**Содержание**

[Введение 4](#_Toc10630669)

[Исходные данные 5](#_Toc10630670)

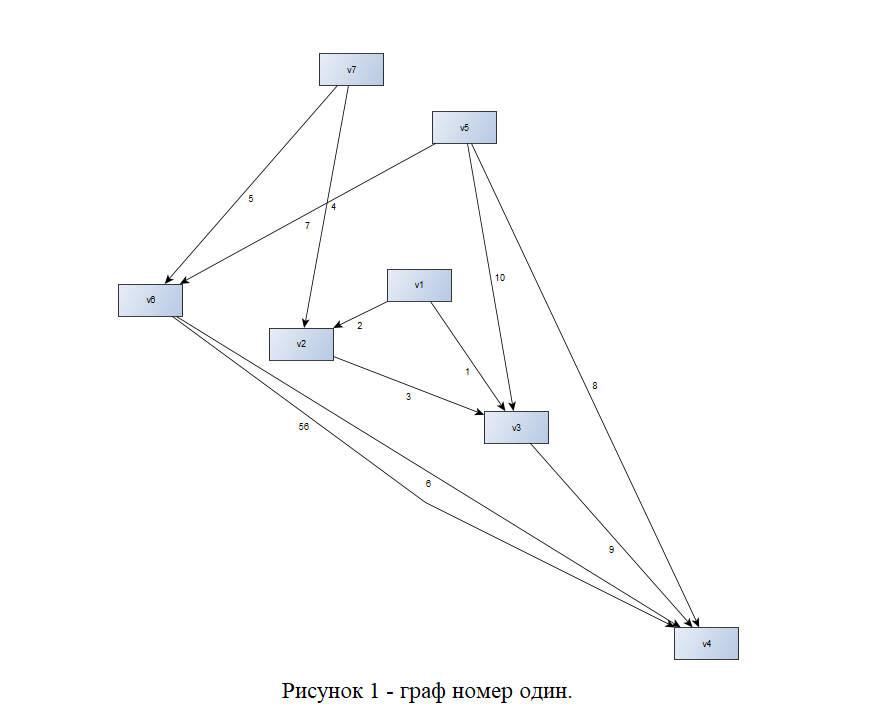
[Алгоритм 7](#_Toc10630671)

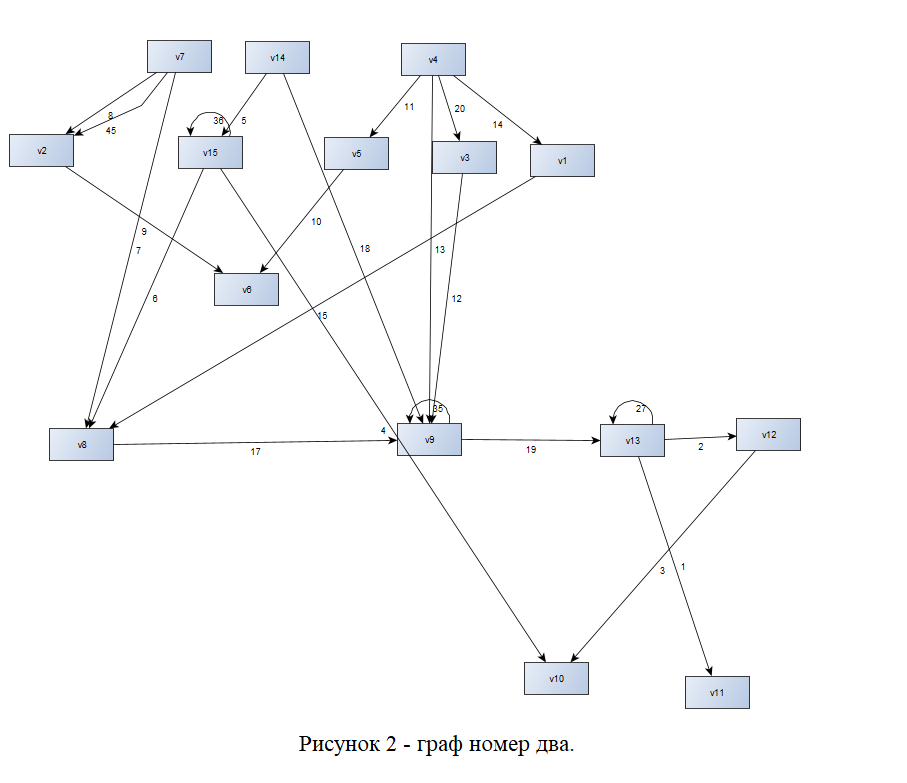
[Cписок использованных источников 15](#_Toc10630672)

