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

ТИПЫ И СТРУКТУРЫ ДАННЫХ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7

“Графы”

ВАРИАНТ 4

Студент Гурова Наталия Алексеевна
фамилия, имя, отчество

Группа ИУ7-34Б

Выполнил _____ Гурова Н.А.
подпись, дата *фамилия, и.о.*

Принял _____ Силантьева А.В.
подпись, дата *фамилия, и.о.*

2021 г.

Цель работы

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.

Указания к выполнению работы

Интерфейс программы должен быть понятен неподготовленному пользователю. При разработке интерфейса программы следует предусмотреть:

- указание формата и диапазона вводимых данных,
- блокирование ввода данных, неверных по типу,
- указание операции, производимой программой:
  - добавление элемента в стек,
  - удаление элемента из стека,
  - вычисление (обработка данных);
- наличие пояснений при выводе результата.

При тестировании программы необходимо:

- проверить правильность ввода и вывода данных (т.е. их соответствие требуемому типу и формату), обеспечить адекватную реакцию программы на неверный ввод данных;
- обеспечить вывод сообщений при отсутствии входных данных («пустой ввод»);
- проверить правильность выполнения операций;
- предусмотреть вывод сообщения при поиске несуществующих путей в графе .

Задание

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

Входные данные

- Для пунктов меню 0, 1, 2, 3, 4, 5 вводится данный пункт в виде целого
- Для пункта 2 вводится также количество вершин, а также матрица дорог
- Для пункта 3 вводится также номер вершины, а также максимальная длина (см. задание)
- Для пункта 5 вводится также параметры исследования памяти и эффективности (количество измерений, количество вершин для каждого измерения)

Выходные данные

- Результат выполнения определенной команды:
    1. Сообщение о том, что данные успешно считаны из файла, либо о том, что в процессе чтения произошла ошибка
    2. Сообщение о том, что граф успешно создан, либо о том, что в процессе ввода произошла ошибка
    3. Информация о длинах путей из введенной вершины во все оставшиеся. Информация о том, какие пути не превышают максимальной длины.
    4. Вывод графа
    5. Отчет о количестве запрашиваемой памяти, эффективности и количестве сравнений.

Возможности программы

0 – выход из программы

1 - Загрузить данные из файла

2 – Создать граф из консоли

3 – Найти вершины, в которые можно попасть из заданной вершины по пути меньше заданного числа

4 – Вывод графа

5 – Проведение исследования

Способ обращения к программе

Программа может быть вызвана через консоль с помощью команды app.exe

Аварийные ситуации

- Некорректный ввод номера команды (введено не число)
- Некорректные данные в файле
- Некорректные данные при вводе графа
- Некорректно введен номер вершины для поиска
- Некорректно введена максимальная длина пути
- Некорректный ввод количества вершин в графе для проведения исследования

Описание алгоритма

1. Выводится меню данной программы.
2. Пользователь вводит номер команды из предложенного меню.
3. Пока пользователь не введет 0 (выход из программы), ему будет предложено вводить номера команд и выполнять действия по выбору.

Структуры данных

Структура для описания графа:

```
typedef struct
{
    int n;
    int **data;
} graph_t;
```

Поля структуры:

n – количество вершин

data – матрица с длиной путей

Оценка эффективности

Время поиска путей (в тактах процессора, 5000 итераций):

TIME(in processor clock cycles)

COUNT TOPS	USED TIME
10	22512
50	1427748
100	6139749
200	28112238
500	451286069

Память (в байтах):

MEMORY

COUNT TOPS	USED MEMORY
10	400
50	10000
100	40000
200	160000
500	1000000

График зависимости времени от количества вершин в графе:

