Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления
Кафедра Интеллектуальных информационных технологий

ОТЧЁТ

по дисциплине «Представление и обработка информации в интеллектуальных системах»
на тему
Двусвязный неориентированный граф

Выполнил: Е. Д. Васильков

Студент группы
121703
Проверил: А. Г. Загорский

Минск 2022

Содержание

Введение

Двусвязный неориентированый граф — это связный граф, в котором отсутвуют точки сочленения и удаление любой вершины не приводит к потере связности. Граф называется связным, если между каждой парой вершин существует ребро. Вершина в неориентированном связном графе является точкой сочленения, если ее удаление разъединяет граф.

По соглашению две вершины, соединенные ребром, образуют двусвязный граф. Для графа с более чем двумя вершинами указанные выше свойства должны присутствовать, чтобы он был двусвязным.

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

Задача: Определить, является ли неориентированный граф двусвязным.

1 Список понятий

1. Для наглядности продемонстрирована иерархия различных типов графовых структур рассмотренных далее (Это неполная иерархия из OSTIS GT)

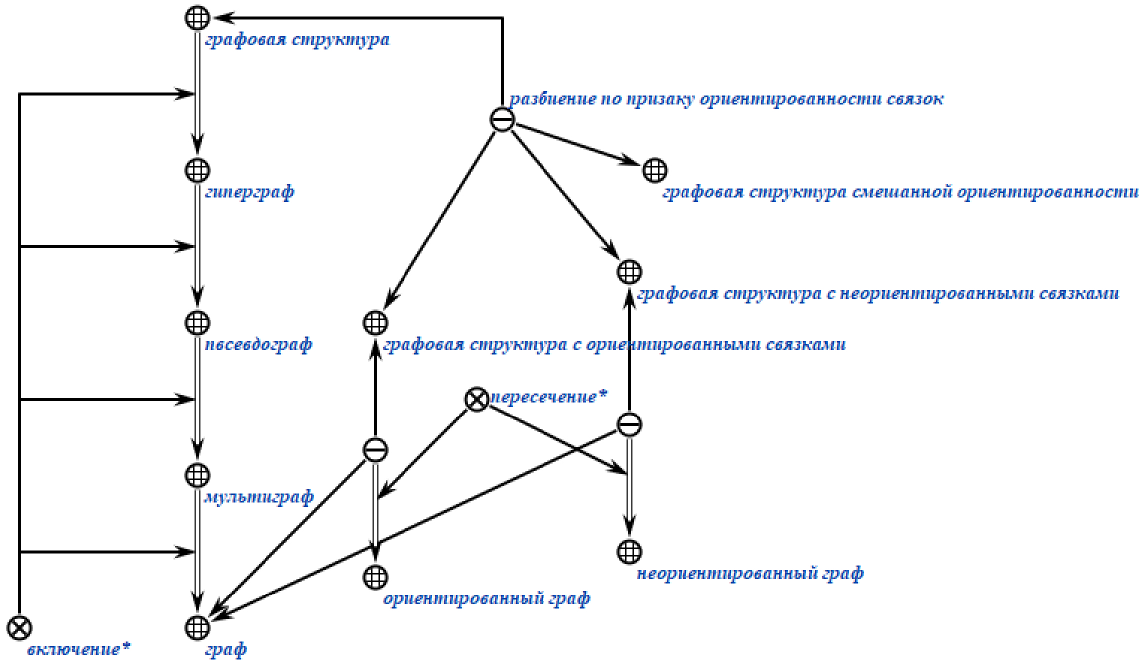

Рисунок *1.1* – Иерархия графовых структур

2. Графовая структура (абсолютное понятие) - это такая одноуровневая реляционная структура, объекты которой могут играть роль либо вершины, либо связки:
   2.1. Вершина (относительное понятие, ролевое отношение).
   2.2. Связка (относительное понятие, ролевое отношение).

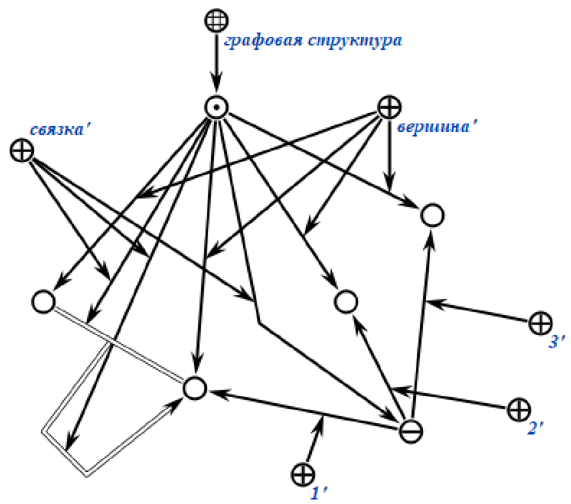

Рисунок *1.2* – Графовая структура

3. Гиперграф (абсолютное понятие) – это такая графовая структура, в которой связки могут связывать только вершины:
   3.1. Гиперсвязка (относительное понятие, ролевое отношение);
   3.2. Гипердуга (относительное понятие, ролевое отношение) - ориентирванная гиперсвязка.
   3.3. Гиперребро (относительноепонятие, ролевоеотношение) - неориентированная гиперсвязка.

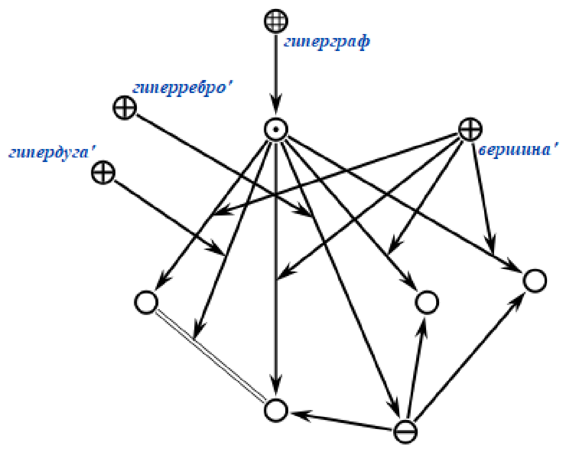

Рисунок *1.3* – Гиперграф

4. Псевдограф (абсолютное понятие) - это такой гиперграф, в котором все связки должны быть бинарными:
   4.1. Бинарная связка (относительное понятие, ролевое отношение) - гиперсвязка арности 2;
   4.2. Ребро (относительное понятие, ролевое отношение) - неориентированная гиперсвязка;

4.3. Дуга (относительное понятие, ролевое отношение) - ориентированная гиперсвязка;

4.4. Петля (относительное понятие, ролевое отношение) - бинарная связка, у которой первый и второй компоненты совпадают.