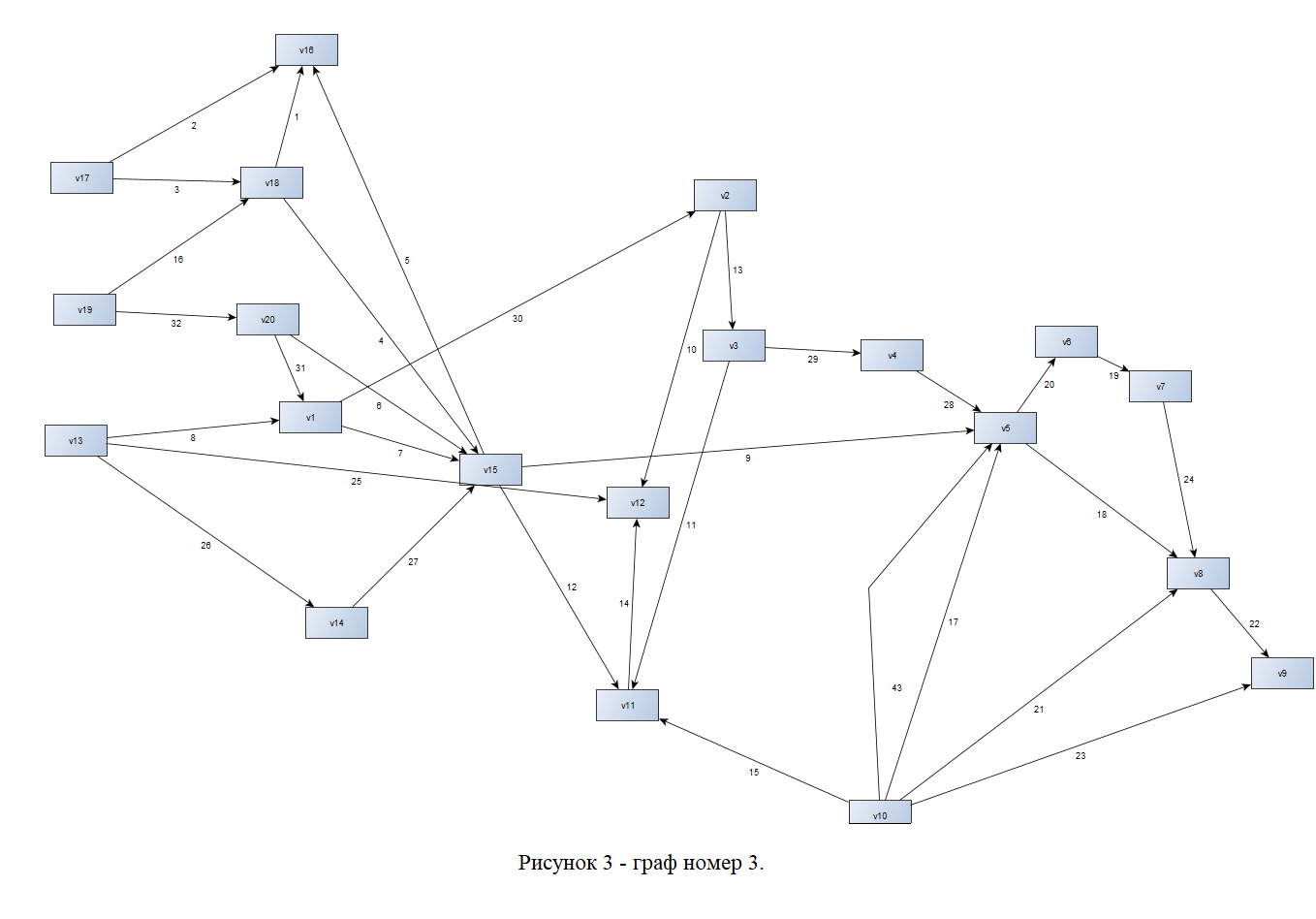
# Введение

Орграф G' = (V, Е') называется минимальным эквивалентным орграфом для орграфа G = (V, Е), если Е' — наименьшее подмножество множества Е (E’ ⊆ E) такое что транзитивные замыкания обоих орграфов G и G' совпадают (причем если граф G ацикличен, то для него существует только один минимальный эквивалентный .Написать программу нахождения минимального эквивалентного орграфа. Представить данный орграф в виде списка дуг.

# Исходные данные

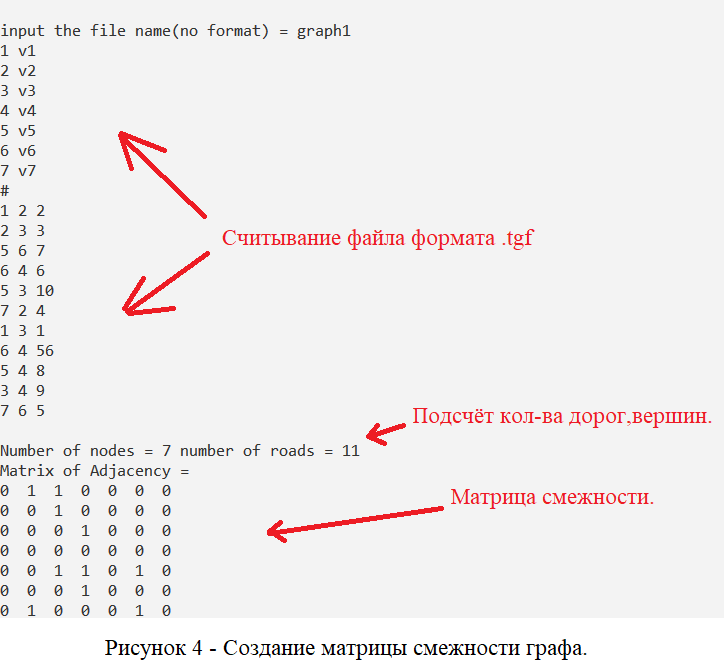






# Алгоритм

Рассмотрим алгоритм нахождения минимального эквивалентного графа для графа 1(рисунок 1).Для начала построим матрицу смежности(рисунок 4):

