

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 кол-во компонент связности в неориентированном графе

Выполнил:

П. И. Кадиков

Студент группы
321702

Проверила:

Н. В. Малиновская

Минск 2024

1 Введение

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

Задача: Найти кол-во компонент связности в неориентированном графе.

2 Список понятий

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

- (a) Вершина (относительное понятие, ролевое отношение);
- (b) Связка (относительное понятие, ролевое отношение).

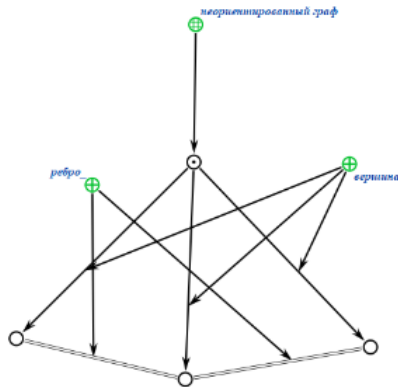


Рис. 1: Абсолютное понятие неориентированного графа

- (а) Вершина (относительное понятие, ролевое отношение);
- (б) Ребро (относительное понятие, ролевое отношение).

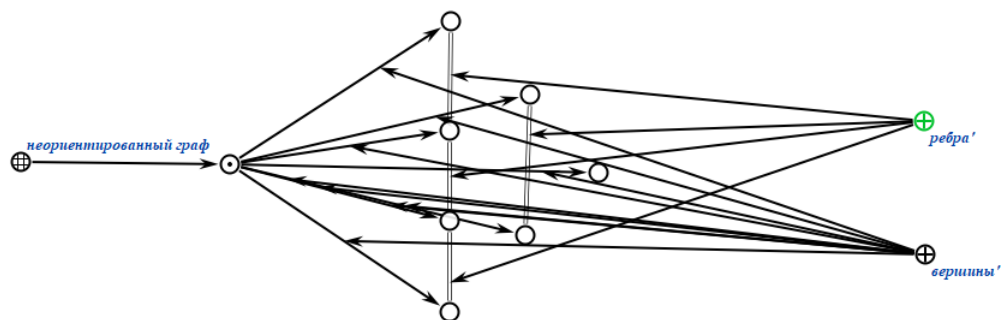


Рис. 2: Компоненты связности(3)

3 Тестовые примеры

Во всех тестах графы будут приведены в сокращенной форме со скрытыми ролями элементов графа.

3.1 Тест 1

Вход: Необходимо найти количество компонент связности неориентированного графа.

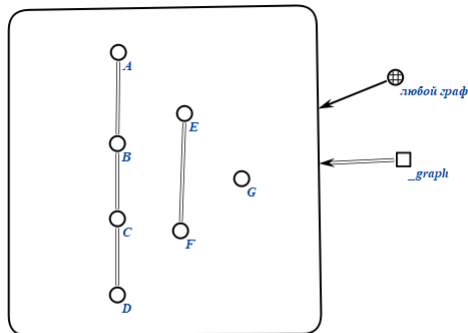


Рис. 3: Вход теста 1

Выход: Результатом станет 3, так как граф имеет только 3 компоненты связности.

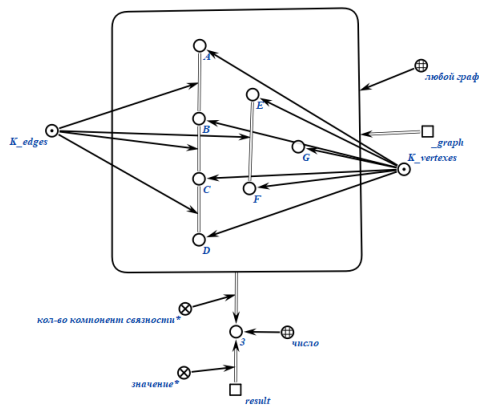


Рис. 4: Выход теста 1

3.2 Тест 2

Вход: Необходимо найти количество компонент связности неориентированного графа.

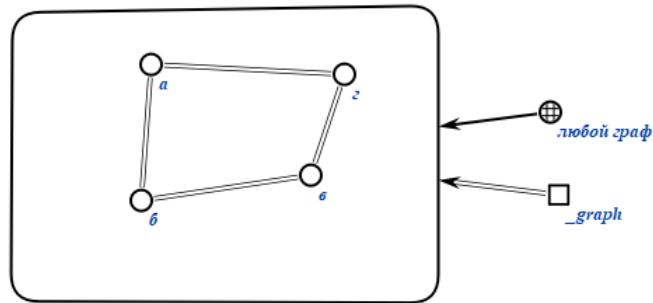


Рис. 5: Вход теста 2

Выход: Результатом станет 1, так как граф имеет только 1-у компоненту связности.

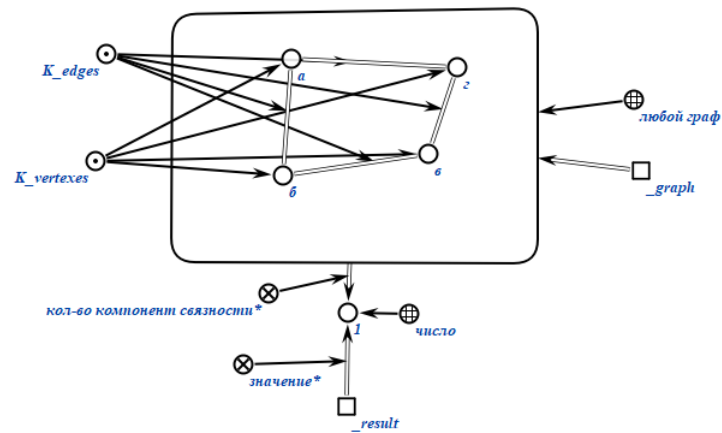


Рис. 6: Выход теста 2

3.3 Тест 3

Вход: Необходимо найти количество компонент связности неориентированного графа.

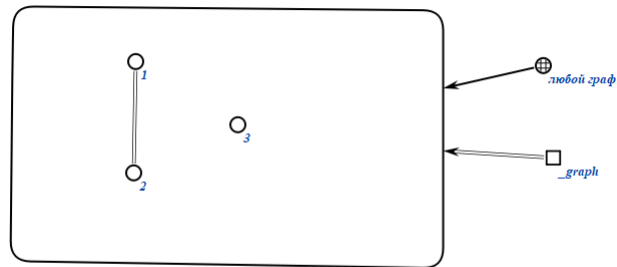


Рис. 7: Вход теста 3

Выход: Результатом станет 2, так как граф имеет только 2 компоненты связности.

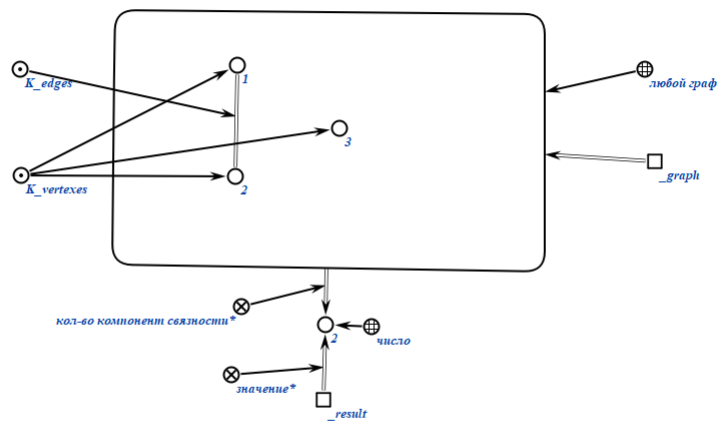


Рис. 8: Выход теста 3

3.4 Тест 4

Вход: Необходимо найти количество компонент связности неориентированного графа.

