Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления
Кафедра Интеллектуальных информационных технологий

РАСЧЕТНАЯ РАБОТА

по дисциплине «Традиционные и интеллектуальные информационные технологии»

на тему

Найти множество вершин, удаление которых приводит к увеличению числа компонентов связности ориентированного графа.

Выполнил: Р. Д. Гергисевич

Студент группы
321702

Проверил: Н. В. Малиновская

Минск 2024

Содержание

1 Введение

Цель: Получить навыки формализации и обработки информации с использованием семантических сетей

Задача: Найти множество вершин, удаление которых приводит к увеличению числа компонентов связности ориентированного графа.

2 Список понятий

1. ***Неориентированный граф***(абсолютное понятие)-граф, в котором все ребра являются звеньями, то есть порядок двух концов ребра графа не существенен

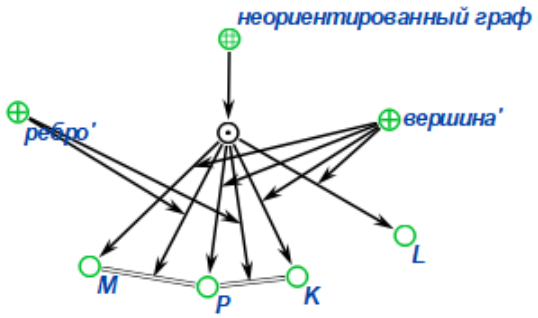

Рис. 1: (Неориентированный гарф) Полный вариант представления

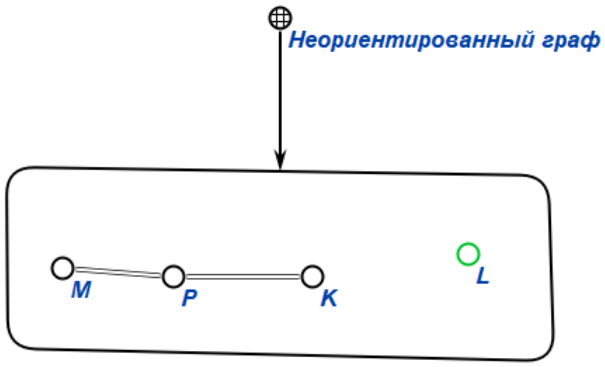

Рис. 2: (Неориентированный гарф) Сокращённый вариант представления

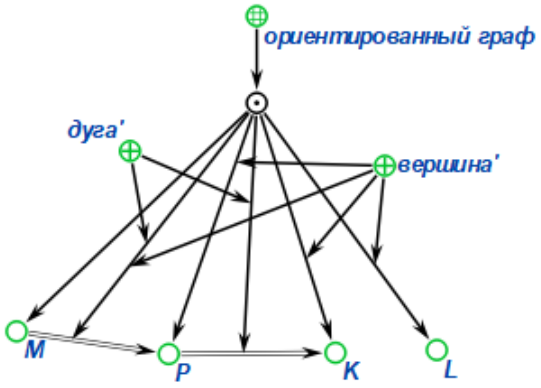

Рис. 3: (Ориентированный гарф) Полный вариант представления

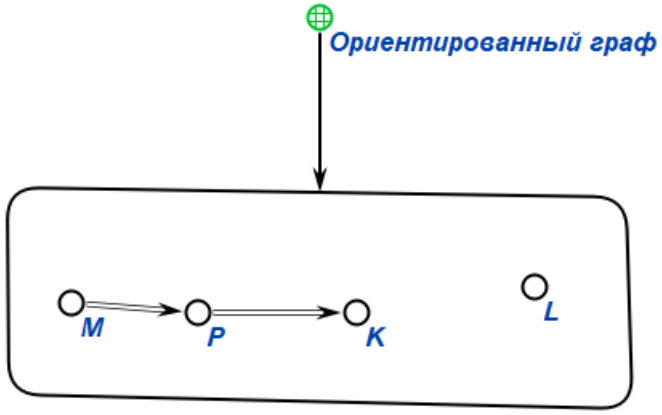

Рис. 4: (Ориентированный гарф) Сокращённый вариант представления

2. ***Компонентами связности*** графа называется множество вершин графа достижимых попарно и рёбра их связывающие.

