**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.

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

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. Во время визуализации пользователь может приостанавливать алгоритм, а также менять скорость визуализации. Входные данные могут быть получены из файла.

### 1.2.2 Требования к визуализации

Должна быть добавлена возможность вызова пояснения о том, как пользоваться программой с помощью ToolBar. Вершины должны добавляться нажатием на соответствующую кнопку. Ребра должны добавляться при перетягивании курсором от одной вершины к другой. Должна быть добавлена возможность выбора считывания из файла. Вершину/ребро можно выбрать и соответствующими кнопками удалить (при выборе должны подсвечиваться). Должна быть кнопка очистить, которая полностью удаляет весь граф. Для запуска алгоритма должна быть добавлена кнопка старт. Также должна быть кнопка стоп, при нажатии которой алгоритм сразу же завершает свою работу, граф возвращается в исходное состояние. Должна быть реализована возможность остановить работу алгоритма (кнопка пауза) и продолжить с места остановки (кнопка старт). Также должна быть добавлена возможность

уменьшать и увеличивать скорость демонстрации работы алгоритма. Изначально кнопки остановки, паузы и изменение скорости демонстрации заблокированы. После нажатия кнопки старт должны стать доступными заблокированные кнопки, а кнопка старт и все кнопки, отвечающие за изменение графа, заблокироваться. При нажатии кнопки паузы становится доступной кнопка старт, кнопка паузы блокируется. При нажатии кнопки стоп все кнопки, кроме остановки, паузы и изменения скорости демонстрации, становятся доступными.

Визуализация работы алгоритма представляет собой: первый поиск в глубину при выходе из вершины окрашивает ее в красный цвет. Затем вершины возвращают исходный цвет. Граф транспонируется, ребра ориентируются в противоположные стороны. Второй запуск поиска в глубину. Вершины также меняют цвет. При этом вершины каждой компоненты связности должны иметь свой цвет, соответствующий компоненте, в которую они входят. По ходу работы алгоритма должны выводиться пояснения, касающиеся первого и второго поиска в глубину, а также о транспонировании графа.

Итоговый прототип интерфейса представлен на рисунке 2.

