|  |  |
| --- | --- |
| 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»**

Студент Пискунов Панте

Группа ИУ7 – 36Б

Вариант №4

Преподаватель [Никульшина Татьяна Александровна](https://e-learning.bmstu.ru/iu7/user/view.php?id=625&course=1)

2022 г.

# **Описание условия задачи**

Обработать графовую структуру в соответствии с заданным вариантом. Обосновать выбор необходимого алгоритма и выбор структуры для представления графов. Ввод данных осуществить на усмотрение программиста. Результат выдать в графической форме.

# **Описание технического задания**

Найти все вершины графа, к которым от заданной вершины можно добраться по пути не длиннее А.

Входные данные:

1. Запуск программы через командную строку

./app.exe input\_file.txt output\_graph.dot output\_task.dot

1. Целое число, представляющее собой номер команды:

целое число в диапазоне от 0 до 5.

1. Файл, в котором находятся вершины графики

**Выходные данные:**

1. Результат выполнения определенной команды.
2. Сравнение работы алгоритма Дейкстры на различных размерах.

Меню программы:

1 — Загрузить граф из файла.

2 — Вывести граф.

3 — Вывести матрицу графа.

4 — Найти все вершины графа, к которым от заданной вершины можно добраться по

пути не длиннее А.

5 — Вывести сравнительную характеристику

0 — Выход из программы

Обращение к программе:

Запуск через терминал ./app.exe input\_file.txt output\_graph.dot output\_task.dot

Аварийные ситуации:

* 1. Некорректный ввод номера команды.

На вход: число, большее чем 5 или меньшее, чем 0.

На выход: Ошибка: *Неправыльно выбрали действие!.*

2. Указан несуществующий файл с входными данными

На вход : ./app.exe не\_существует output\_graph.dot output\_task.dot

На выход: Ошибка: Не удалось прочитать файл!

# **Описание структуры данных**

Граф представлен в виде матрицы смежности.

typedef struct matrix\_t

{

size\_t amount; // количество вершин

double \*\*data; // данные

} matrix\_t;

# **Описание работы алгоритма**

Для представления графа в программе используется матрица смежности. Данные для заполнения матрицы смежности считываются из файла. Далее по полученной матрице создается файл с расширением .dot, содержащий представление графа на языке DOT. Данный файл открывается с помощью программы graphviz.

У пользователя запрашивается номер вершины, от которой производится расчет кратчайших путей до каждой другой вершины с помощью алгоритма Дейкстры. Также запрашивается максимальный путь А. Далее

начальная вершина обозначается нулю, а все остальные вершины получают самое максимальное число. Потом начинаем сравнивать длины ребра с максимальной указаной длиной А. Если нашли такую длину, то ее записываем в отдельный массив а посещенную вершину обозначаем как посещенную(обозначаем ее 1). Результат записывается в файл с расширением .dot на языке DOT для визуализации.

