**Техническое задание к ЛР 4**

Задача: Написать программу на C++, реализующую алгоритмы (поиска) на графах.

В программе должен быть реализован, по меньшей мере, один алгоритм, реализующей решение заданной задачи с помощью графовых структур данных.

При реализации необходимо придерживаться принципа общности: алгоритмы и структуры данных должны быть достаточно общими для решения задач данного класса (в разумных пределах). Например, в некоторых задачах в роли весов дуг могут выступать как целые числа, так и вещественные и дата/время.

Основные реализованные алгоритмы необходимо покрыть тестами.

**Структура данных:**

1. Неориентированный граф
2. Ориентированный граф

**Алгоритм:**

1. Поиск остова графа;
2. Поиск компонент связности неориентированного графа;
3. Поиск компонент сильной связности ор-графа;
4. Поиск пути с наибольшей пропускной способностью:

* Поиск одного наилучшего пути,
* Поиск нескольких путей для достижения заданной суммарной пропускной способности

Баллы: 6 + 5 + 6 + 4 + 5,5 + 7 + 9 = 42,5

**Матрица смежности:**

* Проста в реализации.
* Занимает больше памяти, особенно для разреженных графов, где большинство рёбер отсутствуют.
* Дает быстрый доступ к наличию ребра между двумя вершинами (O(1)).
* Медленный при работе с большими графами из-за избыточности информации.

**Список смежности:**

* Экономит память для разреженных графов, так как хранит только фактические рёбра.
* Доступ к смежным вершинам занимает O(degree), где degree - степень вершины (количество смежных рёбер).
* Удобен для работы с разреженными графами, где многие вершины имеют малую степень.

Я использовал матрицу смежности.

Граф в программировании представляет собой абстрактную структуру данных, используемую для моделирования отношений между объектами. Он состоит из вершин (узлов) и рёбер (связей), соединяющих эти вершины. Графы применяются в различных областях, таких как логистика, сети, социальные сети, компьютерная графика и др.

**Графы можно представить двумя основными способами:**

1. Матрица смежности: Это двумерный массив, где значение graph[i][j] равно весу ребра между вершинами i и j. Если вершины не соединены, то значение может быть задано как бесконечность.
2. Список смежности: Для каждой вершины создается список, содержащий вершины, с которыми эта вершина соединена.

Теперь давайте рассмотрим несколько важных алгоритмов на графах:

**1. Остов графа:**

Остов графа — это подграф, содержащий все вершины исходного графа и являющийся деревом (не содержит циклов). Одним из методов поиска остова графа является алгоритм Прима или алгоритм Крускала.

Алгоритм Прима: На каждом шаге выбирается ребро с минимальным весом, соединяющее уже выбранные вершины и добавляется к остову.

Алгоритм Крускала: Ребра сортируются по весу, затем добавляются к остову в порядке возрастания веса, при этом избегаются циклы.

**2. Компоненты сильной связанности:**

Граф обладает компонентой сильной связанности, если для любых двух вершин u и v существуют пути из u в v и из v в u. Алгоритм Косарайю — один из методов нахождения компонент сильной связанности в ориентированных графах.

**3. Пропускная способность графа:**

Пропускная способность графа измеряет максимальный поток данных между двумя вершинами. Один из наиболее известных алгоритмов для решения задачи максимального потока — алгоритм Форда-Фалкерсона. Вес ребер в графе представляет собой пропускную способность.

**Неориентированный граф:**

Неориентированный граф - это граф, у которого рёбра не имеют направления. В неориентированном графе ребро между вершинами (a, b) равносильно ребру между вершинами (b, a). Вершины в неориентированном графе связаны рёбрами, но порядок их расположения не имеет значения.

**Остовное дерево:**

Остовное дерево неориентированного графа - это подграф, который является деревом (связным ациклическим графом) и включает все вершины оригинального графа. Он может быть получен удалением рёбер из графа таким образом, чтобы он остался связным, но не содержал циклов. Остовные деревья могут использоваться для анализа связности графа и поиска минимальных путей.

**Компоненты связности:**

В неориентированном графе компонентой связности называется максимальный по включению подграф, в котором любая пара вершин достижима друг из друга. Если в графе нет мостов (рёбер, удаление которых разъединяет граф), то он может состоять из одной компоненты связности. Если есть мосты, то каждый мост разъединяет граф на две компоненты связности.

**Пропускная способность графа:**

В неориентированных графах пропускная способность рёбер обычно связана с сетями передачи данных, транспортными сетями и т.д. Как и в ориентированных графах, пропускная способность ребра представляет собой меру того, сколько информации или данных может быть передано через это ребро за единицу времени.

Пропускная способность графа определяется как минимальная пропускная способность рёбер, образующих путь между двумя вершинами. Веса рёбер могут представлять стоимость передачи данных, расстояние, время и т.д. Пропускная способность пути в графе определяется наименьшей пропускной способностью рёбер этого пути.

**В игре крестики-нолики**, я использовал неориентированный граф. Граф строился на основе шагов крестиков и ноликов. То есть использовался для отслеживания ходов.