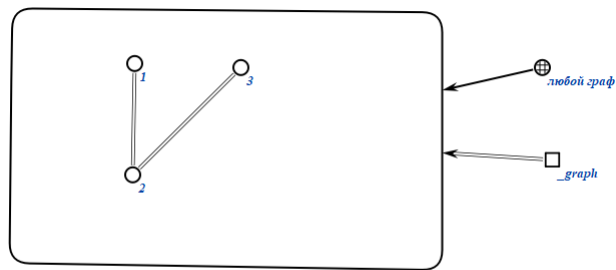


Рис. 9: Вход теста 4

Выход: Результатом станет 1, так как граф имеет только 1 компоненту связности.

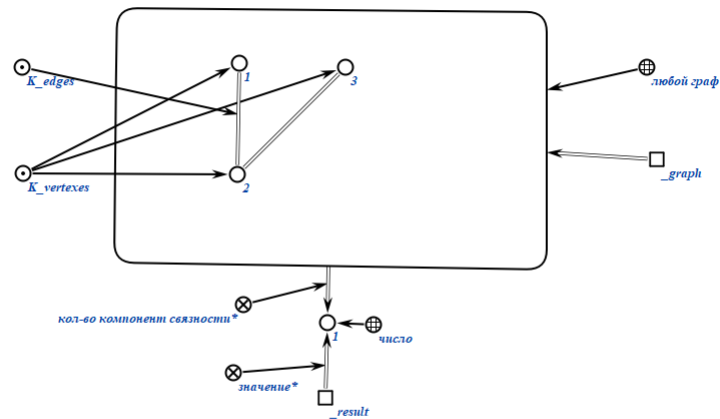


Рис. 10: Выход теста 4

3.5 Тест 5

Вход: Необходимо найти количество компонент связности неориентированного графа.

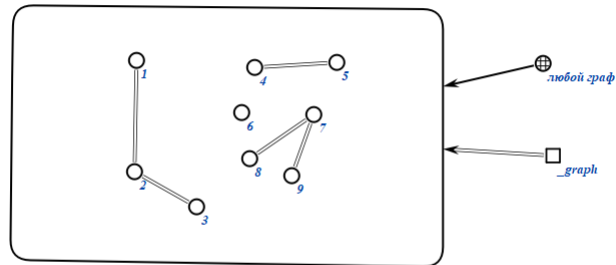


Рис. 11: Вход теста 5

Выход: Результатом станет 4, так как граф имеет только 4 компоненты связности.

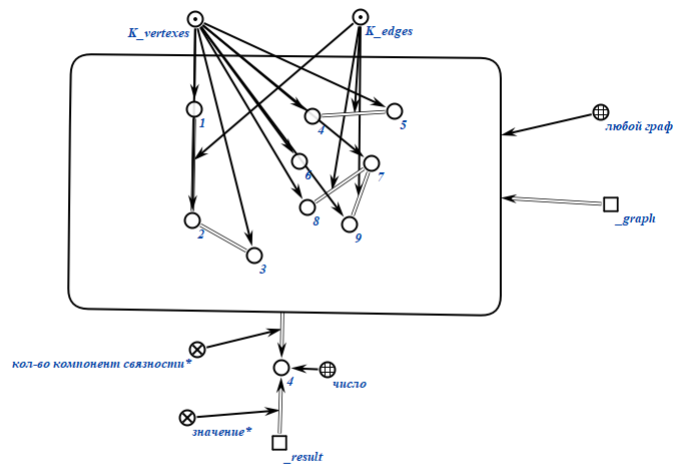


Рис. 12: Выход теста 5

4 Пример работы алгоритма в семантической памяти

(а) Входной граф.

`_graph` получит в качестве значения sc-узел неориентированного графа:

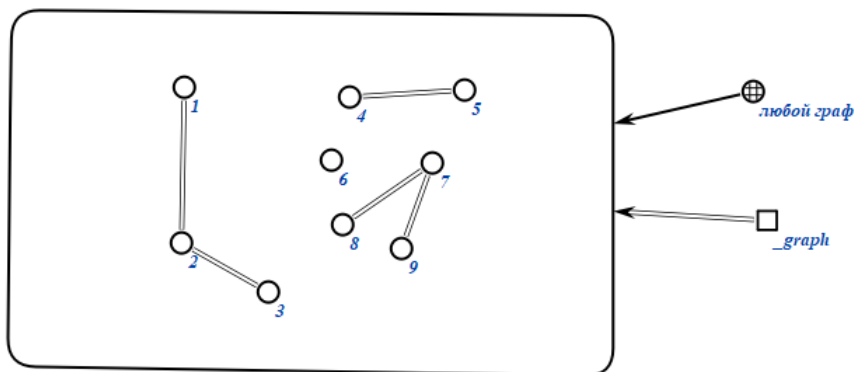


Рис. 13: Входной граф

- (b) Добавляем все вершины графа во множество непосещенных вершин.
 Переменная `_not_checked_vertexes` получит в качестве значения множество непроверенных вершин обрабатываемого графа.

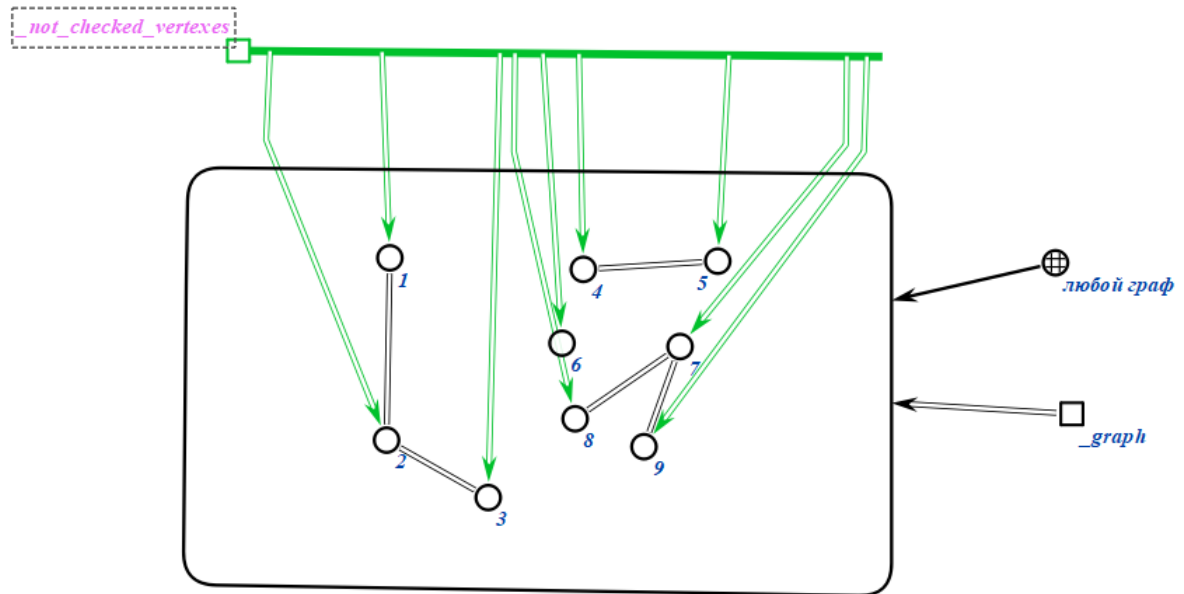


Рис. 14: Непосещённые вершины

- (с) Добавляем все вершины во множество конечных вершин, которые имеют либо 1 ребро, либо вообще не имеют ребер
 Переменная `_finite_vertexes` получит в качестве значения множество конечных вершин обрабатываемого графа.

