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 Маслова Марина Дмитриевна
фамилия, имя, отчество

Группа ИУ7-33Б

2020 г.

Оглавление

Техническое задание

Условие 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.

Найти все вершины заданного орграфа, недостижимые из заданной его вершины.

Входные данные

Целое число, обозначающее пункт меню.

Строка с именем файла.

Целые беззнаковые числа для представления количества ребер, вершин в графе, а также ребер в виде двух вершин, которое это ребро связывают.

Целое беззнаковое число, представляющее номер вершины, для которой определяются недостижимые для нее вершины.

Выходные данные

Изображение графа.

Номера вершин, недостижимых из заданной.

Задачи, реализуемые программой

1. Задание графа с помощью ввода с клавиатуры.
2. Задание графа с помощью выгрузки данных из файла.
3. Создание и показ изображения графа.
4. Поиск вершин, недостижимых из заданной.

Возможные аварийные ситуации и ошибки пользователя

Неверный ввод пункта меню, и других числовых значений.

Ошибка при открытии файла.

Ошибка при выделении памяти.

Описание внутренних структур данных

Для краткости записи используется беззнаковый тип:

```
typedef unsigned int uint;
```

Для представления вершины графа в списке смежности используется структура:

```
typedef struct node node_t;

struct node
{
    uint num;                // номер вершины
    node_t *next_neighbour; //следующий сосед
};
```

Для представления ребра используется структура:

```
typedef struct edge
{
    uint from;  // вершина, из которой выходит ребро
    uint to;    // вершина, в которую входит ребро
} edge_t;
```

Для представления графа в виде списка смежности используется структура:

```
typedef struct graph
{
    node_t **graph; // список смежности
    uint nodes;     // количество вершин в графе
} graph_t;
```

Для представления очереди используется структура:

```
typedef struct
{
    uint *array;   // массив для хранения элементов очереди
```

```
    uint capacity; // вместимость очереди
    uint length;   // текущая длина очереди
    uint in;       // индекс начала очереди
    uint out;      // индекс конца очереди
} queue_t;
```

Описание функций

```
/*
 * Читает граф из файла или стандартного потока ввода в список
 * смежности graph
 */
int read_graph(graph_t *graph);

/*
 * Очищает память из-под списка смежности graph
 */
void free_graph(graph_t *graph);

/*
 * Ищет вершины недостижимые из данной по списку смежности
 * графа graph
 */
int find_by_node(graph_t *graph);

/*
 * Генерирует изображение графа по его списку смежности graph
 */
int graphviz(graph_t *graph);

/*
 * Удаляет элемент из очереди
 */
int aq_pop(queue_t *queue, uint *const node);

/*
 * Помещает элемент в очередь
 */
int aq_push(queue_t *queue, const uint node);

/*
 * Инициализирует очередь
```

```
 */
int queue_init(queue_t *queue, const uint num);
```

Описание алгоритма

Для хранения графа используется список смежности: каждой вершине соответствует индекс в динамическом массиве, по которому хранится список соседей этой вершины. Для ввода графа запрашивается количество вершин для выделения необходимого размера под динамический массив, количество ребер и пары вершин, задающие ребра.

Для поиска вершин, недостижимых из данной используется алгоритм обхода графа в ширину, запущенный из данной вершины. При данном обходе каждая непосещенная вершина записывается в очередь и помечается как посещенная, далее удаляется первая вершина в очереди и происходит переход к каждому непройденному соседу этой вершины. Алгоритм повторяется. Обход заканчивается, когда в очереди не остается вершин, то есть когда были посещены все вершины, достижимые от данной.

Для вывода изображения графа, формируется файл *.gv, представляющий код программы на языке DOT, по заданному файлу создается *.png изображение.

Тесты

№	Что проверяется	Входные данные	Результат
1	Неверный ввод пункта меню (не число)	j	Введенный код не соответствует ни одному действию. Попробуйте ещё раз.
2	Неверный ввод пункта меню (больше количества пунктов)	4	Введенный код не соответствует ни одному действию. Попробуйте ещё раз.
3	Неверный ввод пункта меню (отрицательное значение)	-10	Введенный код не соответствует ни одному действию. Попробуйте ещё раз.
4	Неверный ввод пункта меню при выборе формата задания файла (не число)	j	Такого пункта меня нет! Попробуйте ещё раз!
5	Неверный ввод пункта меню при выборе формата задания файла (больше количества пунктов)	3	Такого пункта меня нет! Попробуйте ещё раз!
6	Неверный ввод пункта меню при выборе формата задания файла (отрицательное)	-6	Такого пункта меня нет! Попробуйте ещё раз!
7	Ввод имени файла, который не существует	file	Ошибка при открытии файла!
8	Неверный ввод количества вершин (не число)	j	Ошибка при чтении числа!
9	Неверный ввод количества вершин (неверное число)	0	Введенное число выходит за допустимый диапазон!
10	Неверный ввод количества ребер (не число)	j	Ошибка при чтении числа!
11	Неверный ввод количества ребер (неверное число)	-1	Введенное число выходит за допустимый диапазон!

12	Неверный ввод номеров вершин при задании ребер (не число)	O 5 2 3	Ошибка при чтении числа!
13	Неверный ввод номеров вершин при задании ребер (неверное число)	3 1 1 2	Ошибка при чтении ребра!
14	Попытка просмотра пустого графа	2	Граф не задан!
15	Попытка нахождения недостижимых вершин в пустом графе	3	Граф не задан!
16	Неверный номер вершины, из которой ищутся недостижимые	8	Номер вершины выходит за допустимый диапазон!

Контрольные вопросы

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).

Как представляются графы в памяти?

С помощью матрицы смежности (элемент с индексами [i][j] равен 1, если есть ребро между вершинами i и j, и равен 0, если ребра нет) или списков смежности (для каждой вершины хранится список её соседей – вершин, к которым идет ребро от заданной).

Какие операции возможны над графами?

Обход графа, поиск путей (кратчайших, эйлеровых, гамильтоновых), удаление/добавление вершины/ребра.

Какие способы обхода графов существуют?

Обход в глубину (просматривается один из непросмотренных соседей вершины до тех пор, пока не встретится просмотренная ранее вершина или не закончится список смежности, если новых вершин нет, происходит возвращение к предыдущей) и обход в ширину (аналогично обходу в глубину только просматриваются сразу все соседи вершины).

Где используются графовые структуры?

Графовые структуры используется при наличии у элементов каких-либо связей, так они могут использоваться при анализе позиционных игр, составлении транспортных путей, генеалогических деревьев и т. п.

Какие пути в графе Вы знаете?

Простые, замкнутые, полные, эйлеровые (один раз через каждое ребро), гамильтоновые (один раз через каждую вершину).

Что такое каркасы графа?

Каркас графа – это подграф, который содержит все вершины графа и является деревом.

Вывод

Список смежности схож по своему строению с матрицей смежности, но не хранит информацию об отсутствующих ребрах, также не затрачивается время на просмотр несуществующих ребер, что ускоряет процесс обхода графа. Для нахождения недостижимых вершин из заданной был использован обход графа в ширину, имеющий эффективность по времени $O(V + E)$, где V– количество вершин, E– количество ребер. Алгоритм нахождения недостижимых вершин из заданной может быть использован для подборки рекомендаций в социальных сетях, онлайн-магазинах и т. п. Например, если человек посетил страницу одного товара, но не был на странице похожего товара, то алгоритм сможет найти такие товары и предложить их пользователю.