

	<p align="center"> <b>Министерство образования и науки Российской Федерации</b>  <b>Федеральное государственное бюджетное образовательное учреждение</b>  <b>высшего образования</b>  <b>«Московский государственный технический университет</b>  <b>имени Н.Э. Баумана</b>  <b>(национальный исследовательский университет)»</b>  <b>(МГТУ им. Н.Э. Баумана)</b> </p>
---	--

ФАКУЛЬТЕТ \_\_\_\_\_ Информатика и системы управления (ИУ) \_\_\_\_\_

КАФЕДРА \_\_\_\_\_ Программное обеспечение ЭВМ и информационные технологии (ИУ7) \_\_\_\_\_

## **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7** **«ОБРАБОТКА ГРАФОВ»**

Студент, группа

**Буланый К., ИУ7-36Б**

2020 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. Определить, является ли связным заданный граф.

## Техническое задание

### Входные данные:

1. **Целое число, представляющее собой количество вершин в рассматриваемом графе:** целое положительное число.
2. **Пара целых чисел, представляющая собой связи между вершинами:** целые неотрицательные числа в диапазоне от 0 до  $V-1$ , где  $V$  – количество вершин графа.
3. **Число для завершения ввода:** -1

### Выходные данные:

1. Графическая визуализация полученного графа с подписью, в зависимости от того, связный ли граф

**Функция программы:** программа находит решение задачи и визуализирует найденное решение при помощи **GraphViz**.

**Обращение к программе:** запускается из терминала.

### Аварийные ситуации:

1. Некорректный ввод количества вершин.  
На входе: неположительное целое число, нецелое число или буква.  
На выходе: сообщение «Wrong. Number.»
2. Некорректный ввод связей вершин в графе.  
На входе: целое число выходящее за диапазон  $[0; V-1]$ , нецелое число, буква или попытка провести путь из вершины в саму себя.  
На выходе: сообщение «Wrong number.»

# Структуры данных

Хранение матрицы смежности:

```
typedef struct
{
    int sz;
    int **m;
} adjmatr_t;
```

Поля структуры:

- **int sz** – количество вершин в графе;
- **int \*\*m** – указатель на массив указателей;

## Алгоритм

После ввода элементов графа граф проверяется на связность. Так как в данной задаче реализуется обработка неориентированного графа, то для проверки его на связность достаточно выполнить поиск в глубину для любой вершины и в случае, если по окончании поиска была посещена каждая вершина графа, граф является связным

## Тесты

	Тест	Пользовательский ввод	Результат
1	Некорректный ввод количества вершин	0	Wrong number.
2	Некорректный ввод количества вершин	X	Wrong number.
3	Некорректный ввод номера вершины	7 (при кол-ве вершин 6)	Wrong number.
4	Некорректный ввод номера вершины	Y	Wrong number.
5	Некорректный ввод связи между вершинами	5 5	Wrong path
6	Корректный ввод характеристик	3; 0 1 / 1 2	Изображение

# Контрольные вопросы

## 1. Что такое граф?

Граф – конечное множество вершин и соединяющих их ребер;  $G = \langle V, E \rangle$ . Если пары  $E$  (ребра) имеют направление, то граф называется ориентированным; если ребро имеет вес, то граф называется взвешенным.

## 2. Как представляются графы в памяти?

С помощью матрицы смежности или списков смежности.

## 3. Какие операции возможны над графами?

Обход вершин, поиск различных путей, исключение и включение вершин.

## 4. Какие способы обхода графов существуют?

Обход в ширину (**BFS – Breadth First Search**), обход в глубину (**DFS – Depth First Search**).

## 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?

Эйлеров путь, простой путь, сложный путь, гамильтонов путь.

## 7. Что такое каркасы графа?

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

## Вывод

В данной реализации алгоритм — поиск в глубину. Если  $V$  — количество вершин графа, а  $E$  — количество ребер, то алгоритм поиска в глубину имеет сложность  $O(V+E)$ . Для списка смежности сложность такая же, но представление графа в таком виде требует больше памяти, так как нужно хранить указатели, номера вершин, а не целые (только в случае сильной разреженности список выиграет), поэтому выбор пал на матрицу смежности.