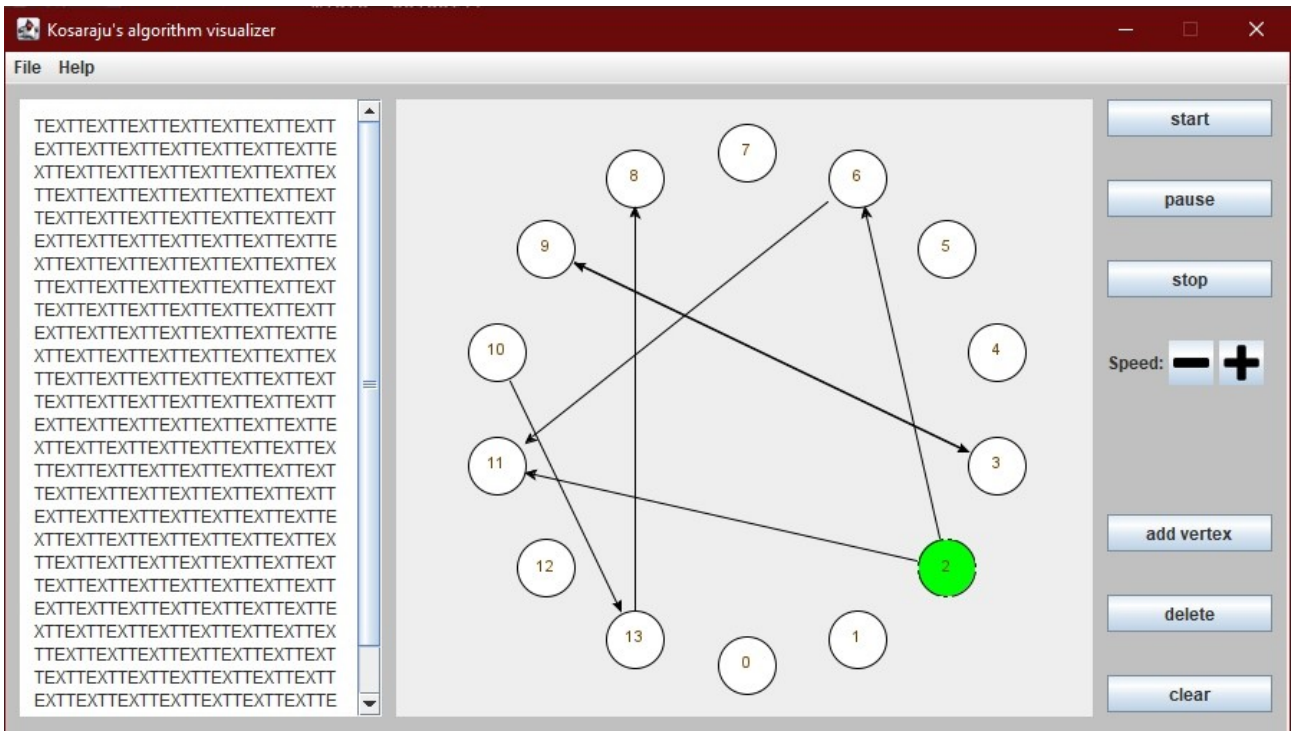


Рисунок 1 - Итоговый прототип интерфейса

### 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 (данные о запуске первого и второго поиска в глубину, информация о посещении вершин), а также о транспонировании графа.

По ходу работы алгоритма должны выводиться следующие пояснения: при первом запуске поиска в глубину сообщение о его запуске и данные о посещении вершин, затем сообщение о том, что все ребра были инвертированы и граф транспонирован, при втором запуске поиска в глубину сообщение о его запуске и данные о посещении вершин, информация о количестве найденных компонент сильной связности и сами компоненты.

Для ввода из файла первой строкой должно идти кол-во вершин в графе. Дальше ребра вида i j, где i и j в диапазоне от 0 до кол-ва вершин.

### 1.2.4 Требования к выходным данным

Выходными данными являются раскрашенные в свой цвет компоненты связности.

## 2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

### 2.1 План разработки

1. Создание прототипа (к 2 июля 2020 года).
    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). Добавить возможность создавать граф и отображать его. Вершины должны добавляться нажатием на соответствующую кнопку. Ребра должны добавляться при перетягивании курсором от одной вершины к другой. Вершину/ребро можно выбрать и соответствующими кнопками удалить (при выборе должны подсвечиваться). Данный пункт задания необходимо выполнить к 2 июля 2020 года.
2. Создание 1-ой версии программы (к 8 июля 2020 года).
    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 6 июля 2020 года.
    5) Реализация основного GUI. (1-я версия). Должна быть реализована визуализация работы алгоритма. Добавить кнопку отчистки, которая

полностью удаляет весь граф. Для запуска алгоритма должна быть добавлена кнопка старт. При ее нажатии все остальные кнопки в процессе работы алгоритма нажать нельзя. Также должна быть реализована возможность полной остановки работы алгоритма (кнопка стоп). Данный пункт задания необходимо выполнить к 7 июля 2020 года.

6) Отладка ошибок. Данный пункт задания необходимо выполнить к 8 июля 2020 года.

3. Создание финальной версии (к 11 июля 2020 года).

1) Улучшение GUI (2-я версия). Добавление возможности остановить работу алгоритма (кнопка пауза) с последующим продолжение работы с места остановки. Добавление возможности увеличивать и уменьшать скорость демонстрации работы алгоритма. Добавление возможности получения пояснения об использовании программы с помощью ToolBar. Данный пункт задания необходимо выполнить к 9 июля 2020 года.
2) Добавление возможности считывания входных данных из файла. Данный пункт задания необходимо выполнить к 9 июля 2020 года.
3) 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