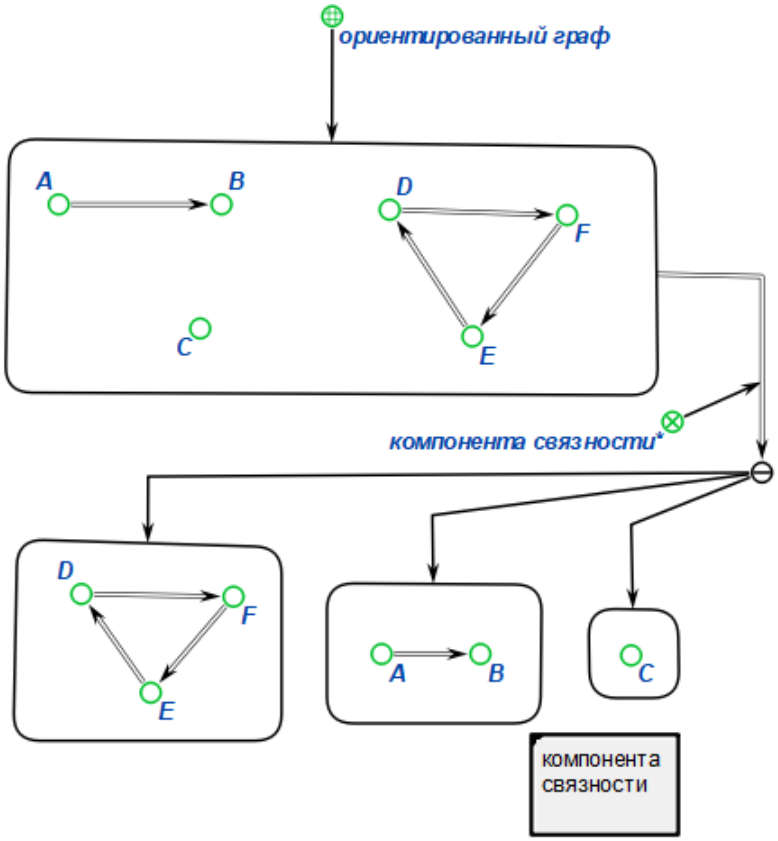

Рис. 5: Абсолютное понятие компонентов связности

3 Тестовые примеры

3.1 Тест 1

Вход: Найти множество вершин, удаление которых приводит к увеличению числа компонентов связности ориентированного графа.

Переменная **Count** хранит число компонетов связности входного графа.

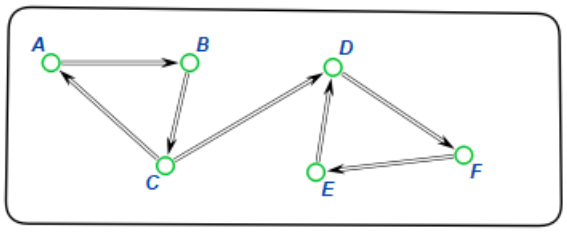

Рис. 6: Вход теста 1

Выход: Удаление вершин C и D приводит к увеличению числа компонентов связности хроняхщихся в переменной **Count**.

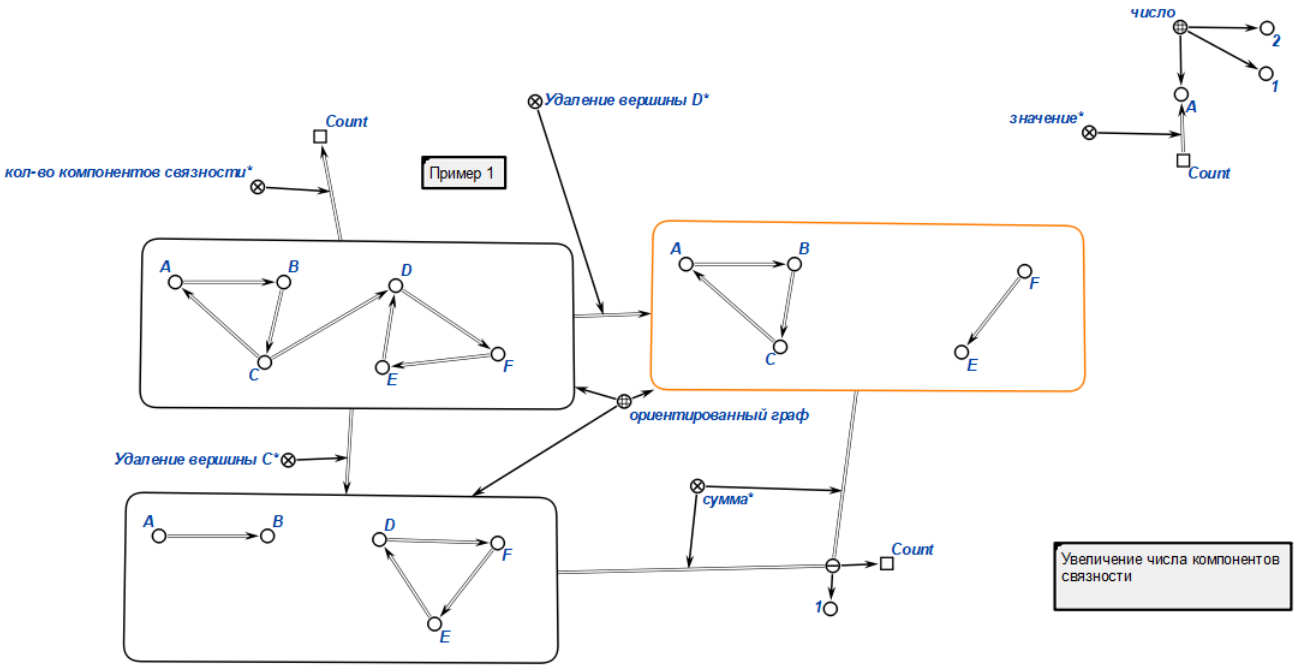

Рис. 7: Выход теста 1

3.2 Тест 2

Вход: Найти множество вершин, удаление которых приводит к увеличению числа компонентов связности ориентированного графа.

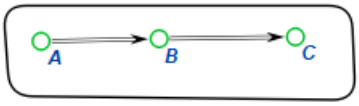

Рис. 8: Вход теста 2

Выход: Вершина B.

