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

ОТЧЕТ

по учебной практике

Тема: VK friends

Студент гр. 7383 _____ Александров Р.А

Студент гр. 7383 _____ Рудоман В.А

Студентка гр. 7383 _____ Ханова Ю.А.

Руководитель _____ Фирсов М.А

Санкт-Петербург
2019

ЗАДАНИЕ
НА УЧЕБНУЮ ПРАКТИКУ

Студент Александров Р.А. группы 7383

Студент Рудоман В.А. группы 7383

Студентка Ханова Ю.А. группы 7383

Тема практики: VK friends

Задание на практику:

Командная итеративная разработка визуализатора алгоритма(ов) на Java с графическим интерфейсом.

Алгоритм: поиск минимального остовного дерева, алгоритм Краскала.

Сроки прохождения практики: 01.07.2019 – 14.07.2019

Дата сдачи отчета:

Дата защиты отчета:

Студент	_____	Александров Р.А.
Студент	_____	Рудоман В.А.
Студентка	_____	Ханова Ю.А.
Руководитель	_____	Фирсов М.А.

АННОТАЦИЯ

В ходе выполнение проекта была реализована задача поиска общих друзей среди группы людей в социальной сети vk.com, построения соответствующего графа и поиска в нем минимального остовного дерева. Поиск друзей осуществляется после авторизации пользователя через GUI, затем выбирается список интересующих друзей и строится граф. В качестве вершин графа принимаются указанные пользователи социальной сети, а весами ребер считается количество общих друзей между ними. Для построения минимального остовного дерева применяется алгоритм Краскала. Программа была разработана на языке Java в среде разработки IntelliJ IDEA.

SUMMARY

During the project, the task of finding common friends among a group of people on the social network vk.com, building the corresponding graph and finding the minimum spanning tree in it was realized. The search for friends is performed after the user is authorized through the GUI, then a list of friends of interest is selected and a graph is constructed. The indicated users of the social network are taken as vertices of the graph, and the number of mutual friends between them is considered to be edge weights. To build the minimum spanning tree, the Kruskal algorithm is used. The program was developed in the Java language in the IntelliJ IDEA development environment.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью выполнения данной практической работы является визуализация графа общих друзей из социальной сети vk.com для заданного пользователем списка участников социальной сети. Вершины графа должны однозначно идентифицировать пользователя социальной сети. У рёбер видно свойство - количество общих друзей. Нажатие на кнопку визуализирует минимальный подграф, объединяющий всех пользователей по рёбрам с наибольшим количеством общих друзей.

Для решения задачи выбран алгоритм поиска минимального (в нашем случае максимального) остовного дерева - алгоритм Краскала В качестве входных данных будут идентификаторы пользователей из vk.com и количество общих друзей между ними.

Краткое описание работы алгоритма: пока в остовном дереве не оказались все вершины, в него по очереди добавляется минимальное (у нас максимальное) ребро, не создающее цикла.

В результате работы алгоритма будет выводиться граф, в вершинах которого пользователи социальной сети, а на ребрах - количество общих друзей.

Из структур данных планируется использовать ArrayList, Hashmap.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Исходные Требования к программе

1.1.1 – Требования к вводу исходных данных

Вводится логин и пароль пользователя, затем из списка появившихся друзей выбирается интересующий набор, по нему будет строиться граф.

1.1.2 – Требования к визуализации

Предполагаемый вид GUI представлен на рис.1-4:

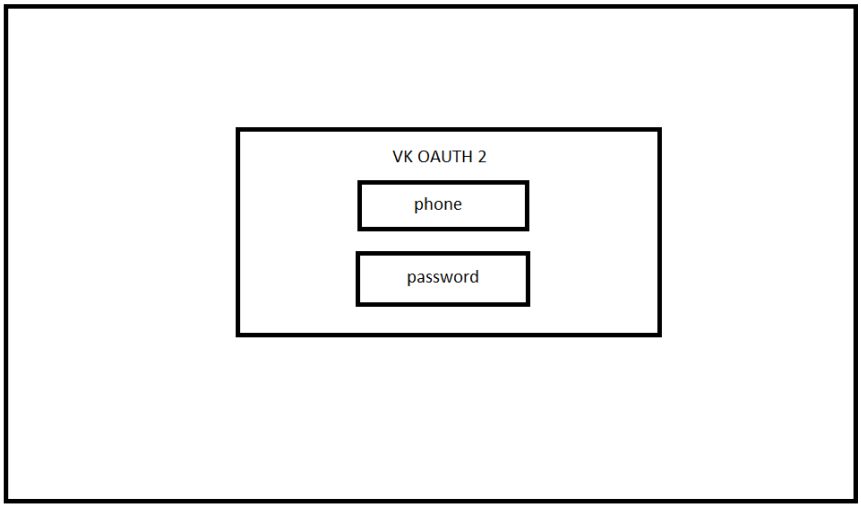

Рисунок 1 - выполняется oauth 2 авторизация на сайт vk.com

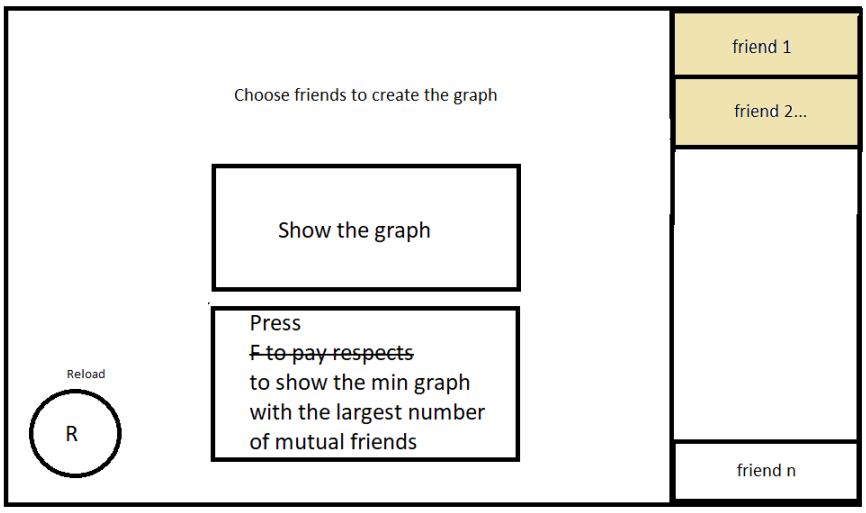

Рисунок 2 - у авторизованного пользователя появляется список друзей, которых можно выбрать для построения графа

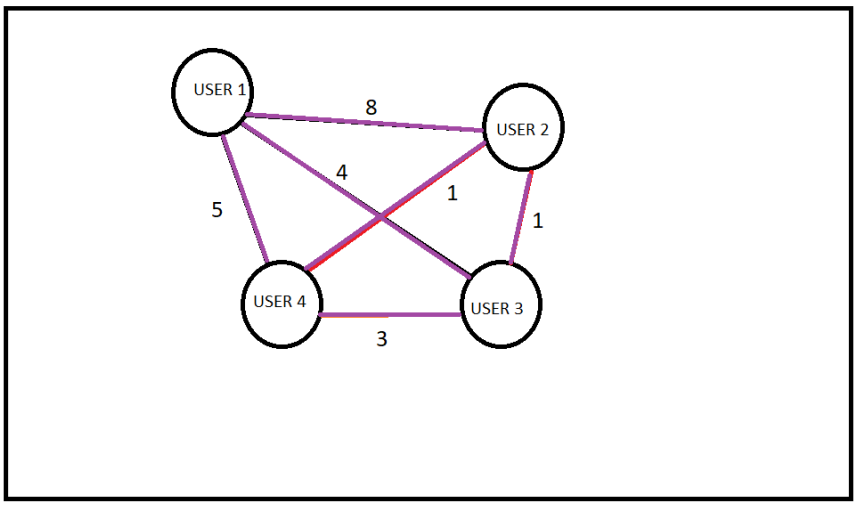

Рисунок 3 - строится граф общих друзей из выбранного пользователем списка.

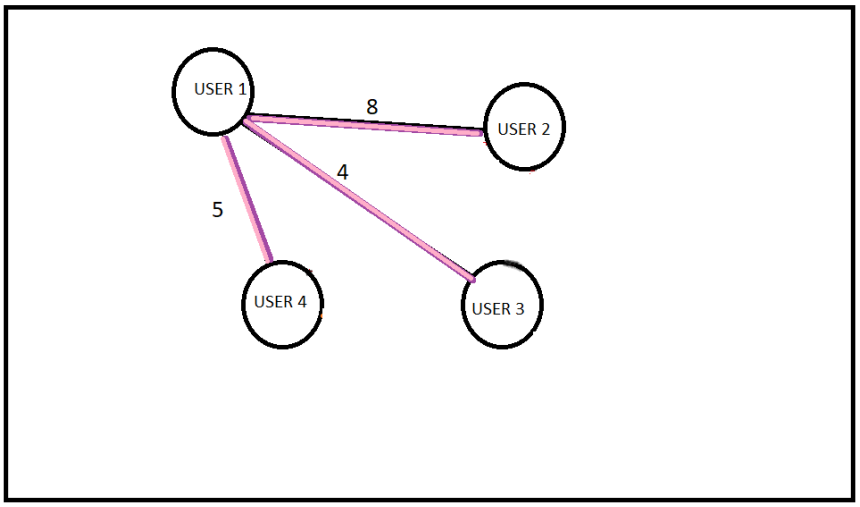

Рисунок 4 - по нажатию кнопки строится максимальное остовное дерево из этого графа

