|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8 «Графы» (ВАР. 6)**

Студент Смирнов Иван Владимирович

Группа ИУ7 – 32Б

Проверил Барышникова Марина Юрьевна

*2023 г.*

Оглавление

[Описание условия задачи 3](#_Toc153272719)

[Описание внутренних структур данных 5](#_Toc153272720)

[Описание алгоритмов 5](#_Toc153272721)

[Сравнение времени работы и ресурсных затрат 6](#_Toc153272722)

[Тесты 8](#_Toc153272723)

[Ответы на контрольные вопросы 10](#_Toc153272724)

[Вывод 12](#_Toc153272725)

# Описание условия задачи

Программа работает с типом данных «граф». Программа должна уметь обрабатывать ввод графа (входные данные выбираются на усмотрение программиста), программа должна выводить граф, представленный в виде множества вершин, и множества ребер, проходящих через определенные пары вершин (визуализация делается с использованием утилиты graphviz). Программа должна уметь находить максимальный путь среди всех возможных пар вершин графа и выводить его значение на экран.  
Описание технического задания

1. **Исходные данные и результат**

**Входные данные:**

Пользователь выбирает пункты меню с соответствующими действиями.

|  |
| --- |
| Меню:  1 - считать граф из файла  2 - вывести граф в graphviz  3 - найти максимальное расстояние между всеми парами вершин  0 - выход  Введите ваш выбор: |

При чтении данных из файла, входные данные имеют следующий вид:

На каждой строке файла должны быть 3 целых числа, записанных через пробел. 1-ое число – номер первой вершины в графе. 2-ое число – номер второй вершины в графе. 3-е число – значение ребра (расстояние) между всеми двумя вершинами.

Ограничения на данные из файла: номер вершины должен быть целым числом, отличным от нуля; значение ребра между вершинами должно быть неотрицательным (можно задать его значением 0, чтобы показать, что данная пара не связана никаким из ребер).

**Выходные данные:**

На экран выводятся результаты команд меню.

1. **Задачи, реализуемой программой**

**Программа умеет выводить граф с помощью утилиты** graphviz, умеет на основе данных из файла с названием <input.txt> строить граф, находить максимальное расстояние между всеми парами вершин.

**Способ обращения к программе**

**Команда для сборки проекта: make release  
Программа запускается через терминал командой: ./app.exe**

1. **Возможные аварийные ситуации и ошибки пользователя**
   1. **Попытка ввести не целое число при выборе пункта меню**

**На выходе сообщение: «Пожалуйста, введите целое число!»**

* 1. **Попытка ввести несуществующий номер пункта меню**

**На выходе сообщение: «Некорректный выбор!»**

* 1. **При попытке считать данные из файла, в файле находятся посторонние символы**

**На выходе сообщение: «Файл имеет некорректные данные!»**

* 1. **Файла с названием** “input.txt” **не существует**

**На выходе сообщение: «Ошибка! Не удалось открыть файл!»**

* 1. **Попытка вывести пустой граф**

**На выходе сообщение: «Граф пуст!»**

* 1. **В файле находится вершина с именем «0»**

**На выходе сообщение: «Граф задан неправильно! (номер вершины должен отличаться от нуля)»**

* 1. **В файле находятся два расстояния между одной парой вершин**

**На выходе сообщение: «Граф задан неправильно! (между одной парой вершин не должно быть 2 и более расстояний)»**

* 1. **В файле расстояние между парой вершин - отрицательное**

**На выходе сообщение: «Граф задан неправильно! (расстояние между вершинами должно быть положительным)»**

* 1. **Попытка найти максимальное расстояние в пустом графе**

**На выходе сообщение: «Граф пуст!»**

# **Описание внутренних структур данных**

Граф представлен в виде следующей структуры:

|  |
| --- |
| typedef struct {  int\* vertices;  int numVertices;  int\*\* adjacencyMatrix;  } graph\_t; |

graph\_t – структура для хранения графа.  
int\* vertices – массив, содержащий номера вершин графа