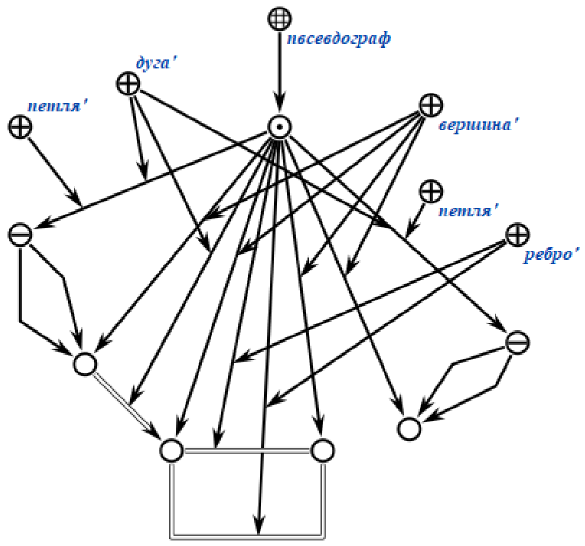

Рисунок *1.4* – Псевдограф

5. Мультиграф (абсолютное понятие) – это такой псевдограф, в котором не может быть петель:

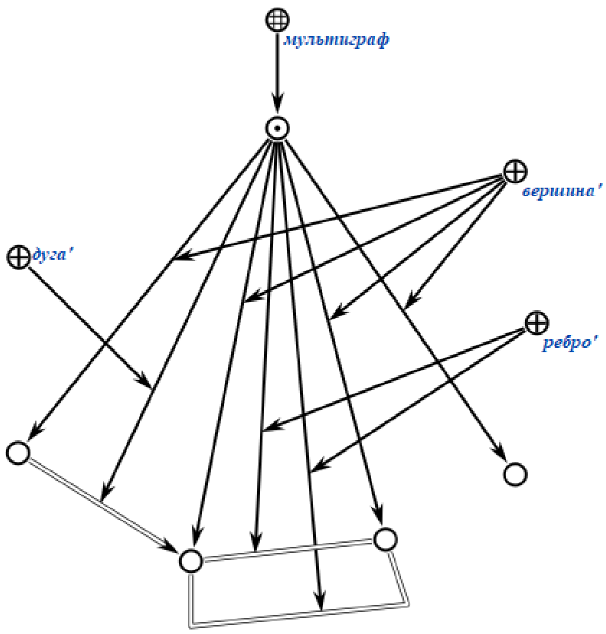

Рисунок *1.5* – Мультиграф

6. Граф (абсолютное понятие) – это такой мультиграф, в котором не может быть кратных связок, т.е. связок у которых первый и второй компоненты совпадают:

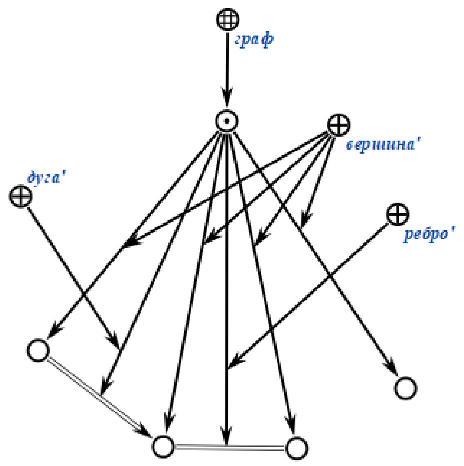

Рисунок *1.6* – Граф

7. Неориентированный граф (абсолютное понятие) – это такой граф, в котором все связки являются ребрами:

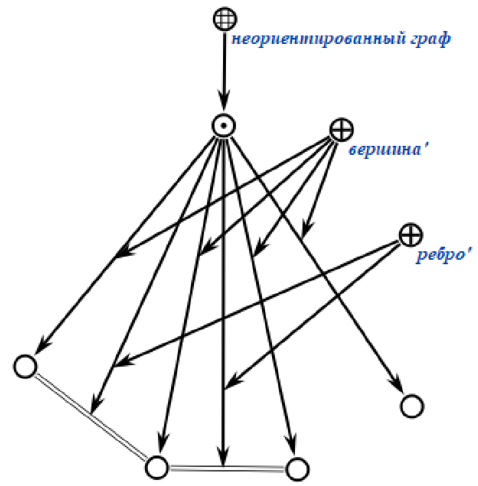

Рисунок *1.7* – Неориентированный граф

8. Ориентированный граф (абсолютное понятие) - это такой граф, в котором все связки являются дугами:

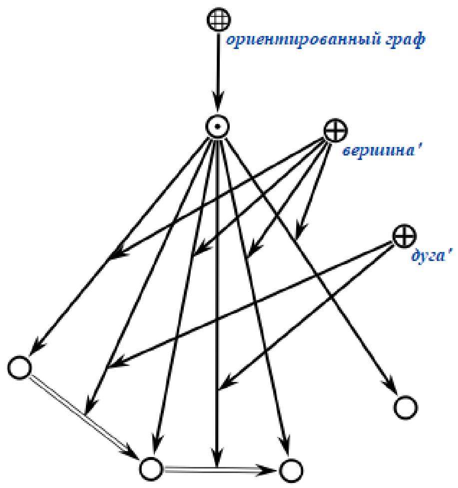

Рисунок *1.8* – Ориентированный граф

9. Связный граф (абсолютное понятие) – граф, содержащий только одну компоненту связности.

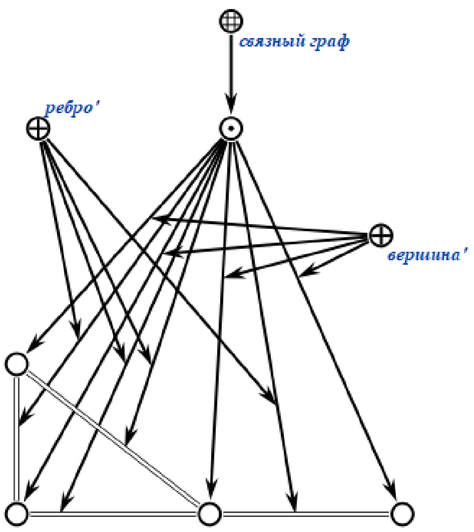

Рисунок *1.9* – Связный граф

10. Компонента связности графа (абсолютное понятие) — такое подмножество вершин графа, для любых двух вершин которого существует путь из одной в другую, и не существует пути из вершины этого подмножества в вершину не из этого подмножества.

