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 «Представление и обработка информации в интеллектуальных системах»

на тему

Задача построения дерева кратчайших путей взвешенного неориентированного графа, заданного списком смежности (инцидентности)

Выполнил: М. И. Семенидо

Студент группы
321702

Проверил: Н. В. Малиновская

Минск 2024

1 ВВЕДЕНИЕ

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

Задача: Построить дерево кратчайших путей взвешенного неориентированного графа, заданного списком смежности (инцидентности)

2 СПИСОК ПОНЯТИЙ

1. Графовая структура (абсолютное понятие) - это такая одноуровневая реляционная структура, объекты которой могут играть роль либо вершины, либо связки:

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

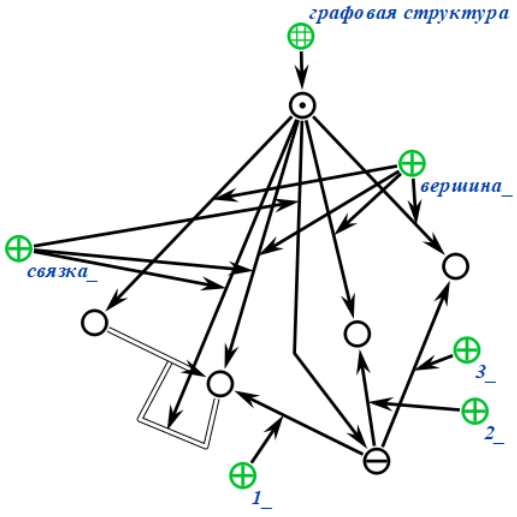

Рисунок *2.1* – Графовая структура

2. Графовая структура с неориентированными связками (абсолютное понятие).

   a. Неориентированная связка (относительное понятие, ролевое отношение) – связка, которая задается неориентированным множеством.

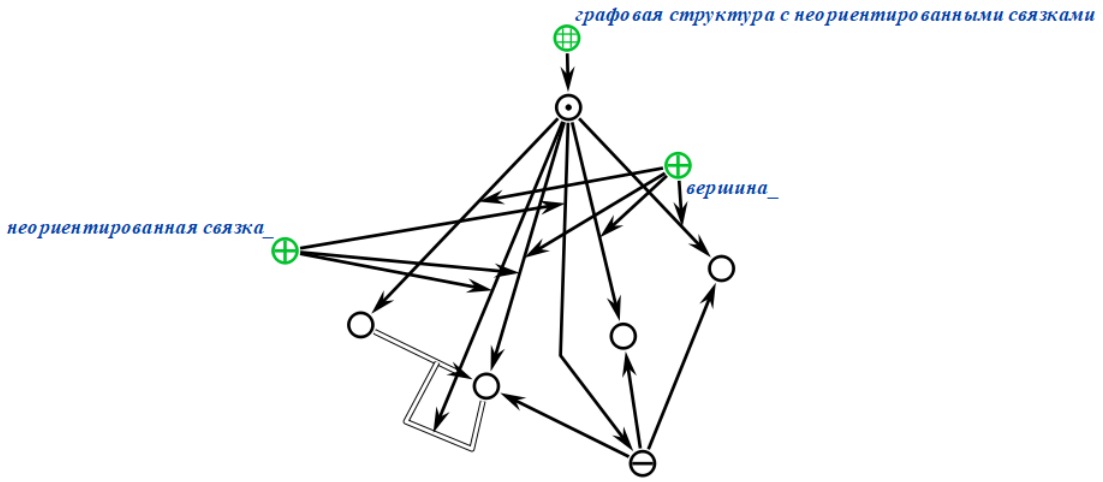

Рисунок *2.2* – Графовая структура с неориентированными связками

3. Графовая структура с ориентированными связками (абсолютное понятие).

   a. Ориентированная связка (относительное понятие, ролевое отношение) – связка, которая задается ориентированным множеством.

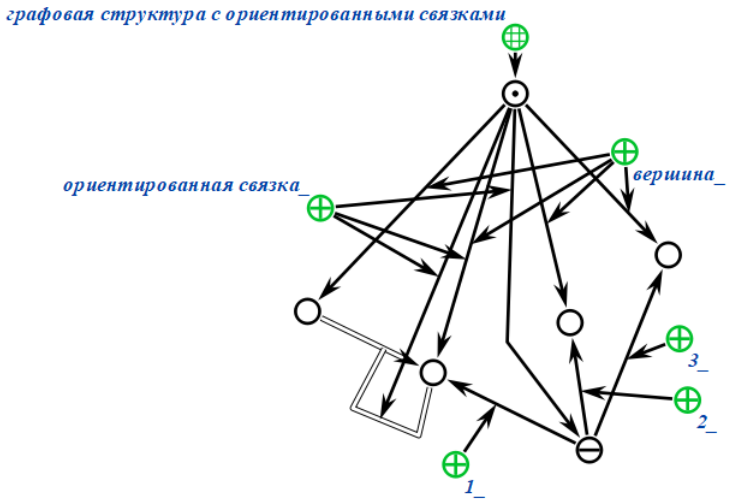

Рисунок *2.3* – Графовая структура с ориентированными связками

4. Гиперграф (абсолютное понятие) – это такая графовая структура, в которой связки могут связывать только вершины:

   a. Гиперсвязка (относительное понятие, ролевое отношение);
   b. Гипердуга (относительное понятие, ролевое отношение) – ориентированная гиперсвязка;
   c. Гиперребро (относительное понятие, ролевое отношение) – неориентированная гиперсвязка.

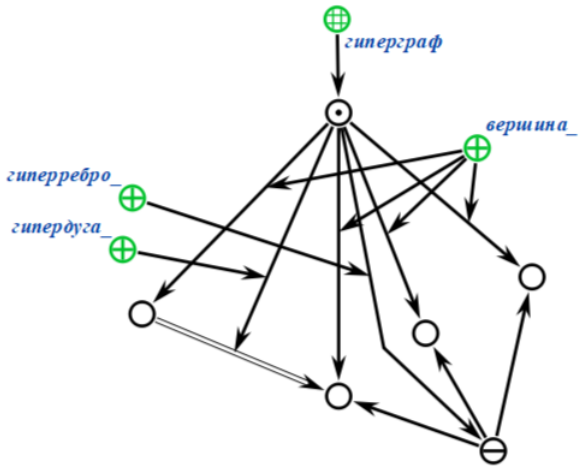

Рисунок *2.4* – Гиперграф

5. Псевдограф (абсолютное понятие) – это такой гиперграф, в котором все связки должны быть бинарными.

   a. Бинарная связка (относительное понятие, ролевое отношение) – гиперсвязка арности 2;
   b. Ребро (относительное понятие, ролевое отношение) – неориентированная гиперсвязка;
   c. Дуга (относительное понятие, ролевое отношение) – ориентированная гиперсвязка;
   d. Петля (относительное понятие, ролевое отношение) – бинарная связка, у которой первый и второй компоненты совпадают.

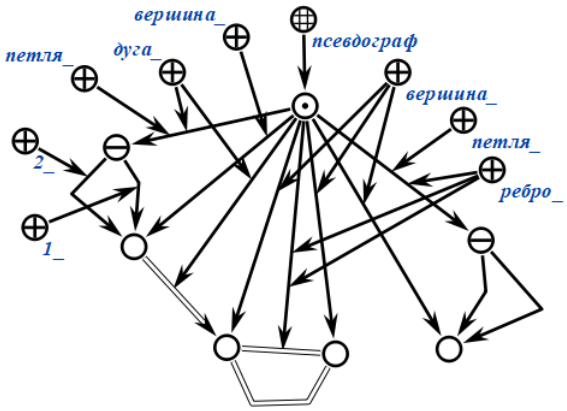

Рисунок *2.5* – Псевдограф

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

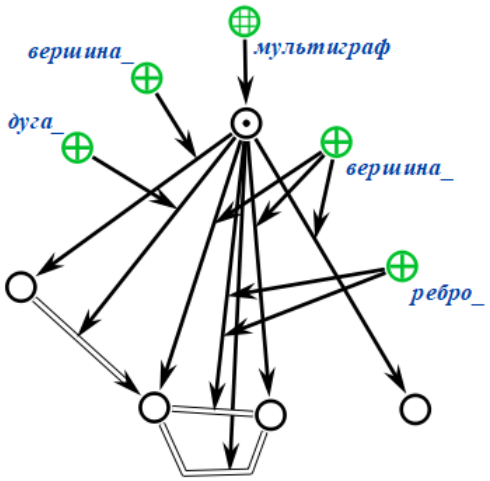

Рисунок *2.6* – Мультиграф

7. Графовая структура с ориентированными связками (абсолютное понятие).

   a. Ориентированная связка (относительное понятие, ролевое отношение) – связка, которая задается ориентированным множеством.

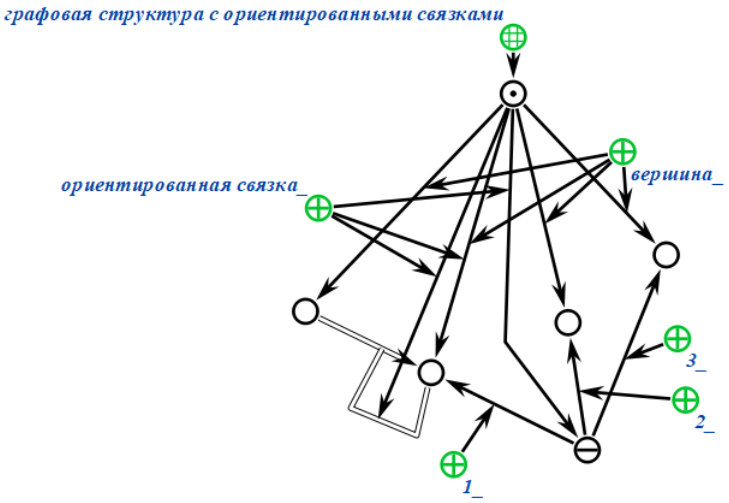

Рисунок *2.7* – Графовая структура с ориентированными связками

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

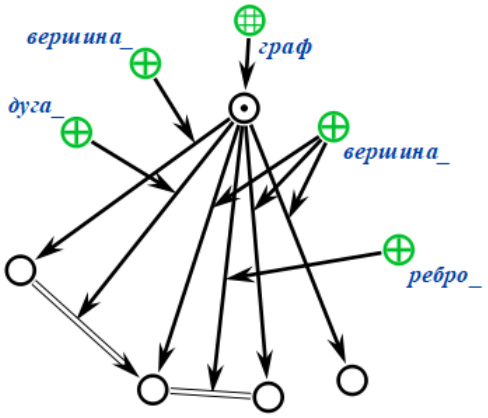

Рисунок *2.8* – Граф

9. Графовая структура с ориентированными связками (абсолютное понятие).

   a. Ориентированная связка (относительное понятие, ролевое отношение) – связка, которая задается ориентированным множеством.

