МИНОБРНАУКИ РОССИИ
Санкт-Петербургский государственный
электротехнический университет
«ЛЭТИ» им. В.И. Ульянова (Ленина)
Кафедра МО ЭВМ

ОТЧЕТ

по учебной практике

Тема: «Поиск компонент сильной связности»

Студентка гр. 8304	_____	Николаева М. А.
Студентка гр. 8304	_____	Мельникова О. А.
Студент гр. 8304	_____	Щука А. А.
Руководитель	_____	Фирсов М. А.

Санкт-Петербург

2020

ЗАДАНИЕ

НА УЧЕБНУЮ ПРАКТИКУ

Студентка Николаева М. А. группы 8304

Студентка Мельникова О. А. группы 8304

Студент Щука А. А. группы 8304

Тема практики: Поиск компонент сильной связности

Задание на практику:

Командная итеративная разработка визуализатора алгоритма на Java с графическим интерфейсом.

Алгоритм: алгоритм Косарайю.

Сроки прохождения практики: 29.06.2020 – 12.07.2020

Дата сдачи отчета: _.07.2020

Дата защиты отчета: _.07.2020

Студентка гр. 8304	_____	Николаева М. А.
Студентка гр. 8304	_____	Мельникова О. А.
Студент гр. 8304	_____	Щука А. А.
Руководитель	_____	Фирсов М. А.

АННОТАЦИЯ

Целью работы является получения навыков работы с такой парадигмой программирования, как объектно-ориентированное программирование. Для получения данных знаний выполняется один из вариантов мини-проекта. В процессе выполнения мини-проекта необходимо реализовать графический интерфейс к данной задаче, организовать ввод и вывод данных с его помощью, реализовать сам алгоритм, научиться работать в команде. В данной работе в качестве мини-проекта выступает поиск компонент сильной связности (визуализация алгоритма Косарайю). Также при разработке выполняется написание тестирования, для проверки корректности алгоритма.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Основная цель практики – реализация мини-проекта, который является визуализацией алгоритма. В данной работе это алгоритм Косарайю. Для выполнения этой цели были поставлены задачи: реализация алгоритма, разработка GUI к проекту, написание тестирования.

Ориентированный граф (орграф) называется сильно связным, если любые две его вершины s и t сильно связаны, то есть если существует ориентированный путь из s в t и ориентированный путь из t в s. Компонентами сильной связности орграфа называются его максимальные по включению сильно связные подграфы. Областью сильной связности называется множество вершин компоненты сильной связности.

Алгоритм Косарайю (в честь американского учёного индийского происхождения Самбасивы Рао Косарайю) — алгоритм поиска областей сильной связности в ориентированном графе. Чтобы найти области сильной связности, сначала выполняется поиск в глубину (DFS) на обращении исходного графа (то есть против дуг), вычисляя порядок выхода из вершин. Затем мы используем обращение этого порядка, чтобы выполнить поиск в глубину на исходном графе (в очередной раз берём вершину с максимальным номером, полученным при обратном проходе). Деревья в лесе DFS, которые выбираются в результате, представляют собой сильные компоненты связности.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1 Исходные требования к программе

Пользователь с помощью графического интерфейса конструирует не взвешенный ориентированный граф. Пользователь может добавлять/удалять вершины графа. Помимо этого, пользователь может задавать ребра в графе, нажав на одну вершину и потянув указатель мыши к другой вершине. После запуска алгоритма программа визуализирует алгоритм, а также выводит текстовые данные, поясняющие ход выполнения алгоритма. Также входные данные могут быть получены из файла.

1.1.1 Требования к входным данным

Для корректной работы алгоритма требуется:

- множество вершин графа
- множество ребер графа

1.1.2 Требования к визуализации

Программа должна обладать простым и понятным интерфейсом. Визуализироваться должны все стадии алгоритма - обход графа в глубину, отображение всех вершин в порядке увеличения времени выхода (массив verticese), обход в глубину на инвертированном графе (каждый раз для обхода будем выбирать ещё не посещенную вершину с максимальным индексом в массиве verticese), при этом на каждой итерации должно быть отображение посещенных вершин - компоненты сильной связности. 0-я версия интерфейса (прототип) должна быть готова к 6 июля, выполнена 28 июня.
Прототип интерфейса представлен на рисунке 1.