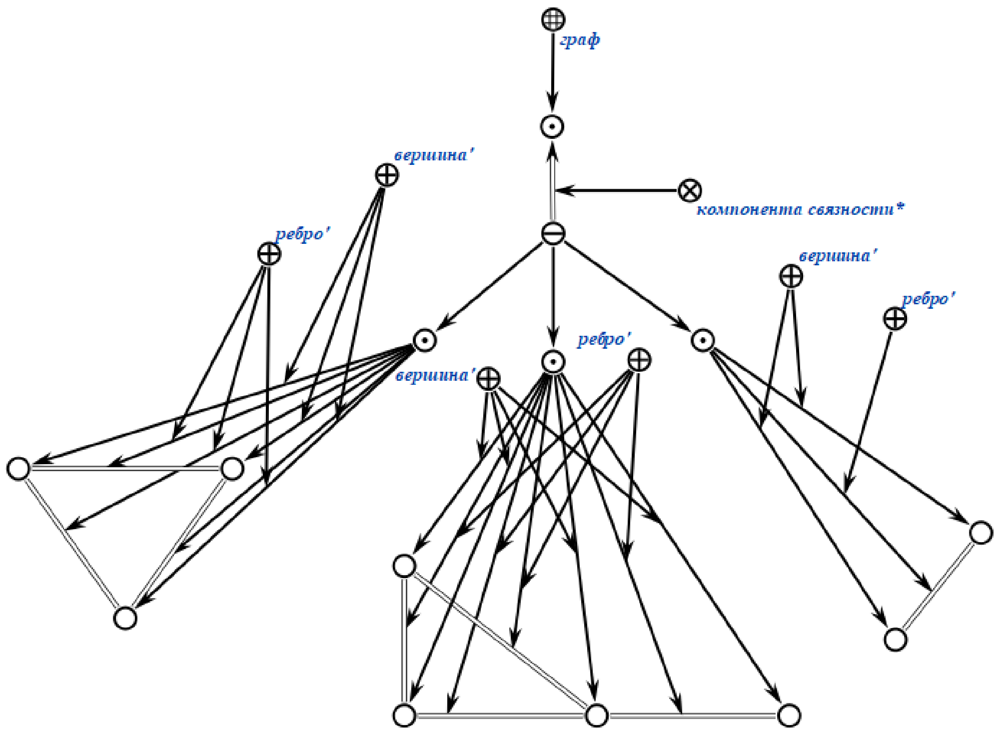

Рисунок *1.10* – Компонента связности

11. Точка сочленения (абсолютное понятие) - вершина графа, при удалении которой количество компонент связности возрастает.

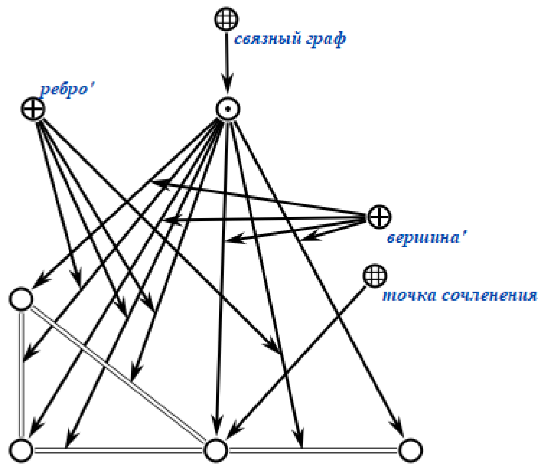

Рисунок *1.11* – Точка сочленения

12. Двусвязный граф (абсолютное понятие) — это связный граф, в котором отсутвуют точки сочленения и удаление любой вершины не приводит к потере связности.

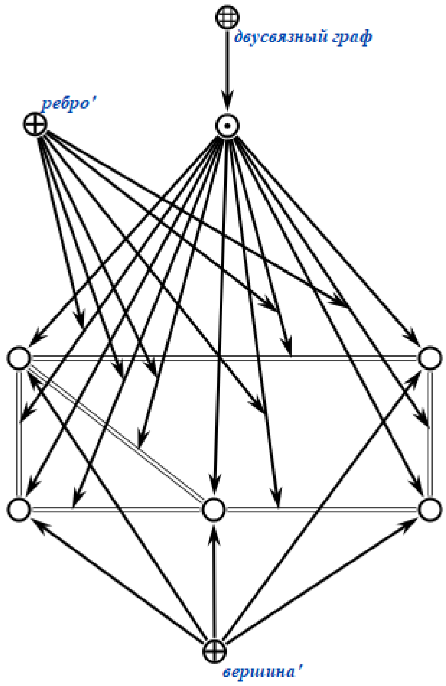

Рисунок *1.12* – Двусвязный граф

2 Алгоритм

График (vertexes) будет хранить информацию о вершинах в следствии обхода графа. Первая компонента кортежа графика хранит порядок посещённых вершин. Вторая компонента хранит минимальное количество шагов, за которое вершина может быть посещена из её поддерева.

1. Создаём пустой график vertexes.
2. Создаём пустой график articulation points, в котором будут хранитья точки сочленения графа.
3. Выбираем первую вершину.
4. Создаём кортеж в графие vertexes.
5. На место первой компоненты записываем порядок посещённой вершины.
   - 5.1. Если у вершины есть ребро, инцидентное с её предком (обратного ребра), то на место второй компоненты записывается первая компонента кортежа предка.
   - 5.2. Иначе, на место второй компоненты записывается первая копонента кортежа.
6. Выбираем следующую вершину, смежную с предыдущей.
   - 6.1. Если смежная вершина отсутствует, то переходим к пункту 7.
   - 6.2. Если смежная вершина присутствует, то переходим к пункту 4.
7. Снова выбираем первую вершину.
   - 7.1. Если существует потомок у которого значение второй компоненты меньше, чем выбранной вершины, то на место второй компоненты выбранной вершины записывается наименьшее значение её потомка.
   - 7.2. Если первая компонента кортежа выбранной вершины меньше или равна второй компоненте потомка этой вершины, то заносим кортеж в график articulation points.
     - 7.2.1. Если выбранная вершина является последней, то переходим к пункту 8.
     - 7.2.2. Иначе берём следующую смежную вершину и переходим к пункту 7.2.
8. Если график articulation points является пустым, то:
   - 8.1. исходный граф является двусвязным.
   - 8.2. Алгоритм завершён.
9. Если график articulation points не является пустым, то:
   - 9.1. в графике присутствуют точки сочленения и исходный граф не является двусвязным.
   - 9.2. Алгоритм завершён.