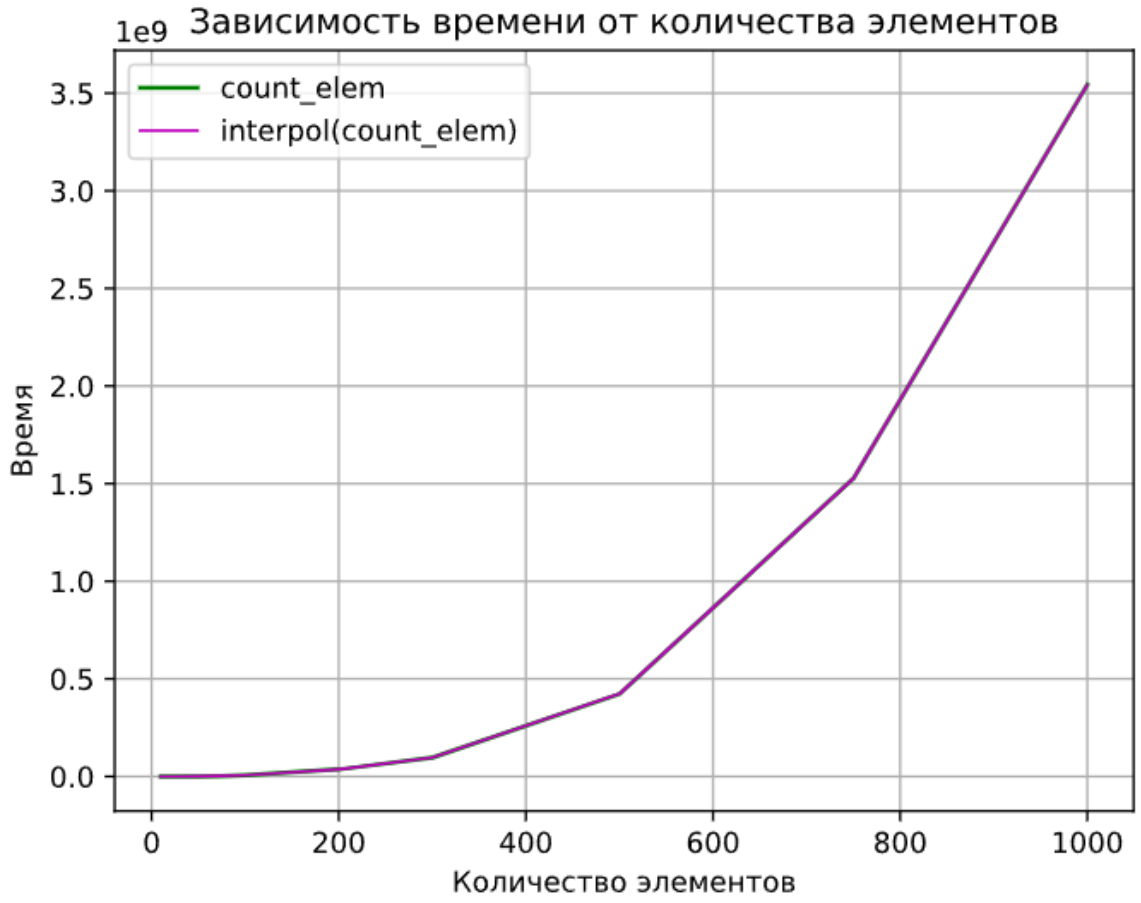
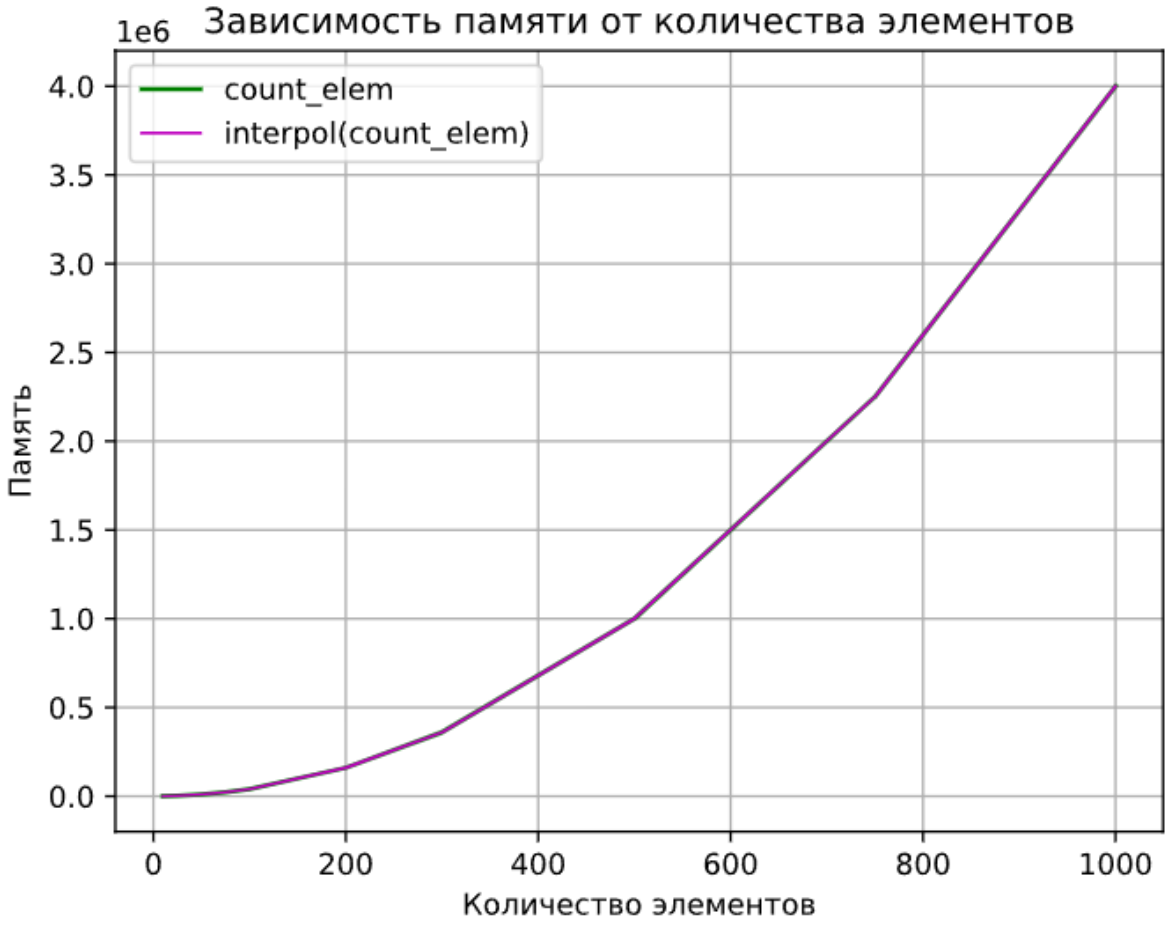