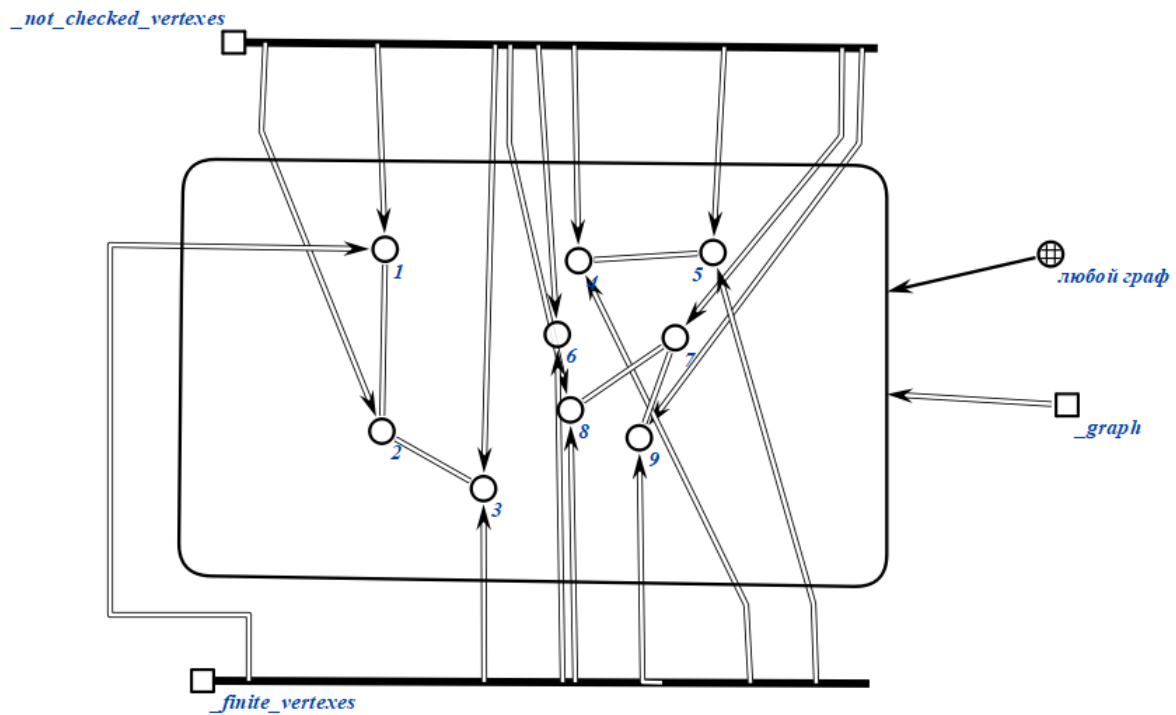


Рис. 15: Конечные вершины

(d) Создаём счетчик кол-ва компонент связности неориентированного графа.

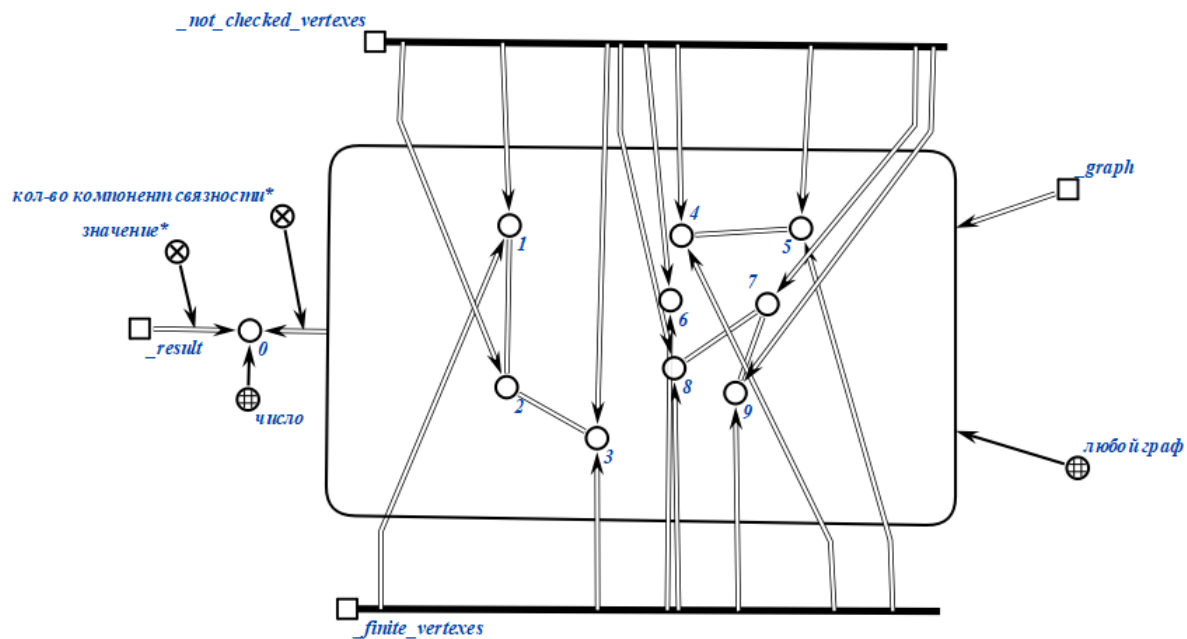


Рис. 16: Счётчик комопнент связности

(е) Создаем переменную `_checked_vertexes`, которая будет хранить в себе посещенные вершины.