3 Тестовые примеры

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

Тест 1

Вход: Определить, является ли граф двусвязным.

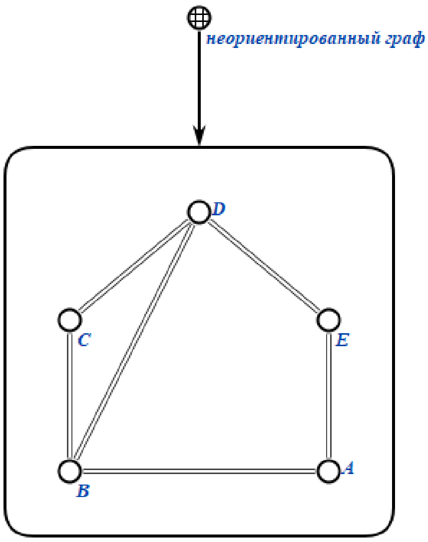

Рисунок *3.1* – Вход теста 1

Шаги:

1. Создаём пустой графику vertexes.
2. Создаём пустой графику articulation points, в котором будут хранитья точки сочленения графа.
3. Выбираем первую вершину (A).
4. Создаём кортеж в графике vertexes.
5. На место первой компоненты записываем порядок посещённой вершины <1, ...>.
6. Так как у вершины нет ребра, инцидентного с её предком (обратного ребра), на место второй компоненты записывается первая копонента кортежа <1, 1>.

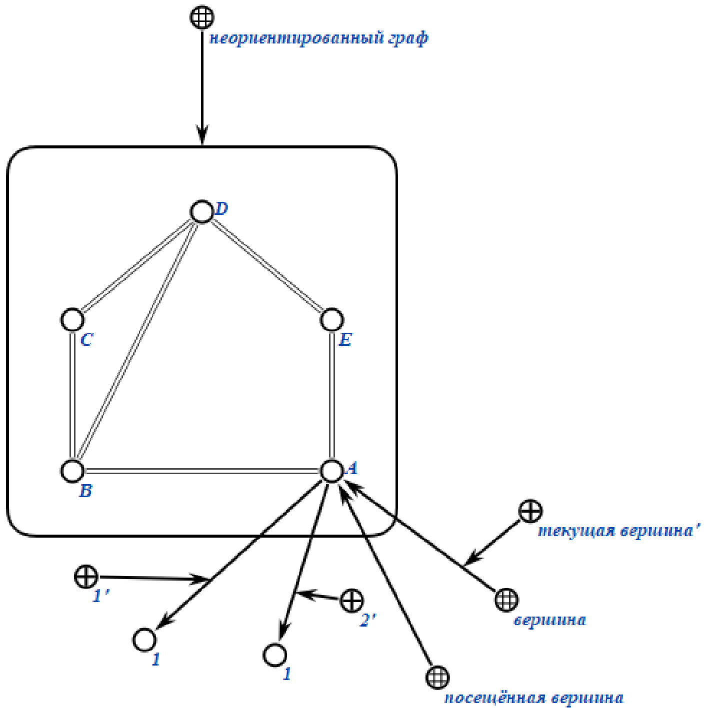

Рисунок *3.2* – Вершина А

7. Выбираем следующую вершину (В), смежную с предыдущей.
8. Создаём кортеж в графике vertexes.
9. На место первой компоненты записываем порядок посещённой вершины <2, ...>.
10. Так как у вершины нет ребра, инцидентного с её предком (обратного ребра), на место второй компоненты записывается первая копонента кортежа <2, 2>.

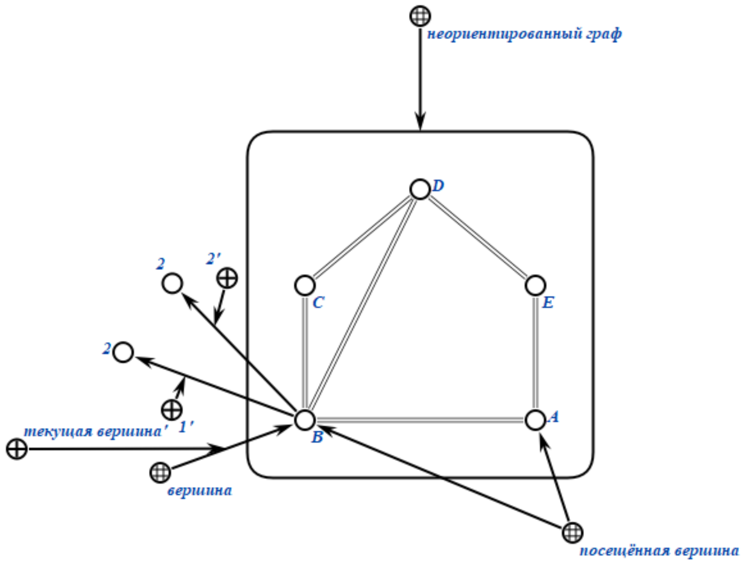

Рисунок *3.3* – Вершина B

11. Выбираем следующую вершину (C), смежную с предыдущей.
12. Создаём кортеж в графике vertexes.
13. На место первой компоненты записываем порядок посещённой вершины <3, ...>.
14. Так как у вершины нет ребра, инцидентного с её предком (обратного ребра), на место второй компоненты записывается первая копонента кортежа <3, 3>.

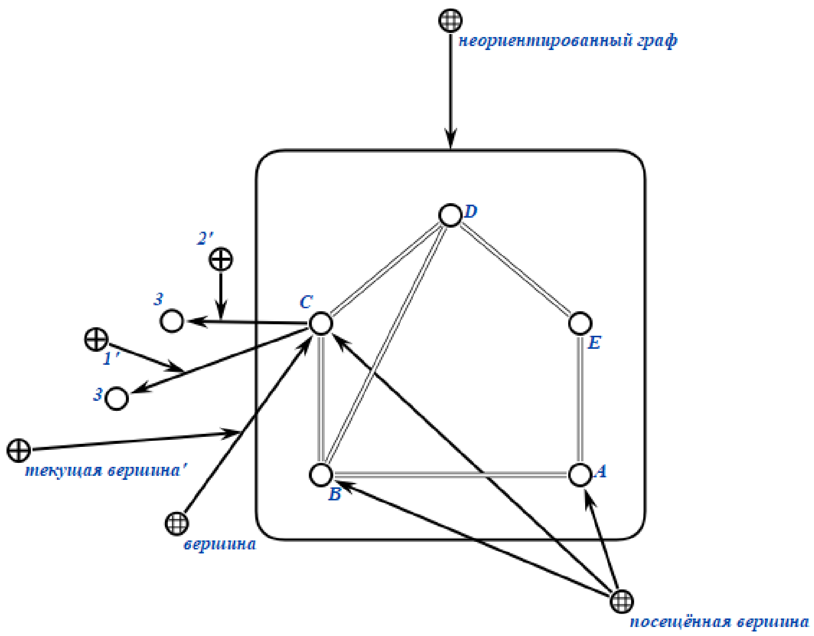

Рисунок *3.4* – Вершина C

15. Выбираем следующую вершину (D), смежную с предыдущей.
16. Создаём кортеж в графике vertexes.

