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

ОТЧЕТ

по учебной практике

Тема: Визуализация алгоритма Краскала

Студентка гр. 9303 _____ Зарезина Е.А.

Студентка гр. 9303 _____ Отмахова М.А.

Студентка гр. 9303 _____ Хафаева Н.Л.

Руководитель _____ Фиалковский М.С.

Санкт-Петербург
2021

ЗАДАНИЕ

НА УЧЕБНУЮ ПРАКТИКУ

Студентка Зарезина Е.А. группы 9303

Студентка Отмахова М.А. группы 9303

Студентка Хафаева Н.Л. группы 9303

Тема практики: визуализация алгоритма Краскала

Задание на практику:

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

Алгоритм: алгоритм Краскала.

Сроки прохождения практики: 01.07.2021 – 14.07.2021

Дата сдачи отчета: 00.07.2021

Дата защиты отчета: 00.07.2021

Студентка _____ Зарезина Е.А.

Студентка _____ Отмахова М.А.

Студентка _____ Хафаева Н.Л.

Руководитель _____ Фиалковский М.С.

АННОТАЦИЯ

Цель практики – научиться работать в команде и улучшить умение писать код в объектно-ориентированном стиле на языке программирования Java. Изучить основы языка программирования Java, а также средства разработки приложений с графическим интерфейсом на данном языке. В рамках практики выполняется мини-проект в команде, суть которого – реализация визуализатора графового алгоритма средствами языка Java. В процессе работы предстоит разработать прототип интерфейса, реализовать сам алгоритм, а также при помощи средств тестирования отладить разработанную программу. Нашей командой был выбран алгоритм Краскала.

SUMMARY

The goal of the practice is to learn how to work in a team and improve the ability to write code in an object-oriented style in the Java programming language. Learn the basics of the Java programming language, as well as tools for developing applications with a graphical interface in this language. As part of the practice, a mini-project is carried out in a team, the essence of which is the implementation of a graph algorithm visualizer using the Java language. In the process of work, it is necessary to develop a prototype of the interface, implement the algorithm itself, and also use testing tools to debug the developed program. Our team chose the Floyd-Warshall algorithm.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Основная цель практики – реализовать визуализатор алгоритма Краскала. Алгоритм предназначен для нахождения кратчайших путей между вершинами во взвешенном графе. Для реализации проекта, необходимо реализовать графический интерфейс, сам алгоритм и объединить данные наработки.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Исходные требования к программе

Приложение будет иметь графический интерфейс, через который будет возможно вводить веса ребер в матрицу смежности. Программа будет реализовывать алгоритм Краскала, и в результате будут выводиться веса ребер, образующих минимальное остовное дерево.

1.1.1 Требования к визуализации

В окне, появляющемся после запуска программы, вводится расстояния между вершинами. Далее, после нажатия кнопки “OK” появляется следующее окно ”Output” с информацией о введенных ранее весах ребер графа. Для вывода результатов работы программы требуется еще раз нажать “OK”, тогда появится окно со списком ребер, добавленных в минимальное остовное дерево и сумма весов данных ребер.

1.1.2 Требования к входным данным

Входные данные задаются вручную в матрицу смежности. Вес ребер – целое число, больше нуля. Количество ребер – 15.

Исходя из алгоритма Краскала, требования к входным данным следующие: веса рёбер следует подавать неотрицательными (алгоритм не предназначен для работы с отрицательными рёбрами) и целыми.

Окно для ввода матрицы смежности выглядит следующим образом:

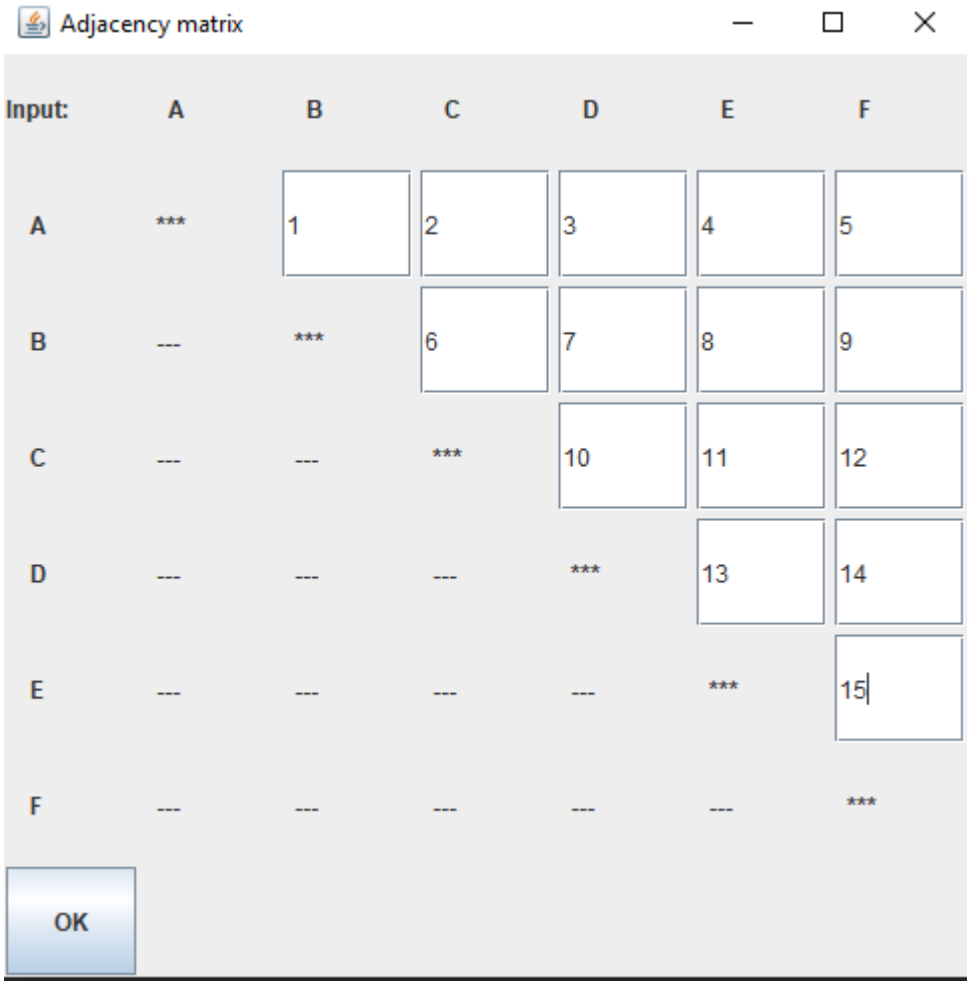

Рисунок 1 - Окно ввода

1.1.3 Требования к архитектуре

Данный проект выполняется средствами языка программирования Java. Для визуализации используется библиотека Swing. Данная библиотека была выбрана потому, что предоставляет большой набор связанных с ней библиотек, которые в свою очередь позволяют визуализировать графы.

В реализации алгоритма Краскала граф хранится в массиве и все действия производятся с ним.

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В БРИГАДЕ

2.1. План разработки

До 01.07.2021 – Распределение по бригадам и выбор темы мини-проекта

До 07.07.2021 – Сдача вводного задания

До 07.07.2021 – Согласование спецификации. Создание прототипа графического интерфейса.

До 10.07.2021 – Сдача второго этапа

До 14.07.2021 – Сдача финальной версии мини-проекта

2.2. Распределение ролей в бригаде

Зарезина Екатерина – Лидер, Фронтенд , Алгоритмист

Отмахова Мария –Документация, Тестировщик

Хафаева Наиля – Алгоритмист, Тестировщик

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

3.1. Структуры данных

В работе используются такая структура данных, как массив объектов типа Edge (один элемент массива – это две вершины и ребро между ними.

Далее действия, необходимые для реализации алгоритма, выполняются над данным массивом. Массив сортируется по полю int w (вес ребра).

3.2. Основные методы

Классы, используемые в работе:

- Edge

  Класс, реализующий граф. Объект класса – две вершины и ребро между ними.

- DSF

  Класс, реализующий систему непересекающихся множеств. Нужен, чтобы проверять, не образуется ли цикл в остовном дереве при добавлении ребра в минимальное остовное дерево. Объект данного класса – массив int[] set, содержащий номера множеств и массив int[] rnk, содержщий ранг каждого дерева.

- SimpleGUI

  Класс, реализующий графический интерфейс алгоритма. В данном классе происходит сортировка массива объектов класса Edge по полю int w (вес ребра) для дальнейшего добавления ребер в минимальное остовное дерево.

UML-диаграмма классов представлена на рис. 2.

