Министерство образования и науки Российской федерации

Санкт-Петербургский государственный электротехнический университет

“ЛЭТИ”

кафедра математического обеспечения и применения ЭВМ

**Спецификация**

**Преподаватель: Фирсов М.А.**

Факультет: КТИ

Выполнили:

|  |
| --- |
| Маратов А. |
| Кожемякин К.Г. |
| Лянгузов Н.Д. |

.

Санкт-Петербург

2017

**План работы:**

**23.06.2016** Встреча группы для обсуждения алгоритмов, создания репозитория и четкого разделения труда.

**26.06.2016** Создание Прототипа(отдельно работающий графический интерфейс и алгоритм).

**28.06.2016** Попытка соединения между собой Алгоритма и граф. интерфейса. Необходимо, чтобы присутствовал хотя бы примитивный способ задания графа (с граф. выводом на экран) и вывод результата в интерфейс(без демонстрации работы алгоритма).

**30.06.2016** Графическая демонстрация работы алгоритма, с реализованным алгоритмом выделения различных видов ребер, и несколькими способами задания графа пользователем(в том числе с случайной генерацией).

**Разделение труда:**

**Группа :** Обсуждение выбора алгоритма и тонкостей его реализации

Алмас: Разработка алгоритма на Java

Project manage (слежение за планом, создание репозитория)

Николай: Создание интерфейса и графического способа задания графа, визуализация различных видов ребер

Кирилл: Графическая визуализация алгоритма и создание ввода графа из файла и генерации графа

**Список терминов:**

**Граф** — основной объект изучения математической теории графов, совокупность непустого множества вершин и наборов пар вершин (связей между вершинами).

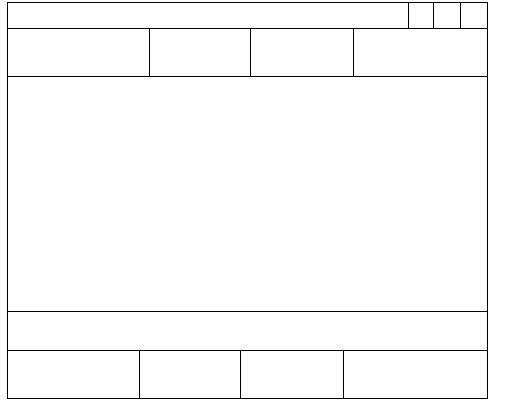
**Остовное дерево** графа состоит из минимального подмножества рёбер графа, таких, что из любой вершины графа можно попасть в любую другую вершину, двигаясь по этим рёбрам.

**Минимальное остовное дерево** (или **минимальное покрывающее дерево**) в связанном взвешенном [неориентированном графе](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84) — это [остовное дерево](https://ru.wikipedia.org/wiki/%D0%9E%D1%81%D1%82%D0%BE%D0%B2%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE" \o "Остовное дерево) этого графа, имеющее минимальный возможный вес, где под весом дерева понимается сумма весов входящих в него рёбер.

**Алгоритм Крускала:**

Вначале текущее множество рёбер устанавливается пустым. Затем, пока это возможно, проводится следующая операция: из всех рёбер, добавление которых к уже имеющемуся множеству не вызовет появление в нём цикла, выбирается ребро минимального веса и добавляется к уже имеющемуся множеству. Когда таких рёбер больше нет, алгоритм завершён. Подграф данного графа, содержащий все его вершины и найденное множество рёбер, является его [остовным деревом](https://ru.wikipedia.org/wiki/%D0%9E%D1%81%D1%82%D0%BE%D0%B2%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE" \o "Остовное дерево) минимального веса.

**Сложность:**  O(E \* log(E))

**Интерфейс:** 

Считывание графа из файла

Случайная генерация графа

Визуализация графа на текущем шаге алгоритма

Вывод сообщения об ошибке

Отобразить следующий шаг алгоритма

Отобразить конечный результат

Описание полей интерфейса:

Кнопки

1. Считывание графа из файла: считывает данные с файла и отображает граф (или выводит сообщение об ошибки считывания).
2. Случайная генерация графа: случайным образом генерирует граф
3. Отобразить следующий шаг алгоритма: выполняет один шаг алгоритма и отображает граф (рассматриваемое на данном шаге ребро окрасится в зелёный, если оно не входит в цикл, в красный, если оно образует цикл с уже рассмотренными рёбрами)
4. Отобразить конечный результат: отображает конечный результат работы алгоритма

**Ввод данных:**

Ввод данных производится через текстовый документ.

Формат ввода:

|  |  |  |  |
| --- | --- | --- | --- |
| Строка № | Значение | | |
| 1 | n-число вершин графа | | |
| 2 | k-число рёбер графа | | |
| 3 | l-вершина u | m-вершина v | p-вес ребра |
| … | … | | |
| 3+k |  |  |  |

Символ разделения строк: ‘\n’

Символ разделения значений в одной строке: ‘ ’

Здесь следует учесть все возможные ошибки, которые могут возникнуть при вводе данных:

1. n, k, l, m, p – целые положительные числа
2. k <= n(n-1)/2
3. попытка создать ребро между вершинами, у которых уже есть ребро
4. строк в файле меньше, чем 3+k
5. считанный граф не связный

**Пример:**

|  |  |
| --- | --- |
| **Изображение** | **Описание** |
| [Kruskal Algorithm 1.svg](https://commons.wikimedia.org/wiki/File:Kruskal_Algorithm_1.svg?uselang=ru) | Ребра **AD** и **CE** имеют минимальный вес, равный 5. Произвольно выбирается ребро **AD** (выделено на рисунке). |
| [Kruskal Algorithm 2.svg](https://commons.wikimedia.org/wiki/File:Kruskal_Algorithm_2.svg?uselang=ru) | Теперь наименьший вес, равный 5, имеет ребро **CE**. Так как добавление **CE** не образует цикла, то выбираем его в качестве второго ребра. |
| [Kruskal Algorithm 3.svg](https://commons.wikimedia.org/wiki/File:Kruskal_Algorithm_3.svg?uselang=ru) | Аналогично выбираем ребро **DF**, вес которого равен 6. |
| [Kruskal Algorithm 4.svg](https://commons.wikimedia.org/wiki/File:Kruskal_Algorithm_4.svg?uselang=ru) | Следующие ребра — **AB** и **BE** с весом 7. Произвольно выбирается ребро **AB**, выделенное на рисунке. Ребро **BD** выделено красным, так как уже существует путь (зелёный) между **B** и **D**, поэтому, если бы это ребро было выбрано, то образовался бы цикл **ABD**. |
| [Kruskal Algorithm 5.svg](https://commons.wikimedia.org/wiki/File:Kruskal_Algorithm_5.svg?uselang=ru) | Аналогичным образом выбирается ребро **BE**, вес которого равен 7. На этом этапе красным выделено гораздо больше ребер: **BC**, потому что оно создаст цикл **BCE**, **DE**, потому что оно создаст цикл **DEBA**, и **FE**, потому что оно сформирует цикл **FEBAD**. |
| [Kruskal Algorithm 6.svg](https://commons.wikimedia.org/wiki/File:Kruskal_Algorithm_6.svg?uselang=ru) | Алгоритм завершается добавлением ребра **EG** с весом 9. [Минимальное остовное дерево](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BD%D0%B8%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BE%D1%81%D1%82%D0%BE%D0%B2%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE) построено. |