**Содержание**

[Введение 3](#_Toc197717353)

[Постановка задачи «Вариант 23.Оптимальной размещение столовой в университете» 4](#_Toc197717354)

[Ход решения задачи «Вариант 23.Оптимальное размещение столовой в университете» 5](#_Toc197717355)

[Заключение 9](#_Toc197717356)

[Список литературы 10](#_Toc197717357)

**Введение**

Дискретная математика – это важная область математики, которая исследует отдельные, не связанные между собой объекты и отношения между ними. В отличие от непрерывной математики, где рассматриваются, например, бесконечные множества чисел, дискретная математика изучает конечные или счетные множества. Она играет фундаментальную роль в информатике и вычислительной технике, предоставляя теоретические основы для решения различных задач. Благодаря дискретной математике мы можем: Анализировать данные: Разрабатывать методы обработки и анализа информации, представленной в дискретном виде (например, тексты, изображения, базы данных). Строить алгоритмы: Создавать эффективные и корректные компьютерные программы, определяющие последовательность шагов для решения конкретных задач. Оптимизировать процессы: Находить наилучшие решения в различных задачах, например, оптимизировать маршруты доставки, расписание работы или использование ресурсов

Данная работа направлена на практическое применение знаний дискретной математики. Мы будем решать задачу выбора оптимального места для столовой, учитывая все остальные здания на территории. Задача заключается в минимизации максимального расстояния от столовой до любого другого здания. Представьте себе карту: здания – это точки (вершины графа), а дороги между ними – линии (ребра графа) с указанной длиной. Нам нужно найти "центр" этого графа – точку, наиболее близко расположенную ко всем остальным. Для решения этой задачи мы воспользуемся алгоритмом Флойда, реализовав его на языке программирования C#. Алгоритм Флойда будет наиболее эффективен для нахождения кратчайших путей между всеми парами вершин в графе, что позволит нам точно определить оптимальное расположение столовой. В итоге, работа продемонстрирует не только знание теории, но и умение применять её на практике с помощью программирования.

**Постановка задачи «Вариант 23. Оптимальной размещение столовой в университете»**

В одном из зданий университета решили открыть столовую. Для этих целей требуется выбрать одно из зданий, в котором и будет располагаться столовая. Чтобы студенты как можно меньше отвлекались от учёбы, было решено выбрать такое здание, чтобы максимальное расстояние от него до всех остальных зданий было как можно меньше. Необходимо разработать программу, которая определяет здания, для размещения столовой.

**Формат входных данных**

В первой строке входного файла находятся два целых числа N и M – количество зданий и количество дорог, соединяющих здания (1 ≤ N ≤ 100, 0 ≤ M ≤ N×(N−1)/2). Далее в M строках расположены описания дорог: 3 целых числа si, ei, li – здания, в которых начинается и заканчивается дорога и длина дороги соответственно (1 ≤ si, ei ≤ N, 0 ≤ li ≤ 100, дороги двунаправленные). Гарантируется, что между любыми двумя зданиями существует путь.

**Формат выходных данных**

В выходной файл вывести одно число – номер искомого здания. Если есть несколько зданий удовлетворяющих поставленным критериям, выбрать среди них здание с наименьшим номером.

**Ход решения задачи «Вариант 23. Оптимальной размещение столовой в университете»**

Для решения задачи мы начинаем с инициализации входных данных. Сначала считываем количество зданий, обозначаемое как N, а также количество дорог между ними, обозначаемое как M. Далее создаем матрицу расстояний размером N на N, которая будет использоваться для хранения минимальных расстояний между каждой парой зданий. При заполнении этой матрицы мы устанавливаем нули на диагонали, поскольку расстояние от здания до самого себя равно нулю, а для всех остальных элементов задаем значение бесконечности (INF), чтобы обозначить отсутствие прямого пути на начальном этапе.