1.2. Уточнение требований после первой итерации

При изображении графов в качестве вершин выступают фотографии пользователей социальной сети

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1. План разработки

Первая итерация:

1. Выбор необходимого алгоритма – 03.07.19
2. Реализация алгоритма – 04-09.07.19
3. Написание запросов к api – 09.07.19
4. Написание юнит-тестов – 10.07.19

Вторая итерация:

5. Составление UML диаграммы – 10.07.19
6. Реализация графического интерфейса – 11.07.19
7. Написание отчета – 12.07.19

2.2. Распределение ролей в бригаде

Александров Р.А. – реализация алгоритма, GUI, UML диаграмма, работа с запросами api.

Рудоман В.А. – реализация алгоритма, тестирование.

Ханова Ю.А. – реализация алгоритма, написание отчета.

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

3.1. Использованные структуры данных

Для реализации алгоритма использовались такие структуры данных, как:

1. HashMap<K,V>;
2. ArrayList<T>;
3. HashSet<T>;
4. Array.

3.2. Основные методы

1. Методы addEdge(T, T, long, long, long) и sortEdgeByDesc() из интерфейса IGraph<T> предоставляют возможность добавлять ребра и сортировать их по убыванию веса соответственно;

2. Метод findMaxTree(IGraph<T>) из интерфейса IKruskal<T> принимает на вход готовый граф и находит максимальное остовное дерево алгоритмом Краскала;

3. Метод getRequest(String) интерфейса IHttpSenter осуществляет HTTP GET запрос по переданному URL;

4. Методы getFriends(String) и getMutualFriends(long, ArrayList<MinPersonDTO> container, String) интерфейса IFirendGetter осуществляют получение всех друзей пользователя и общих друзей среди выбранных соответственно;

5. Методы getEdgesForStartGraph(HashMap<Long, ArrayList<MinPersonDTO>>) и getEdgesForMinTree(HashMap<Long, ArrayList<MinPersonDTO>>) интерфейса IFriendGraphGetter возвращают ребра для графа выбранных друзей и ребра для максимального остовного дерева;

6. Метод PersonContainer getFriends(), ArrayList<MinPersonDTO> getStartGraph(ArrayList<MinPersonDTO>) и ArrayList<MinPersonDTO> getMaxTree() интерфейса IGuiConnector, осуществляя взаимодействовие GUI и бизнес-логики, возвращают всех друзей, ребра для графа выбранных друзей и ребра для максимального остовного дерева;

7. Методы createGraphEvent(javafx.scene.input.MouseEvent) и doKruskalEvent(javafx.scene.input.MouseEvent) из класса MainController по нажатию левой кнопки мыши строят граф выбранных друзей и осуществляют поиск максимального остовного дерева соответственно.

UML-диаграмма представлена в приложении А.

4. ТЕСТИРОВАНИЕ

4.1. Тестирование графического интерфейса

Разработка велась и тестировалась в среде Intellij Idea с OpenJDK 12 в системе Windows 10.

Пример работы графического интерфейса представлен в приложении Б.

4.2. Тестирование алгоритма

В работе использовались юнит-тесты для проверки работоспособности алгоритма Краскала при помощи библиотеки JUnit.

На вход был подан граф указанный на рис.5 с 8 ребрами.

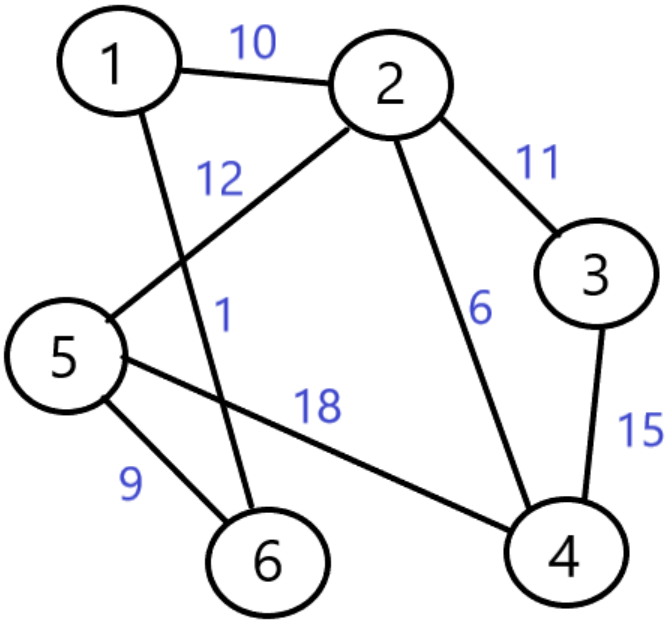

Рисунок 5 – Граф, используемый для тестирования

1 тест проверял, что добавится ровно 8 ребер.

