|  |  |
| --- | --- |
|  | **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

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7**

Название предмета: Типы и структуры данных

Студент: Гриценко Алексей Михайлович

Группа: ИУ7-31Б

*2020г.*

1. **Описание условия задачи**

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

Задана система двусторонних дорог. Для каждой пары городов найти длину кратчайшего пути между ними.

1. **Техническое задание**
2. ***Описание исходных данных***
3. **Размер матрицы:** число – размер матрицы.
4. **Имя файла с матрицей:** строка, содержащая имя файла.
5. ***Описание результата программы***

С разницей в четыре секунды, каждый раз открывается изображение, которое отображает кратчайший путь из одной вершины в другую, в это время в терминале на всякий случай выводится, какие пары городов рассматриваются на данный момент

1. ***Задача, реализуемая программой***

Поиск кратчайшего пути между вершинами графа

1. ***Обращения к программе***

Способ обращения к программе происходит через консоль.

1. ***Описание возможных аварийных ситуаций и ошибок пользователя***
2. Некорректный ввод имени файла.  
   На входе: имя файла, несуществующего в системе.   
   На выходе: сообщение «Неверное имя файла! Повторите попытку.»
3. Некорректный ввод размера графа.  
   На входе: буква или, любой другой нечисловой символ или отрицательное число.  
   На выходе: сообщение «Введено недопустимое значение! Повторите попытку.»
4. На входе: пустой файл

На выходе: сообщение “Файл пуст! Проверьте файл”.

1. На входе: матрица, из которой получается несвязанный граф.

На выходе: сообщение “Ошибка, граф не является связанным.”

1. **Описание внутренних структур данных**

**Структура графа**

typedef struct graph

{

int size;

int \*\*matrix;

char \*\*paths;

} graph\_t;

int size — кол-во вершин в графе.

int \*\*matrix — матрица связности графа.

char \*\*paths — кратчайшие пути.

Данная структура наиболее удобна для работы, но неэффективна по памяти, поскольку матрица связности может быть заполнена в основном нулями, то есть неиспользуемыми элементами. Поэтому для более эффективного использования памяти можно воспользоваться связным списком.

1. **Описание алгоритма**

Для поиска кратчайшего пути используется алгоритм Дейкстры. Он подходит под условие задачи, так как в графе не может быть отрицательных весов. Алгоритм Дейкстры не подходит для несвязного графа, и так же не подходит для графа, где есть отрицательные весы у дуг. Но, так как в моей задаче таких случаев быть не может, то данный алгоритм идеально подходит для решения моей задачи. Так как вершины хранятся в простом массиве и для поиска минимума используется линейный алгоритм, то сложность алгоритма составляет O(V \* V + E) = O(V2).

**Функции, использующиеся в программе**

void create\_string\_path(graph\_t graph, char \*graph\_visual) - функция создания пути для отображения графа

static void print\_path(char \*graph\_visual, char \*path) – функция печати пути.

static void print\_solution(char \*solution\_path) – функция печати решения.

void paint\_graphs(const graph\_t graph, char \*graph\_visual, char \*copy\_graph\_visual, const int start) – функция печати рассматриваемого на данный момент пути в терминал, и, соответственно, вывод.

static int min\_distance(int \*distance, char \*is\_visited, const int size) – функция вычисления минимальной дистанции.

static void save\_path(graph\_t graph, const int start\_vertex, char \*path)

– функция записи пути в массив строк.

int deijkstra(graph\_t graph, int start\_vertex) – функция самого алгоритма дейкстры.

graph\_t create\_graph(int size) – функция создания графа.

static int file\_read(const graph\_t graph, FILE \*f) – функция чтения файла

int read\_graph(const graph\_t graph, FILE \*f) – функция чтения графа из файла.

1. **Тестирование**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Тест | Пользовательский ввод | Результат |
| 1 | Некорректный ввод имени файла | graph01.txt (файл не существует) | Неверное имя файла! Повторите попытку. |
| 2 | Пустой файл | graph01.txt (пустой файл) | Файл пуст! Проверьте файл. |
| 3 | Некорректный ввод размера матрицы | A | Введено недопустимое значение! |
| 4 | Некорректный ввод размера матрицы | 0 | Введено недопустимое значение! |
| 5 | Некорректный ввод размера матрицы | -1 | Введено недопустимое значение! |
| 6 | Ввод матрицы, из которого получим несвязанный граф | 0 5 0 0  5 0 0 0  0 0 0 7  0 0 7 0 | Ошибка, граф не является связанным. |
| 7 | Корректный ввод всех характеристик | Корректный файл, корректный ввод размера матрицы | Картинка с дорогами |

1. **Выводы по проделанной работе**

Алгоритм Дейкстры удобен тем, что во время работы можно добавить или исключить вершину, дополнительно стоит сказать, что по сравнению с другими алгоритмами он относительно прост в реализации. Он будет лучше работать при больших(полных) графах. Но, существует множество других алгоритмов для решения этой задачи, например алгоритм Белмана-Форда, который может рассматривать ребра с отрицательным весом.

1. **Ответы на вопросы**

**Что такое граф?**Граф – конечное множество вершин и соединяющих их ребер; G = <V, E>.Если пары Е (ребра) имеют направление, то граф называется ориентированным; если ребро имеет вес, то граф называется взвешенным.

**Как представляются графы в памяти?**

С помощью матрицы смежности или списков смежности.

**Какие операции возможны над графами?**

- поиск вершин в графе  
- поиск кратчайших путей от Vk до Vm  
- поиск Эйлерова пути  
- поиск Гамильтонова пути  
- поиск кратчайших путей между всеми вершинами

Так-же : поиск различных путей, исключение и включение вершин.

**Какие способы обхода графов существуют?**

Обход в ширину и обход в глубину.

**Где используются графовые структуры?**Графовые структуры могут использоваться в задачах, в которых между элементами могут быть установлены произвольные связи, необязательно иерархические.

**Какие пути в графе Вы знаете?**Эйлеров путь, непростой путь, гамильтонов путь.

**Что такое каркасы графа?**

Каркас графа – дерево, в которое входят все вершины графа, и некоторые (необязательно все) его рёбра.