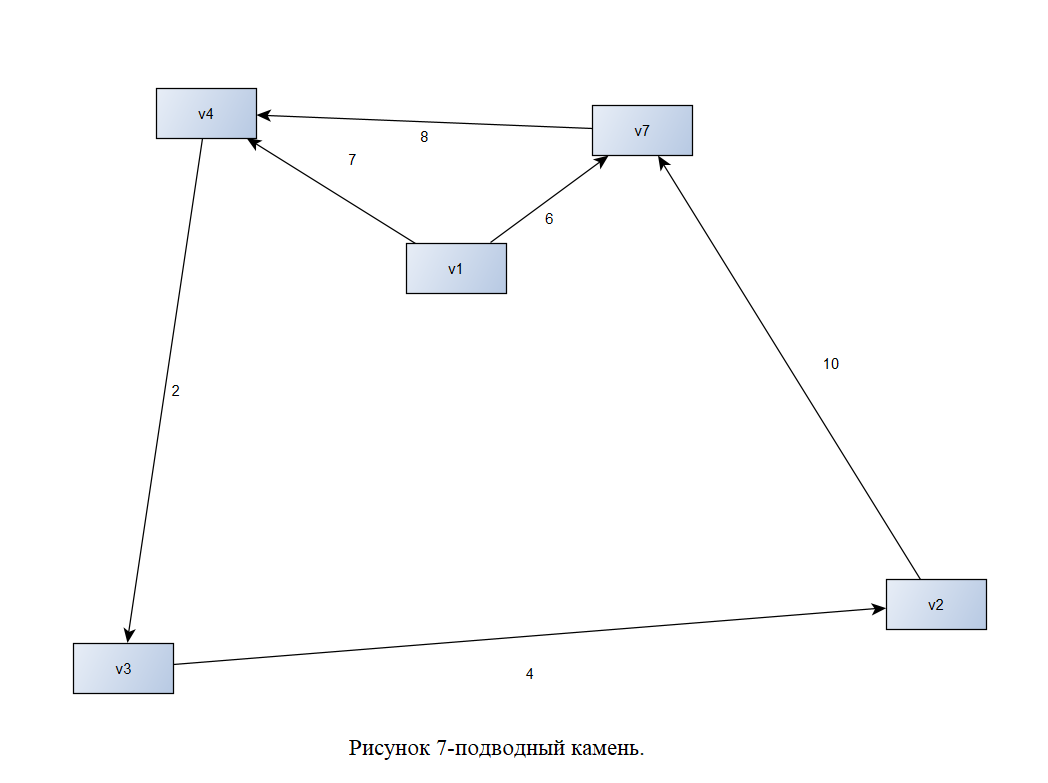


Я считываю из файла данные(рисунок 4) и заношу их в матрицу смежности. Аналогично я делаю и для таблицы инцидентности(рисунок 5), а затем, по алгоритму Уоршелла(рисунок 6) , я строю матрицу достижимости (рисунок 5):

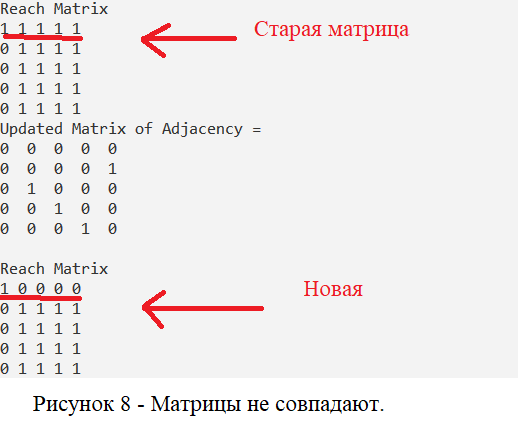




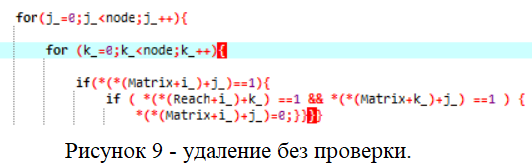
После того как я закончил строил матрицы, непосредственно начинаю выполнять задание.Я удаляю “вспомогательные ненужные дуги”. Например, если есть дуги из v1 в v5 и