int numVertices – количество вершин в графе

int\*\* adjacencyMatrix – матрица смежности графа

# Описание алгоритмов

1. Пользователь вводит пункт меню как целое число.
2. При выборе пункта меню – вывод – выводится граф через утилиту graphviz. Картинка с деревом автоматически открывается.
3. При заполнении графа на основе данных из файла, сначала осуществляется проход по файлу для того, чтобы определить, какое максимальное место необходимо выделить под массив вершин и соответственно под матрицу смежности. Затем происходит второй проход по файлу, в ходе которого проверяется корректность данных (описано в пункте 3 в разделе «Описание технического задания»).
4. При нахождении максимального пути между всеми возможными парами вершин в графе используется модифицированный алгоритм Флойда-Уоршалла с рекурсивным обходом в глубину. При этом путь подразумевает собой прохождение через какие-либо вершины только один раз (без повторного захода в пройдённую вершину).
5. При выборе пункта 0 – программа завершается.

# Сравнение времени работы и ресурсных затрат

В программе предусмотрена функция, которая замеряет время (в мкс.) определенного процесса 100 раз и в качестве результата выдает среднее из 100 замеров. В данной задаче программа выводит среднее время поиска максимального пути в графе, а также выводит размер графа, занимаемый в байтах. Пример работы алгоритма поиска максимального пути, а также время нахождения пути и вывод занимаемого места в памяти приведено ниже:

|  |
| --- |
| Образец файла «input.txt»:  1 2 1  4 2 11  3 5 6  2 5 4  99 1 1  4 3 2 |

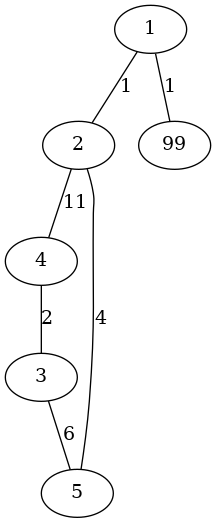


Рисунок 1. Изначальный граф

|  |
| --- |
| … Введите ваш выбор: 3  Максимальное расстояние между вершинами 1 и 2: 1  Максимальное расстояние между вершинами 1 и 4: 13  Максимальное расстояние между вершинами 1 и 3: 14  Максимальное расстояние между вершинами 1 и 5: 20  Максимальное расстояние между вершинами 1 и 99: 1  Максимальное расстояние между вершинами 2 и 4: 12  Максимальное расстояние между вершинами 2 и 3: 13  Максимальное расстояние между вершинами 2 и 5: 19  Максимальное расстояние между вершинами 2 и 99: 2  Максимальное расстояние между вершинами 4 и 3: 21  Максимальное расстояние между вершинами 4 и 5: 15  Максимальное расстояние между вершинами 4 и 99: 14  Максимальное расстояние между вершинами 3 и 5: 17  Максимальное расстояние между вершинами 3 и 99: 15  Максимальное расстояние между вершинами 5 и 99: 21  Поиск пути занял: 6.130000мкс.  Граф занимает: 128Б  … |

Для поиска кратчайших путей между всеми вершинами используется алгоритм Флойда-Уоршалла. Немного его изменив, можно получить модифицированный алгоритм Флойда-Уоршалла, который находит максимальный путь между вершинами в графе.

Сложность алгоритма – o(n^3), где n – количество вершин в графе. Связано это с тем, что оригинальный алгоритм Флойда-Уоршалла также имеет кубическую сложность из-за того, что для каждой пары (n^2) находится максимальный путь с помощью DFS (n).

В качестве реальной задачи данный алгоритм можно использовать, чтобы подобрать максимально насыщенный экскурсионный маршрут для туриста (то есть за определенное время (сутки или 5 часов) маршрут должен содержать как можно больше достопримечательностей, при этом они не должны повторяться). Или, например, у сети пиццерий есть несколько точек доставок в городе. Однако каждая точка может доставлять только на определенное (максимальное) расстояние, иначе еда остынет. Поэтому от каждой точки вычисляется максимальное расстояние для доставки. При этом граф подразумевает собой систему дорог, которые соединены и по ним необходимо не проезжать дважды и более раз.

# Тесты

В следующих тестах символ «>» обозначает ввод пользователем пункта меню, а символ Х – полученное значение в ходе программы.