2 тест проверял нахождение максимального остовного дерева, результат совпадает с рис. 6.

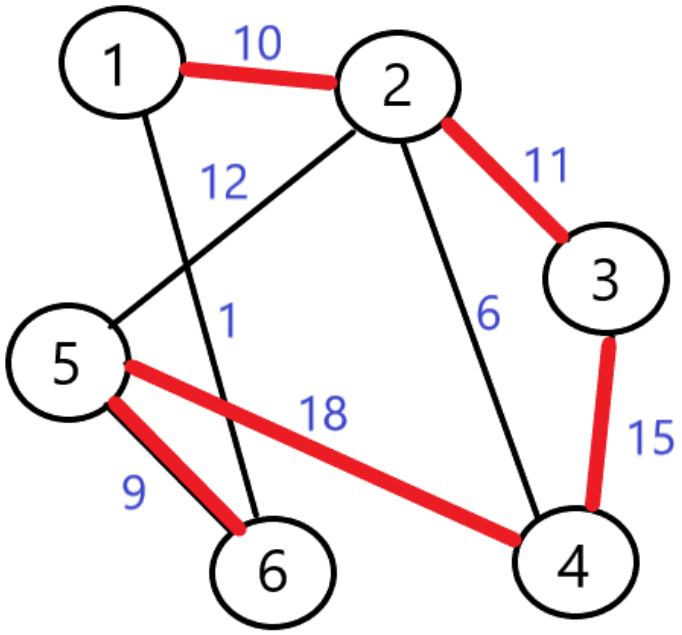

Рисунок 6 – Результат тестирования максимального остовного дерева, число ребер в нем равняется 5

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной практической работы были освоены основные принципы объектно-ориентированного программирования на языке Java, получены навыки создания графического интерфейса пользователя и реализации алгоритмов на графах. Также была разработана программа, визуализирующая граф общих друзей из социальной сети vk.com для заданного пользователем списка участников социальной сети.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления