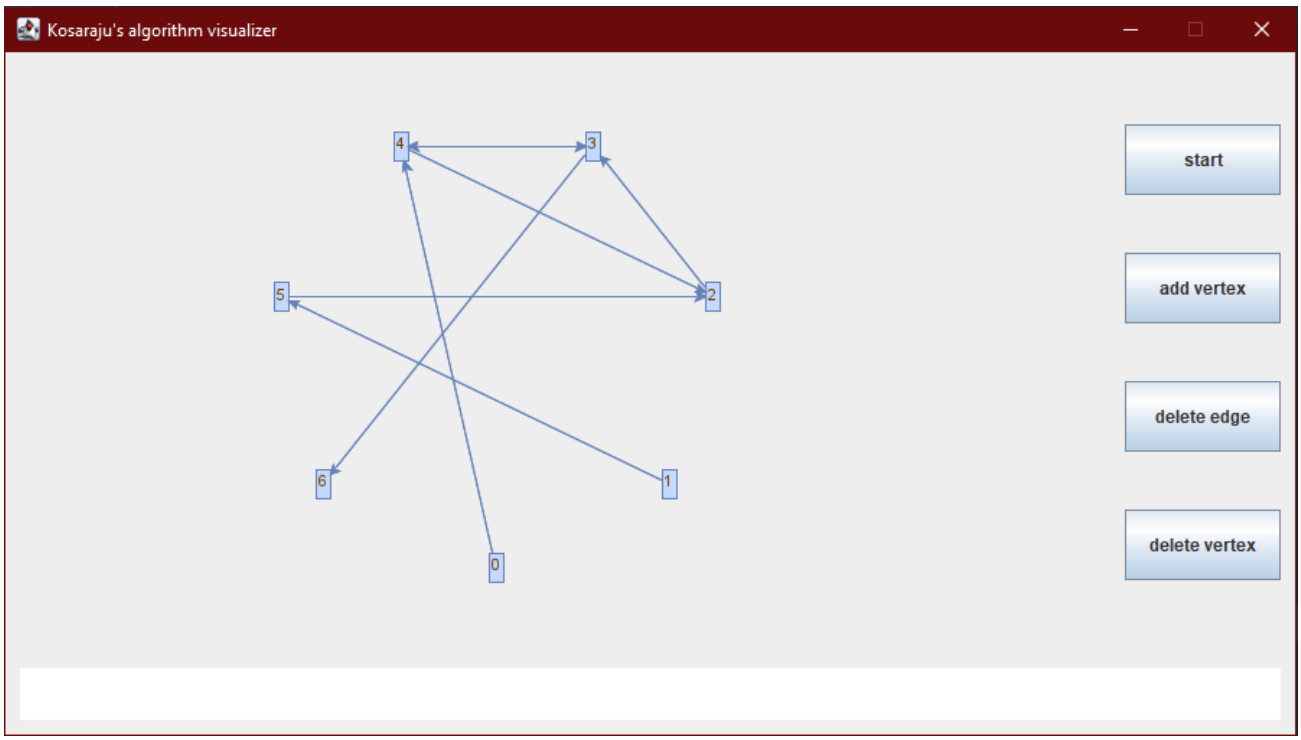

Рисунок 1 - Прототип интерфейса

Программа имеет интерактивную область с графом с возможностью выбора вершин, ребер и запуска алгоритма. С правой стороны расположены функциональные кнопки, снизу расположено поле для вывода текстовой информации для пояснения алгоритма.

1.1.3 Требования к алгоритму и данным

Алгоритм получает на вход граф, заданный пользователем, и в процессе выполнения передает промежуточные состояния графа для визуализации.

1.1.4 Требования к выходным данным

Выходные данные: компоненты сильной связности в исходном графе.

1.2 Требования к программе после уточнения работы

1.2.1 Требования к входным данным

Для ввода из файла первая строка - кол-во вершин. Дальше ребра вида i , j , где i и j от нуля до н (реализация во второй версии). Либо ввод графа в приложении через кнопку добавить вершину и перетягивания вершин для

создания ребер.

