Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский химико-технологический университет имени Д.И. Менделеева»

Кафедра информационных компьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

Выполнил студент группы КС-36 Полковникова Д.Д.

Ссылка на репозиторий: https://github.com/MUCTR-IKT-CPP/Polkovnikova\_CS-36

Приняли: Пысин Максим Дмитриевич

Краснов Дмитрий Олегович

Лобанов Алексей Владимирович

Крашенинников Роман Сергеевич

Дата сдачи: 17.03.2025

Оглавление

[Описание задачи. 2](#_Toc192415473)

[Описание метода/модели. 3](#_Toc192415474)

[Выполнение задачи. 4](#_Toc192415475)

[Заключение. 11](#_Toc192415476)

# Описание задачи.

В рамках лабораторной работы необходимо реализовать генератор случайных графов, генератор должен содержать следующие параметры:

* Максимальное/Минимальное количество генерируемых вершин
* Максимальное/Минимальное количество генерируемых ребер
* Максимальное количество ребер связанных с одной вершины
* Генерируется ли направленный граф
* Максимальное количество входящих и выходящих ребер

Сгенерированный граф должен быть описан в рамках одного класса(этот класс не должен заниматься генерацией), и должен обладать обязательно следующими методами:

* Выдача матрицы смежности
* Выдача матрицы инцидентности
* Выдача список смежности
* Выдача списка ребер

В качестве проверки работоспособности, требуется сгенерировать 10 графов с возрастающим количеством вершин и ребер(количество выбирать в зависимости от сложности расчета для вашего отдельно взятого ПК). На каждом из сгенерированных графов требуется выполнить поиск кратчайшего пути или подтвердить его отсутствие из точки А в точку Б, выбирающиеся случайным образом заранее, поиском в ширину и поиском в глубину, замерев время требуемое на выполнение операции. Результаты замеров наложить на график и проанализировать эффективность применения обоих методов к этой задаче.

# Описание метода/модели.

Под графом в математике понимается абстракция реальной системы объектов безотносительно их природы, обладающих парными связями.

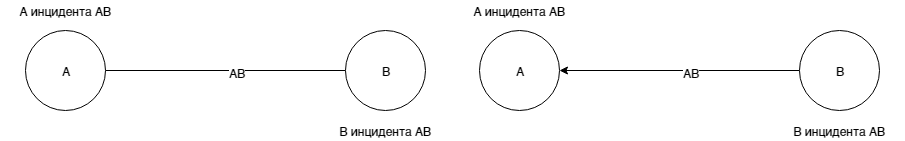
**Вершина графа** – это некоторая точка связанная с другими точками

**Ребро графа** – это линия соединяющая две точки и олицетворяющая связь между ними

**Граф** – это множество вершин соединённых друг с другом произвольным образом множеством ребер

Граф описывает:

* Взаимоотношение между людьми(Социальные связи)
* Иерархические отношения(Подчиненность людей, подразделений и прочего)
* Пути перемещения в любой местности(Карта метро, сеть дорог)
* Взаимозависимости поставщиков услуг или товаров(Поставщики для сборки одного автомобиля)
* Распределенные системы(Любая микросервисная архитектура)
* Распределенные данные(Реляционные базы данных)



Связанную с каким либо ребром вершину называются инцидентной, это такая вершина которая каким либо образом принадлежит ребру.

Соседними или смежными называются те вершины которые соединены между собой ребром.

Графы бывают простыми и сложными. При этом простыми называют такие графы которые не содержат в себе петли и кратные ребра, которые требуют большей тщательности при разработке и реализации алгоритмов на них.

Обход графа в ширину и в глубину имеет общие цели, но разные характеристики, вот несколько из них:

1. Поиск в ширину и в глубину — это и есть обход графа.
2. Поиск в глубину — это поиск по ребрам графа туда-обратно, а поиск в ширину — это плавный «обход по соседям».
3. В DFS главное — стек, а в BFS главное — очередь.
4. Результат алгоритма поиска в глубину — это некий маршрут, который открывается от стартовой вершины и до искомой. А в алгоритме поиска в ширину маршрут не всегда является результатом.
5. DFS является рекурсивным алгоритмом, а BFS — нет.

**Представления графа в памяти:**

**1. Матрица смежности:**

o Двумерный массив, где элемент в строке i и столбце j равен 1 (или весу ребра), если существует ребро между вершинами i и j, и 0 в противном случае. o Подходит для плотных графов, когда много рёбер.

**2. Список смежности:**

o Это массив или список, где для каждой вершины хранится список её соседей. o Подходит для разреженных графов, когда количество рёбер значительно меньше количества возможных рёбер.

**3. Список рёбер:**

o Это список всех рёбер графа, где каждое ребро представлено парой вершин, которые оно соединяет. o Подходит для хранения рёбер без необходимости в быстром поиске соседей.

**Алгоритмы работы с графами:**

1. Поиск в глубину (DFS) — исследует граф, начиная с вершины и углубляясь по пути, пока не встретит тупик.

2. Поиск в ширину (BFS) — исследует граф, посещая вершины на одном уровне (сначала все вершины, соседние с текущей, потом все вершины, соседние с ними)..

# Выполнение задачи.

**1. Класс Graph**:

* + Содержит матрицу смежности для хранения графа.
  + Методы для добавления ребер, получения матриц смежности и инцидентности, списков смежности и ребер.
  + Реализованы алгоритмы BFS и DFS для поиска пути между вершинами.

**2. Класс GraphGenerator**:

* + Генерирует случайные графы на основе заданных параметров.
  + Использует генерацию случайных чисел для выбора количества вершин и ребер.

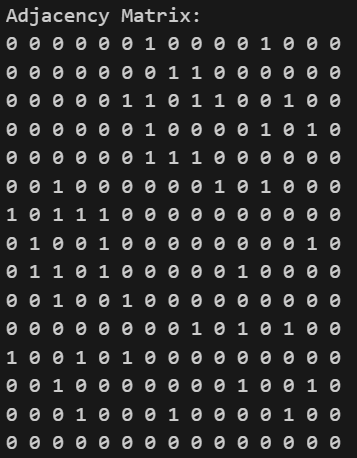
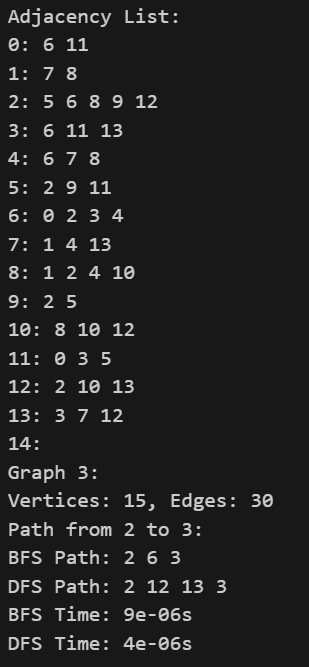
**3. Тестирование**:

* + Генерируется 10 графов с возрастающим количеством вершин и ребер.
  + Для каждого графа случайным образом выбираются вершины start и end.
  + Выполняется поиск пути с использованием BFS и DFS, замеряется время выполнения.

**Запуск и проверка:**

* Программа выводит матрицы смежности, списки смежности, пути, найденные BFS и DFS, а также время выполнения для каждого графа.
* Вы можете изменить параметры генерации (например, количество вершин и ребер) в конструкторе GraphGenerator.

**Вывод об одном графе:**

# Заключение.