2. Официальная документация к Java: https://docs.oracle.com/en/java/javase/

3. http://graphstream-project.org

4. Учебный курс по основам Java на Stepik: https://stepik.org/course/187/

5. Википедия: https://ru.wikipedia.org

6. https://ru.stackoverflow.com/

7. https://habr.com/ru/

8. https://openjdk.java.net/

9. https://gradle.org/

ПРИЛОЖЕНИЕ А

UML-ДИАГРАММА КЛАССОВ

Диаграмма классов представлена на рис. 7-8

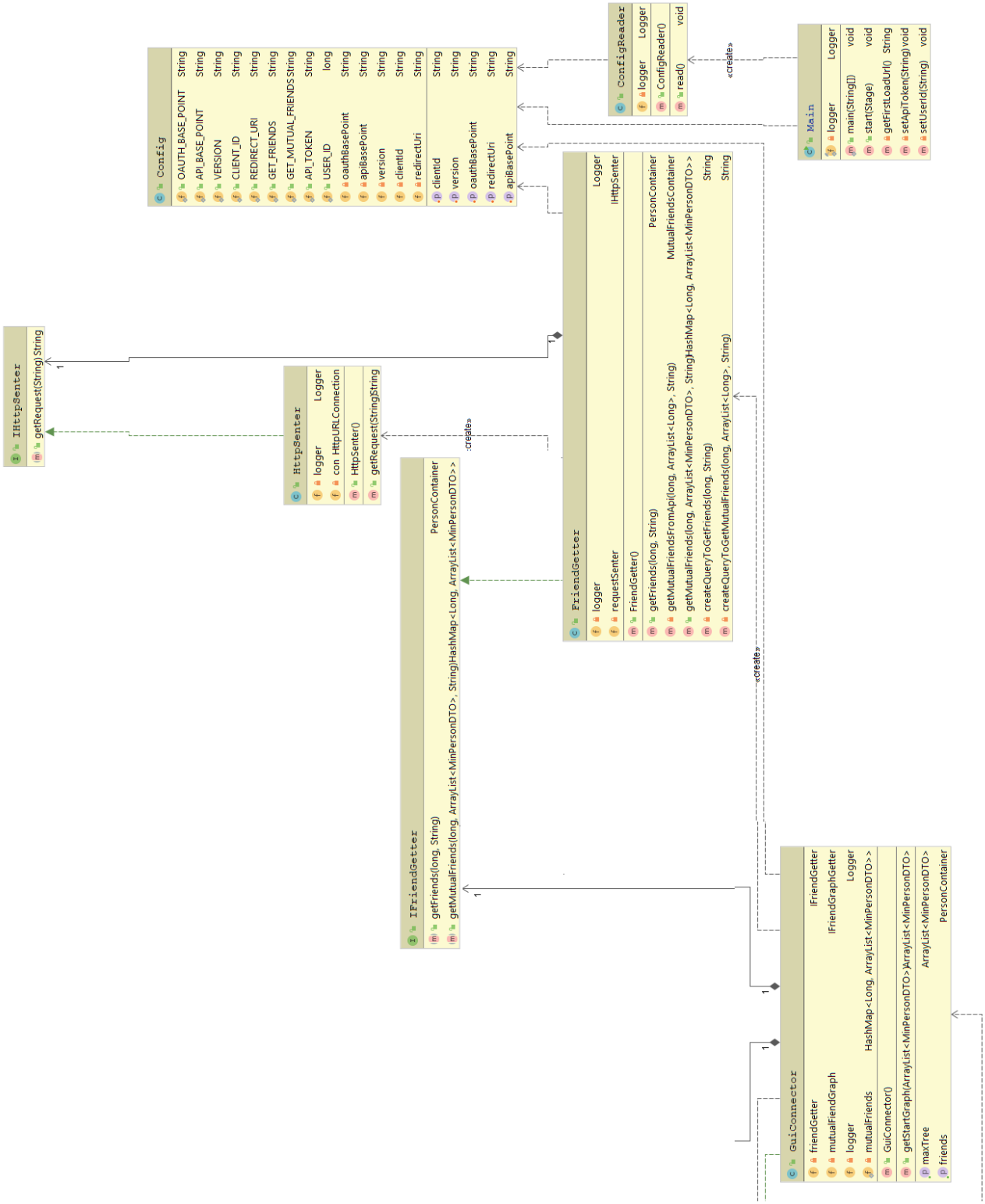

Рисунок 7

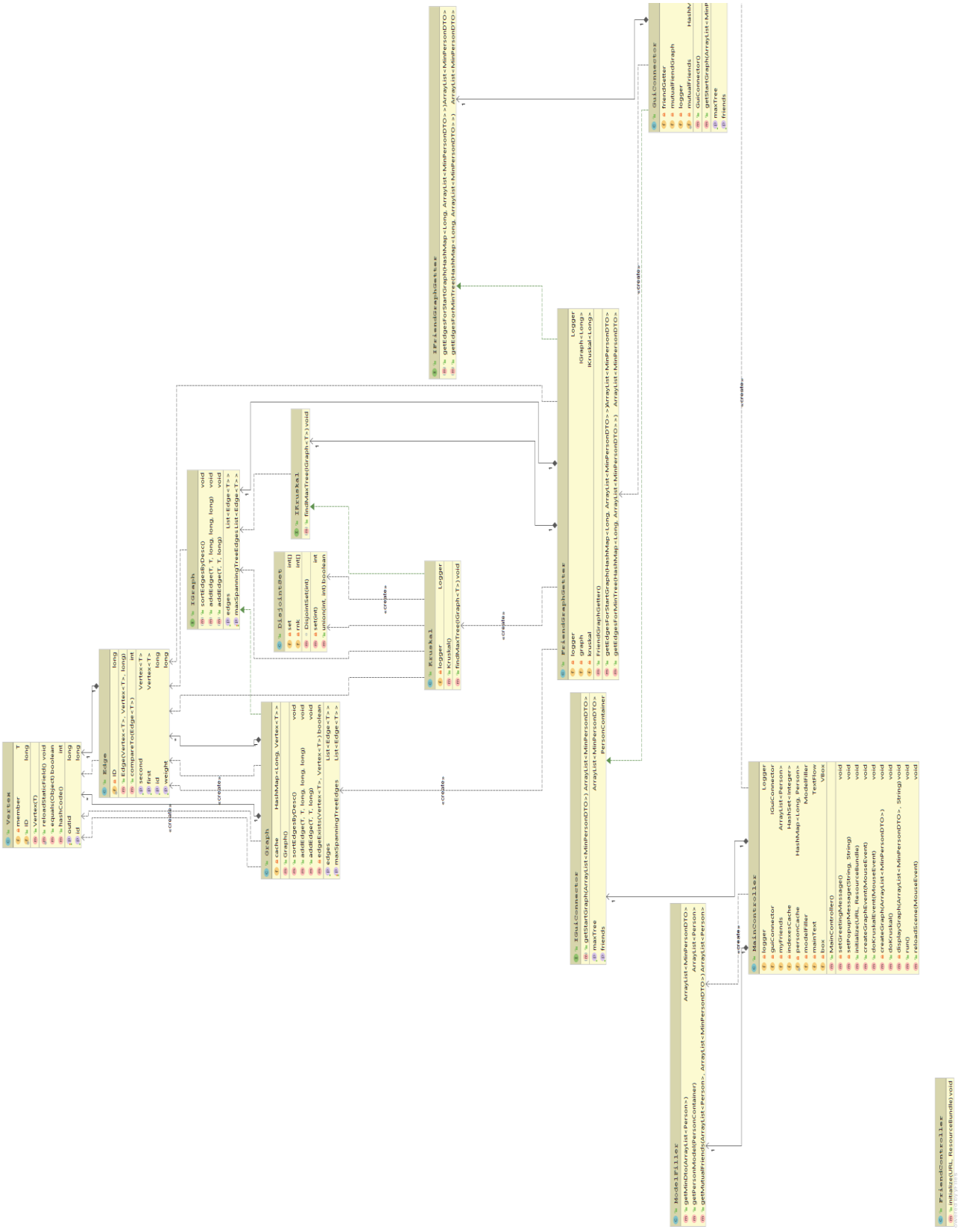

Рисунок 8

ПРИЛОЖЕНИЕ Б

ТЕСТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА

Пошаговая проверка работы GUI представлена на рис.9-13.