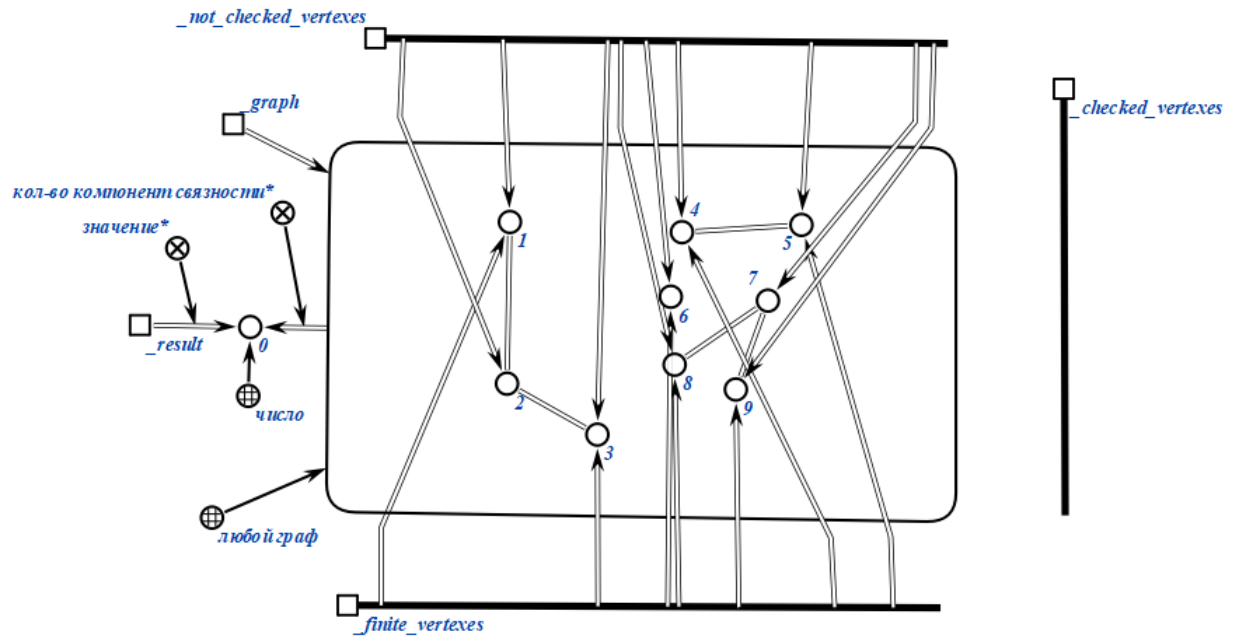


Рис. 17: Посещенные вершины

- (f) Начинаем поиск кол-ва компонент связности с вершины под индексом 1, одновременно занося ее во мн-во посещенных вершин и удаляя из множества непосещенных вершин.

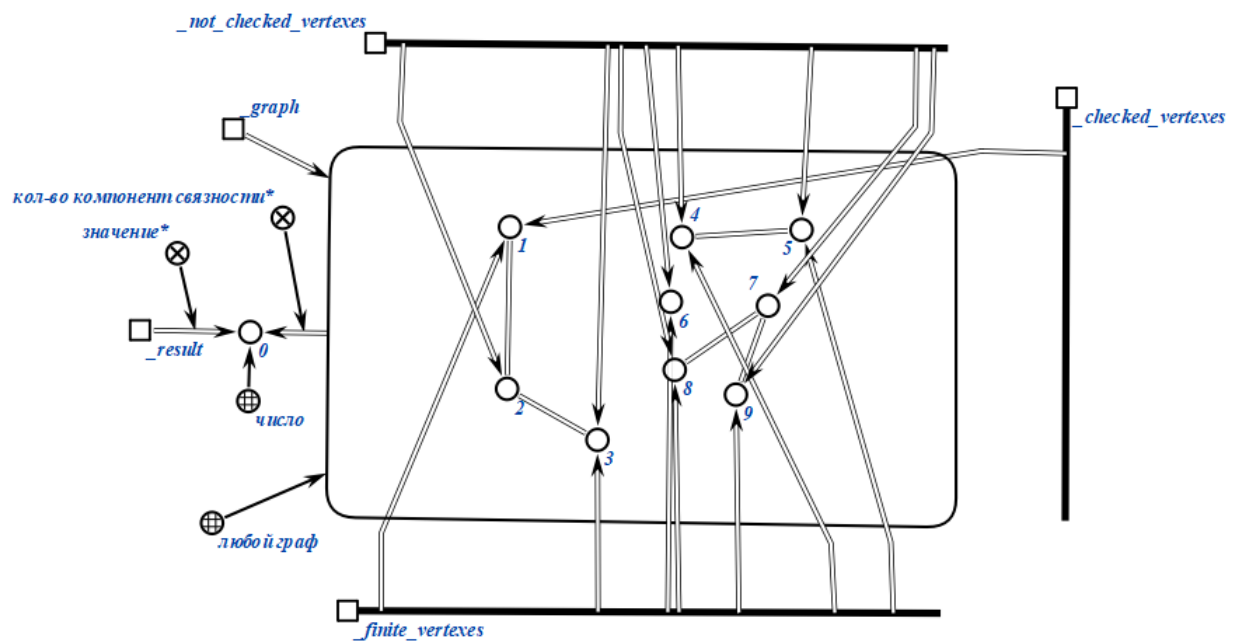


Рис. 18: Начало поиска из вершины с индексом 1

- (g) Идём из вершины с индексом 1 к вершине с индексом 2. Заносим вершину с индексом 2 во множество посещенных вершин и удаляем ее же из мн-ва непосещенных вершин.

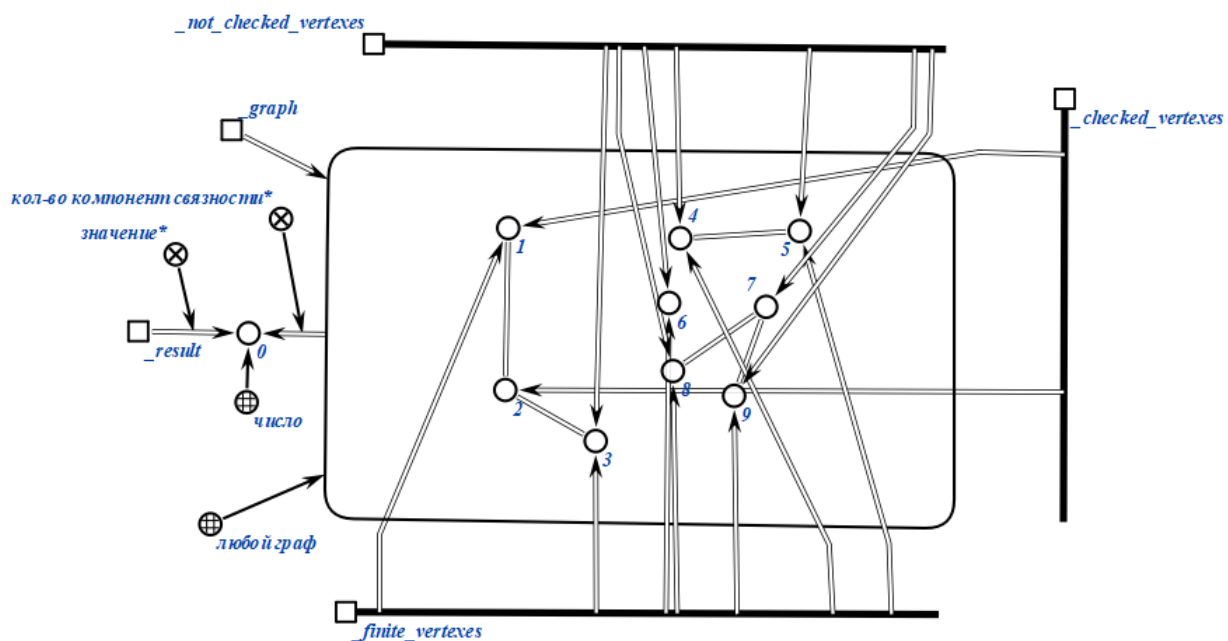


Рис. 19: Идём в вершину с индексом 2

- (h) Затем программа идет из вершины с индексом 2 в вершину с индексом 3, так как между ними имеется ребро и вершина с индексом 3 еще не внесена во множество посещенных вершин. Соответственно удаляем эту вершину из мн-ва непосещенных вершин и вносим во мн-во посещенных вершин.