# **Пример работы программы**

**Граф по входным данным**:

**Входные данные**

5

0 1 2 3 0

0 0 0 2 0

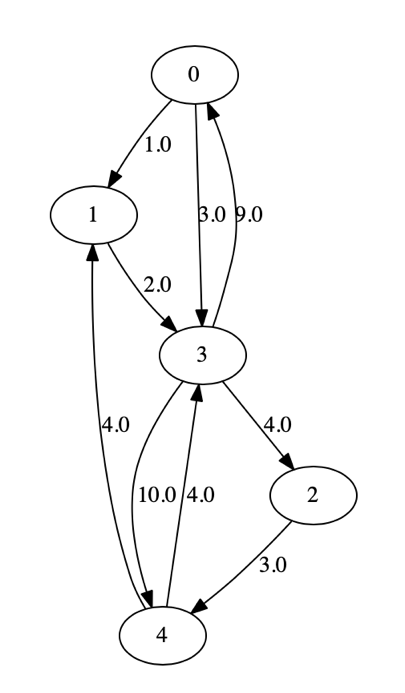
2 0 0 0 3

9 0 4 0 10

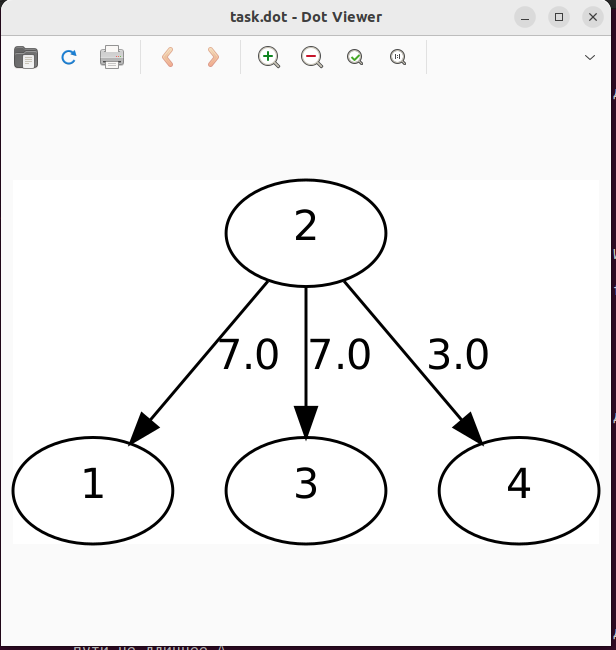
0 4 0 4 0

**Ключевая вершина**: 2

**Расстояние**: 10



**Граф по ключевой вершине**:



# **Набор тестов**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название теста** | **Пользовательский ввод** | **Результат** |
| 1 | Некорректный ввод команды | SADASD | Ошибка: Неправильно выбрали действие! |
| 3 | Отрицательное количество вершин |  | Ошибка: Не удалось выделить память под матрицой! |
| 4 | Вывод графа, вывод матрицы графа или поиск в несозданном графе | 3/4/5 | Данные еще не были загружены. Выберите пункт 1. |
| 6 | Ввод максимально возможной длины, меньше, чем самый маленький путь | 4 | Не успели найти такой путь! |
| 7 | Ввод максимально возможной длины, большей, чем самый большой путь | 4 | Граф отображается. |
| 8 | Файла, который необходимо открыть, не существует |  | Ошибка: Не удалось прочитать файл! |
| 9 | Значение пути отрицательно |  | Число должно быть неотрицательным. |

# **Сравнительная характеристика**

Сравнивается время и память, затраченные при поиске кратчайших путей от выбранной вершины до каждой другой с помощью алгоритма Дейкстры. Время измеряется в секундах, память – в байтах.

|  |  |  |
| --- | --- | --- |
| **Количество\*** | **Время (такты)** | **Память (байты)** |
| 10 | 0.000006 | 816 |
| 50 | 0.000065 | 20016 |
| 100 | 0.000229 | 80016 |
| 500 | 0.002838 | 2000016 |

Временная сложность алгоритма – O().

# **Разработанная задача**

Необходимо предложить вариант задачи, для которой можно было бы использовать разработанную программу.

Поскольку графы популярны при решении заданий о путях и дорогах, подобных задач придумано очень много:

- автоматическое построение маршрута на онлайн-карте;

- поиск системой бронирования наиболее быстрых или дешевых билетов, в том числе с возможными пересадками;

- моделирование движения робота, который перемещается по местности;

# **Контрольные вопросы**

*1. Что такое граф?*

Граф – это конечное множество вершин и ребер, соединяющих их, G = < V,E >, где V – конечное непустое множество вершин; Е – множество ребер (пар вершин).

Если пары Е (ребра) имеют направление, то граф называется ориентированным (*орграф*), если иначе - неориентированный (*неорграф*). Если в пары Е входят только различные вершины, то в графе нет петель. Если ребро графа имеет вес, то граф называется *взвешенным*.

Неорграф называется *связным,* если существует путь из каждой вершины в любую другую.

*2. Как представляются графы в памяти?*

Граф в памяти представляется в виде матрицы смежности или списка смежности.

*Матрица смежности* B(n\*n) – элемент b[i,j]=1, если существует ребро, связывающее вершины i и j, и =0, если ребра не существует.

*Список смежностей* содержит для каждой вершины из множества вершин V список тех вершин, которые непосредственно связаны с ней. Входы в списки смежностей могут храниться в отдельной таблице, либо же каждая вершина может хранить свой список смежностей.

*3. Какие операции возможны над графами?*

* поиск кратчайшего пути от одной вершины к другой (если он есть);
* поиск кратчайшего пути от одной вершины ко всем другим;
* поиск кратчайших путей между всеми вершинами;
* поиск эйлерова пути (если он есть);
* поиск гамильтонова пути (если он есть).

*4. Какие способы обхода графов существуют?*

*Обход в ширину (BFS – Breadth First Search)* - обработка вершины V осуществляется путём просмотра сразу всех «новых» соседей этой вершины, которые последовательно заносятся в очередь просмотра.

*Обход в глубину (DFS – Depth First Search)* - начиная с некоторой вершины v0, ищется ближайшая смежная ей вершина v, для которой в свою очередь осуществляется поиск в глубину до тех пор, пока не встретится ранее просмотренная вершина, или не закончится список смежности вершины v (то есть вершина полностью обработана). Если нет новых вершин, смежных с v, то вершина v считается использованной, идет возврат в вершину, из которой попали в вершину v, и процесс продолжается до тех пор, пока не получим v = v0. При просмотре используется стек.

*5. Где используются графовые структуры?*

Графовые структуры могут использоваться в задачах, где между элементами могут быть установлены произвольные связи. Распространенное применение — решение задачах о путях.

*6. Какие пути в графе Вы знаете?*

*Эйлеровый путь* - путь в графе, проходящий через каждое ребро ровно один раз. Если путь проходит по некоторым вершинам несколько раз – он называется *непростым*, иначе – *простым*.

*Гамильтонов путь* - путь, проходящий через каждую вершину ровно один раз.

*7. Что такое каркасы графа?*

*Каркас графа* – дерево, в которое входят все вершины графа, и некоторые (не обязательно все) его рёбра. Для построения каркасов графа используются алгоритмы Крускала и Прима.

# **Вывод**

Для решения данной задачи был выбран способ хранения графа в виде матрицы смежности. Плюсы данного метода заключаются в возможности сразу визуально понять, насколько плотно связан граф, а также в легкости перевода на язык DOT связей вершин, что позволяет быстро получить графическое изображение графа.

Также для решения задачи использовался алгоритм Дейкстры – достаточно быстрый, особенно при небольших размерах графа и простой метод нахождения кратчайших путей между вершинами. Алгоритм Дейкстры основан на выборе для включения в путь всякий раз той вершины, которая имеет наименьшую оценку кратчайшего пути (по весам ребер), то есть наименьший путь до этой вершины из всех возможных путей, которые были рассмотрены ранее.