Следующий шаг – заполнение графа на основе введенных данных о дорогах. Для каждой дороги обновляем соответствующие элементы в матрице расстояний, учитывая длину дороги между двумя зданиями. При этом если между некоторыми зданиями существует несколько дорог, мы фиксируем минимальное из известных расстояний, чтобы гарантировать, что матрица отражает самые короткие пути.

После подготовки матрицы применяем алгоритм Флойда, который позволяет найти кратчайшие пути между всеми парами вершин в графе. Это классический алгоритм, состоящий из тройного вложенного цикла, где перебираются все возможные промежуточные вершины. В процессе работы алгоритма мы обновляем значения в матрице расстояний, проверяя, может ли путь через промежуточное здание улучшить текущие известные расстояния между двумя другими зданиями. Таким образом, по окончании алгоритма матрица содержит минимальные расстояния между всеми парами зданий.

Наконец, переходим к поиску оптимального здания по заданному критерию. Для каждого здания вычисляем максимальное расстояние до всех остальных зданий, то есть находим, насколько далеко может быть самый удаленный объект от рассматриваемого здания. Затем выбираем то здание, для которого этот максимальный показатель минимален, что обеспечивает минимальное наибольшее расстояние до других зданий и обеспечивает лучшую доступность. При возникновении нескольких зданий с одинаковым значением максимального расстояния отдаем предпочтение зданию с меньшим номером согласно условию задачи.

Итоговый код программы представлен на рисунке 1.

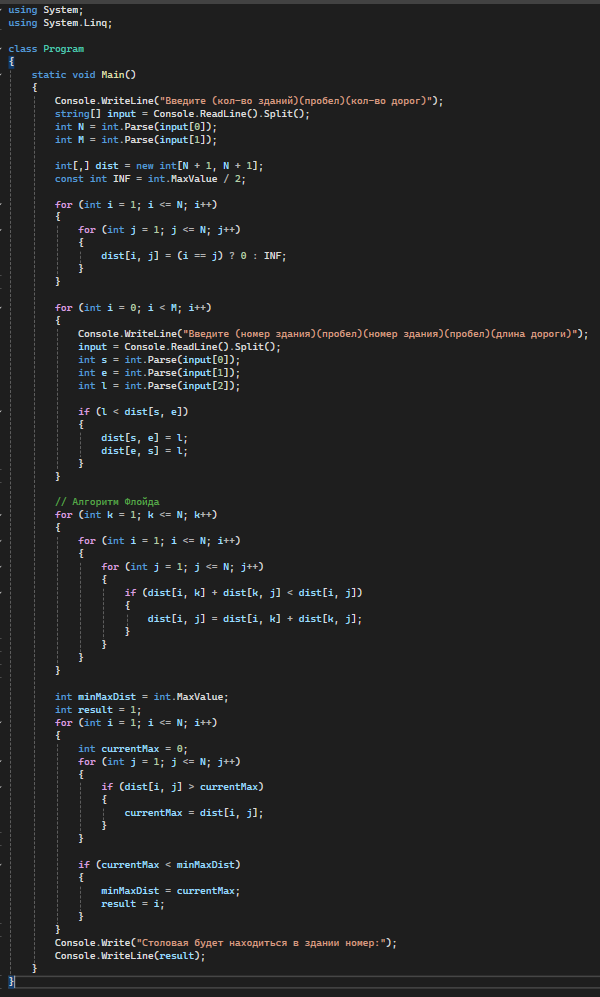


Рисунок №1 – Код программы.

Проверим правильность работы программы на тестах.