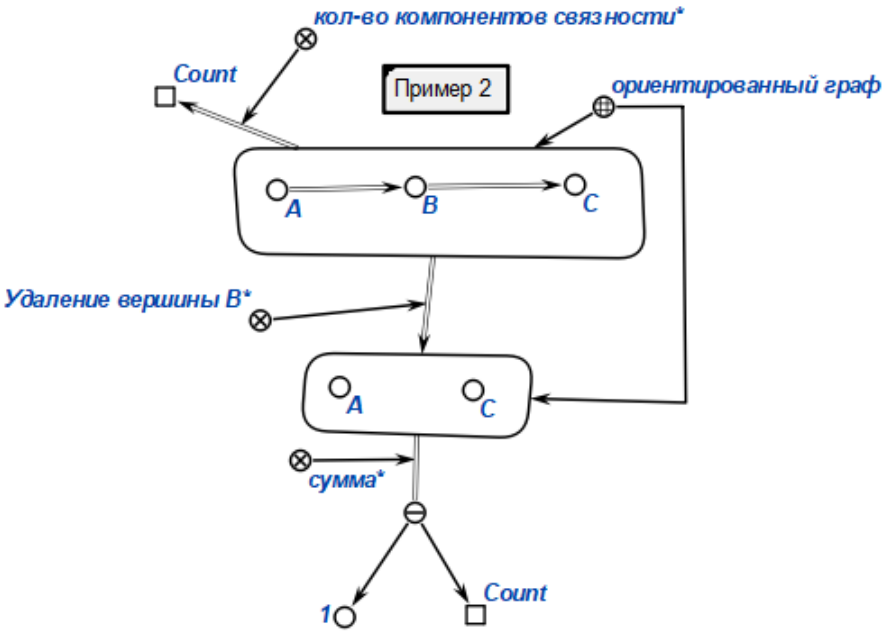

Рис. 9: Выход теста 2

3.3 Тест 3

Вход: Найти множество вершин, удаление которых приводит к увеличению числа компонентов связности ориентированного графа.

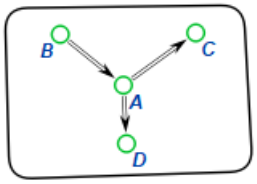

Рис. 10: Вход теста 3

Выход: Вершина A.

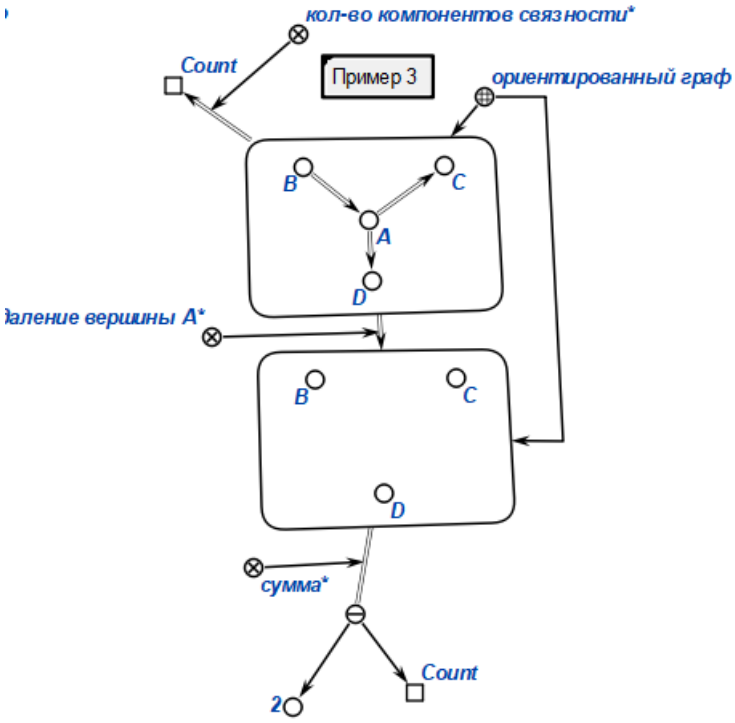

Рис. 11: Выход теста 3

3.4 Тест 4

Вход: Найти множество вершин, удаление которых приводит к увеличению числа компонентов связности ориентированного графа.

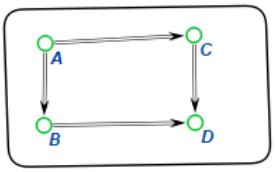

Рис. 12: Вход теста 4

Выход: Удаление любой из вершин приводит к сохранению того же числа компонентов связности и равно **Count**.