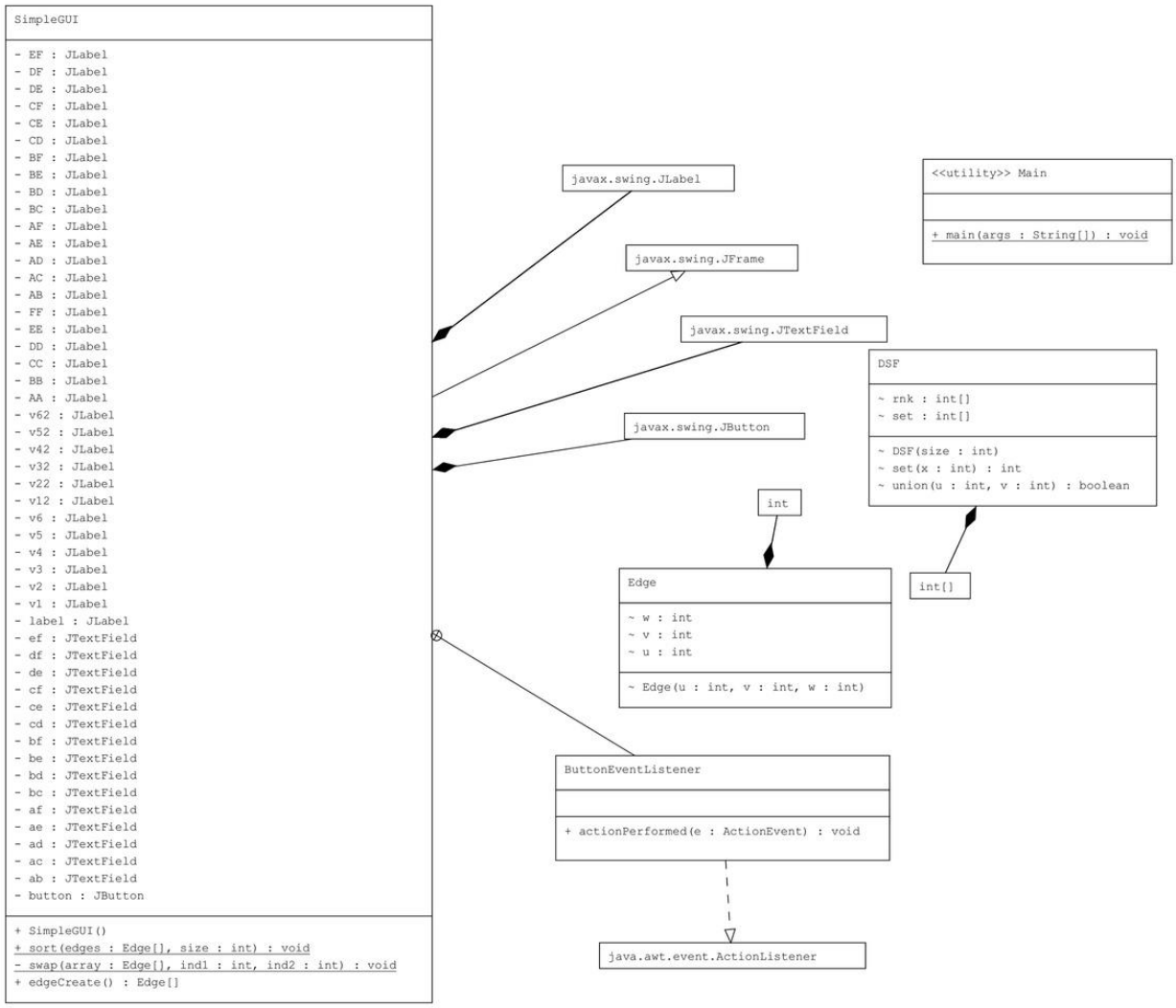

Рисунок 2 - UML-диаграмма классов

4. ТЕСТИРОВАНИЕ

4.1. Тестирование графического интерфейса

Было проведено тестирование графического интерфейса. Работа графического интерфейса соответствует ожиданиям.

4.2. Тестирование итогового проекта

Было проведено тестирование итогового проекта. Результаты тестирования представлены ниже.

Тест 1.

Входные данные представлены на рис. 3.