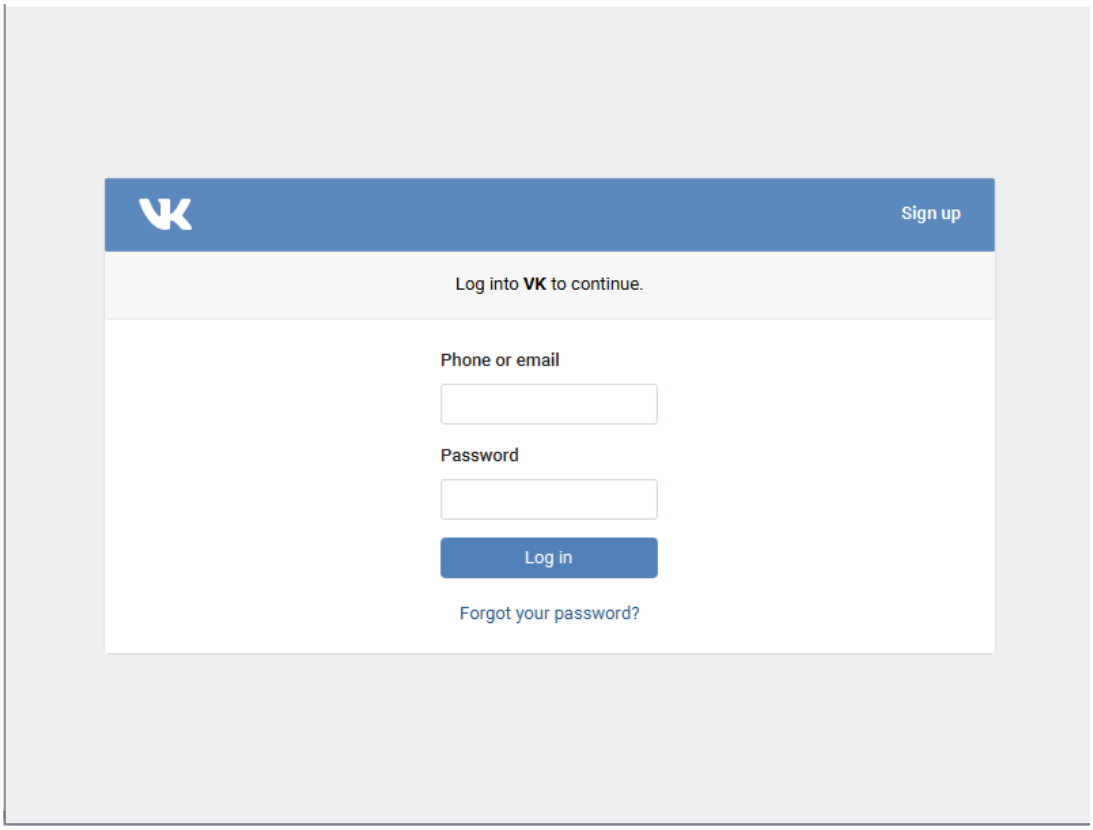

Рисунок 9 – Стартовое меню аутентификации ВК

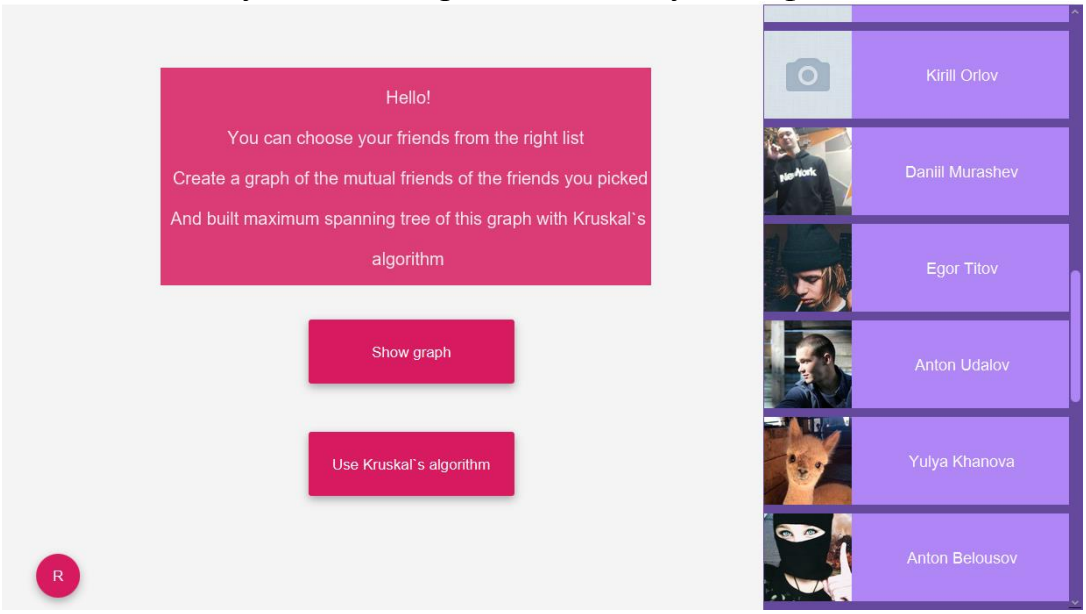

Рисунок 10 – Основное меню программы

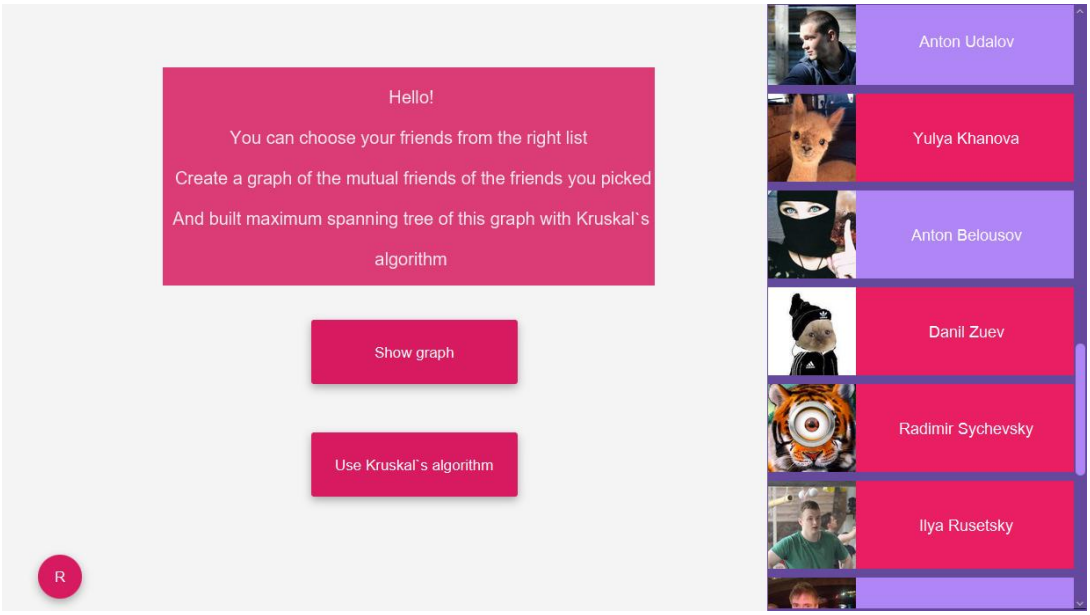

Рисунок 11 – Результат выбора друзей из списка справа

