**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по учебной практике**

Тема: VKFriends

|  |  |  |
| --- | --- | --- |
| Студент гр. 7382 |  | Гаврилов А.В. |
| Студент гр. 7382 |  | Гиззатов А.С. |
| Студент гр. 7382 |  | Государкин Я.С. |
| Руководитель |  | Фирсов М.А. |

Санкт-Петербург

2019

**ЗАДАНИЕ**

**на учебную практику**

|  |  |  |
| --- | --- | --- |
| Студент Гаврилов А.В. группы 7382 | | |
| Студент Гиззатов А.С. группы 7382 | | |
| Студент Государкин Я.С. группы 7382  Тема практики: VKFriends | | |
| Задание на практику:  Командная итеративная разработка визуализатора алгоритма(ов) на Java с графическим интерфейсом.  Алгоритм: Поиск общих друзей двух пользователей во вконтакте. | | |
| Сроки прохождения практики: 01.07.2019 – 14.07.2019 | | |
| Дата сдачи отчета: 00.07.2019 | | |
| Дата защиты отчета: 00.07.2019 | | |
|  | | |
| Студент |  | Гаврилов А.В. |
| Студент |  | Гиззатов А.С. |
| Студент |  | Государкин Я.С. |
| Руководитель |  | Фирсов М.А. |

**Аннотация**

Темой данной учебной практики является командная итеративная разработка визуализатора алгоритма на языке программирования Java.

Цель работы научиться писать на языке программирования java, также обучиться налаживанию коммуникации между участниками практики, разобраться с декомпозицией обязанностей. Визуализировать любой алгоритм для работы с графами. В работе представлена визуализация поиска общих друзей в ВК.

**Содержание**

[**ВВЕДЕНИЕ** 5](#_Toc13676558)

[**1. ТРЕБОВАНИЯ К ПРОГРАММЕ** 6](#_Toc13676559)

[**1.1. Исходные Требования к программе** 6](#_Toc13676560)

[**1.1.1. Примеры работы программы.** 6](#_Toc13676561)

[**1.1.2. Примеры тестирования программы.** 6](#_Toc13676562)

[**2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ** 7](#_Toc13676563)

[**2.1. План разработки** 7](#_Toc13676564)

[**2.1.1. Создать базовый gui с кпопками добавления/удаления вершины, удаления всего графа, графическое поле для графа и поле/окно для вывода информации об аккаунте.** 7](#_Toc13676565)

[**2.1.2. Добавить возможность строить граф, где вершина - аккаунт, а ребро - кол-во общ. Друзей.** 7](#_Toc13676566)

[**2.1.3. Добавить возможность добавлять/удалять новые вершины.** 7](#_Toc13676567)

[**2.1.4. Реализовать базовый набор unit тестов.** 7](#_Toc13676568)

[**2.1.5. Свести все наработки в один проект (реализовать взаимодействие между частями проекта).** 7](#_Toc13676569)

[**2.1.6. Выводить некоторую информацию об аккаунте по нажатию на вершину.** 7](#_Toc13676570)

[**2.1.7. Строить минимальное/максимальное остовное дерево на основе базового графа.** 7](#_Toc13676571)

[**3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ** 8](#_Toc13676572)

[**ЗАКЛЮЧЕНИЕ** 9](#_Toc13676573)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 10](#_Toc13676574)

[**ПРИЛОЖЕНИЕ А** 11](#_Toc13676575)

# **ВВЕДЕНИЕ**

Научиться программировать на языке java, а конкретно:

1. Разобраться с созданием графического интерфейса при помощи встроенных библиотек.
2. Построить граф друзей, ребра которого показывает количество общих друзей между двумя людьми связанными этим ребром, а вершина содержит краткую информацию о пользователе.
3. Написать алгоритм, который ищет всех общих друзей между двумя выбранными пользователями.
4. Написать юнит тесты для проверки корректности работы программы.

# **1. ТРЕБОВАНИЯ К ПРОГРАММЕ**

* 1. **Исходные Требования к программе**

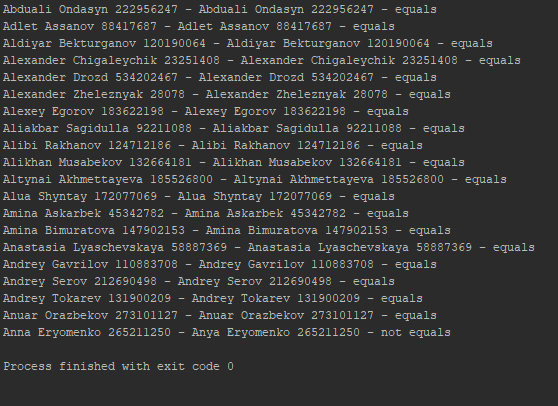
При запуске проекта будет создаваться отдельное окно. Данное окно будет иметь 3 поля:

1. Поле графического представления графа и визуализации алгоритма. Граф строится по данным, введенными пользователем с помощью графического интерфейса.
2. Поле с элементами управления:

* Добавление вершины в граф.
* Перестройка графа.
* Удаление вершины.
* Удаление графа.