17. На место первой компоненты записываем порядок посещённой вершины <4, ...>.
18. Так как у вершины есть ребро, инцидентное с её предком (обратное ребро), то на место второй компоненты записывается первая компонента кортежа предка (вершины B) <4, 2>.

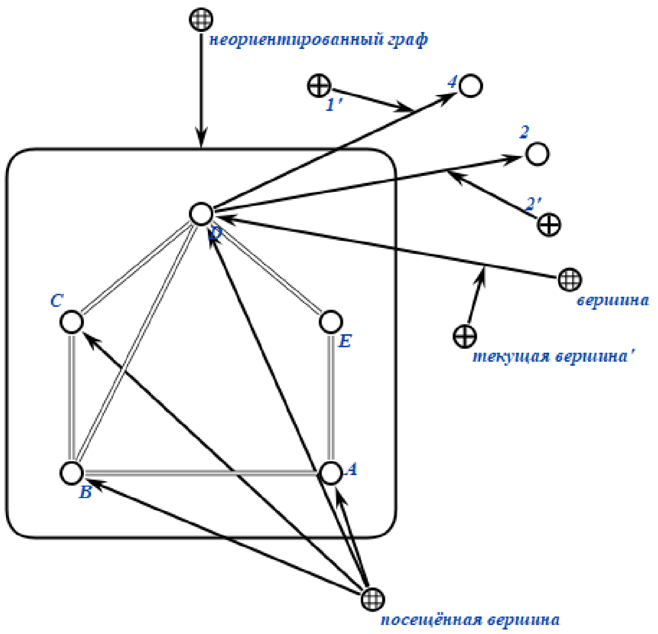

Рисунок *3.5* – Вершина D

19. Выбираем следующую вершину (E), смежную с предыдущей.
20. Создаём кортеж в графике vertexes.
21. На место первой компоненты записываем порядок посещённой вершины <5, ...>.
22. Так как у вершины есть ребро, инцидентное с её предком (обратное ребро), то на место второй компоненты записывается первая компонента кортежа предка (вершины A) <5, 1>.

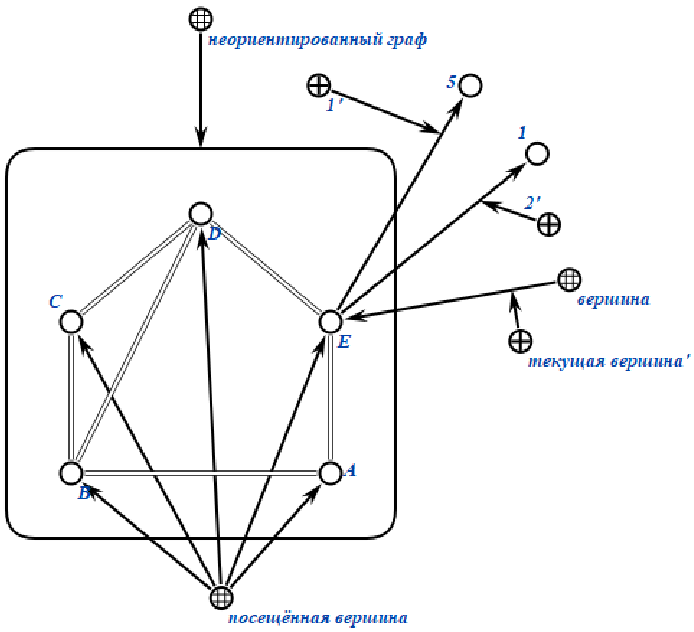

Рисунок *3.6* – Вершина E

23. Снова выбираем первую вершину.
24. Так как вершина (A) является корнем, то не существует потомка, у которого значение второй компоненты меньше, чем у корня. Поэтому значение второй компоненты остаётся неизменной <1, 1>.
25. Выбираем следующую вершину (B).
26. Так как существует потомок, у которого значение второй компоненты меньше, чем выбранной вершины (B), то на место второй компоненты выбранной вершины записывается наименьшее значение её потомка <2, 1>.
27. В связи что перва компонента выбранной вершины больше второй компоненты, то данная вершина не может быть точкой сочленения.
28. Выбираем следующую вершину (C).
29. Так как существует потомок, у которого значение второй компоненты меньше, чем выбранной вершины (C), то на место второй компоненты выбранной вершины записывается наименьшее значение её потомка <3, 1>.
30. В связи что перва компонента выбранной вершины больше второй компоненты, то данная вершина не может быть точкой сочленения.
31. Выбираем следующую вершину (D).
32. Так как существует потомок, у которого значение второй компоненты меньше, чем выбранной вершины (D), то на место второй компоненты выбранной вершины записывается наименьшее значение её потомка <4, 1>.
33. В связи что перва компонента выбранной вершины больше второй компоненты, то данная вершина не может быть точкой сочленения.

34. Выбираем следующую вершину (E).
35. Так как вершина (E) является листом,значение второй компоненты остаётся неизменной <5, 1>.
36. В связи с тем, что в график articulation points не было добавлено ни одного кортежа, то исходный граф является двусвязным.

Выход: Граф является двусвязным

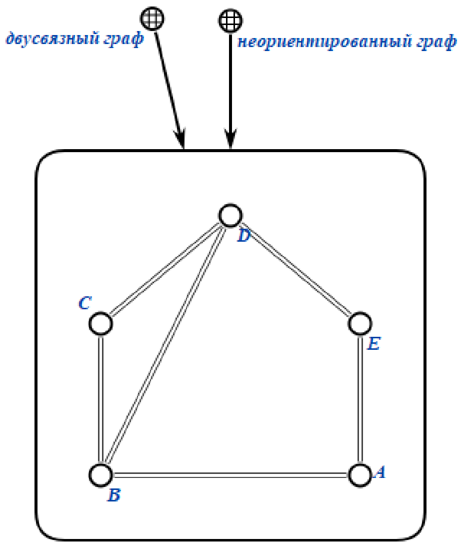

Рисунок *3.7* – Выход теста 1

Тест 2
Вход: Определить, является ли граф двусвязным.

