|  |  |
| --- | --- |
| 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 №7**

**«Графы»**

Студент ???

Группа ИУ7 – 3?Б

Преподаватель Силантьева А.В.

*2021 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.

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 .

**Описание технического задания**

Найти самый длинный простой путь в графе.

Входные данные:

1. Команда: целое число от 0 до 4.
2. Граф: файл с графом, хранящимся в виде списка смежности.

Выходные данные:

1. Картинка графа в формате png.
2. Самый длинный путь.

Функции программы:

1. Загрузить граф из файла.
2. Вывести граф с помощью GRAPHVIZ.
3. Вывести на экран список смежности.
4. Найти самый длинный путь.

0. Выход из программы.

Обращение к программе:

Запускается через терминал.

Аварийные ситуации:

1. Некорректный ввод номера команды.

На входе: число, большее чем 4 или меньшее, чем 0.

На выходе: сообщение «Invalid menu code.»

2. Совершение команды с графом, до его загрузки.

На входе: команда 2, до команды 1.

На выходе: сообщение «Graph is empty.»

**Описание структуры данных**

Граф храниться в виде списка смежности.

***struct node\_t***

***{***

***int num;***

***node\_t \*next;***

***};***

***num*** — значение вершины   
***\*next*** – указатель на вершину, смежную с данной

Входы в списки смежностей для каждой вершины графа хранятся в массиве:

**node\_t arrb[MAX\_N];**

**MAX\_N = 500**

**Описание алгоритма**

1. Выводится меню.
2. Пока пользователь не введет 0, он может выбирать программы по своему усмотрению.

**Набор тестов**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название теста** | **Пользовательский ввод** | **Результат** |
| 1 | Некорректный ввод команды | -5 | Invalid menu code |
| 2 | Некорректный ввод команды | asd | Invalid menu code |
| 3 | Несуществующий файл | Команда 1 | Error opening file |
| 4 | Вывод не считанного графа | 2 | Graph is empty |
| 5 | Поиск самого длинного пути в пустом графе | 4 | Grapg is empty |

**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; 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?*

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 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. Наиболее распространено 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.

**Вывод**

Для решения данной задачи был использован алгоритм поиска в глубину, т.к. он просматривает каждый раз следующую вершину графа, стремясь в «глубину» графа. Так как для алгоритма требуется все вершины, смежные с рассматриваемой, был использован список смежности для хранения графа.