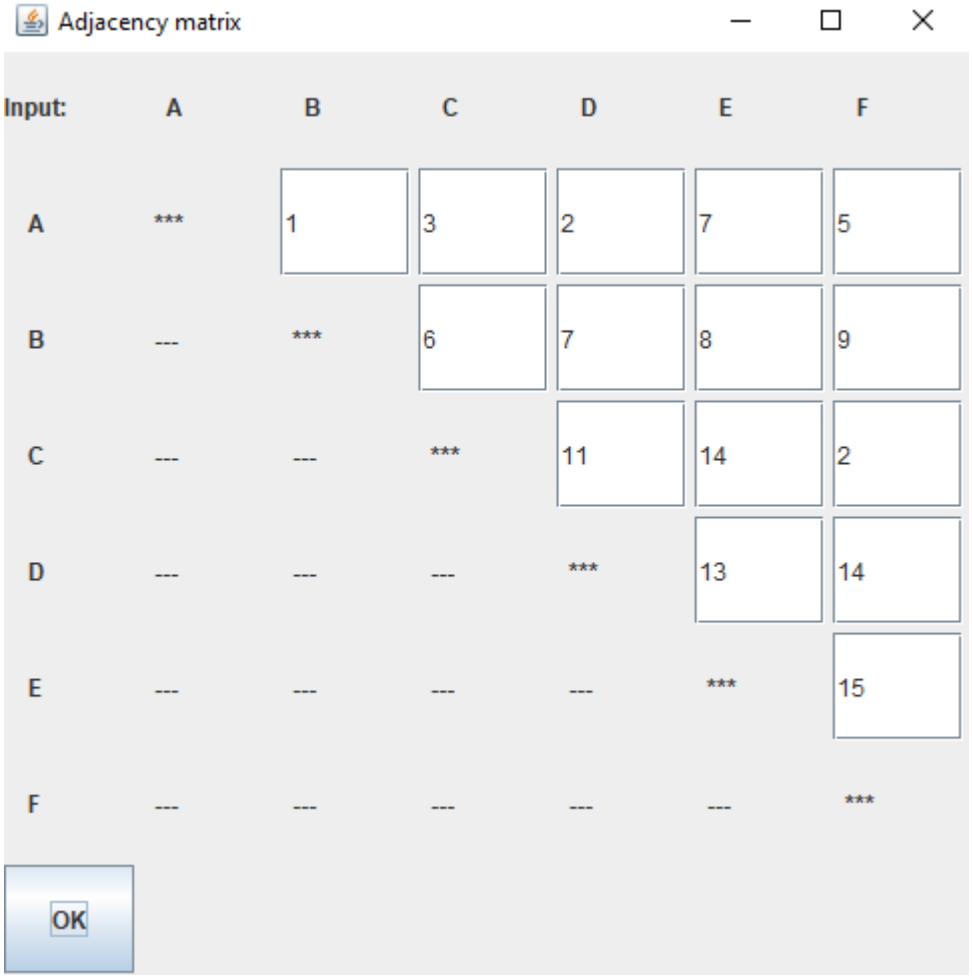

Рисунок 3 - Входные данные. Тест 1.

Далее программа выводит окно, в котором указаны все ребра графа с соответствующими им весами. Окно представлено на рис.4.

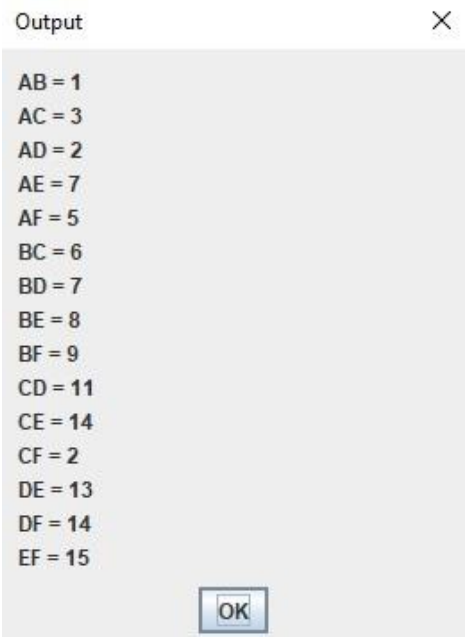

Рисунок 4 - Окно "Output"

После чего программа выводит окно с результатом. Окно представлено на рис. 5.

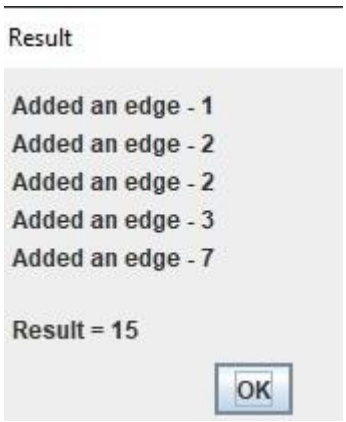

Рисунок 5 - Результат работы программы

Корректность работы программы была проверена с помощью сайта https://graphonline.ru. Был введен граф, изображенный на рис.6. Результат работы алгоритма на ресурсе изображен на рис.7.