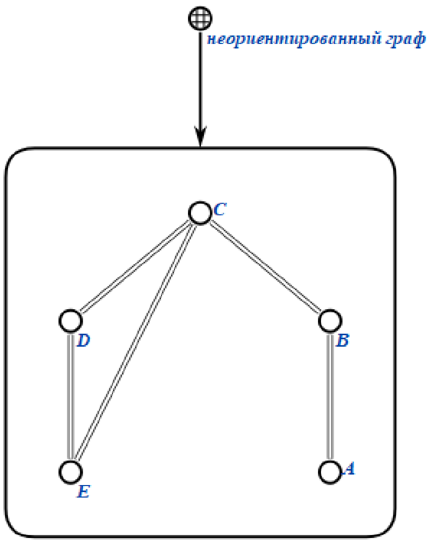

Рисунок *3.8* – Вход теста 2

Шаги:
1. Создаём пустой график vertexes.

2. Создаём пустой график articulation points, в котором будут хранитья точки сочленения графа.
3. Выбираем первую вершину (A).
4. Создаём кортеж в графике vertexes.
5. На место первой компоненты записываем порядок посещённой вершины <1, ...>.
6. Так как у вершины нет ребра, инцидентного с её предком (обратного ребра), на место второй компоненты записывается первая копонента кортежа <1, 1>.

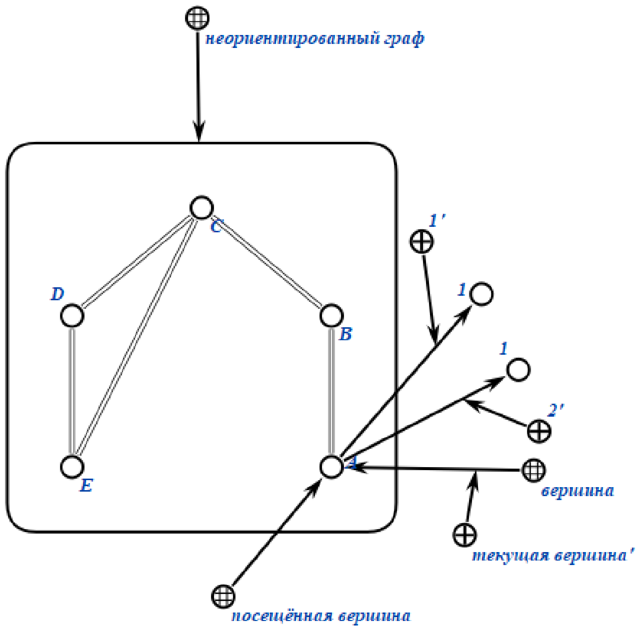

Рисунок *3.9* – Вершина A

7. Выбираем следующую вершину (B), смежную с предыдущей.
8. Создаём кортеж в графике vertexes.
9. На место первой компоненты записываем порядок посещённой вершины <2, ...>.
10. Так как у вершины нет ребра, инцидентного с её предком (обратного ребра), на место второй компоненты записывается первая копонента кортежа <2, 2>.

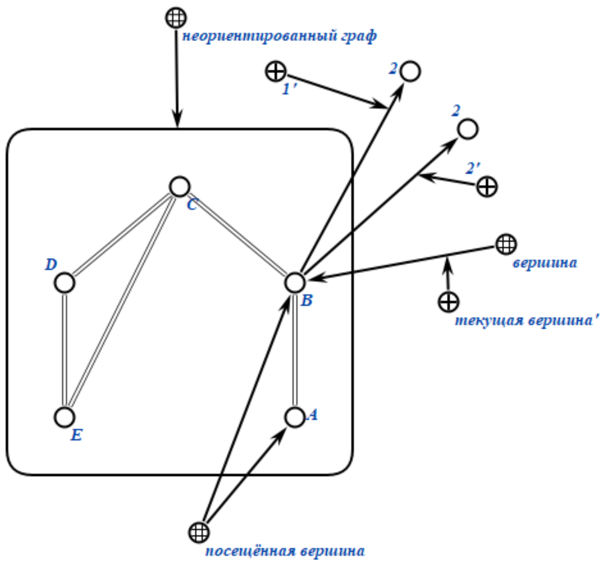

Рисунок *3.10* – Вершина B

11. Выбираем следующую вершину (C), смежную с предыдущей.
12. Создаём кортеж в графике vertexes.
13. На место первой компоненты записываем порядок посещённой вершины <3, ...>.
14. Так как у вершины нет ребра, инцидентного с её предком (обратного ребра), на место второй компоненты записывается первая копонента кортежа <3, 3>.

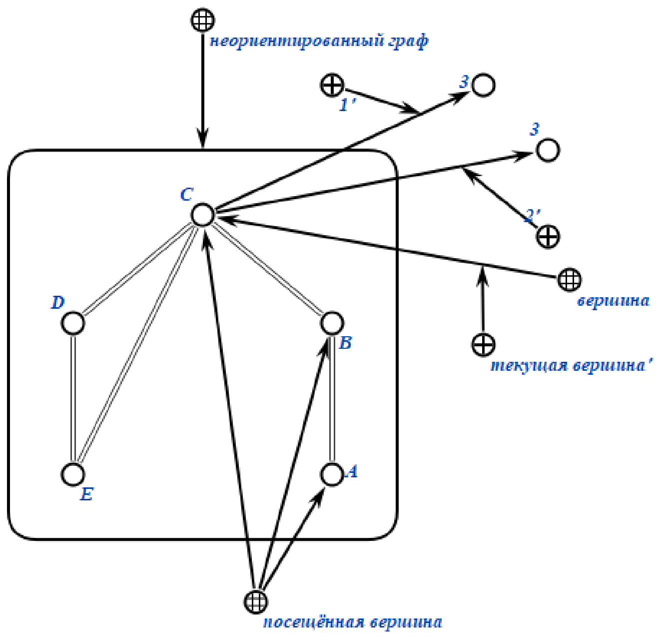

Рисунок *3.11* – Вершина C

15. Выбираем следующую вершину (D), смежную с предыдущей.

16. Создаём кортеж в графике vertexes.
17. На место первой компоненты записываем порядок посещённой вершины <4, ...>.
18. Так как у вершины нет ребра, инцидентного с её предком (обратного ребра), на место второй компоненты записывается первая копонента кортежа <4, 4>.

