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 ПО ДИСЦИПЛИНЕ ТИПЫ И СТРУКТУРЫ ДАННЫХ

Студент **Простев Тимофей Александрович**

Группа **ИУ7-33Б**

Название предприятия **НУК ИУ МГТУ им. Н. Э. Баумана**

Студент _____ **Простев Т. А.**

Преподаватель _____ **Никульшина Т. А.**

Преподаватель _____ **Барышникова М. Ю.**

Оценка _____

2023 г.

1. Описание условия задачи.

В графе найти максимальное расстояние между всеми парами его вершин.

2. Техническое задание.

Исходные данные и правила их ввода:

- Номер пункта меню: целое число от 1 до 5:
  - 1 – Считать граф из файла.
  - 2 – Считать граф с клавиатуры.
  - 3 – Вывести граф на экран.
  - 4 – Вывести матрицу наибольших расстояний между всеми парами вершин графа.
  - 5 – Выход.
- Имя файла: строка, не больше 255 символов.
- Ввод графа:
  - Количество вершин - положительное целое число, больше 0.
  - Количество рёбер - положительное целое число, больше 0.
  - Вершина-начало ребра – положительное целое число от 0 до количества вершин графа – 1.
  - Вершина-конец ребра – положительное целое число от 0 до количества вершин графа – 1.
  - Алгоритм поиска кратчайших путей будет работать корректно, если в графе будут отсутствовать положительные циклы.

Выходные данные:

- png-изображение графа.
- Матрица максимальных расстояний.

Способ обращения к программе:

Запуск исполняемого файла. В рабочей директории выполнить команду ./app.exe.

Аварийные ситуации:

•Ввод некорректных данных.

•Ошибка выделения памяти.

В случае аварийной ситуации выводится сообщение об ошибке.

3. Описание внутренних структур данных.

Структура матрицы:

```
typedef struct graph {
    int **adj_matrix;
    size_t vertices_count;
} graph_t;
```

Структура содержит 3 поля:

- `adj_matrix` – массив указателей на строки матрицы смежности графа – указатель на указатель типа `int`.
- `vertices_count` – количество вершин графа – беззнаковое целое числа типа `size_t`.

Структура матрицы расстояний.

```
typedef struct
{
    int **paths;
    size_t size;
} matrix_t;
```

Структура содержит 2 поля:

- `paths` – массив указателей на строки матрицы расстояний – указатель на указатель типа `int`.
- `size` – размер матрицы.

Для работы со структурой используются функции:

`graph_t* graph_new(size_t vertices_count, size_t edges_count);` - выделяет память под граф.

`void graph_delete(graph_t *graph);` - освобождает память, выделенную под граф.

`int graph_input(FILE *file, graph_t *graph);` - ввод графа из потока file.

`int graph_to_dot(FILE *file, graph_t *graph);` - перевести граф в dot формат.

`matrix_t* graph_longest_paths(graph_t *graph);` - создать матрицу максимальных расстояний.

`void paths_matrix_print(FILE *file, matrix_t *paths);` - вывести матрицу расстояний.

4. Описание алгоритма.

Выводится меню программы. Пользователь вводит номер команды, после чего выполняется действие.

Для расчета максимальных расстояний между всеми вершинами графа используется алгоритм Флойда-Уоршелла. Данный алгоритм – алгоритм поиска кратчайших путей во взвешенном графе. Перед выполнением самого алгоритма создается копия матрицы смежности, после чего все её значения умножаются на -1, а всем нулевым значениям, кроме значений на диагонали матрицы, присваивается максимальное возможное значение.

Алгоритм:

Для каждой вершины i от 0 до числа вершин в графе - 1 выполняется следующее:

Для каждой пары вершин j, k проверяется, уменьшится ли расстояние между j и k, если пройти через i. Если да – обновить расстояния от j до k на сумму расстояний от j до i и от i до k.

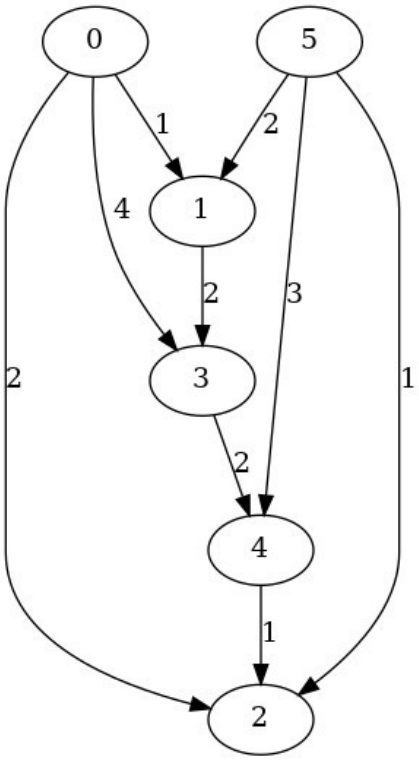
Алгоритм имеет время выполнения $O(V^3)$, где V – число вершин графа, а также занимаемую дополнительную память $O(v^2)$.

После выполнения алгоритма все значения в матрице снова умножаются на -1. Умножение значений на -1 превращает минимальные значения в максимальные и наоборот, соответственно, алгоритм будет искать наибольшие пути.

Алгоритм работает корректно, если в графе нет циклов отрицательной величины.

5. Пример работы.

На выход подается файл с описанием графа. Граф выглядит следующим образом:



Получившаяся матрица наибольших путей:

	0	**1**	**2**	**3**	**4**	**5**
0	0	1	7	4	6	
1		0	5	2	4	
2			0			
3			3	0	2	
4			1		0	
5		2	7	4	6	0

6. Выводы.

При выполнении задачи для хранения графа была выбрана матрица смежности, так как алгоритм Флойда-Уоршелла производит операции именно над матрицами смежностей. Алгоритм был выбран, так как для вычисления длиннейших путей между всеми парами вершин он имеет наименьшую сложность по времени для данной задачи.

7. Контрольные вопросы.

1. Что такое граф?

Граф – конечное множество вершин V и соединяющих их рёбер E. Если рёбра имеют направление, то граф называется ориентированным.

2. Как представляются графы в памяти?

Основные способы хранения: матрица смежности и список смежности.

3. Какие операции возможны над графами?

Обходы графа, добавление вершины в граф, удаление вершины из графа, добавление ребра в граф, удаление ребра из графа, поиски кратчайших путей между всеми вершинами и между двумя вершинами, поиск эйлерова пути, поиск гамильтонова пути.

4. Какие способы обхода графов существуют?

Обход в глубину, обход в ширину.

5. Где используются графовые структуры?

Графовые структуры могут использоваться в задачах, в которых между элементами могут быть установлены произвольные связи, необязательно иерархические.

6. Какие пути в графе вы знаете?

Простой путь, непростой путь, эйлеров путь, гамильтонов путь.

7. Что такое каркасы графа?

Графовые структуры могут использоваться в задачах, в которых между элементами могут быть установлены произвольные связи, необязательно иерархические