График зависимости памяти от количества вершин в графе:



Вывод

Для реализации данной задачи был использован алгоритм Дейкстры, который находил все пути, которые меньше заданного значения, от одной вершины до всех остальных.

Алгоритм Дейкстры очень удобен, и его можно использовать в различных сферах жизни. Например, для поиска кратчайшего расстояния от одного города до другого, или для определения, какой маршрут по стоимости из одного города в другой самый дешевый.

Хранение графа в виде матрицы смежности удобно тем, что по матрице можно понять расстояние между вершинами, узнать, существует ли ребро между вершинами.

Ответы на контрольные вопросы

1. Что такое граф?

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

Если пары E (ребра) имеют направление, то граф называется ориентированным (орграф), если иначе - неориентированный (неорграф). Если в пары E 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?

В памяти удобно представлять граф 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?

Обход вершин и поиск различных путей: поиск кратчайшего пути от одной вершины к другой (если он есть), поиск кратчайшего пути, поиск эйлерова пути, поиск гамильтонова пути.

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 последоватеьно 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 стуктуры могут использоваться в задачах, где между элементами могут быть установлены произвольные связи. Наиболее распространеное использование таких структур — при решении различных задачах о путях.

6. Какие пути в графе Вы знаете?

Эйлеровый путь - путь в графе, проходящий через каждое ребро ровно один раз. (путь может проходить по некоторым вершинам несколько раз – в этом случае он является непростым)

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

Такие пути могут не существовать в графах.

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.