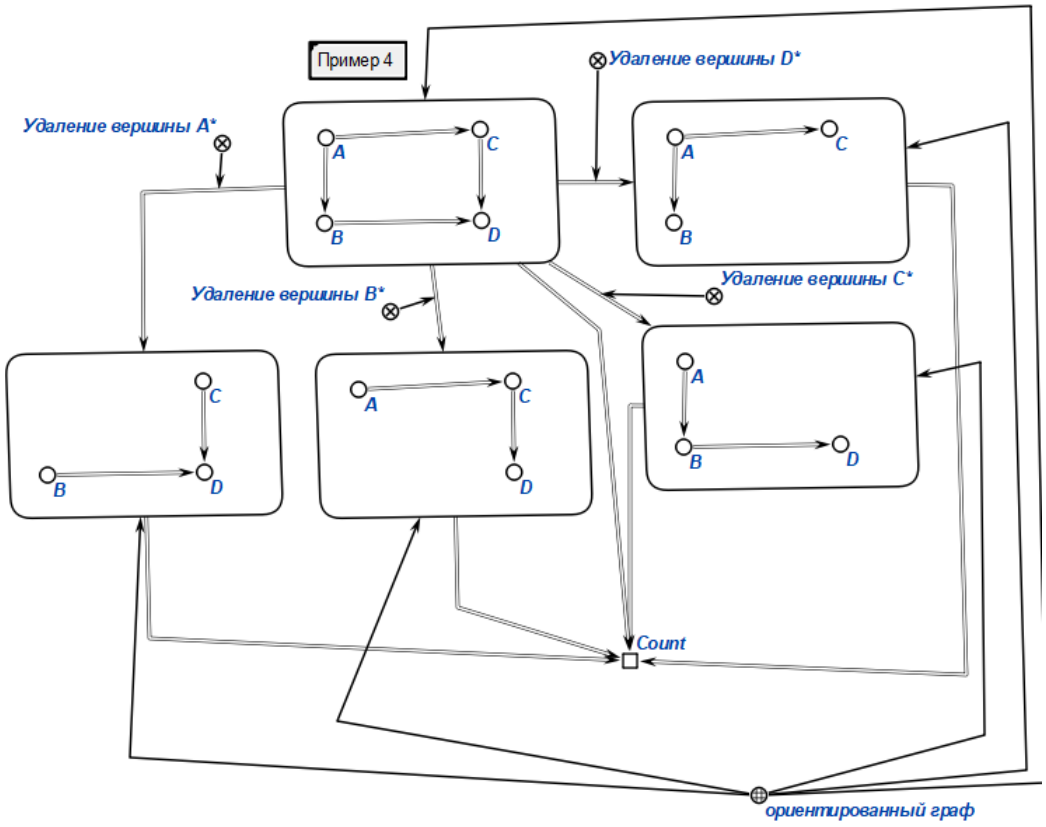

Рис. 13: Выход теста 4

3.5 Тест 5

Вход: Найти множество вершин, удаление которых приводит к увеличению числа компонентов связности ориентированного графа.

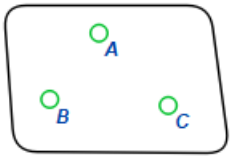

Рис. 14: Вход теста 5
В пользовательском графе 3 отдельно стоящие вершины, что значит что **Count** хранит 3 компоненты связнсти.

Выход: Удаление вершин не приводит к увеличению числа компонентов связности.

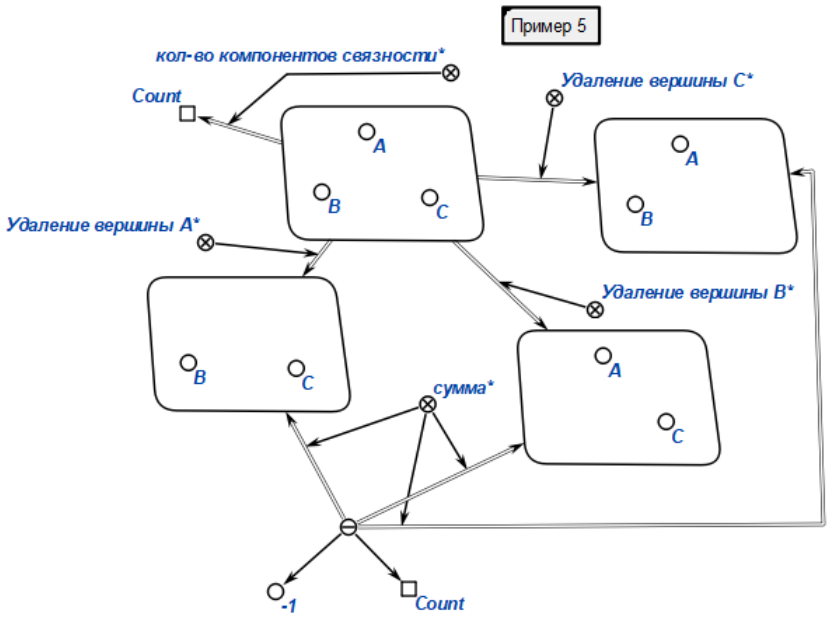

Рис. 15: Выход теста 5

3.6 Демонстрация работы алгоритма на пользовательском графе

1. Получаем в качестве входного графа пользовательский граф. Создаём неориентированный граф из входного.
2. Создаём переменную *BaseConectCount* Которая будет хранить количество компонентов связности во входном графе, счетчик *ConectCount* котрый будеи считать количество компонентов связности в кажлом графе и *ResultVertex* хранящий результат.