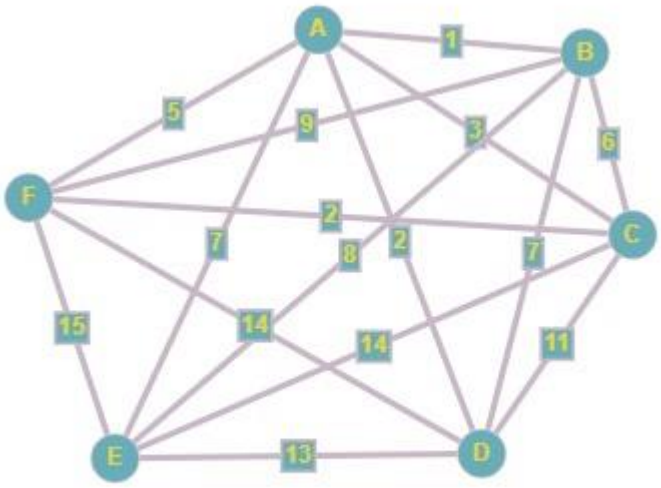

Рисунок 6 - Исходный граф

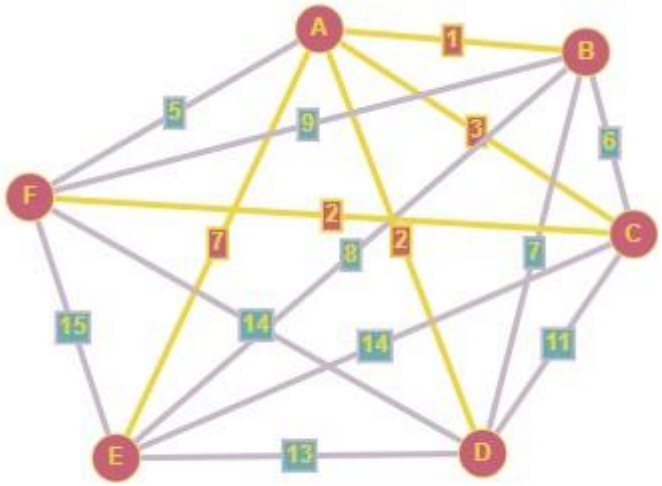

Рисунок 7 - Результат работы

Результат работы программы совпадает с результатом, получившимся при проверке.

Тест 2.

Входные данные представлены на рис. 8.

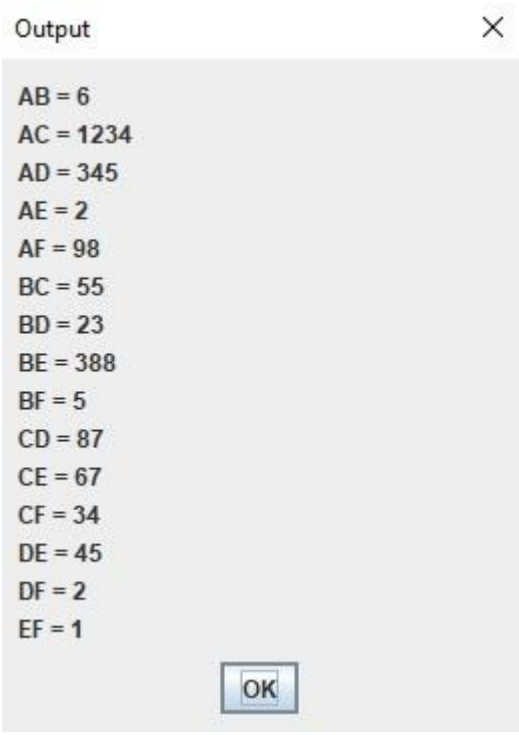

Рисунок 8 - Входные данные. Тест 2.

После чего программа выводит окно с результатом. Окно представлено на рис. 9.