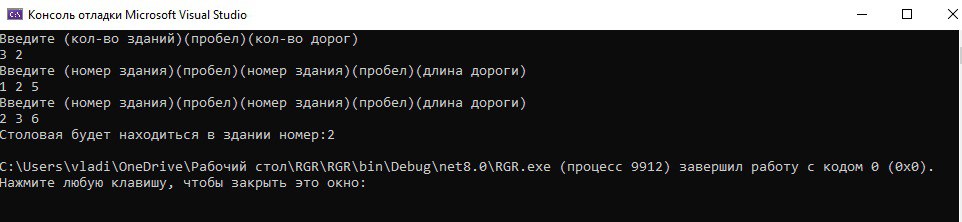
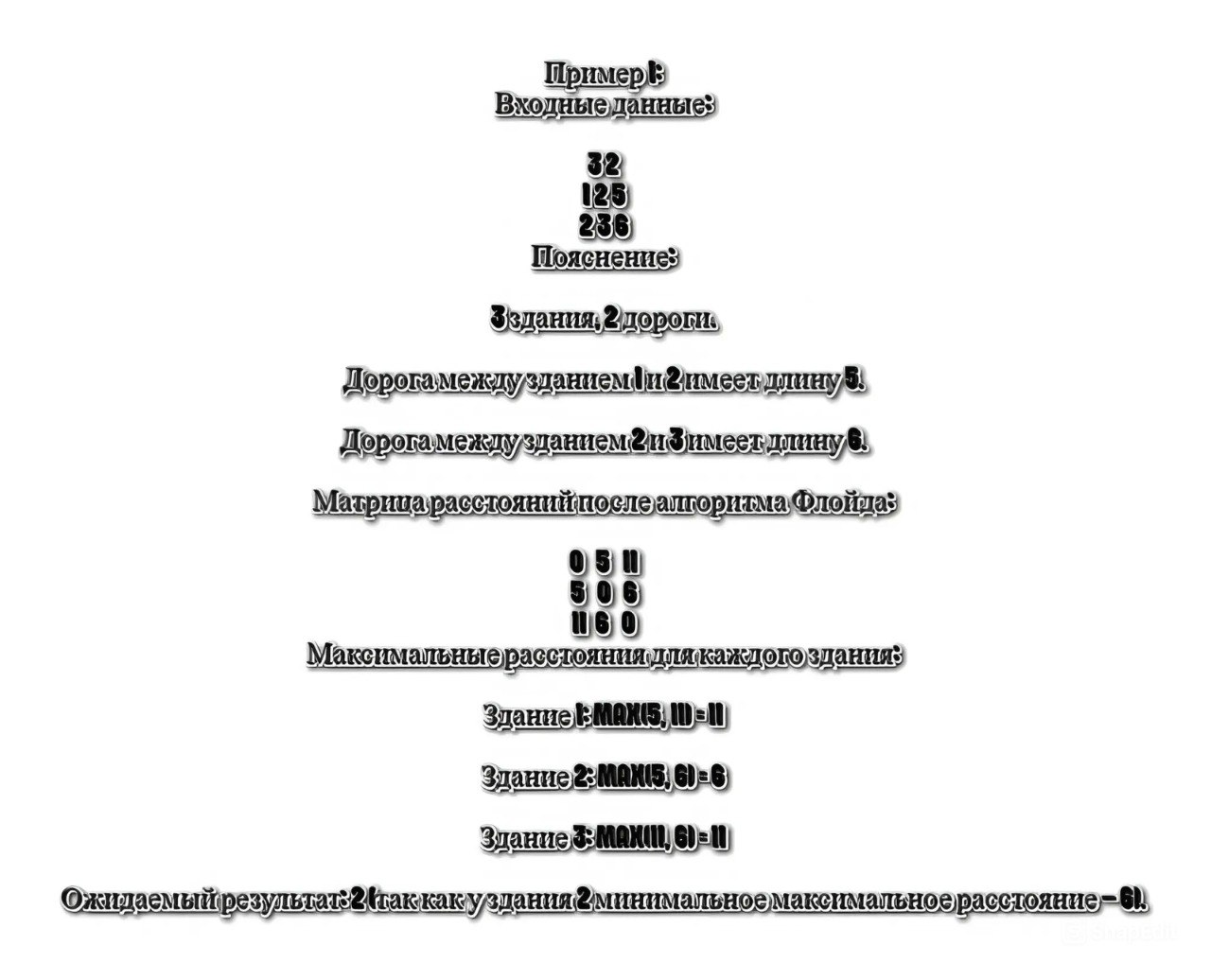


Рисунок №2 – Тест 1.