1.2.2 Требования к визуализации

1-я версия визуализации должна быть готова к 8 июля. Должна быть реализована визуализация алгоритма. Пояснения о том, как пользоваться программой должны вызываться кнопкой помощь/справка. Нажатием на кнопку должна быть возможность добавлять вершины. При перетягивании от одной вершины к другой мышкой, необходимо добавлять ребра. Вершину/ребро можно выбрать и соответствующими кнопками и удалить (при выборе должны подсвечиваться). Должна быть кнопка очистить, которая удаляет все. Должна быть добавлена кнопка старт для запуска алгоритма. В первой версии визуализироваться будет только результат алгоритма. Во 2-ой версии также должны быть кнопки стоп, продолжить, алгоритм будет визуализироваться по шагам, непрерывно, с возможностью остановить и продолжить. Поиск в глубину окрашивает вершины в другой цвет, выводятся текстовые данные о выходе из вершины. Вершины становятся старого цвета. На рисунке транспонируется граф. Снова запускается поиск в глубину (тоже самое все, с отличием в том, что теперь надо компоненты отделять) Вершины каждой компоненты связности имеют свой цвет (для каждой компоненты разный). Должна быть кнопка считывания с файла и вывода в файл. Таким образом во 2-ой версии будет улучшен интерфейс, повышение подробности и качества визуализации, а также добавлено юнит тестирование.

1.2.3 Требования к алгоритму и данным

Алгоритм, для визуализации, должен при поиске компонент связности возвращать промежуточные значения, касающиеся первого и второго обходов в глубину (где находимся, данные о выходе из вершины, выделенные компоненты), граф с инвертированными ребрами.

1.2.4 Требования к выходным данным

Выходными данными должны являться раскрашенные в свой цвет компоненты связности, либо вывод в файл соответствующих компонент.

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1 План разработки

1. Создание прототипа
    1) Обсудить задание, распределить роли, выбрать необходимые средства разработки и структуры данных. Данный пункт задания необходимо выполнить до 1 июля 2020 года.
    2) Создать прототип GUI (0-я версия визуализации). Должна быть возможность создавать граф и отображать его. Данный пункт задания необходимо выполнить к 2 июля 2020 года.
2. Создание основного GUI
    1) Реализовать структуры данных, необходимые для алгоритма. Данный пункт задания необходимо выполнить к 3 июля 2020 года.
    2) Реализовать алгоритм без использования GUI. Данный пункт задания необходимо выполнить к 4 июля 2020 года.
    3) Добавить JavaDoc комментарии для генерации документации к алгоритму и структуре данных. Данный пункт задания необходимо выполнить до 5 июля 2020 года.
    4) Связывание структур данных алгоритма и визуализации. Данный пункт задания необходимо выполнить к 7 июля 2020 года.
    5) Реализация основного GUI. (1-я версия). Должна быть реализована визуализация алгоритма. Данный пункт задания необходимо выполнить к 6 июля 2020 года.
3. Сборка проекта
    1) Первичная сборка проекта и первичное тестирование его функций. Данный пункт задания необходимо выполнить к 8 июля 2020 года.
4. Окончательная сборка проекта. Финальная версия
    1) Отладка ошибок, улучшение GUI (2-я версия) улучшен интерфейс, повышение подробности и качества визуализации, добавление юнит

тестирования и добавление возможности считывания входных данных из файла. Данный пункт задания необходимо выполнить к 9 июля 2020 года.

2) Окончательная сборка проекта и добавление полноценного тестирования всех модулей программы. Данный пункт задания необходимо выполнить до 11 июля 2020 года.

2.2 Распределение ролей в бригаде

- Николаева М. А.:
  - Тестирование программы;
  - Реализация ввода-вывода;
  - Проектирование, организация командной работы.
- Щука А. А.:
  - Создание основного GUI;
  - Объединение отдельных модулей программы;
  - Рефракторинг кода.
- Мельникова О. А.:
  - Создание алгоритма и структур данных;
  - Оформление пояснительной записки;
  - Расширение возможностей GUI.

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

3.1 Использованные структуры данных

3.1 Основные методы

4. ТЕСТИРОВАНИЕ

4.1 Написание UNIT Tests

4.2 Ручное тестирование программы

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ А. UML ДИАГРАММА

ПРИЛОЖЕНИЕ Б. ИСХОДНЫЙ КОД