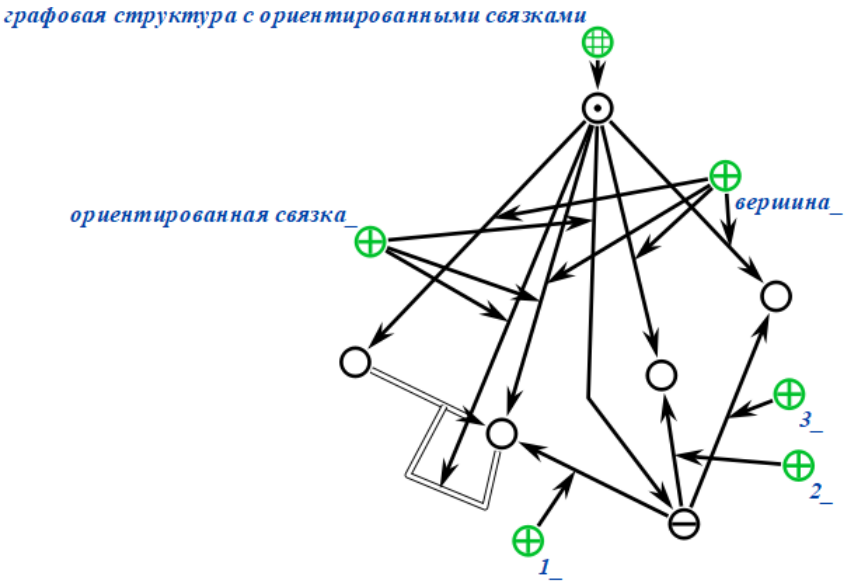

Рисунок *2.9* – Графовая структура с ориентированными связками

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

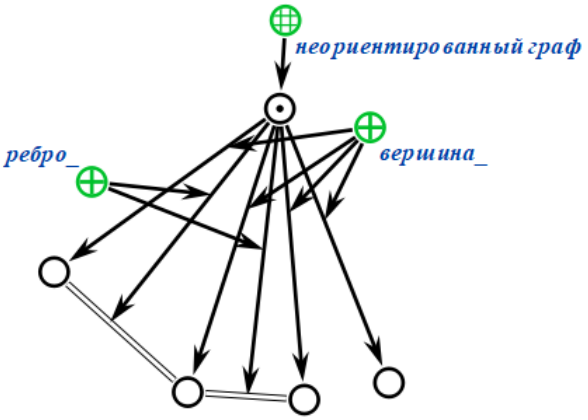

Рисунок *2.10* – Неориентированный граф

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

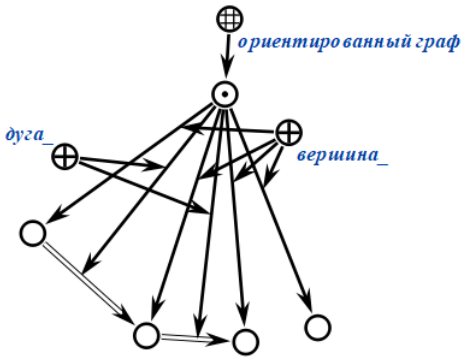

Рисунок *2.11* – Ориентированный граф

12. Маршрут (относительное понятие, бинарное ориентированное отношение) – это чередующаяся последовательность вершин и гиперсвязок в гиперграфе, которая начинается и кончается вершиной, и каждая гиперсвязка последовательности инцидентна двум вершинам, одна из которых непосредственно предшествует ей, а другая непосредственно следует за ней.

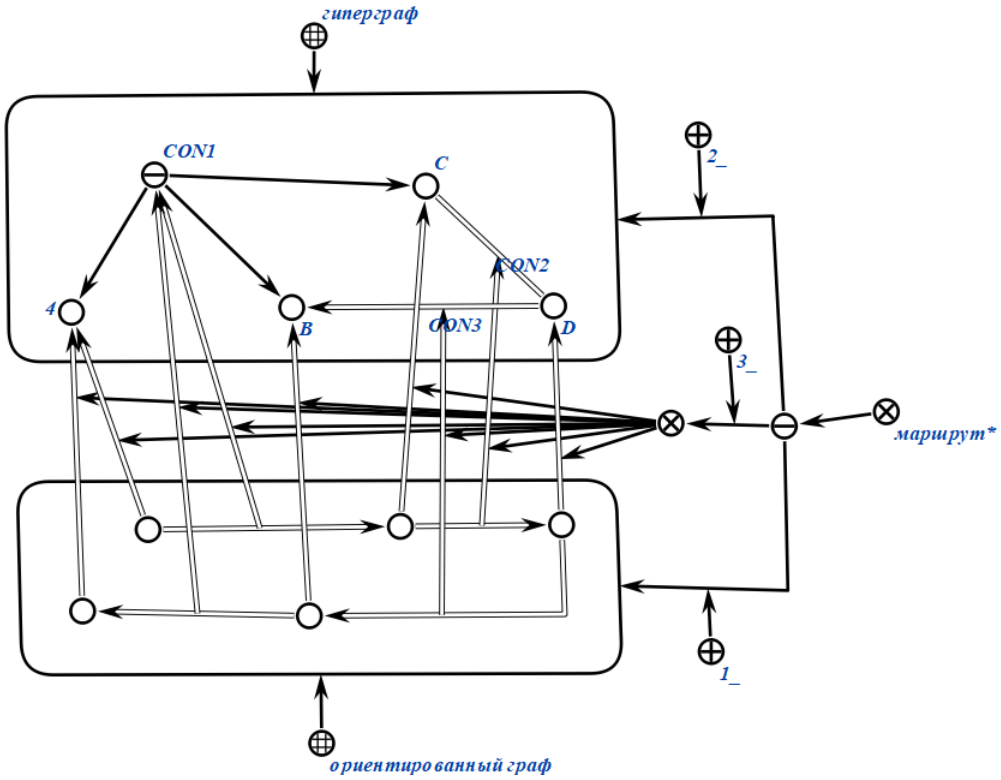

Рисунок *2.12* – Маршрут

13. Цепь (относительное понятие, бинарное ориентированное отношение) – это маршрут, все гиперсвязки которого различны.

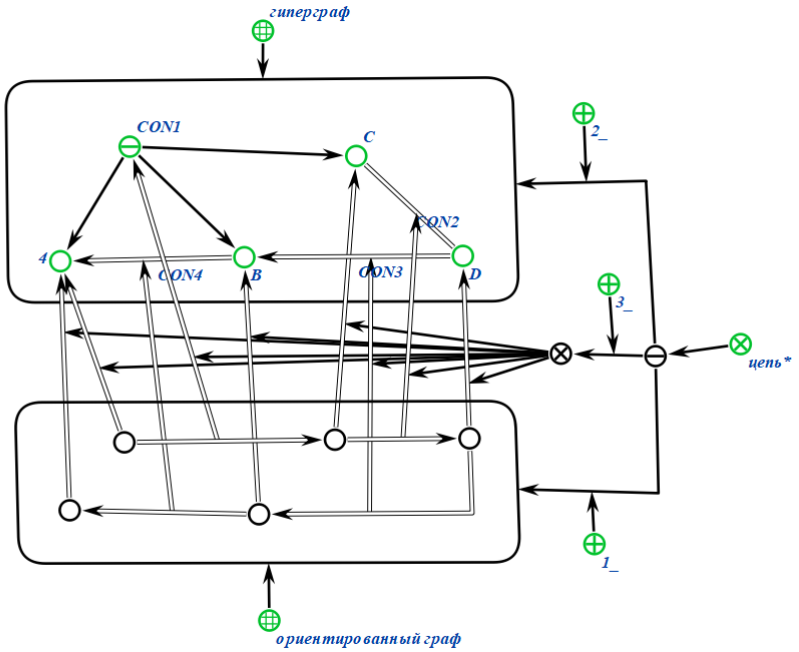

Рисунок *2.13* – Цепь

14. Простая цепь, путь (относительное понятие, бинарное ориентированное отношение) – это цепь, в которой все вершины различны.

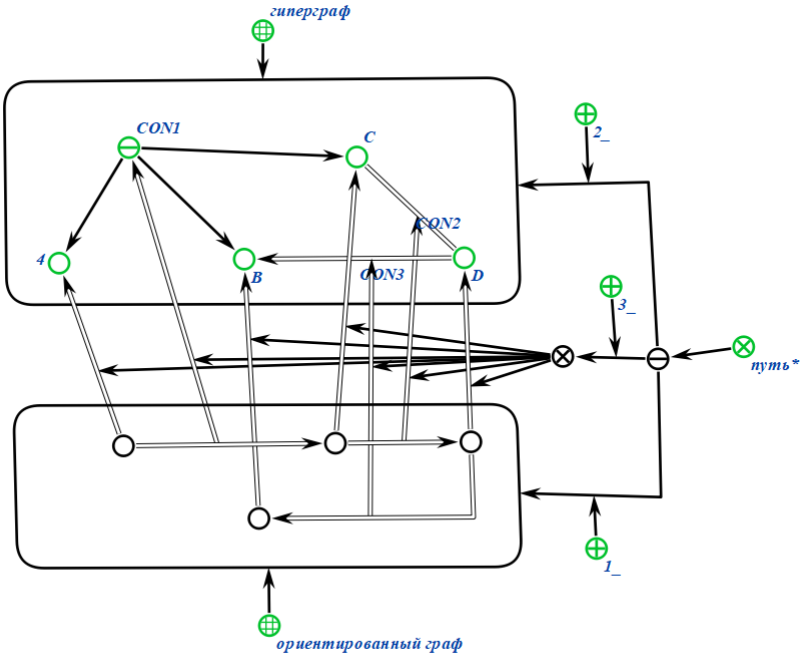

Рисунок *2.14* – Путь

15. Связный граф (абсолютное понятие) — это граф, содержащий ровно одну компоненту связности. Это означает, что между любой парой вершин этого графа существует как минимум один путь.

    a. Компонента связности (относительное понятие, ролевое отношение) – это часть графа, являющаяся связной.