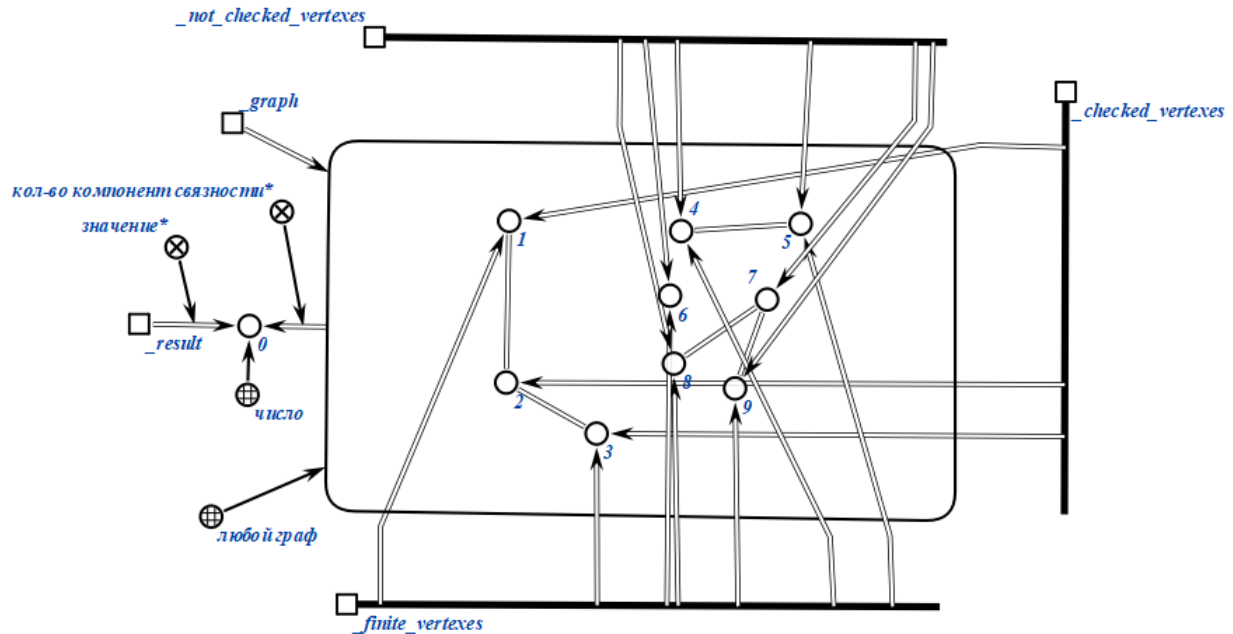


Рис. 20: Идём в вершину с индексом 3

- (i) Больше у вершины с индексом 3 нету других вершин, с которыми она была бы связана ребром и которые бы входили во множество непосещенных вершин. Такая же ситуация и с вершиной с индексом 1. Также программа видит, что вершины с индексом 1 и 3 входят во множество конечных вершин (`_finite_vertexes`). Соответственно программа понимает, что счетчик кол-ва компонент связности нужно увеличить на 1 (что она собственно и делает).

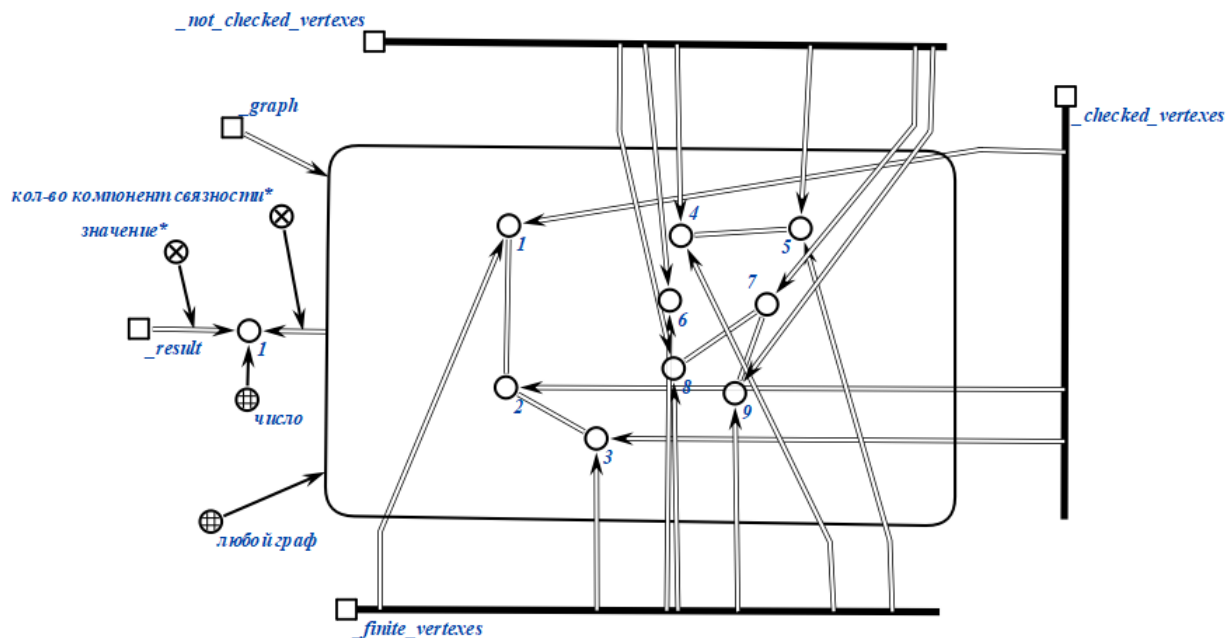


Рис. 21: Увеличиваем счётчик кол-ва компонент связности на 1 значение

- (j) Затем программа начинает выполнение этого же алгоритма, но начинает с другой непосещенной и крайней(конечной) вершины(например, с индексом 4), удаляет ее из мн-ва непосещенных вершин и вносит во мн-во посещенных вершин.