1. Диалог создания графа:

* Ввод URL/ID пользователя.
* Ввод URL/ID двух или более пользователей для нахождения общих друзей.
  + 1. **Примеры работы программы.**
    2. **Примеры тестирования программы.**

Рис.1 – Пример тестирования метода parseFriendsJson.

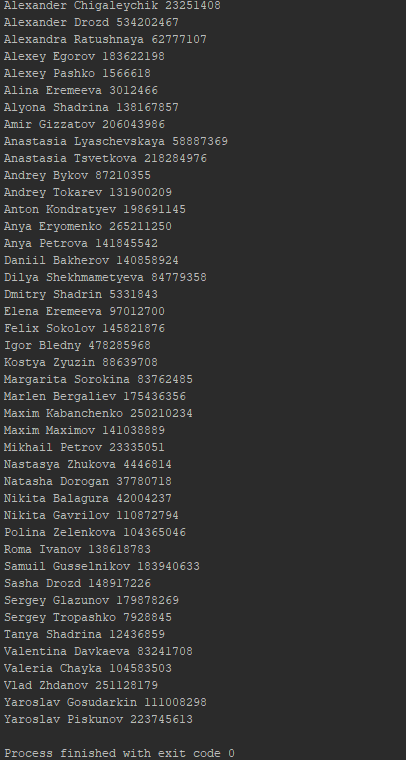
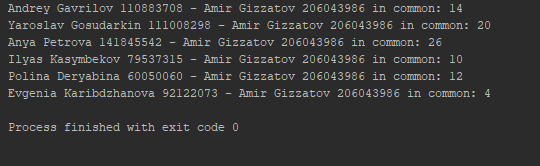


Рис.2 – Пример тестирования метода getUser.

Рис.3 – Пример тестирования метода getCommonFriends.

# **2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ**

# **2.1. План разработки**

* + 1. **Создать базовый gui с кпопками добавления/удаления вершины, удаления всего графа, графическое поле для графа и поле/окно для вывода информации об аккаунте.**
    2. **Добавить возможность строить граф, где вершина - аккаунт, а ребро - кол-во общ. Друзей.**
    3. **Добавить возможность добавлять/удалять новые вершины.**
    4. **Реализовать базовый набор unit тестов.**
    5. **Свести все наработки в один проект (реализовать взаимодействие между частями проекта).**
    6. **Выводить некоторую информацию об аккаунте по нажатию на вершину.**
    7. **Строить минимальное/максимальное остовное дерево на основе базового графа.**

**2.2. Распределение ролей в бригаде**:

* **Государкин** **Ярослав - ответственный за алгоритмы.**
* **Гаврилов Андрей - ответственный за графический интерфейс.**
* **Гиззатов Амир** - **ответственный за юнит тестирование и отчёт.**

# **3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ**

* 1. **Используемые структуры данных.**

Граф для реализации алгоритма представляется

* 1. **Описание классов.**

Программа состоит из классов: ApplicationMain, VKClient, VKUser, MainWindow.

* Класс ApplicationMain отвечает за выполнение программы в целом. Он создает граф и вызывает методы для нахождения общих друзей.
* Класс VKClient отвечает за получение списка друзей, списка общих друзей, информации о пользователе, отправка запроса в вк api.

Поля класса VKClient предоставлены в таблице 1, а методы перечислены в таблице 2.

Таблица 1-Поля класса VKClient

|  |  |
| --- | --- |
| Поле | Значение поля |
| public static final String[] basicArgs | Информация о пользователе |
| private static final String accessVkApiToken | Ключ для отправки запроса к вк api |
| private static final String versionVkApi | Версия api вк |
| private static final String beginVkApi | Начало запроса к api вк |
| private static final String endVkApi | Конец запроса к api вк |

|  |
| --- |
| методы |
| public ArrayList<VKUser> parseFriendsJson(String str) |
| public ArrayList<int[]> getCommonFriends(int srcId, int[] targetIds) |
| public VKUser getUser(int userId, String[] args) |
| public String getUserFriends(int userId, String order, String[] args) |
| public ArrayList<VKUser> getFriends( int userId, String order, String[] args ) |
| private String createGetRequest(String method, int id, String order, String[] args) |
| private String getRequest(String url) |

* Класс VKUser отвечает за информацию о пользователе во вконтакте. Имеет методы для получения строки из полей класса(userdId, firstName, lastName), получение строки из списка друзей, поиска и удаления пользователя в списке. Поля класса представлены в таблице 3.

Таблица 3 – Поля класса VKUser.

|  |  |
| --- | --- |
| Поле | Значение поля |
| public int userId | Уникальный номер пользователя. |
| public String firstName | Имя пользователя. |
| public String lastName | Фамилия пользователя. |

|  |
| --- |
| Методы |
| public String toString() |
| public static String[] arrayToStrings(ArrayList<VKUser> list) |
| public static void findAndRemove(ArrayList<VKUser> list, int id) |

* Класс MainWindow отвечает за графический интерфейс, строит граф друзей и граф общих друзей пользователя. Поля и методы класса представлены в таблицах 4 и 5.