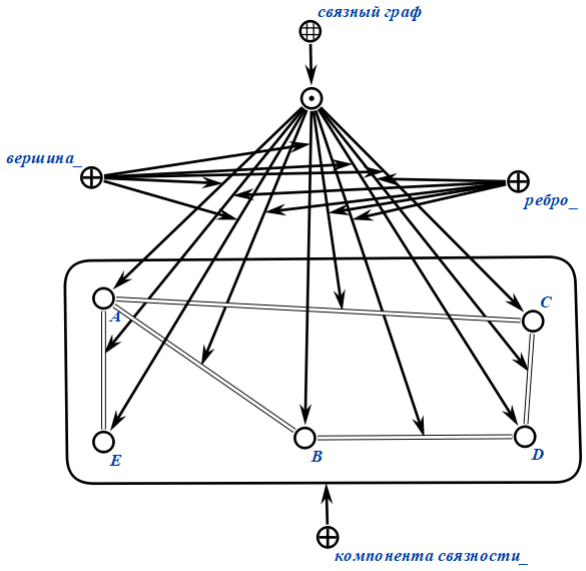

Рисунок *2.15* – Связный граф

16. Ациклический граф (абсолютное понятие) – это граф без циклов.

    a. Цикл (относительное понятие, бинарное ориентированное отношение) – это замкнутая цепь.

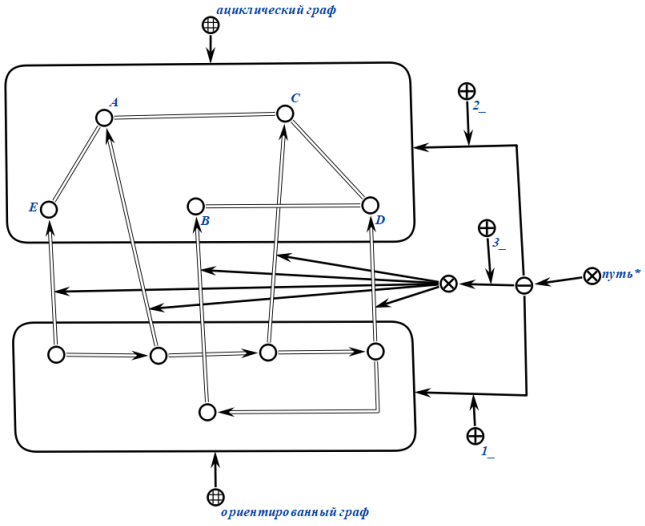

Рисунок *2.16* – Ациклический граф

17. Дерево (абсолютное понятие) – это связный ациклический граф.

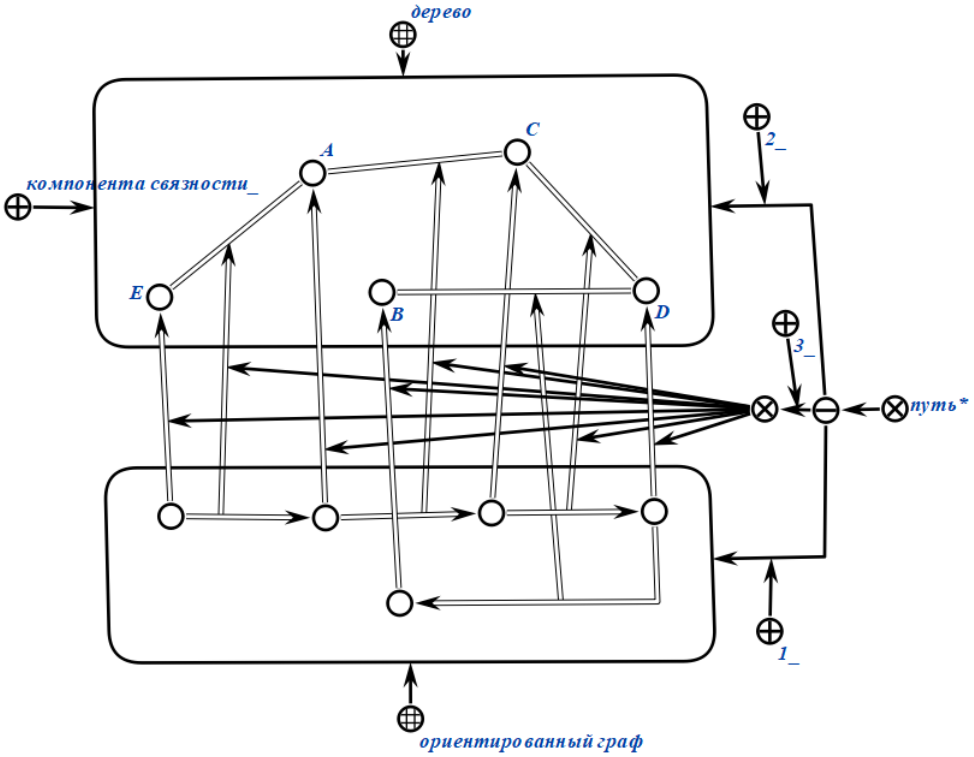

Рисунок *2.17* – Дерево

18. Взвешенный граф (абсолютное понятие) – это граф, каждому ребру которого поставлено в соответствие некоторое значение (вес ребра).

    a. Вес ребра (относительное понятие, не ролевое отношение) – свойство ребра иметь некоторое значение.
    b. Длина ребра (относительное понятие, не ролевое отношение) – значение веса ребра, которое является вещественным числом.

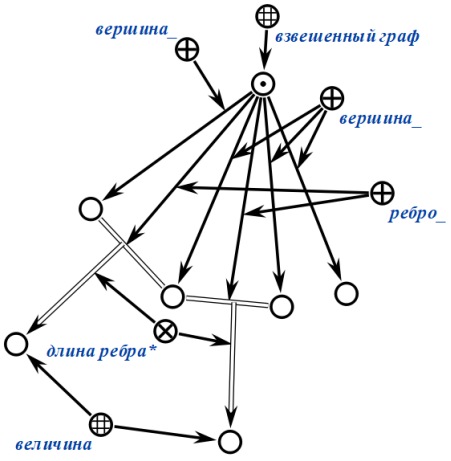

Рисунок *2.18* – Взвешенный граф

19. Дерево кратчайших путей (абсолютное понятие) – это дерево, состоящее из кратчайших путей от заданной вершины (корня) до любой вершины графа.

    a. Корень дерева (относительное понятие, ролевое отношение) – вершина, относительно которой строится дерево.

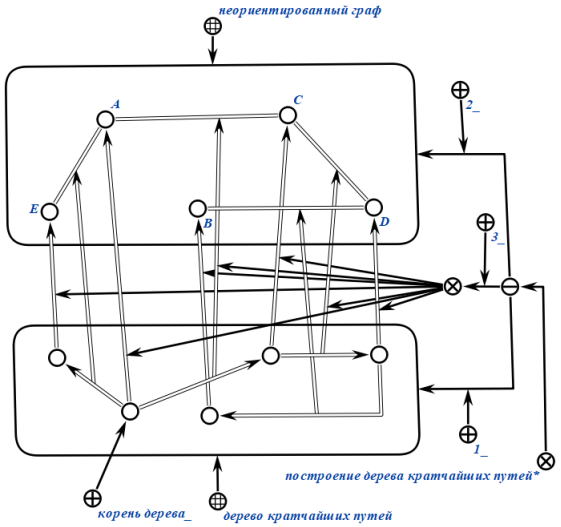

Рисунок *2.19* – Дерево кратчайших путей

3 ТЕСТОВЫЕ ПРИМЕРЫ

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

3.1 Тест 1

Вход:

Необходимо построить дерево кратчайших путей взвешенного неориентированного графа относительно корня в вершине 0.

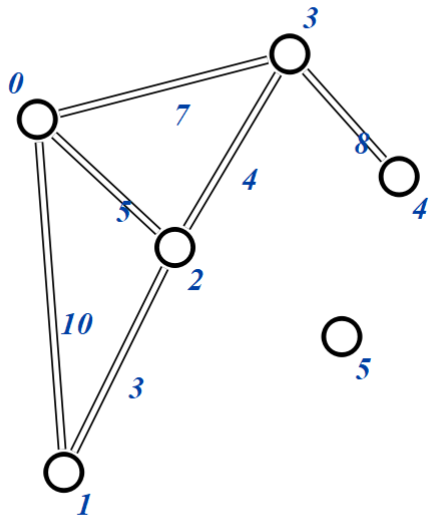

Рисунок *3.1* – Вход теста 1

Выход:

Дерево кратчайших путей для заданного графа и корневой вершины.

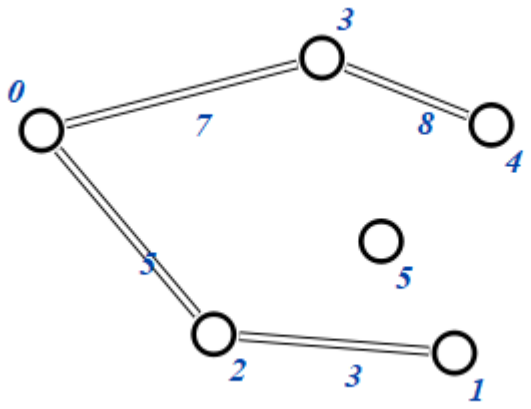

Рисунок *3.2* – Выход теста 1

3.2 Тест 2

Вход:

Необходимо построить дерево кратчайших путей взвешенного неориентированного графа относительно корня в вершине 0.

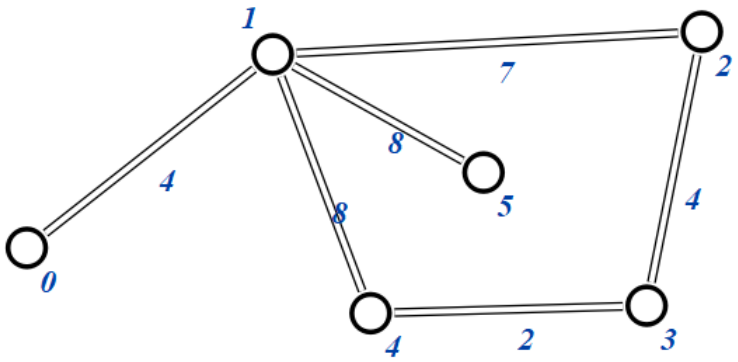

Рисунок *3.3* – Вход теста 2

Выход:

Дерево кратчайших путей для заданного графа и корневой вершины.