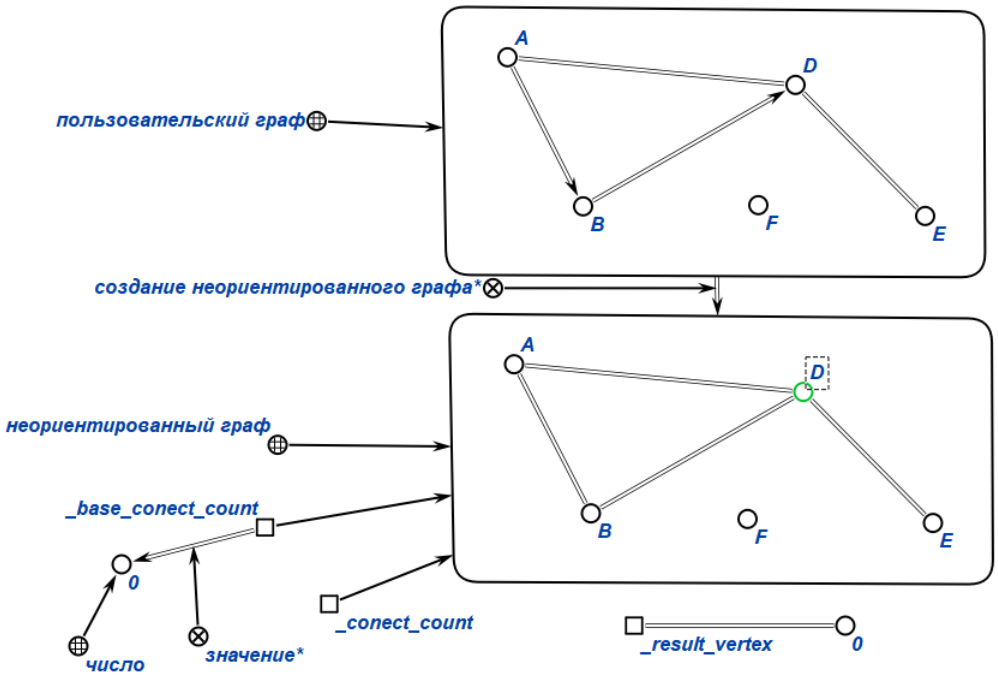

Рис. 16: Действие 1 и 2

3. Создаём пременную ***UnvisitedVertex***- стэк из непосещённых вершин, ;

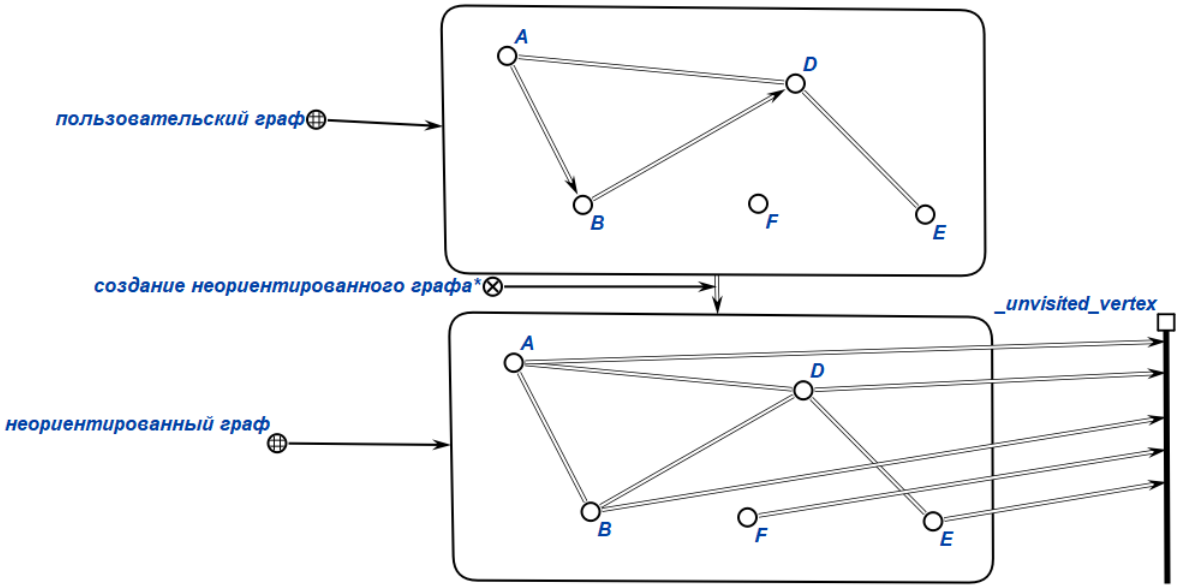

Рис. 17: Действие 3

4. Создаём стэк *deleteVetexCount* который будет поочереди удалять каждую вершину входного графа;

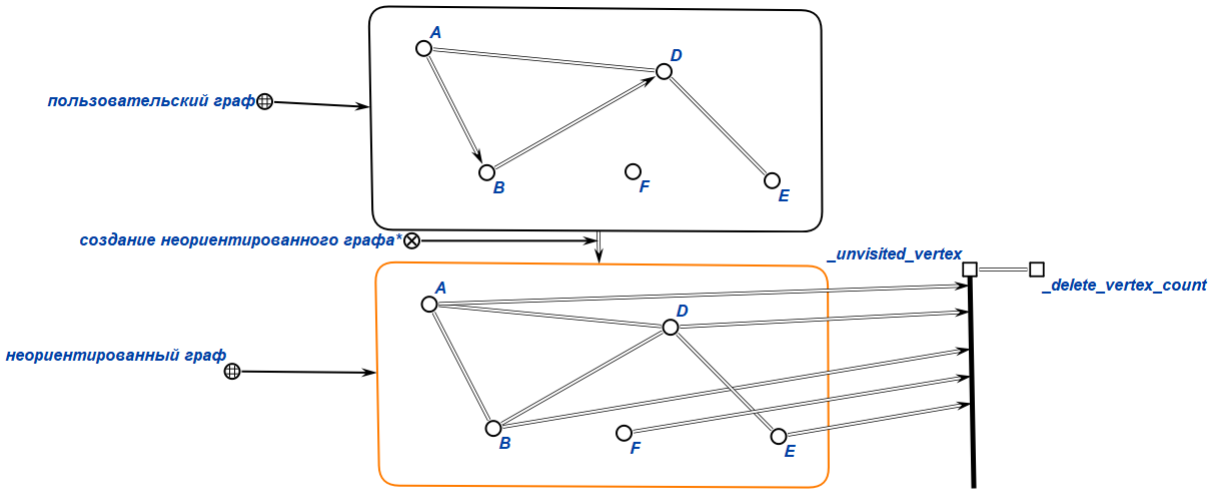

Рис. 18: Действие 4

5. Выбираем и удаляем из стэка первую вершину **inputGraph**; **endVertex** показывает последнюю вершину до которой есть связаные с нашей вершиной рёбра.