Таблица 4 – Поля класса MainWindow.

|  |  |
| --- | --- |
| Поле | Значение поля |
| private final int leftPanelWidth | Высота левой панели |
| private final int colNum | Количество столбцов в textfild |
| private JPanel toolBar | Панель инструментов |
| private JPanel buttonBar | Панель кнопок |
| private JPanel LeftPanel | Левая панель |
| private final Color colorForTools | Цвет инструментов |
| private final Color colorForOther | Цвет графа и текстового поля |
| private JList<String> mList | Динамический список пользователей |
| private JTextField InputField | Поле ввода id или URL пользователя |
| private mxGraph graph | Граф |
| private VKClient mClient | Информация о пользователе |
| private ArrayList<VKUser> mUsers | Список пользователей типа VKUser |

|  |
| --- |
| Методы |
| protected mxGraphComponent makeGraph() |
| protected void makeToolbar() |
| public void actionPerformed(ActionEvent actionEvent) |
| protected void makeLeftPane(mxGraphComponent graphComponent) |
| private void setUpVKClient() |

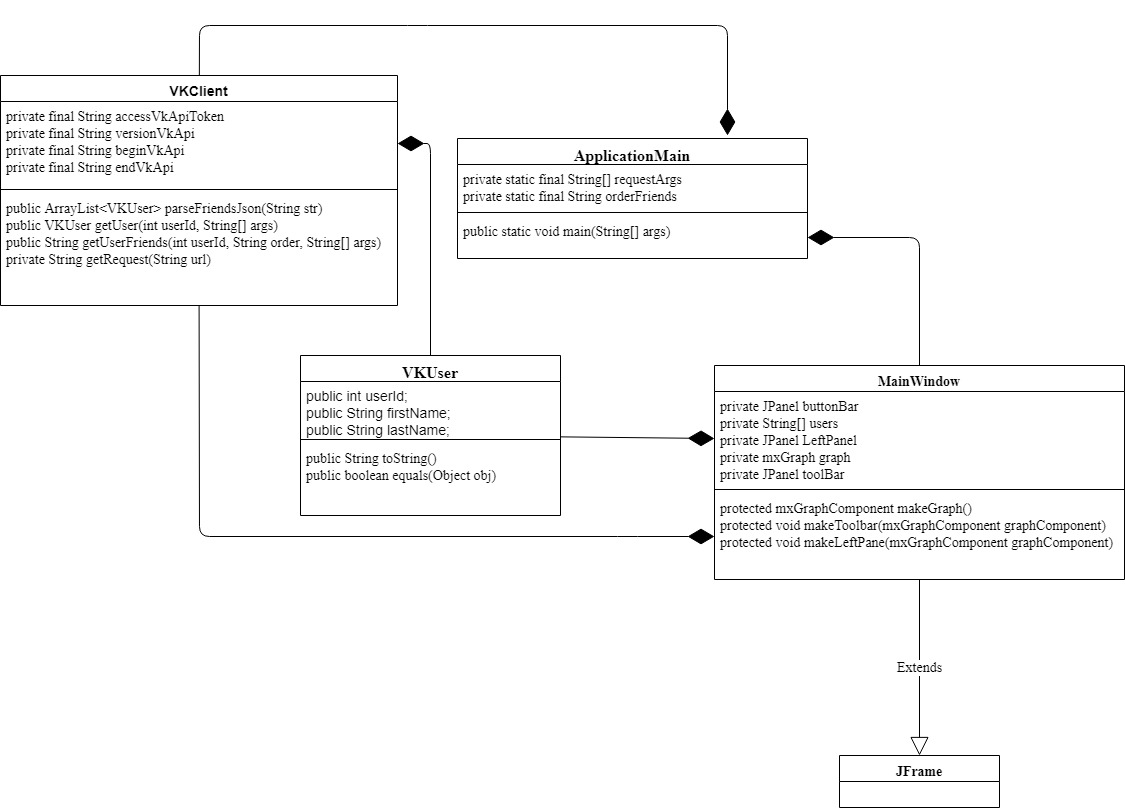
**** **3.3 UML – диаграмма классов.**

Рис.3 - диаграмма классов.

# **ЗАКЛЮЧЕНИЕ**

Таким образом, был реализован алгоритм поиска общих друзей во вконтакте с визуализацией. Был разработан интерфейс удобный и понятный для пользователя. Так же в ходе учебной практики были получены знания по языку программирования Java, получены навыки в визуализации алгоритмов и работе в команде.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. VK API [Электронный ресурс] URL:

<https://vk.com/dev/manuals>

1. HABR[Электронный ресурс] URL:

<https://habr.com/>

1. ORACLE JFRAME[Электронный ресурс] URL:

<https://docs.oracle.com/javase/7/docs/api/javax/swing/JFrame.html>

1. JUNIT ON JAVARUSH[Электронный ресурс] URL:

<https://javarush.ru/groups/posts/605-junit>

# **ПРИЛОЖЕНИЕ А**

**Название приложения**

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