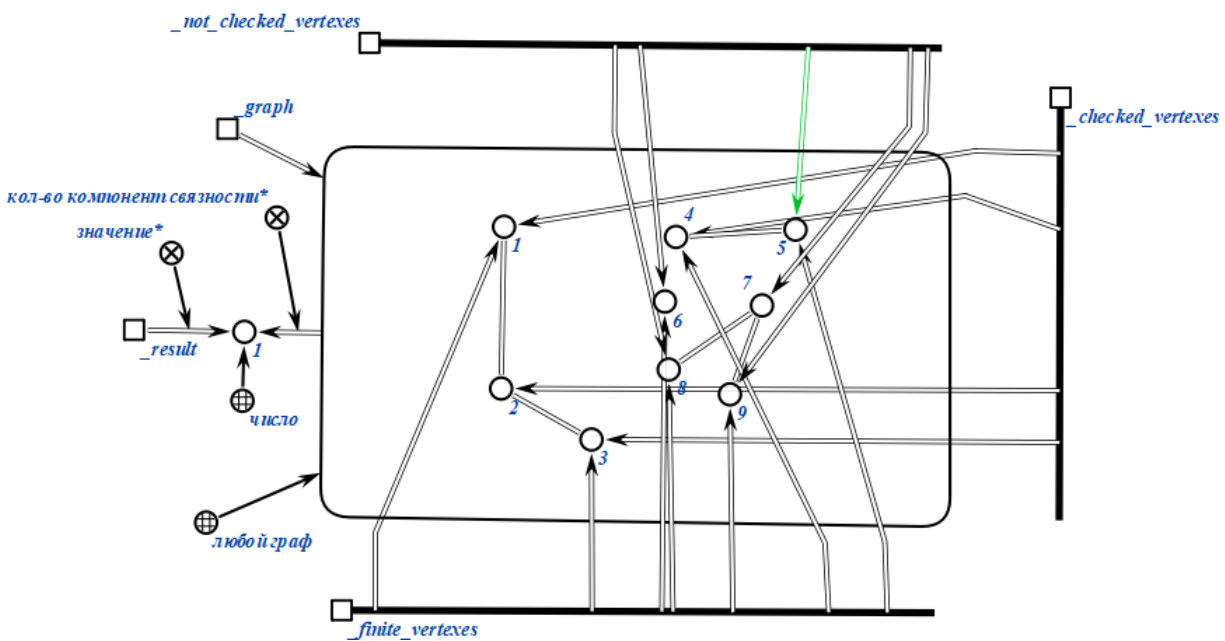


Рис. 22: Продолжение этого же алгоритма с другой компонентой связности

- (k) Программа идёт далее к вершине с индексом 5, удаляет ее из мн-ва непосещенных вершин и вносит во мн-во посещенных вершин.

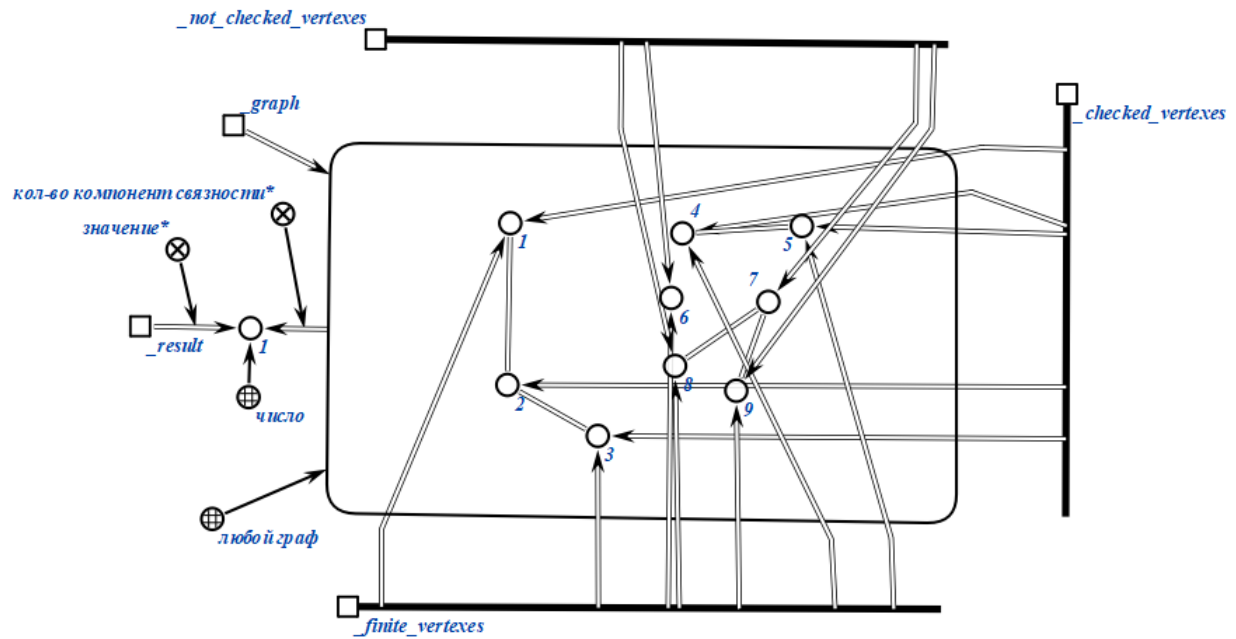


Рис. 23: Идём к вершине с индексом 5

- (1) Больше у вершины с индексом 5 нету других вершин, с которыми она была бы связана ребром и которые бы входили во множество непосещенных вершин. Такая же ситуация и с вершиной с индексом 4. Также программа видит, что вершины с индексом 4 и 5 входят во множество конечных вершин(`_finite_vertexes`). Соответственно программа понимает, что счетчик кол-ва компонент связности нужно увеличить на 1(что она собственно и делает).

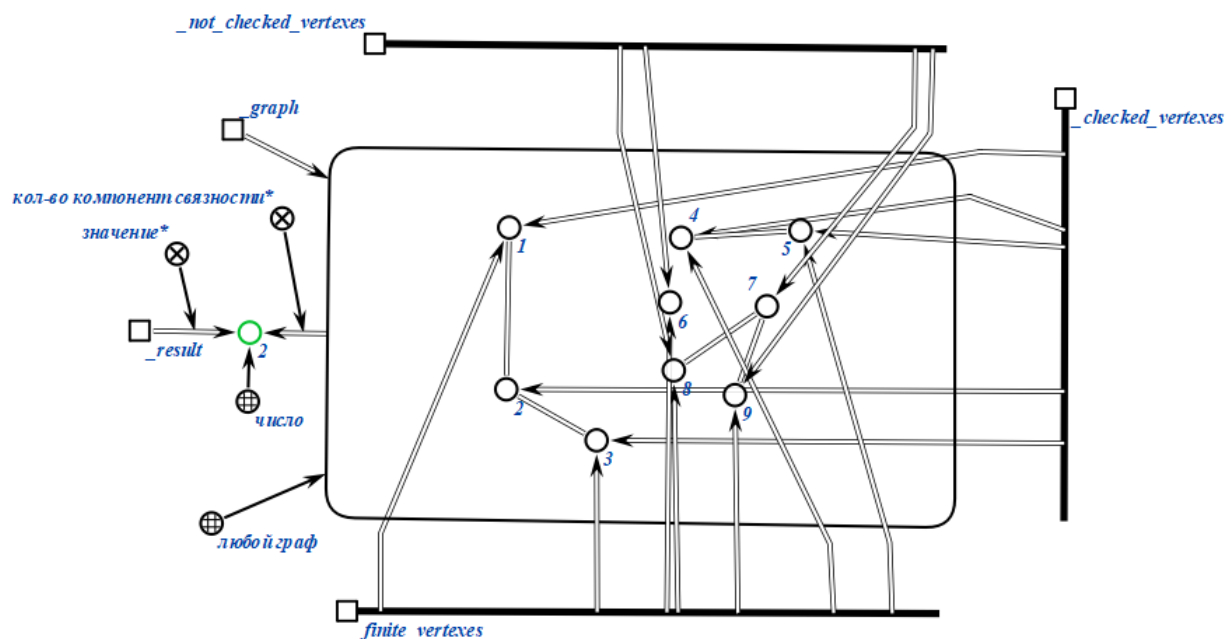


Рис. 24: Увеличение счётчика компонент связности на одно значение

- (m) Переходим к вершине с индексом 6 (т.к. ее индекс минимален среди непосещенных оставшихся вершин и она находится во множествах таких, как конечная (крайняя) вершина и непосещенные вершины). Удаляем ее из мн-ва непосещенных вершин и добавляем во мн-во посещенных вершин. У вершины с индексом 6 нету других вершин, с которыми она была бы связана ребром и которые бы входили во множество непосещенных вершин. Также программа видит, что вершина с индексом 6 входит во множество конечных вершин (`_finite_vertexes`). Соответственно программа понимает, что счетчик кол-ва компонент связности нужно увеличить на 1 (что она собственно и делает).