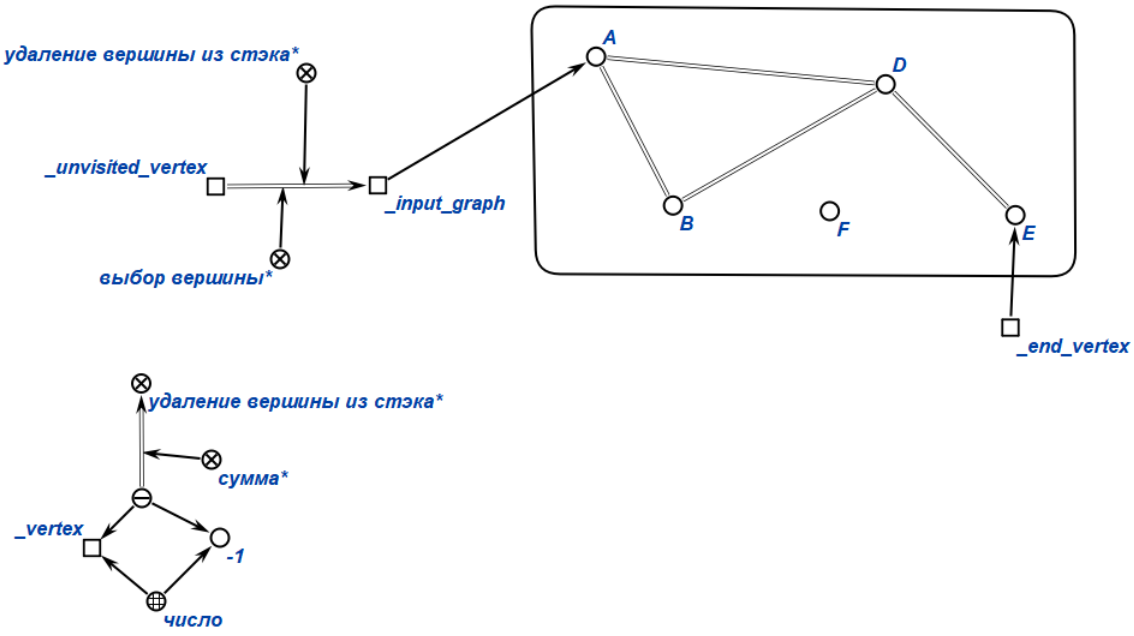

Рис. 19: Действие 5

6. После достижения конечной вершины количество компонентов связности *conectCount* уведичивается на 1 (изначально в переменной хранится 0); А также каждая посещённая вершина удаляется из стэка.

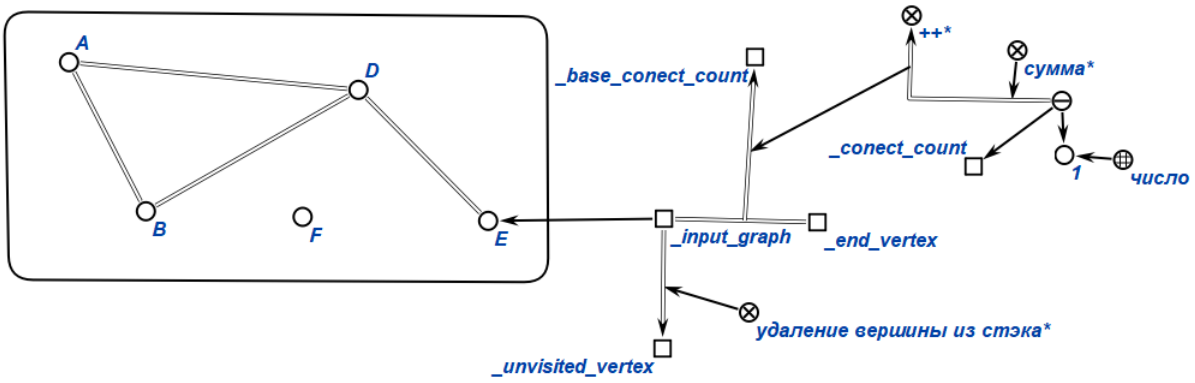

Рис. 20: Действие 6 (увеличение conectCount ври достижении endVertex)

7. При нахождении отдельно стоящей вершины количество компонентов связеости также увеличивается на 1;

8. При достижении стэком **unvisitetVertex** нуля количество компонентов связности из **conectCount** преходит в переменную **baseConectCount**. Это означает что теперь входной граф содержит данное количество компонентов связности