из v5 в v3,значит у нас уже есть пути из v1 в v5 и из v1 в v3. Следоватеьно,если есть такая дуга v1 v3, то её можно удалить.Однако не всё так просто (рисунок 7):



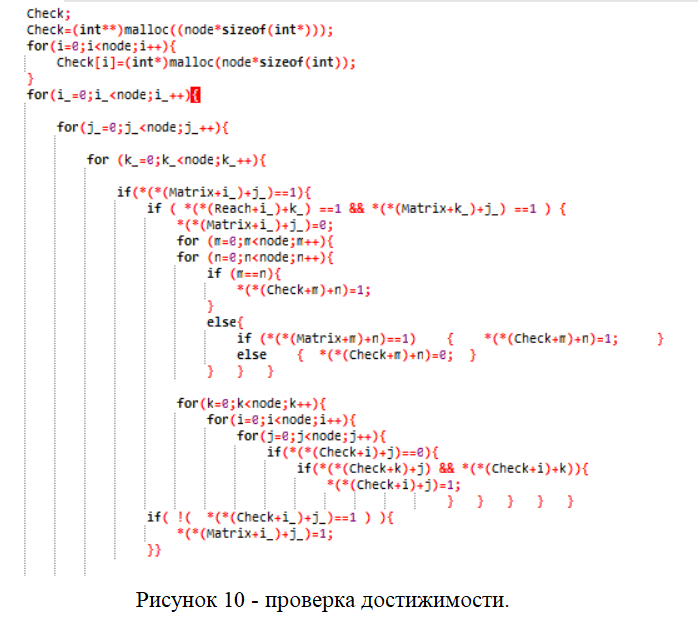
Моя программа записывает пошагово новую матрицу смежности и достижимости (Рисунок 8), и первая таблица достижимости,в данном случае принадлежит СТАРОЙ матрице смежности,а значит если мы будем просто удалять дуги, и сравнивать со старой матрицей, тогда у нас получится следующий результат для рисунка 7:



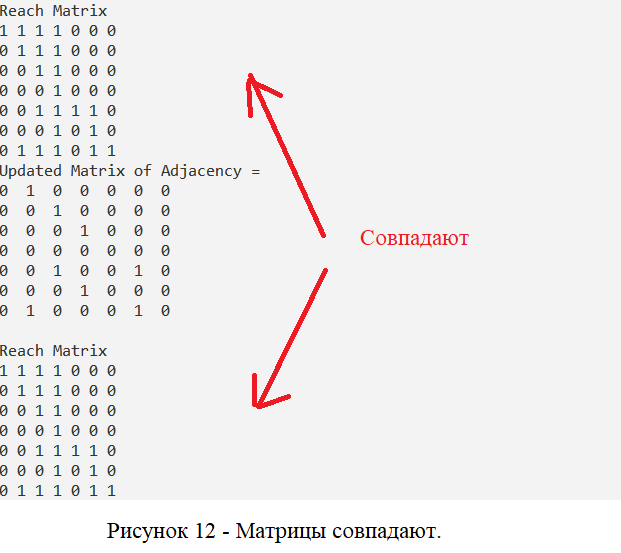
Это потому что программа сверяется со старой,неизменяемой матрицей достижимости,как показано на рисунке 9(Reach-старая матрица достижимости, Matrix- матрица смежности):



Следовательно (Рисунок 7), программа удалит по алгоритму сначала дугу 7,так как есть уже другой путь из v1 в v3 (v1-v7-v2-v3-v4) и дугу в v4,а потом дугу 6,так как у нас всё ещё есть путь из v1 в v7(?) (v1-v4-v3-v2-v7) и дуга из v7 в v4.Но тогда получается,что из v1 нет больше путей.Поэтому необходимо проводить Матрицу проверки достижимости(Check). Обратим внимание на рисунок 10:



Я создаю новую матрицу достижимости каждый раз и сверяюсь с полученными значениями,чтобы не оплошать. Результат налицо(рисунок 12):

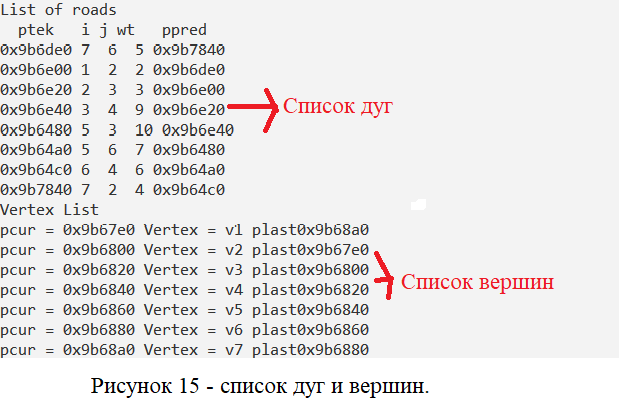


Примечание: Updated Matrix of Adjacency-это новая матрица смежности с удалёнными вспомогательными дугами, петлями, дополнительными дугами и т.д. В итоге мы реализуем список дуг и список вершин,как список элементов(рисунки 13 и 14):





На рисунке 15 показан вывод этих списков в консоль:



Также мною были реализованы функции добавления/ удаления/изменения вершин/дуг,работа с динамической памятью, а также конструкторы/деструкторы классов и наследование. Файл с работой прилагается.

# Cписок использованных источников

1.Ссылка на скачивание yEd Graph Editor: <https://www.yworks.com/products/yed>

2.Dev-C++ 5.11,ссылка на скачивание: <https://soft.mydiv.net/win/files-DEV-C.html>