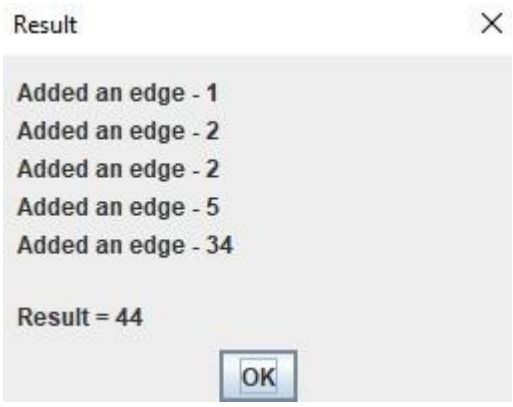

Рисунок 9 - Результат работы программы

Корректность работы программы была проверена с помощью сайта https://graphonline.ru. Был введен граф, изображенный на рис.10. Результат работы алгоритма на ресурсе изображен на рис.11.

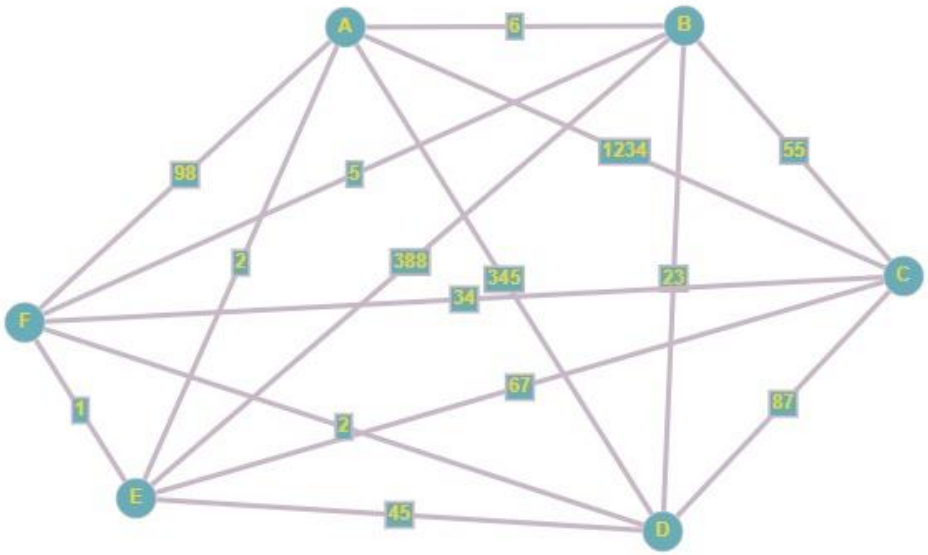

Рисунок 10 - Исходный граф

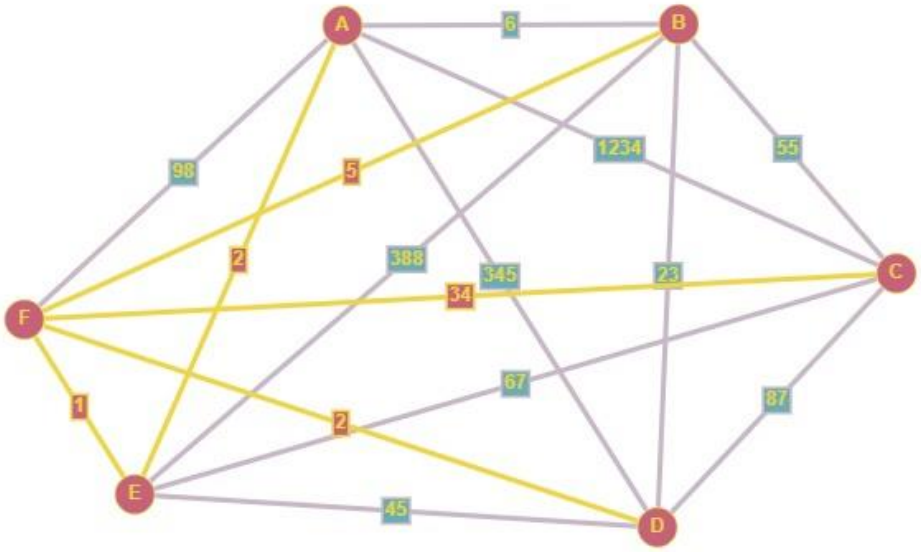

Рисунок 11 - Результат работы

Результат работы программы совпадает с результатом, получившимся при проверке.

ЗАКЛЮЧЕНИЕ

В течение практики был изучен язык программирования Java.

В результате учебной практики был разработан мини-проект, реализующий алгоритм Краскала и визуализацию данного алгоритма.