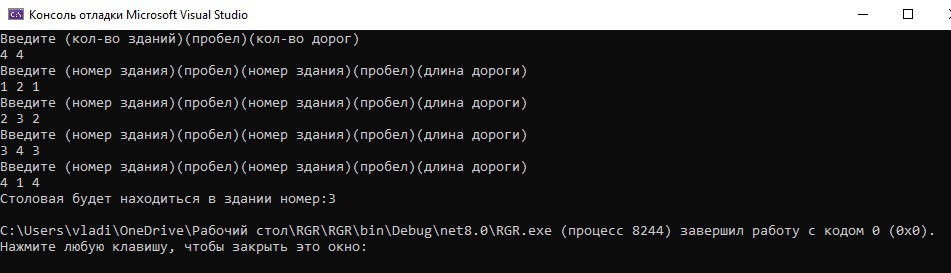
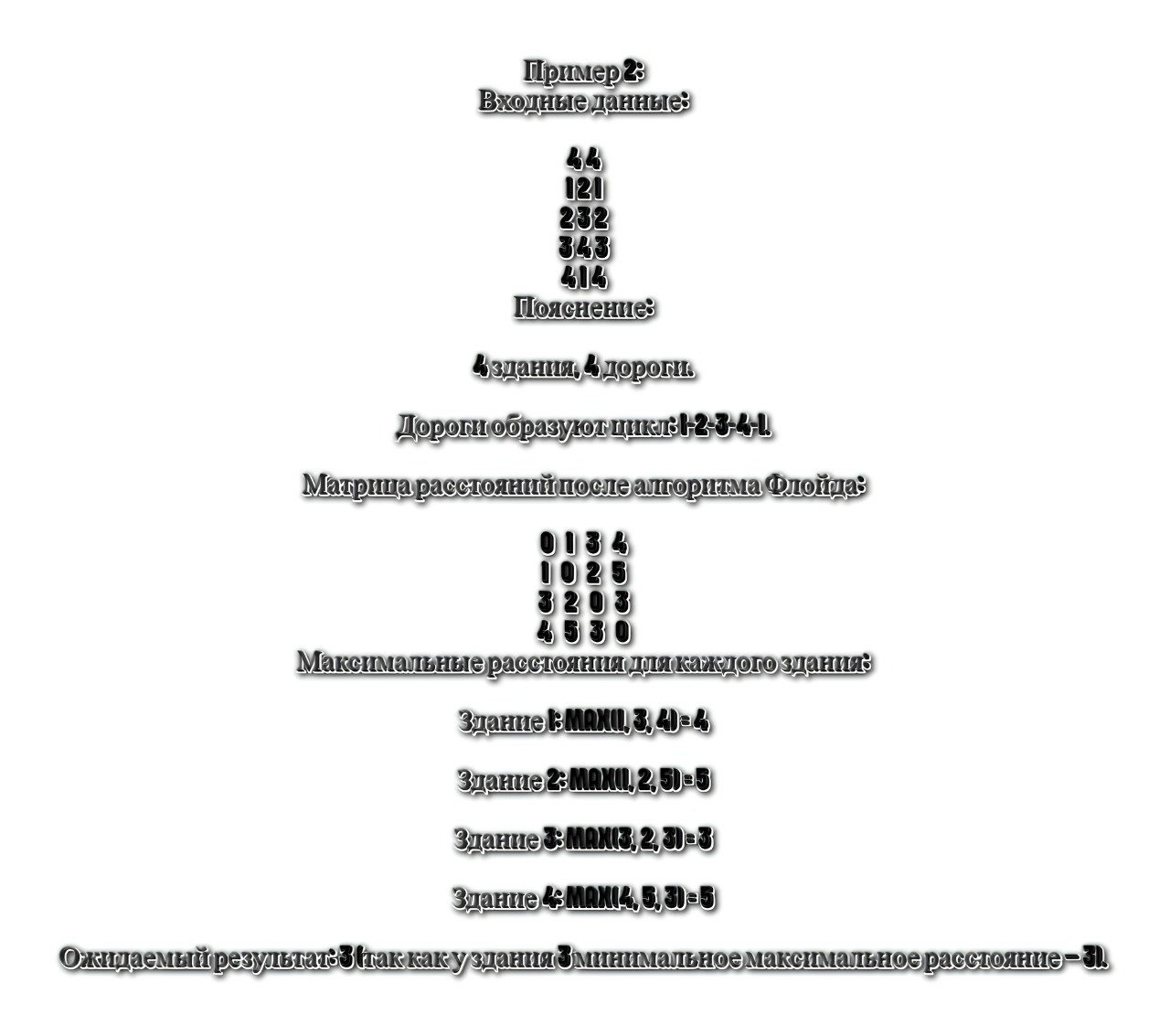