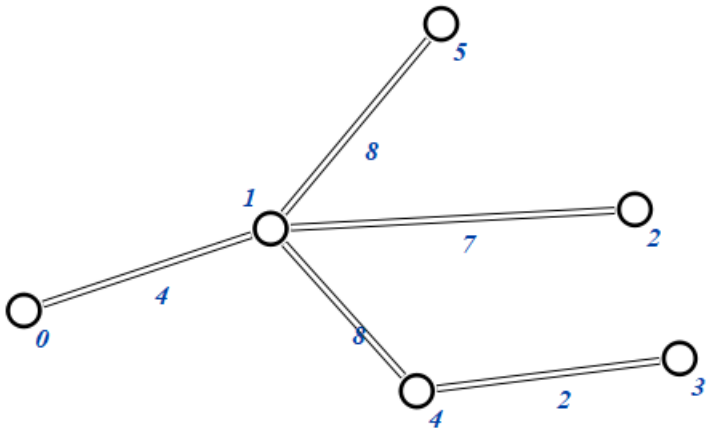

Рисунок *3.4* – Выход теста 2

3.3 Тест 3

Вход:

Необходимо построить дерево кратчайших путей взвешенного неориентированного графа относительно корня в вершине 0.

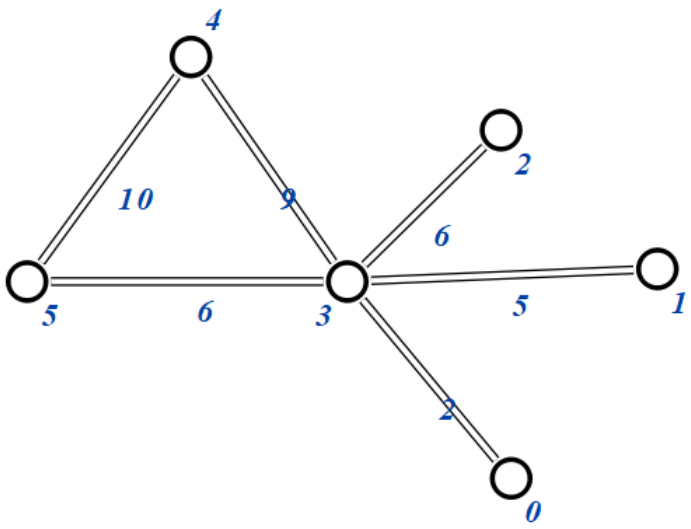

Рисунок *3.5* – Вход теста 3

Выход:

Дерево кратчайших путей для заданного графа и корневой вершины.

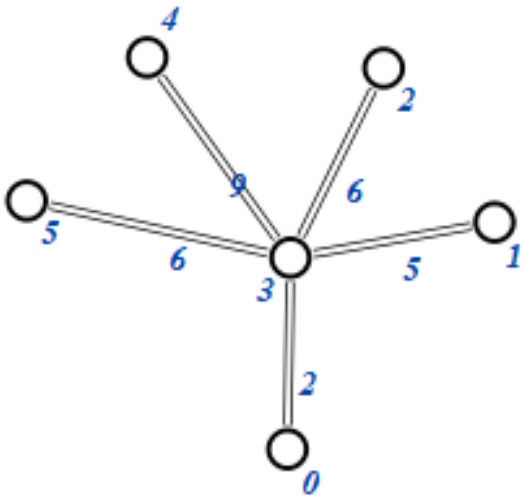

Рисунок *3.6* – Выход теста 3

3.4 Тест 4

Вход:

Необходимо построить дерево кратчайших путей взвешенного неориентированного графа относительно корня в вершине 0.

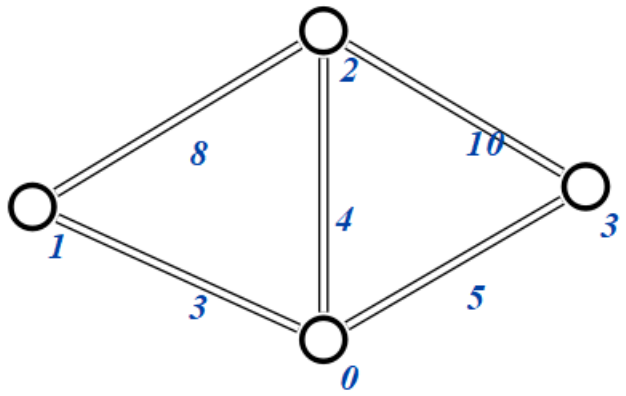

Рисунок *3.7* – Вход теста 4

Выход:

Дерево кратчайших путей для заданного графа и корневой вершины.

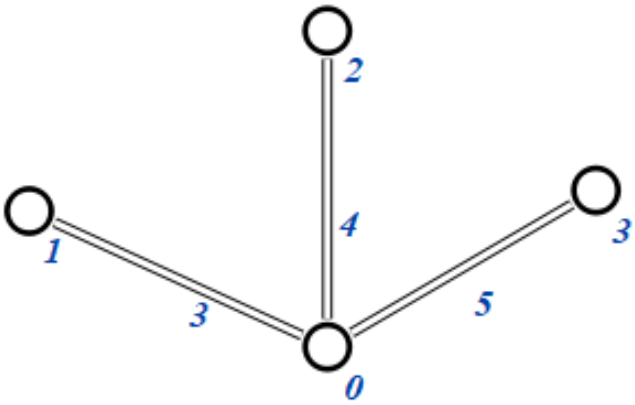

Рисунок *3.8* – Выход теста 4

3.5 Тест 5

Вход:

Необходимо построить дерево кратчайших путей взвешенного неориентированного графа относительно корня в вершине 0.

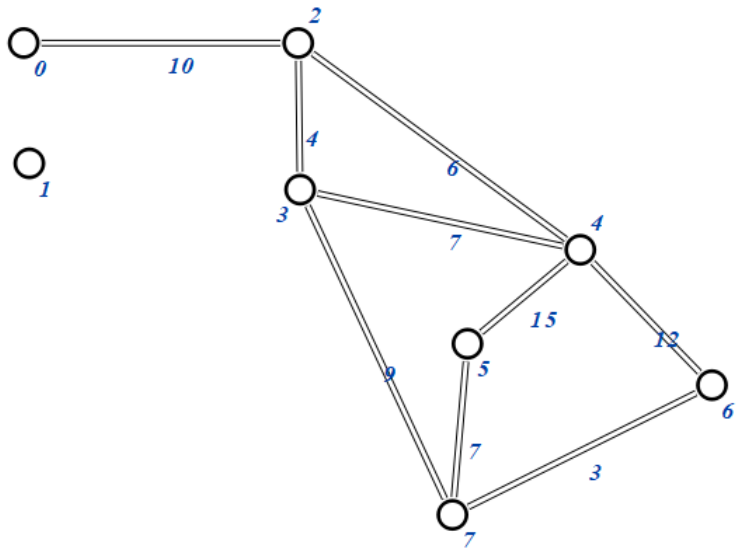

Рисунок *3.9* – Вход теста 1

Выход:

Дерево кратчайших путей для заданного графа и корневой вершины.

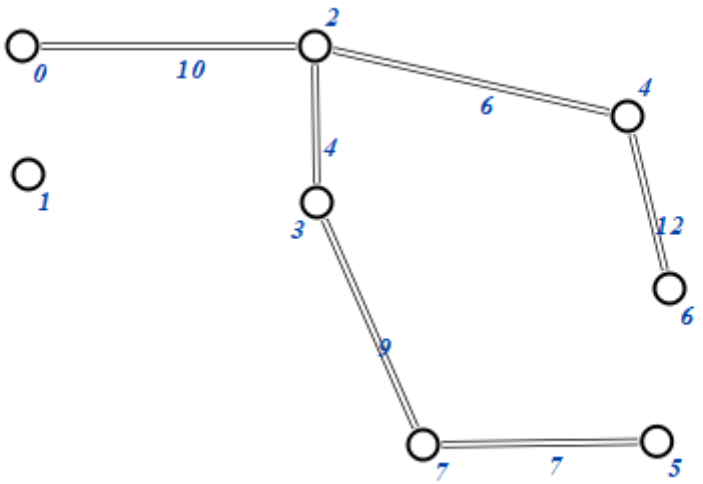

Рисунок *3.10* – Выход теста 5

4 ПРИМЕР РАБОТЫ АЛГОРИТМА В СЕМАНТИЧЕСКОЙ ПАМЯТИ

1. Задание входного графа и корневой вершины

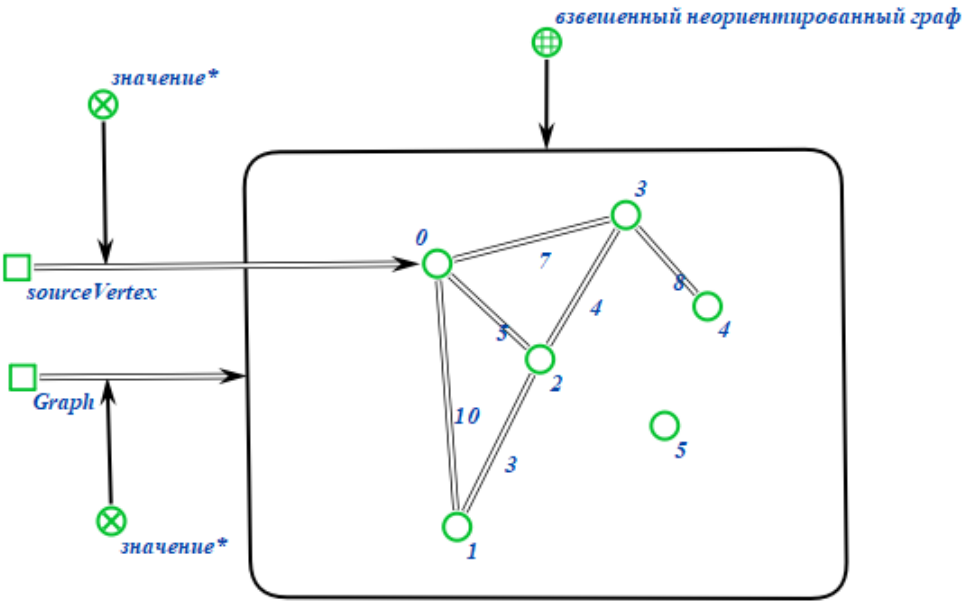

Рисунок *4.1* – Шаг 1

– Переменная *sourceVertex* получит в качестве значения узел, указанный в качестве корня будущего дерева кратчайших путей (узел, относительно которого будут определяться длины кратчайших путей до каждой из остальных вершин.