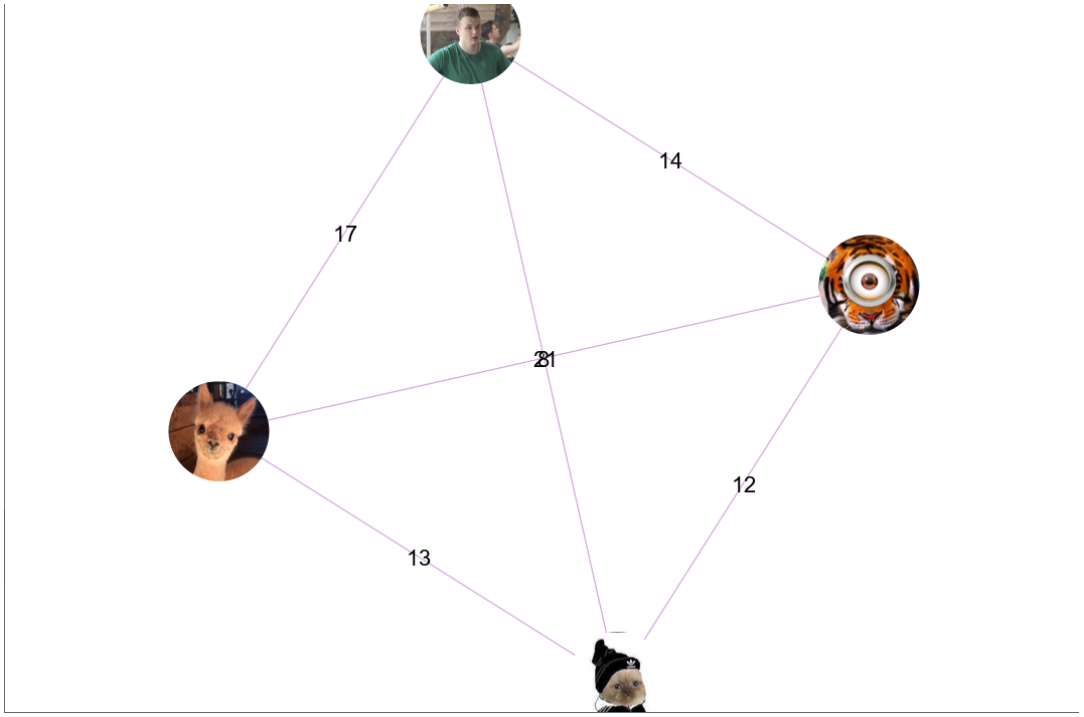

Рисунок 12 – Результат построения графа общих друзей между выбранными пользователями

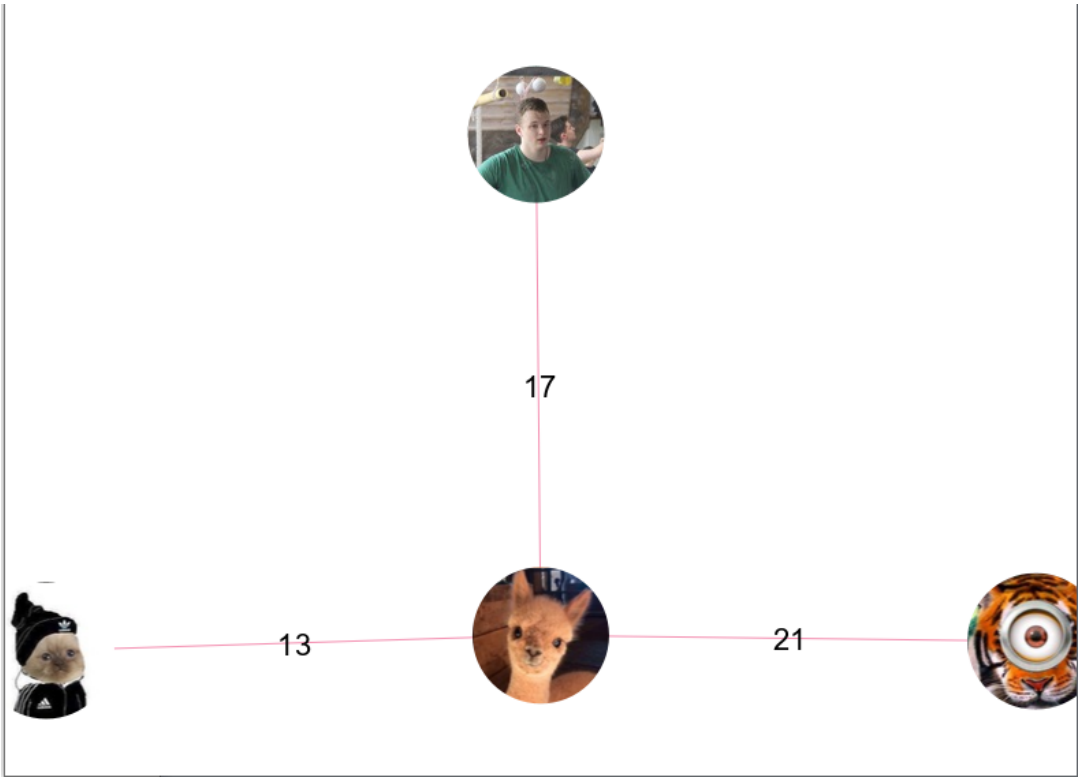

Рисунок 13 – Результат работы алгоритма Краскала, построение максимального остовного дерева.

ПРИЛОЖЕНИЕ В
ИСХОДНЫЙ КОД ПРОГРАММЫ