Рисунок №3 – Тест 2.

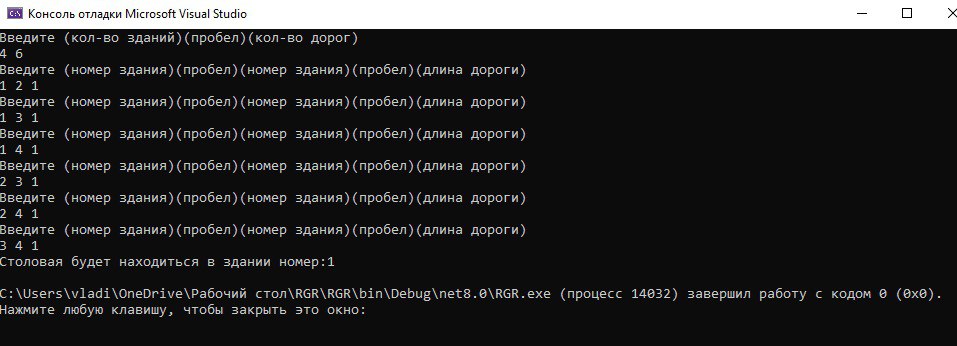


Рисунок №4 – Тест 3.

**Заключение**

В ходе выполнения расчётно-графической работы была решена задача оптимального размещения столовой в университете. Для этого требовалось определить здание, из которого максимальное расстояние до всех остальных зданий минимально.

Реализация алгоритма Флойда на языке C# позволила эффективно вычислить кратчайшие расстояния между всеми парами вершин графа. Этот алгоритм был выбран благодаря его простоте и высокой эффективности для графов с небольшим количеством вершин. В ходе работы были проанализированы входные данные, корректно построена матрица смежности, а также учтена двунаправленность дорог между зданиями.

Тестирование программы на различных наборах данных подтвердило её работоспособность и корректность расчётов. Алгоритм успешно определил оптимальное здание для размещения столовой, учитывая критерий минимального максимального расстояния. В случае нескольких подходящих вариантов программа выбирает здание с наименьшим номером, что соответствует условию задачи.

Данная работа позволила закрепить навыки работы с графами, алгоритмами поиска кратчайших путей и обработки матричных структур данных. Применение алгоритма Флойда продемонстрировало его универсальность и удобство для решения задач, требующих анализа всех возможных путей в графе.

**Список литературы**

1. Калиберда Е. А., Федотова И. В. Применение алгоритмов на графах для решения прикладных задач – Издательство ОмГТУ, Омск, 2021 – 22.
2. [Алгоритм Флойда — Уоршелла — Википедия](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A4%D0%BB%D0%BE%D0%B9%D0%B4%D0%B0_%E2%80%94_%D0%A3%D0%BE%D1%80%D1%88%D0%B5%D0%BB%D0%BB%D0%B0).
3. [Алгоритм Флойда — Уоршелла / Хабр](https://habr.com/ru/articles/105825/).