– Переменная *Graph* получит в качестве значения sc-узел неориентированного взыешенного графа.

2. Создание множеств соседей текущей вершины (корневой), пустого множества длин да каждой из вершин, пустого множества предков текущей вершины

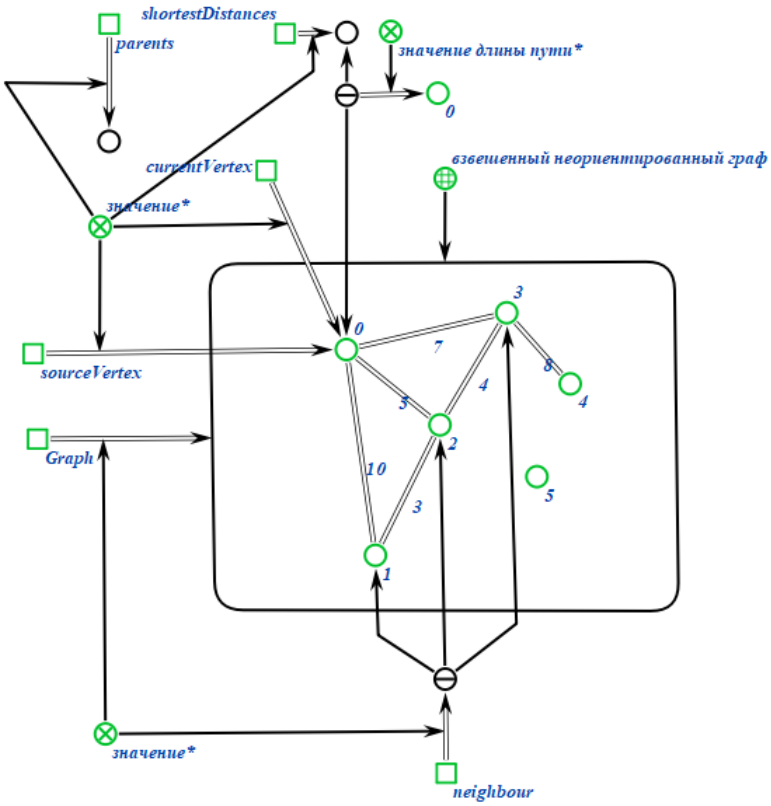

Рисунок *4.2* – Шаг 2

– Переменная *currentVertex* получит в качестве значения первую (корневую) вершину графа.

– Переменная *parents* для текущей вершины - пустое множество.

– Переменная *shortestDistances* для текущей вершины - пустое множество.

– Переменная - *neighbour* получает множество значений вершин графа - соседей *currentVertex*.

3. Определение расстояния от *currentVertex* до первой соседней вершины

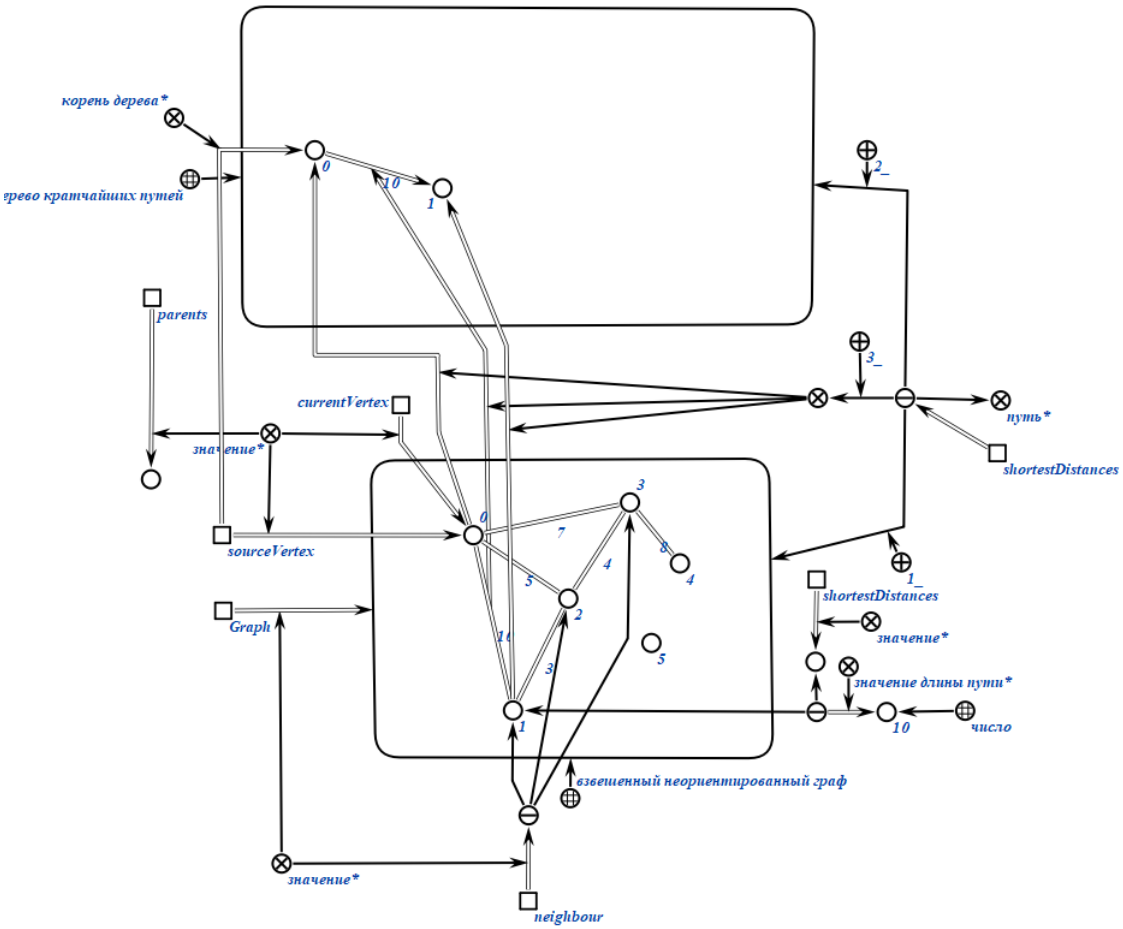

Рисунок *4.3* – Шаг 3

– Переменная*shortestDistances* получает в качестве значения sc-узел будущего дерева кратчайших путей (добавление вершин 0 и 1, а также связи между ними).

– Относительно данной вершины из *neighbour* для *currentVertex* во множестве *shortestDistances* за вершиной номер 1 графа закрепляется длина пути равная 10.

4. Определение расстояния от *currentVertex* до второй соседней вершины

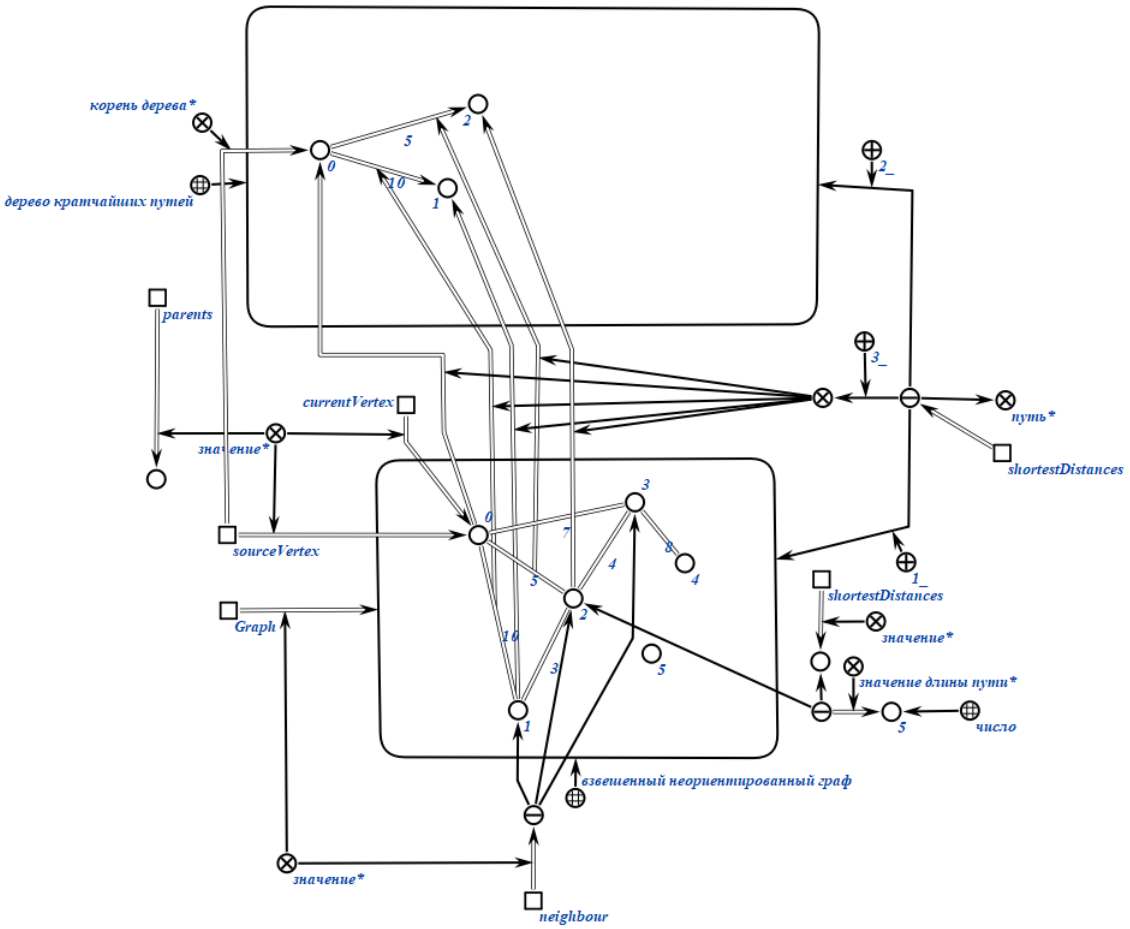

Рисунок *4.4* – Шаг 4

– Переменная*shortestDistances* обновляет sc-узел дерева кратчайших путей новой вершиной 2 (указание её связи с *currentVertex*).

– Относительно данной вершины из *neighbour* для *currentVertex* во множестве *shortestDistances* за вершиной номер 2 графа закрепляется длина пути равная 5.