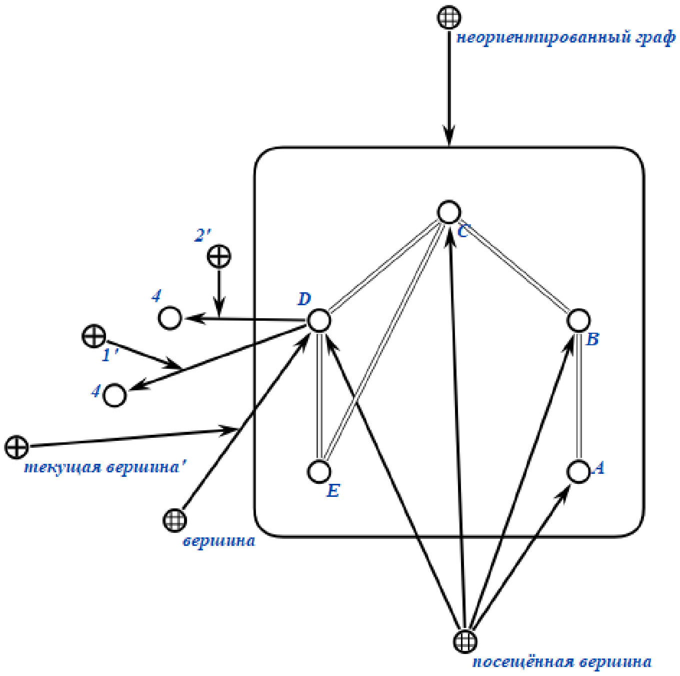

Рисунок *3.12* – Вершина D

19. Выбираем следующую вершину (E), смежную с предыдущей.
20. Создаём кортеж в графике vertexes.
21. На место первой компоненты записываем порядок посещённой вершины <5, ...>.
22. Так как у вершины есть ребро, инцидентное с её предком (обратное ребро), то на место второй компоненты записывается первая компонента кортежа предка (вершины C) <5, 3>.

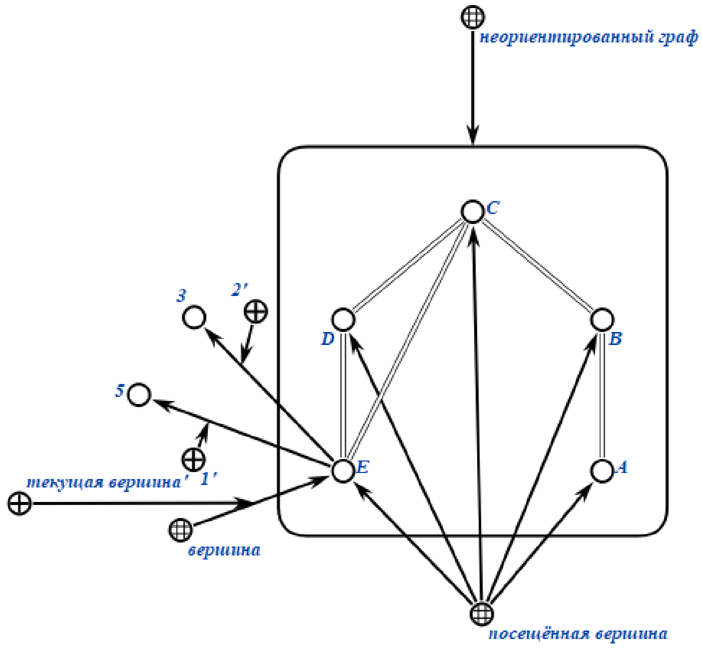

Рисунок *3.13* – Вершина E

23. Снова выбираем первую вершину.
24. Так как вершина (A) является корнем, то не существует потомка, у которого значение второй компоненты меньше, чем у корня. Поэтому значение второй компоненты остаётся неизменной <1, 1>.
25. Выбираем следующую вершину (B).
26. Так как не существует потомока, у которого значение второй компоненты меньше, чем выбранной вершины (B), то на место второй компоненты выбранной вершины записывается значение первой компоненты <2, 2>.
27. В связи что первая компонента выбранной вершины равна второй компоненте, то данная вершина является точкой сочленения и добавляем кортеж <2,2> в график articulation points.
28. Выбираем следующую вершину (C).
29. Так как не существует потомока, у которого значение второй компоненты меньше, чем выбранной вершины (C), то на место второй компоненты выбранной вершины записывается значение первой компоненты <3, 3>.
30. В связи что первая компонента выбранной вершины равна второй компоненте, то данная вершина является точкой сочленения и добавляем кортеж <3,3> в график articulation points.
31. Выбираем следующую вершину (D).
32. Так как существует потомок, у которого значение второй компоненты меньше, чем выбранной вершины (D), то на место второй компоненты выбранной вершины записывается наименьшее значение её потомка <4, 3>.
33. В связи что первая компонента выбранной вершины больше второй ком-

поненты, то данная вершина не может быть точкой сочленения.

34. Выбираем следующую вершину (E).
35. Так как вершина (E) является листом,значение второй компоненты остаётся неизменной <5, 3>.
36. В связи с тем, что в график articulation points было добавлено два кортежа <2, 2> и <3, 3>, то в исходном графе есть точки сочленения B и C и он не является двусвязным.

Выход: Граф не является двусвязным.

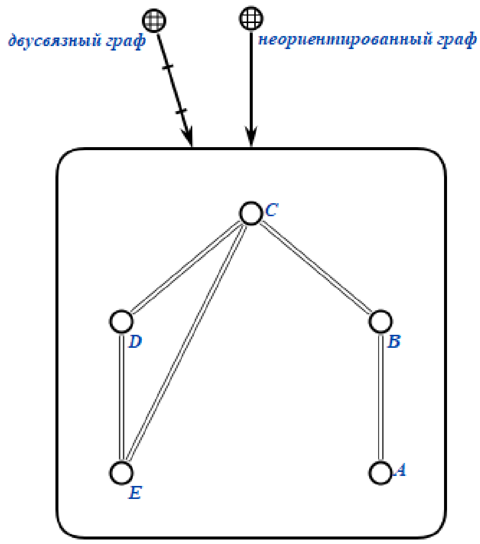

Рисунок *3.14* – Выход теста 2

Тест 3

Вход: Определить, является ли граф двусвязным.