1. **Позитивные тесты**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Входные данные | Выходные данные | Что проверяется |
| pos\_01 | >1 | Граф успешно прочитан из файла! | Умеет ли программа составлять граф на основе данных из файла |
| pos\_02 | >2 | (модель графа в graphviz) | Умеет ли программа выводить граф с помощью graphviz |
| pos\_03 | >3 | Максимальное расстояние между вершинами Х и Х: Х  Поиск пути занял: Х мкс.  Граф занимает: Х Б | Умеет ли программа находить максимальный путь в графе |
| pos\_04 | >0 | Выход из программы | Умеет ли программа завершаться |

1. **Негативные тесты**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Входные данные | Выходные данные | Что проверяется |
| neg\_01 |  | Сообщение: «Пожалуйста, введите целое число!» | Пустой ввод |
| neg\_02 | >-1 | Сообщение: «Неверный выбор.» | Правильность ввода пункта меню |
| neg\_03 | >100 | Сообщение: «Неверный выбор.» | Правильность ввода пункта меню |
| neg\_04 | >А | Сообщение: «Пожалуйста, введите целое число!» | Правильность ввода пункта меню |
| neg\_05 | >1  (буквы или другие посторонние символы в файле) | Сообщение: «Файл имеет некорректные данные!» | Корректность данных в файле |
| neg\_06 | >3  (пустой граф) | Cообщение: «Граф пуст!» | Существование графа |
| neg\_07 | >1  (отсутствие файла с названием “input.txt”) | Сообщение: «Ошибка! Не удалось открыть файл!» | Существование файла |
| neg\_08 | >2  (пустой граф) | Cообщение: «Граф пуст!» | Существование графа |
| neg\_09 | >1  (в файле присутствуют отрицательные расстояния) | Сообщение: «Граф задан неправильно! (расстояние между вершинами должно быть положительным)» | Корректность данных в файле |
| neg\_10 | >1  (в файле найдено несколько путей от одной пары вершин) | Сообщение: «Граф задан неправильно! (между одной парой вершин не должно быть 2 и более расстояний)» | Корректность данных в файле |
| neg\_11 | >1  (номер вершины в файле равен 0) | Сообщение: «Граф задан неправильно! (номер вершины должен отличаться от нуля)» | Корректность данных в файле |

# Ответы на контрольные вопросы

1. **Что такое граф?**

Граф - это абстрактная математическая структура, используемая для моделирования отношений между объектами. Он состоит из вершин (или узлов) и ребер (или связей), которые соединяют эти вершины.

1. **Как представляются графы в памяти?**

Графы могут быть представлены в памяти различными способами, такими как:

* Список смежности: каждая вершина имеет список своих соседних вершин.
* Матрица смежности: двумерный массив, в котором значение в позиции [i][j] указывает на наличие ребра между вершинами i и j.

1. **Какие операции возможны над графами?**

Операции, возможные над графами, включают:

* Добавление и удаление вершин и ребер.
* Поиск кратчайшего (максимального) пути между вершинами.
* Проверка наличия циклов в графе.
* Поиск минимального остовного дерева.
* И многие другие.

1. **Какие способы обхода графов существуют?**

Существуют различные способы обхода графов:

* Обход в глубину (DFS).
* Обход в ширину (BFS).
* Топологическая сортировка.

1. **Где используются графовые структуры?**

Графовые структуры используются в различных областях, таких как информационные технологии (например, для моделирования сетей), транспортное планирование (например, для нахождения оптимального пути в метро), социальные сети, биоинформатика и так далее.

1. **Какие пути в графе Вы знаете?**

В графе существуют различные типы путей, такие как:

* Простой путь: путь, в котором все вершины различны.
* Цикл: путь, в котором начальная и конечная вершины совпадают.
* Кратчайший путь: путь между двумя вершинами с минимальной суммой весов ребер.
* Эйлеров путь - это путь в графе, который проходит через каждое ребро графа ровно один раз.
* Эйлеров цикл - это эйлеров путь, который начинается и заканчивается в одной и той же вершине.
* Гамильтонов путь - это путь в графе, который проходит через каждую вершину графа ровно один раз.
* Гамильтонов цикл - это гамильтонов путь, который начинается и заканчивается в одной и той же вершине.

1. **Что такое каркасы графа?**

Остовное дерево (или каркас) графа - это подграф, содержащий все вершины и некоторые (или все) ребра исходного графа, образующий дерево без циклов.

# Вывод

В рамках задания был реализован тип «граф». Концептуально граф представлен в виде матрицы смежности, в программе можно задать его на основе данных из файла, вывести используя graphviz, а также найти макс. путь.