5. Определение расстояния от *currentVertex* до третьей соседней вершины

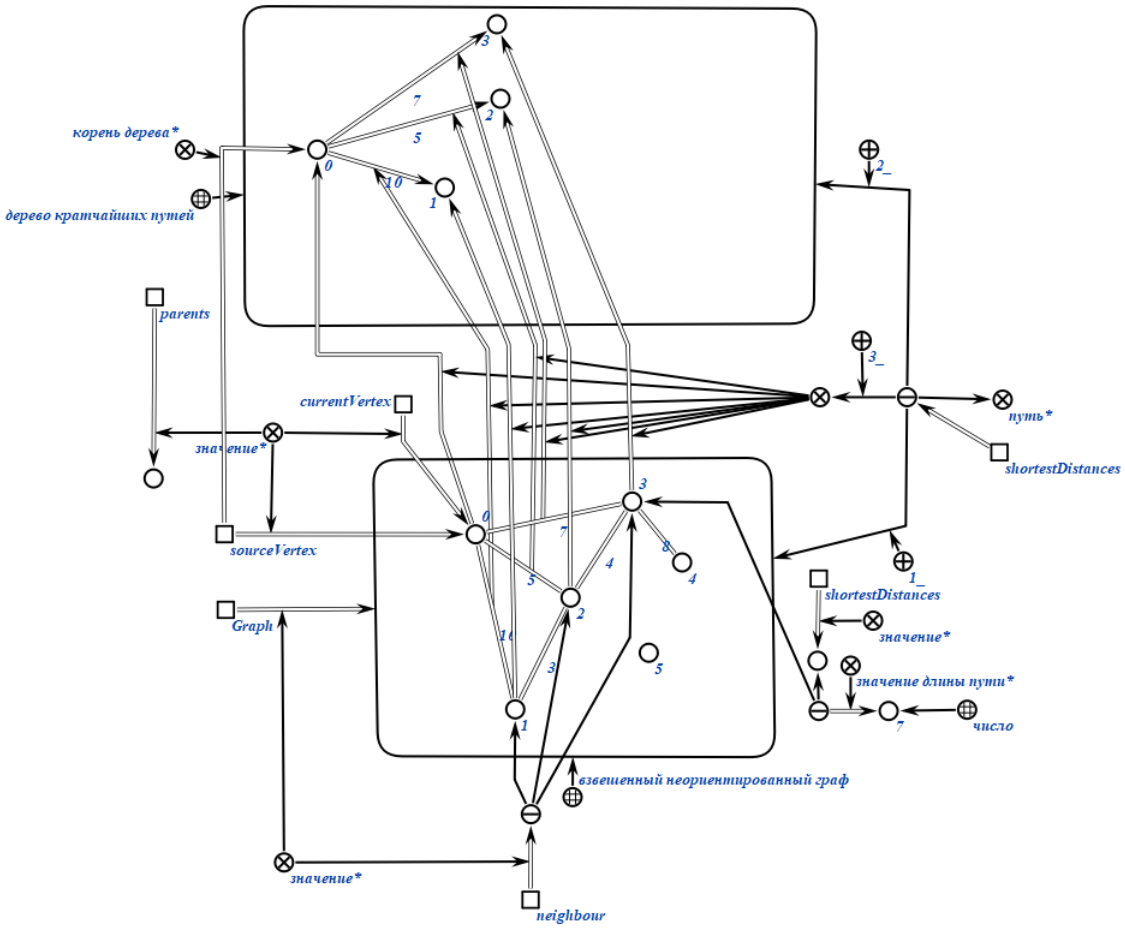

Рисунок *4.5* – Шаг 5

– Переменная*shortestDistances* обновляет sc-узел дерева кратчайших путей новой вершиной 3 (указание её связи с *currentVertex*).

– Относительно данной вершины из *neighbour* для *currentVertex* во множестве *shortestDistances* за вершиной номер 3 графа закрепляется длина пути равная 7.

6. Переход к определению путей для следующей вершины графа, обновление значения *currentVertex*

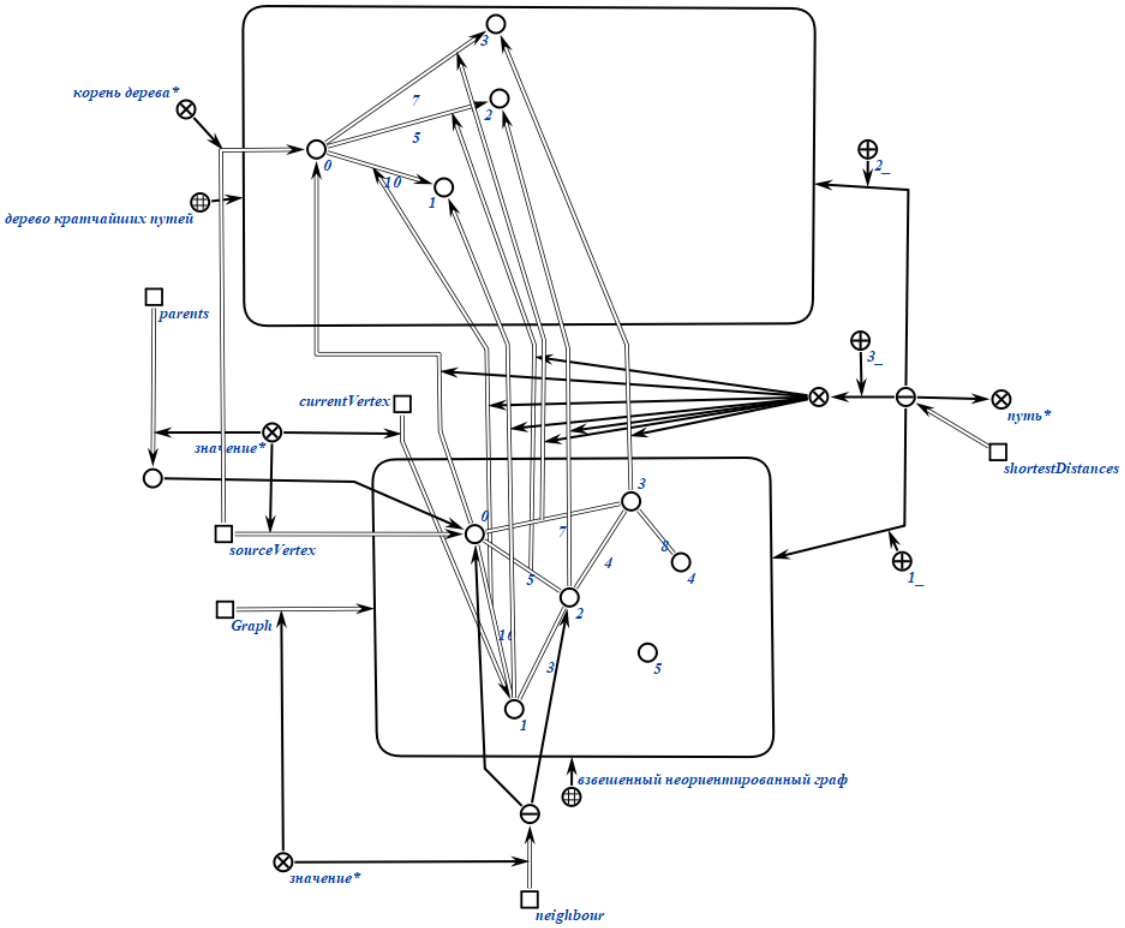

Рисунок *4.6* – Шаг 6

– Переменная *currentVertex* принимает значение следующей вершины графа, равное 1.

– Переменная *neighbour* принимает значение множества соседей *currentVertex*.

– Для *currentVertex* переменная *parents* принимает значение равное 0.

– Т.к. суммарные длины путей через *currentVertex* к остальным вершинам из *neighbour* больше текущих длин путей от *sourceVertex*, то значение *shortestDistances* не изменяется.

7. Переход к определению путей для следующей вершины графа, обновление значения *currentVertex*, определение пути к вершине 1

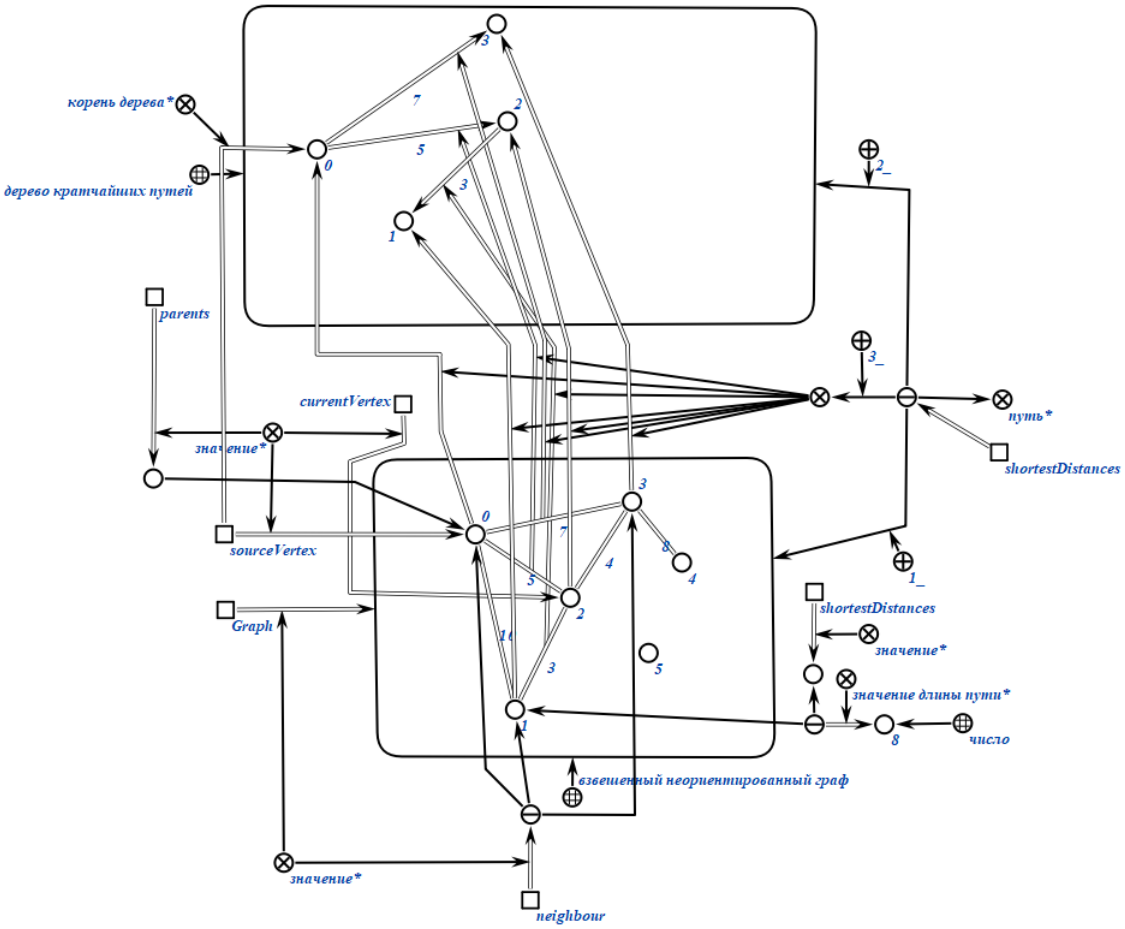

Рисунок *4.7* – Шаг 7

– Переменная *currentVertex* принимает значение следующей вершины графа, равное 2.