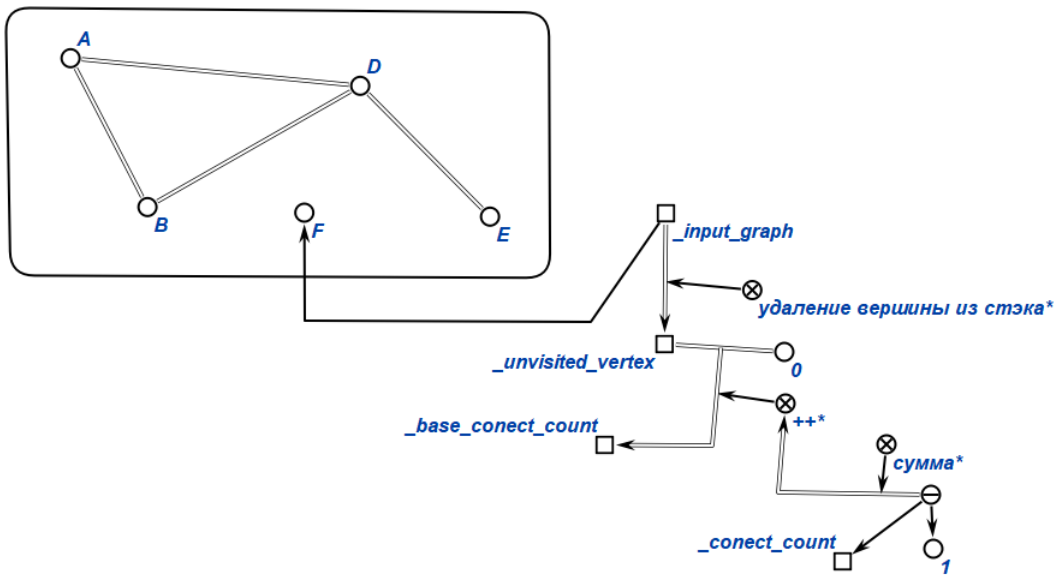

Рис. 21: Действие 7 и 8

9. Количество компонентов связеости в взодном графе равно 2.

Рис. 22: Действие 9

10. Выбераем первую вершину из стэка **deleteVertexCount** и удаляем ее из входного графа для проверки на количество компонентов связности.
11. С полученный графом повторяем теже действия что и с входным. Полученное значение **conectCount** сравниваем с **baseConectCount**. До тех пор пока количество компонентов связности данного графа не больше чем у входного -переходим к последующей вершине из **deleteVertexCount**.

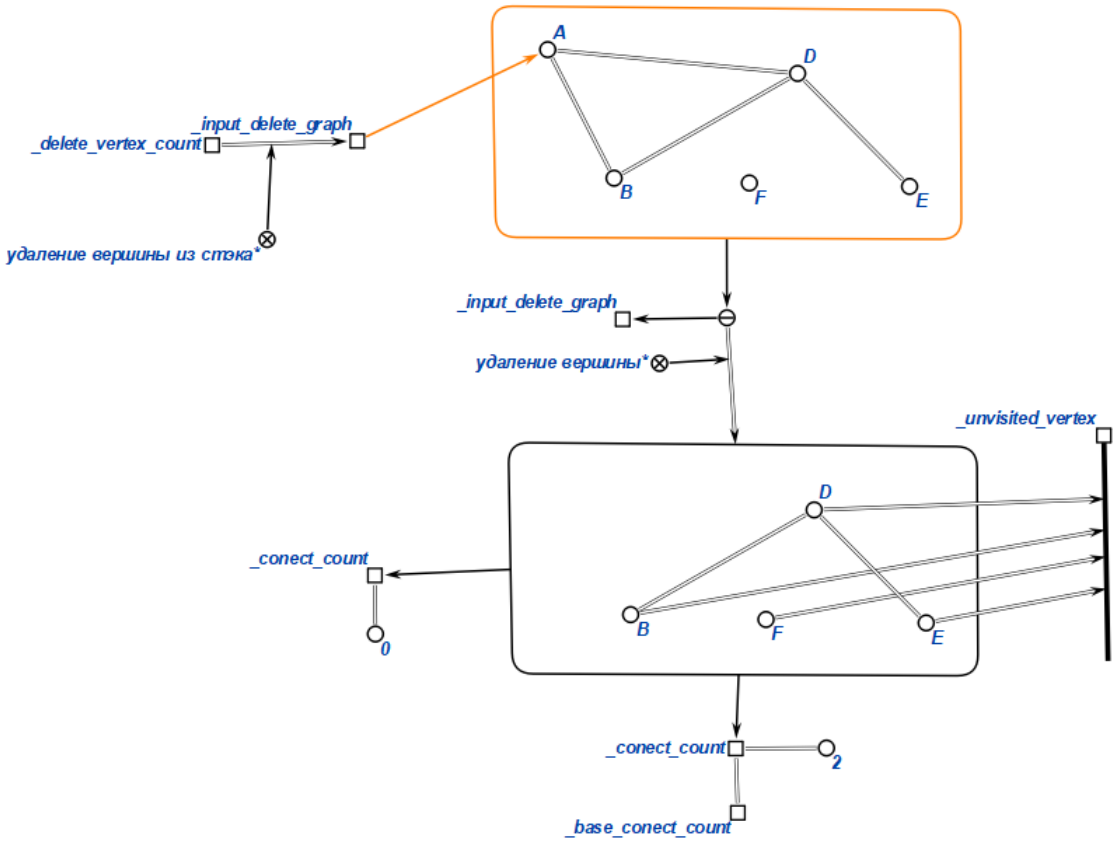

Рис. 23: Действие 9 и 10

12. Повторяем алгоритм пока стэк **deleteVertexCount** не опустеет. Если значение в каком-либо графе окажется больше чем во входном - то эта вершина добавляется в **resultVertex** для дальнейшего вывода результата