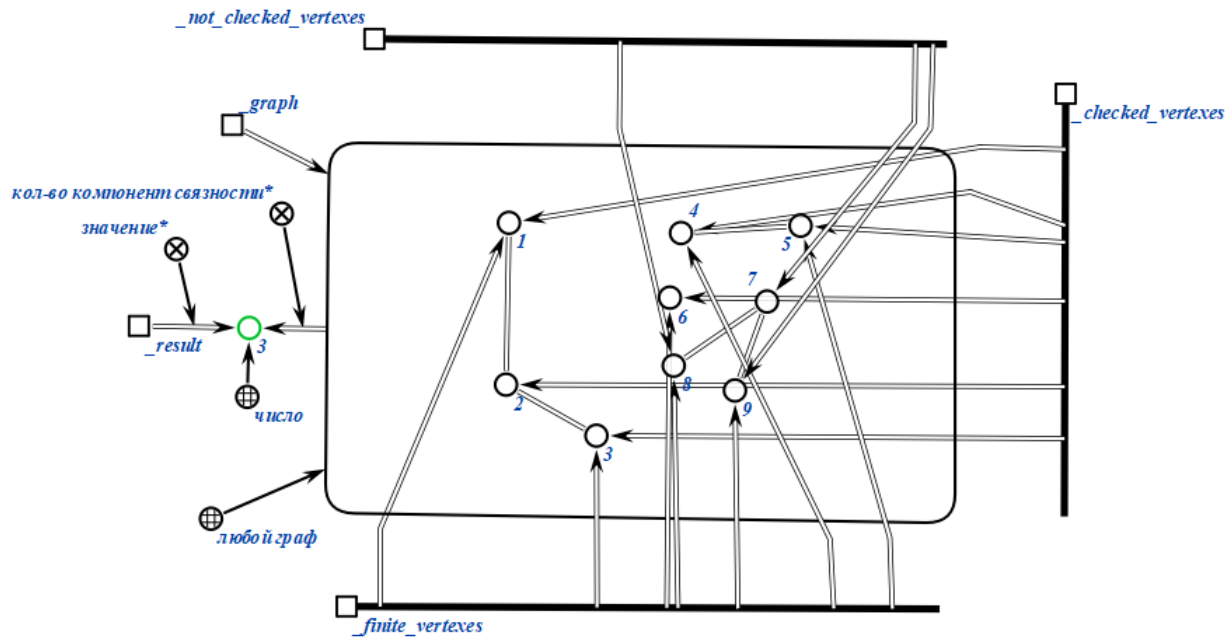


Рис. 25: Переход к вершине с индексом 6 и увеличение счетчика на одно значение

- (n) Переходим к вершине с индексом 8 (т.к. ее индекс минимален среди непосещенных оставшихся вершин входящих во множества, как конечная (крайняя) вершина и непосещенные вершины). Удаляем ее из мн-ва непосещенных вершин и добавляем во мн-во посещенных вершин.

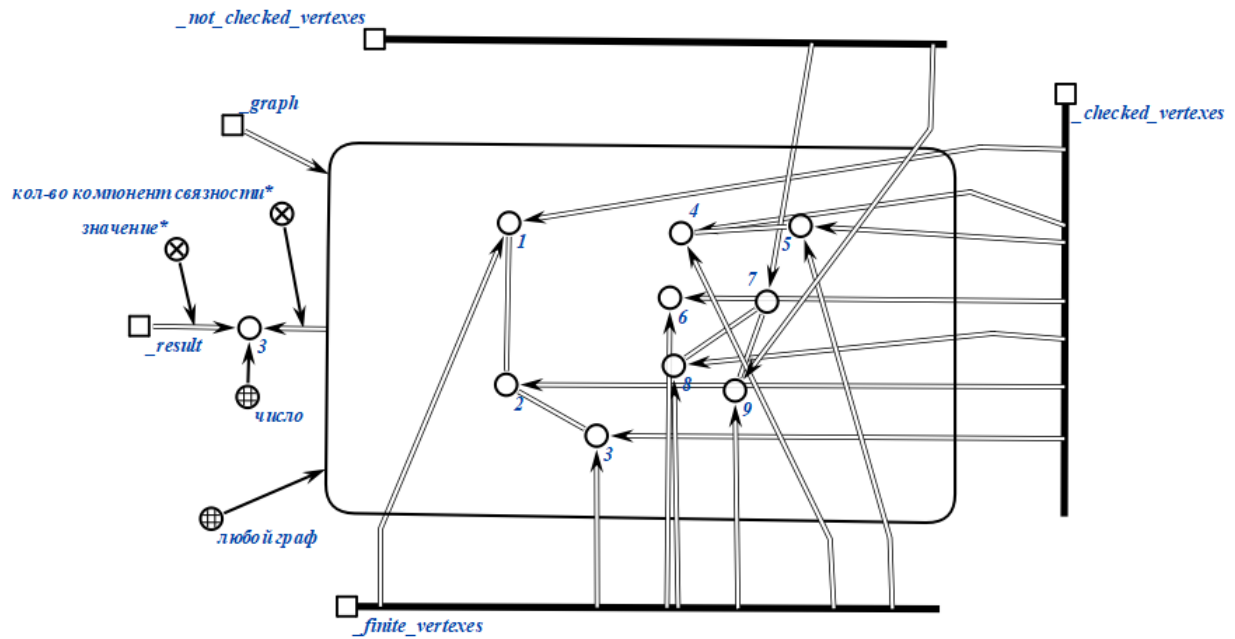


Рис. 26: Переход к вершине с индексом 8

- (о) Далее программа видит, что вершина с индексом 8 связана с вершиной с индексом 7 ребром, которая входит во мн-во непосещенных вершин, следовательно она идёт к вершине 7, удаляем ее(7) из мн-ва непосещенных вершин и добавляем во мн-во посещенных вершин.