– Переменная *neighbour* принимает значение множества соседей *currentVertex*.

– Для *currentVertex* переменная *parents* принимает значение равное 0.

– Т.к. суммарная длина пути через *currentVertex* к вершине 1 меньше, чем значение закреплённое за ней в *shortestDistances*, то обновляем её значение на 8.

8. Определение длины пути к вершине 3 через *currrentVertex*

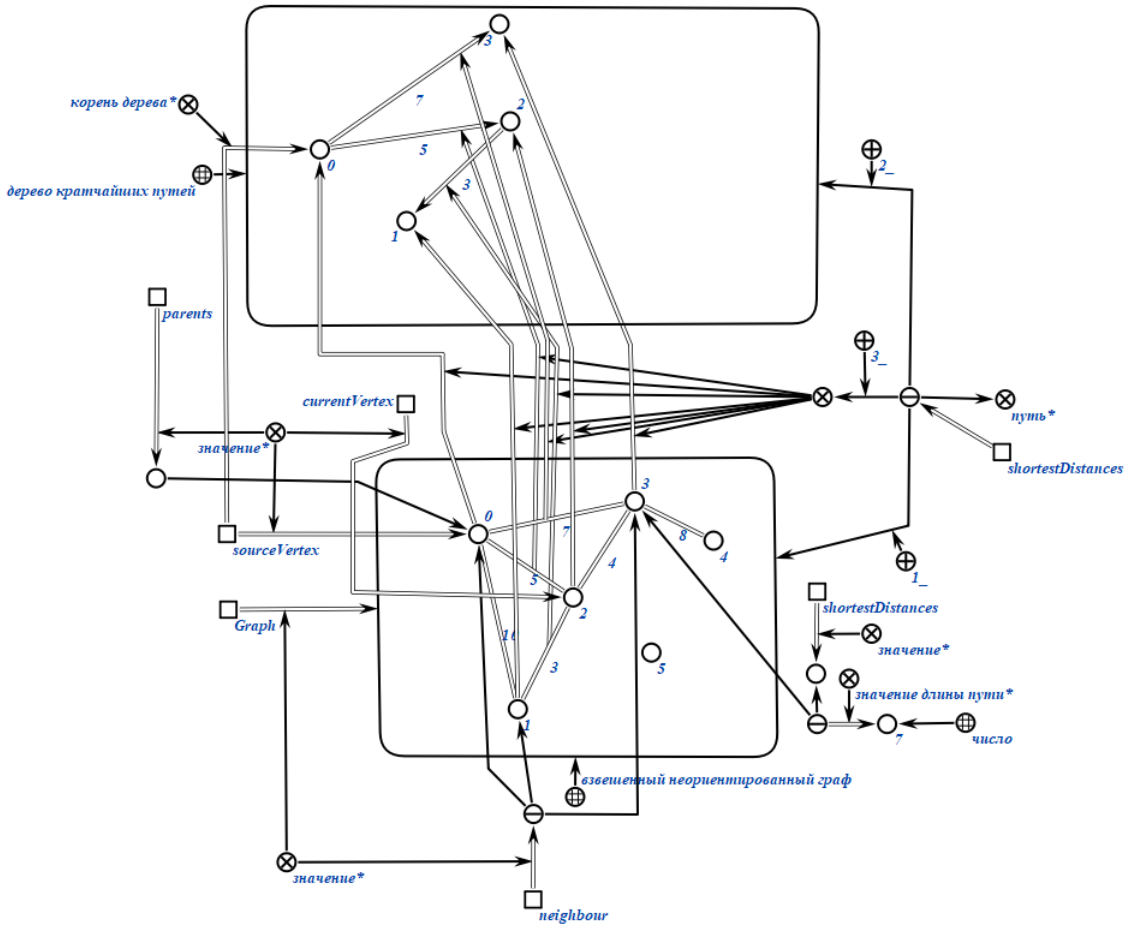

Рисунок *4.8* – Шаг 8

– Т.к. суммарная длина пути через *currentVertex* к вершине 3 больше, чем значение закреплённое за ней в *shortestDistances*, то её значение не изменяется и остаётся равным 8.

9. Переход к определению путей для следующей вершины графа, обновление значения *currentVertex*, определение длины пути к вершине 4

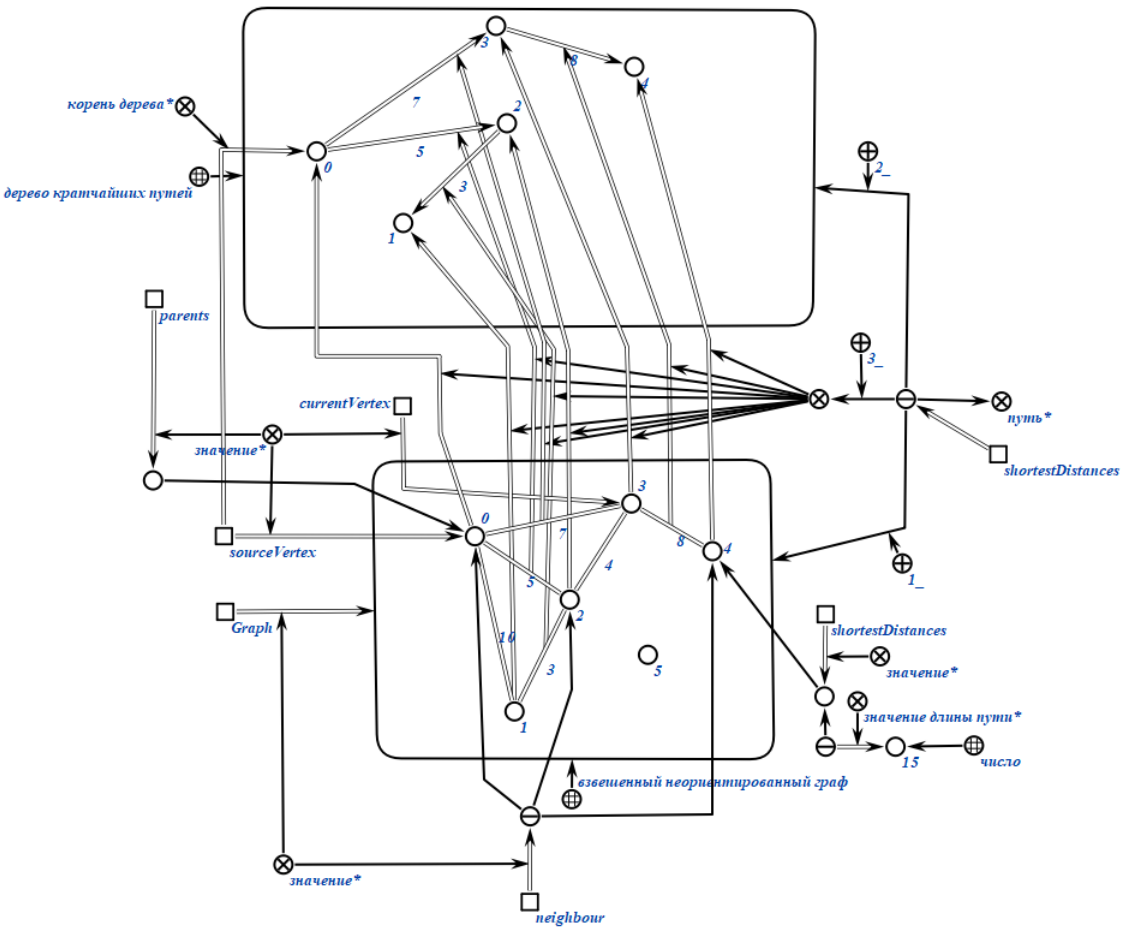

Рисунок *4.9* – Шаг 9

– Переменная *currentVertex* принимает значение следующей вершины графа, равное 3.

– Переменная *neighbour* принимает значение множества соседей *currentVertex*.

– Для *currentVertex* переменная *parents* принимает значение равное 0.

– Относительно данной вершины из *neighbour* для *currentVertex* во множестве *shortestDistances* за вершиной номер 4 графа закрепляется длина пути равная 15.

10. Переход к определению путей для следующей вершины графа, обновление значения *currentVertex*

