характеристик вершин в графическом поле и отрисовка орентированных ребёр на первом этапе работы алгоритма (см. требования к визуализации алгоритма), вывод информативных сообщений об ошибках, реализация сохранения графа в файл.

2.2. Распределение ролей в бригаде.

- Ахримов А.М. — разработка графического интерфейса пользователя и графического представления графа;

- Эйсвальд М.И. — связь графических компонент и внутренней логики приложения; управление визуализацией алгоритма.
- Максимов Е.А. — разработка внутренней логики приложения и реализация алгоритма Тарьяна поиска мостов в графе;

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

4. ТЕСТИРОВАНИЕ

ЗАКЛЮЧЕНИЕ

Список литературы

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