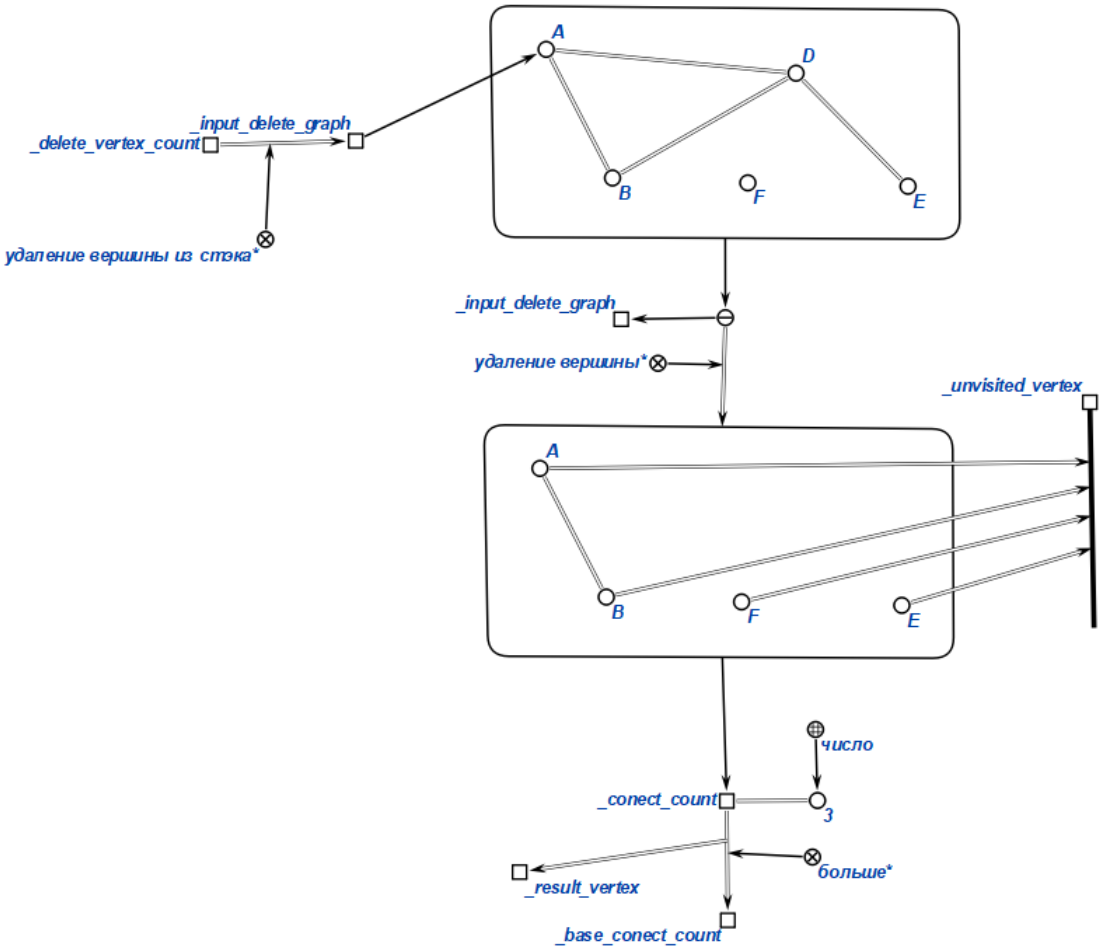

Рис. 24: Действие 11 (Повторение алгоритма)

13. Вывод хранящейся в переменной **resulVertex** информации как результат выполнения задачи.

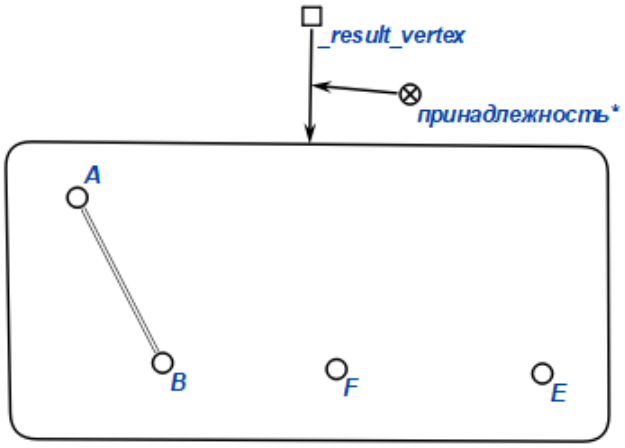

Рис. 25: Действие 13 (Вывод результата)

4 Заключение

В заключении у нас получилось формализовать поставленную задачу. Мы нашли все необходимые вершины, удаление которых приводит к увеличениб числа компонентов связности. Реализовали алгоритм его создания, который работает на любом графе.

5 Список используемых источников

[1] Кормен, Д. Алгоритмы. Построение и анализ / Д. Кормен. — Вильямс, 2015. — P. 1328.i
[2] Кузнецов, О. П. Дискретная математика для инженера / О. П. Кузнецов, Г. М. Адельсон-Вельский. — Энергоатомиздат, 1988. — P. 480.
[3] Оре, О. Теория графов / О. Оре. — Наука, 1980. — P. 336.
[4] Харарри, Ф. Теория графов / Ф. Харарри. — Эдиториал УРСС, 2018. — P. 304.