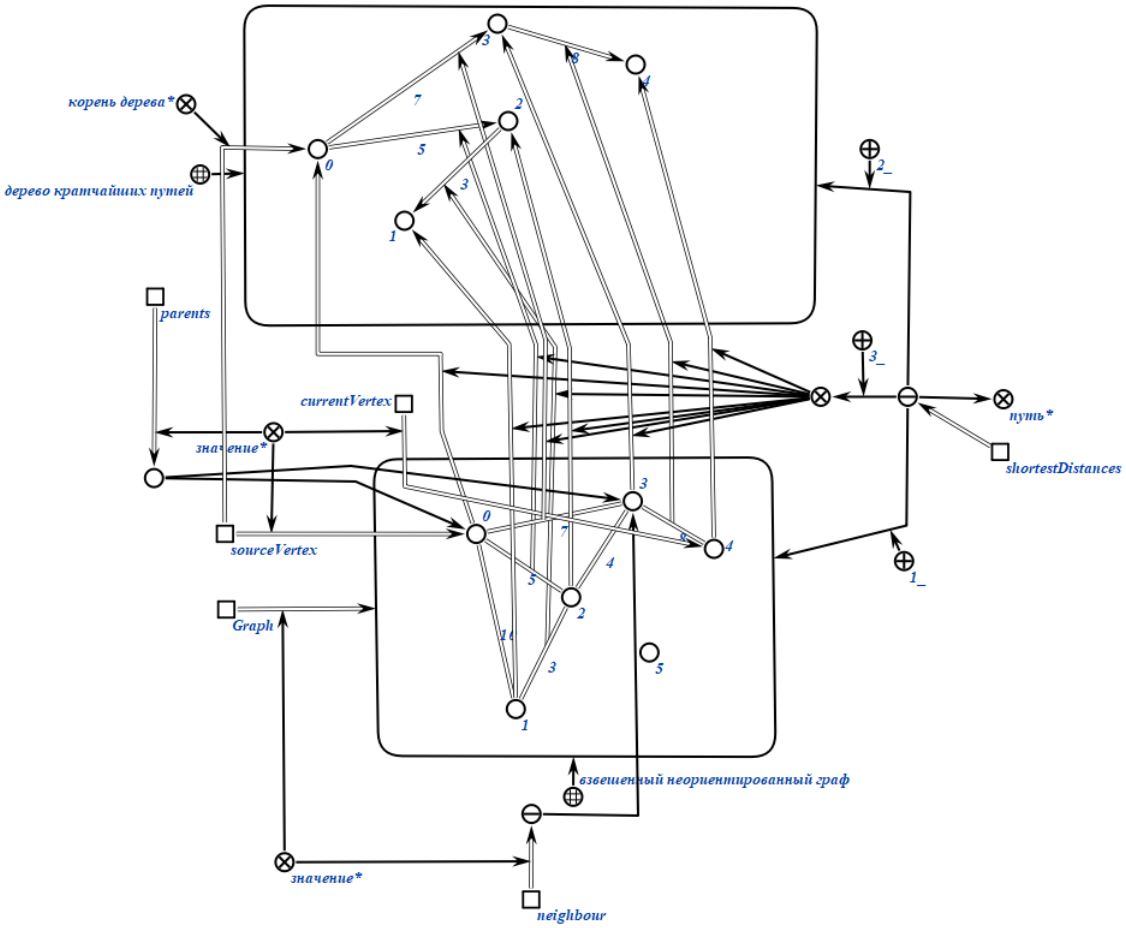

Рисунок *4.10* – Шаг 10

– Переменная *currentVertex* принимает значение следующей вершины графа, равное 4.

– Переменная *neighbour* принимает значение множества соседей *currentVertex*.

– Для *currentVertex* переменная *parents* принимает множества, включающее вершины 0, 3 графа.

11. Переход к определению путей для следующей вершины графа, обновление значения *currentVertex*

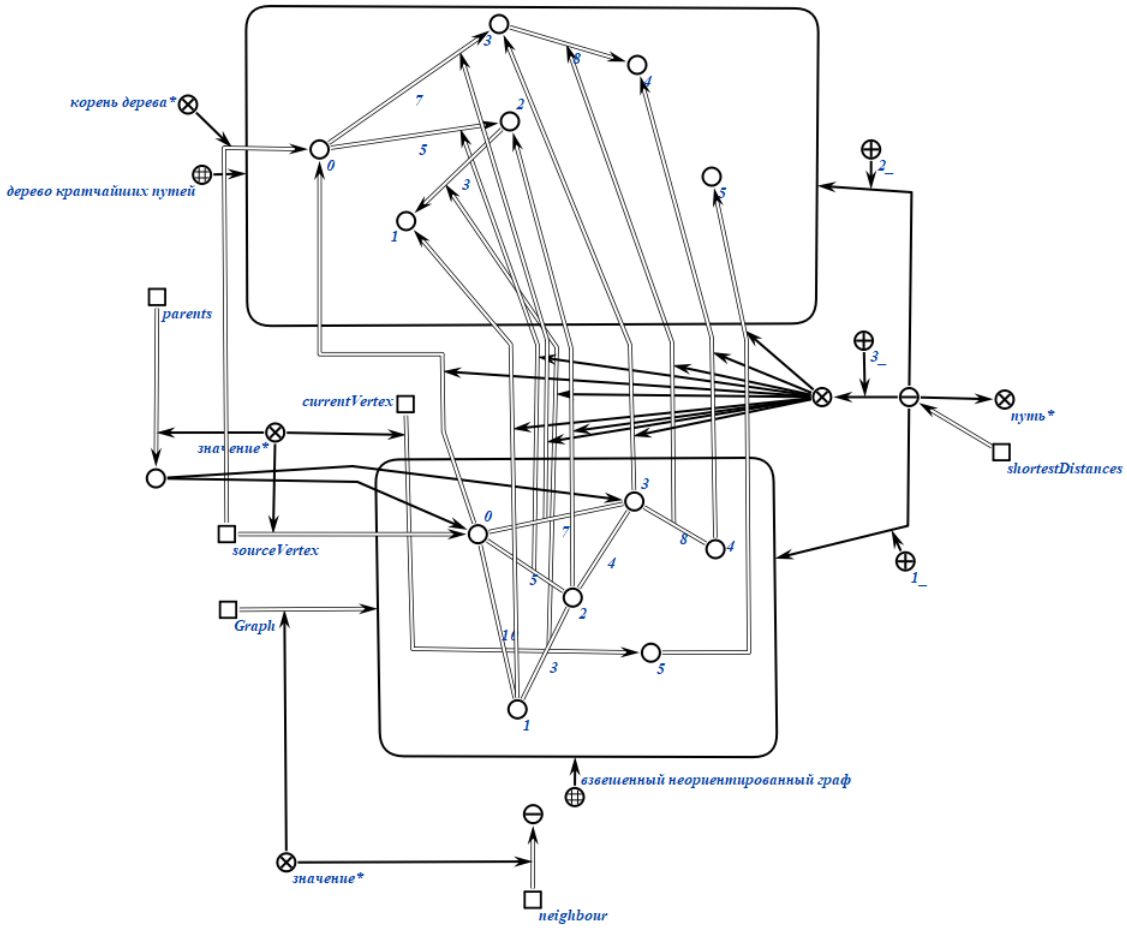

Рисунок *4.11* – Шаг 11

– Переменная *currentVertex* принимает значение следующей вершины графа, равное 5.

– Переменная *neighbour* принимает значение множества соседей *currentVertex* - пустое множество.

– Для *currentVertex* переменная *parents* является пустым множеством.

– Т.к. переменные *parents* и *neighbour* являются пустыми множествами, то вершина 5 графа является изолированной, т.е. до неё не существует пути - на основании этого обновляем значение *shortestDistances* закреплённое за данной вершиной.

Вывод ответа: На основании множества *shortestDistances* и закреплённых за каждой вершиной значений длин путей выводим построенное дерево кратчайших путей.

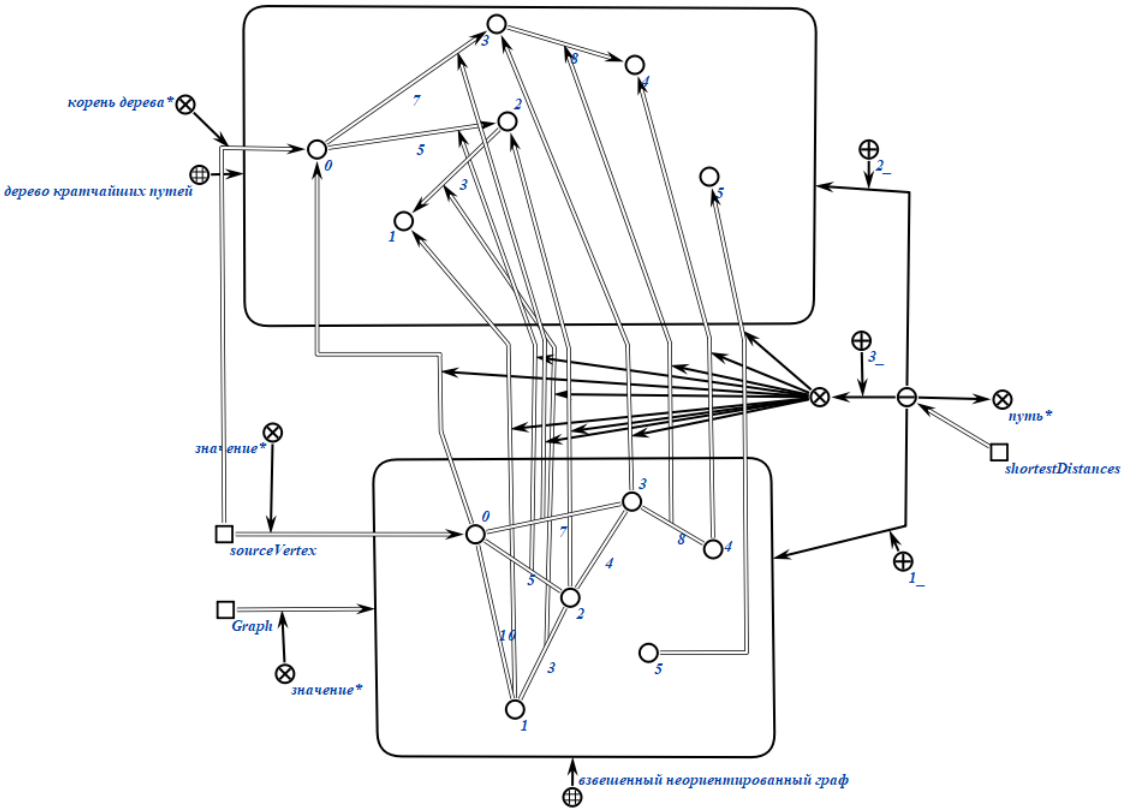

Рисунок *4.12* – Вывод - в семантической памяти

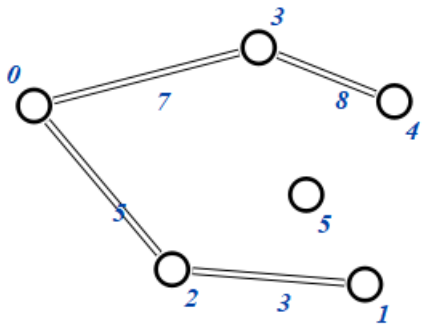

Рисунок *4.13* – Вывод

5 ЗАКЛЮЧЕНИЕ

5.1 Вывод:

В ходе выполнения данной расчётной работы была построена модель онтологии по решению графовой задачи ***« Поиск дерева кратчайших путей взвешенного неориентированного графа »***, содержащая описание данной работы, тестовые примеры, а также представление алгоритма выполнения данной задачи в семантической памяти.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Дистель, Р. Теория графов Пер. с англ. / Р. Дистель. — Изд-во Ин-та математики, 2002. — Р. 336.

[2] Харарри, Ф. Теория графов / Ф. Харарри. — Эдиториал УРСС, 2018. — Р. 304.