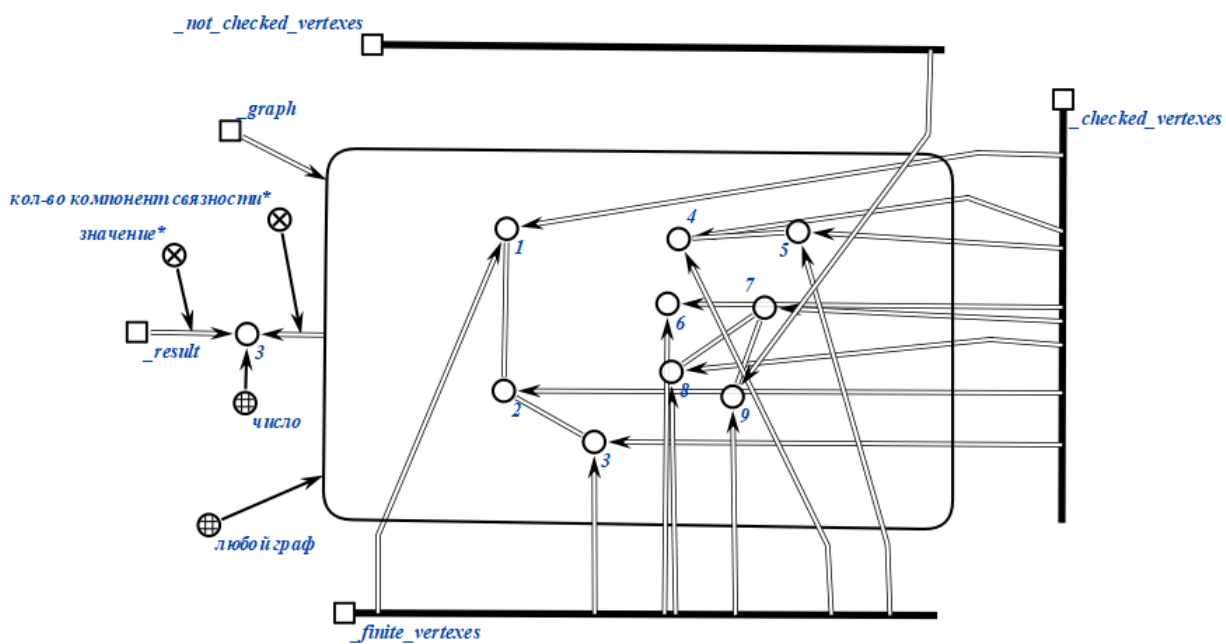


Рис. 27: Переход к вершине с индексом 7

- (р) Далее программа видит, что вершина с индексом 7 связана с вершиной с индексом 9 ребром, которая входит во мн-во непосещенных вершин, следовательно она идёт к вершине 9, удаляем ее(9) из мн-ва непосещенных вершин и добавляем во мн-во посещенных вершин.

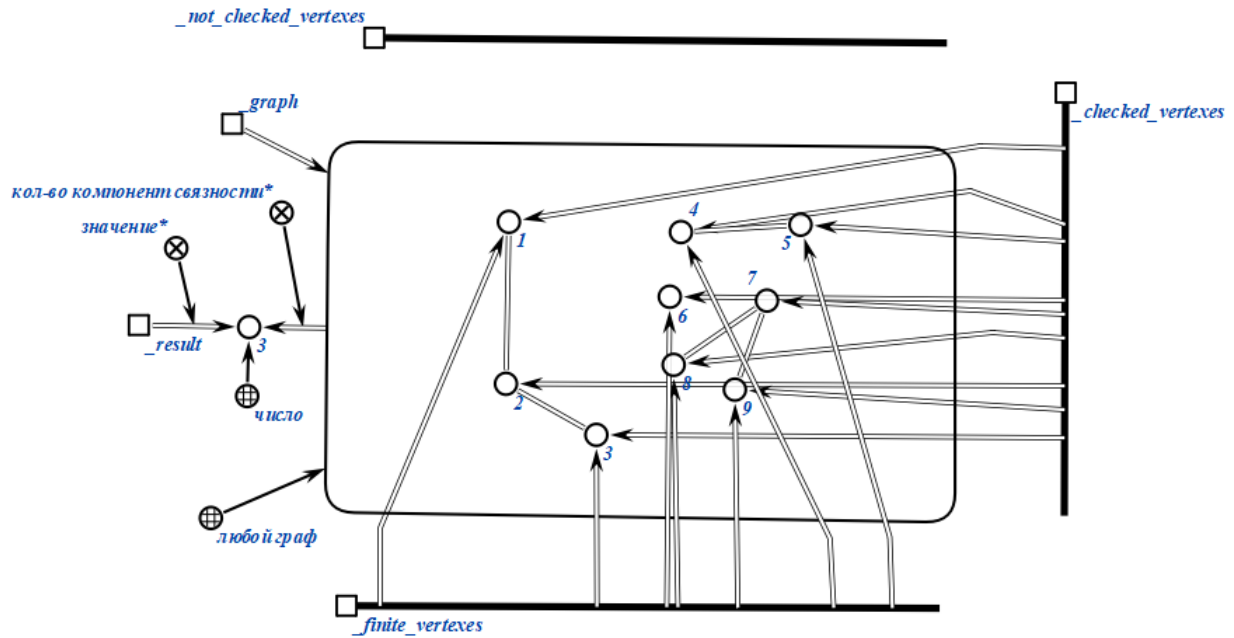


Рис. 28: Переход к вершине с индексом 9

- (q) Больше у вершины с индексом 9 нету связей с вершинами, которые бы входили во множество непосещенных вершин. Так же программа видит, что вершины с индексом 8 и 9 входят в множество крайних вершин. Следовательно программа добавляет одно значение в счетчик компонент связности.

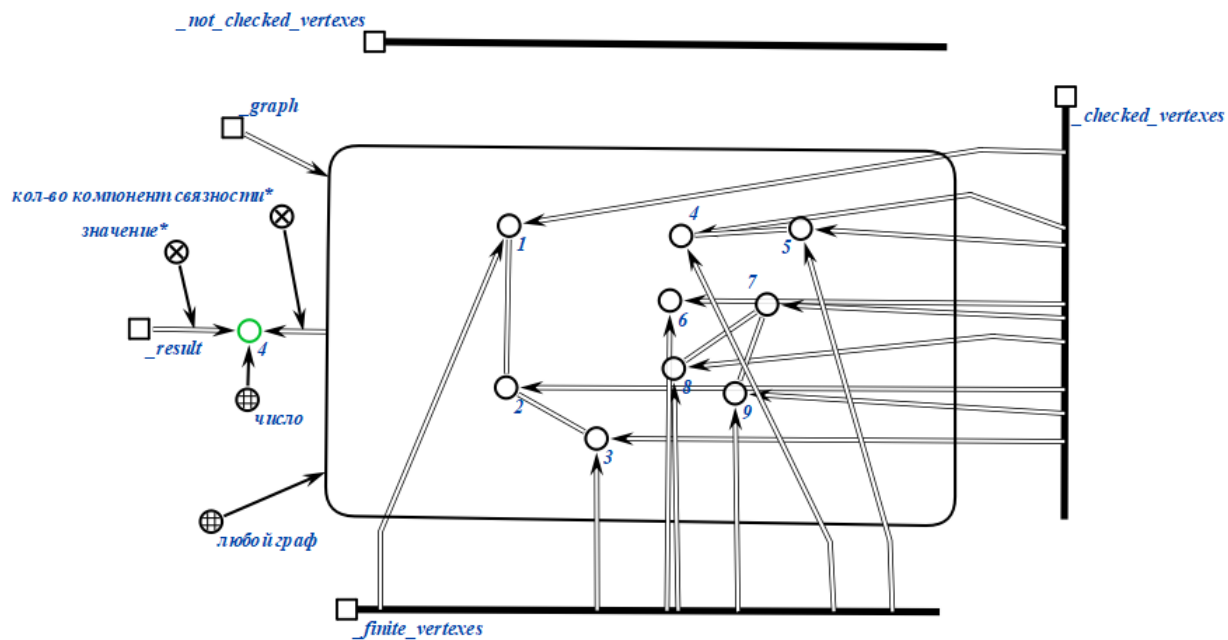


Рис. 29: Увеличение показания счетчика компонент связности на одно значение

- (г) Программа после каждого добавления в значение счетчик всегда смотрела, не пустое ли мн-во непосещенных вершин, так как оно до этого момента было не пустым, то программа выполняла свою работу. На данном этапе мн-во непосещенных вершин пусто, что гласит нам о том, что программа успешно выполнила свою работу с ответом 4(компоненты связности в неориентированном графе).

5 Заключение

Формализовал алгоритм поиска количества компонент связности в неориентированном графе.