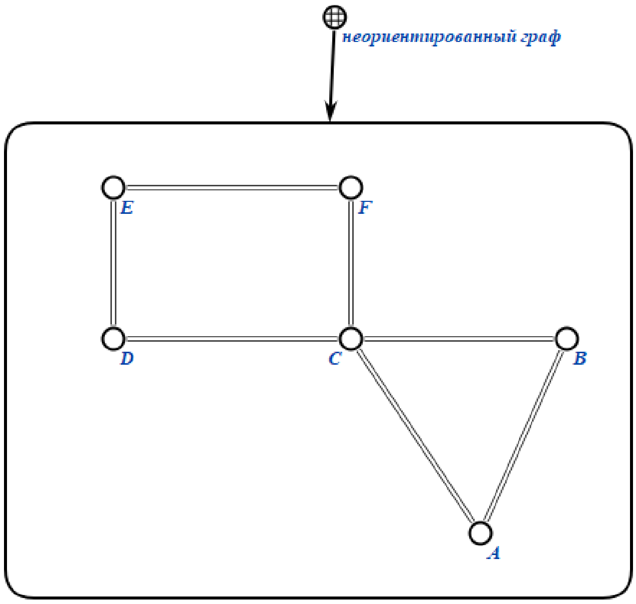

Рисунок *3.15* – Вход теста 3

Выход: Граф не является двусвязным.

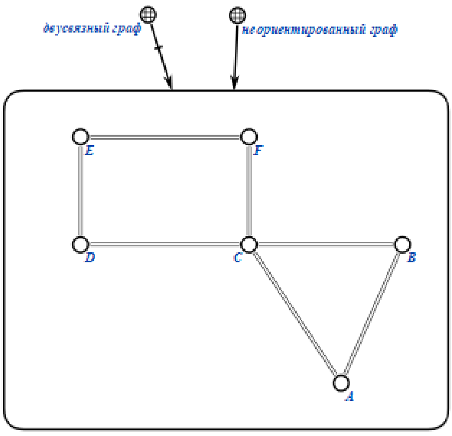

Рисунок *3.16* – Выход теста 3

Тест 4

Вход: Определить, является ли граф двусвязным.

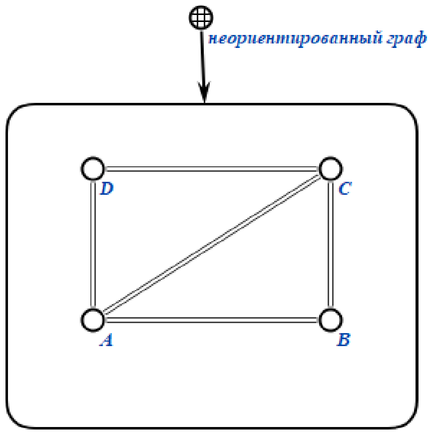

Рисунок *3.17* – Вход теста 4

Выход: Граф является двусвязным.

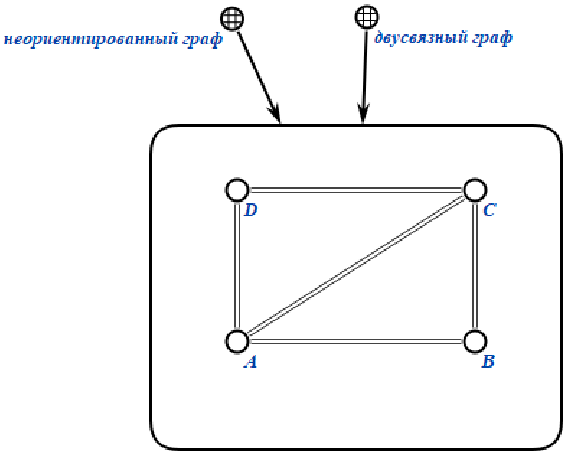

Рисунок *3.18* – Выход теста 4

Тест 5

Вход: Определить, является ли граф двусвязным.

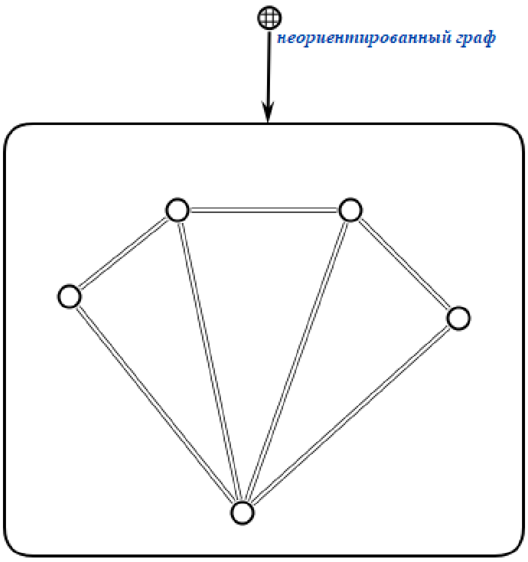

Рисунок *3.19* – Вход теста 5

Выход: Граф является двусвязным.

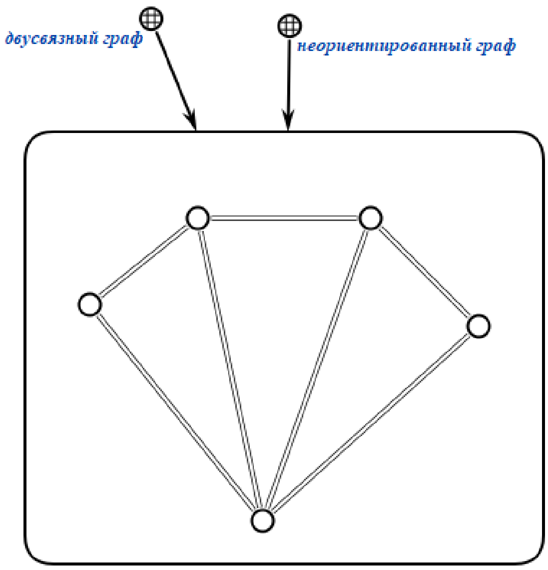

Рисунок *3.20* – Выход теста 5

Заключение

В заключении отчета сделаем краткие выводы по результатам проделанной работы:

– Получили навыки формализации и обработки информации с использованием семантических сетей.В частности формализавали различные типы графовых структур.

– Рассмотрели актуальную задачу, является ли неориентированный граф двусвязным.

Список использованных источников

[1] База знаний по теории графов OSTIS GT [Электронный ресурс] / проект OSTIS, 2022. — Режим доступа: http://ostisgraphstheo.sourceforge.net. (Дата обращения: 18.05.2022).

[2] Лазуркин, Д.А. Руководство к выполнению расчетной работы по курсам ОИИ и ППвИС / Д.А. Лазуркин. — 2013. — P. 126.

[3] Оре, О. Теория графов / О. Оре. — Наука, 1980. — P. 336.