Проект на c++ использующий фреймворк qt 5 для отрисовки и редактирования специальных графов-деревьев через qt компонент QGraphicsScene(система Graphics View Framework в qt).

(С минимальным использованием qml)

Что можно использовать(посмотреть как устроено в nodeeditor):

1) https://github.com/paceholder/nodeeditor

Так же будет предоставлена код попытки использования кода из nodeeditor

Основная краткая идея программы:

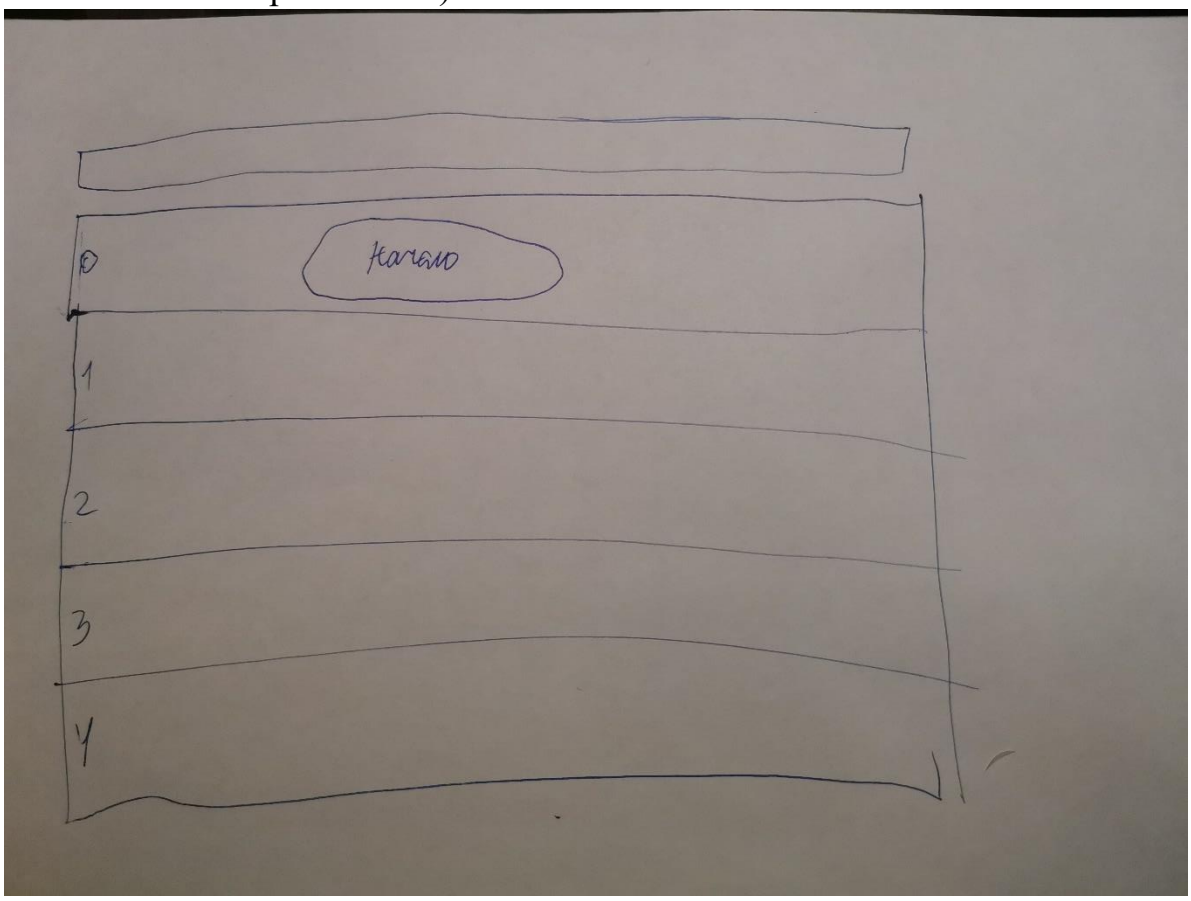
Программа редактор специальных деревьев. Состоит из нод, которые становятся на уровни и автоматически соединяются стрелочками в зависимости от того, куда поставлены. Ноды создаются двойным нажатием на место на каком то уровне. Колёсиком мыши можно приближать – удалять сцену. Так же можно перемещать ноды (изменять их присоединение к какой то группе узлов) (Это описано в Подробная идея программы).

В программе есть 2 типа нод: действие и вывод.

Можно приближать – удалять окно сцену с графом(деревом)?,

Подробная идея программы:

У нас Окно программы с виджетом-сценой на все окно программы (Окно можно сжимать-разжимать).

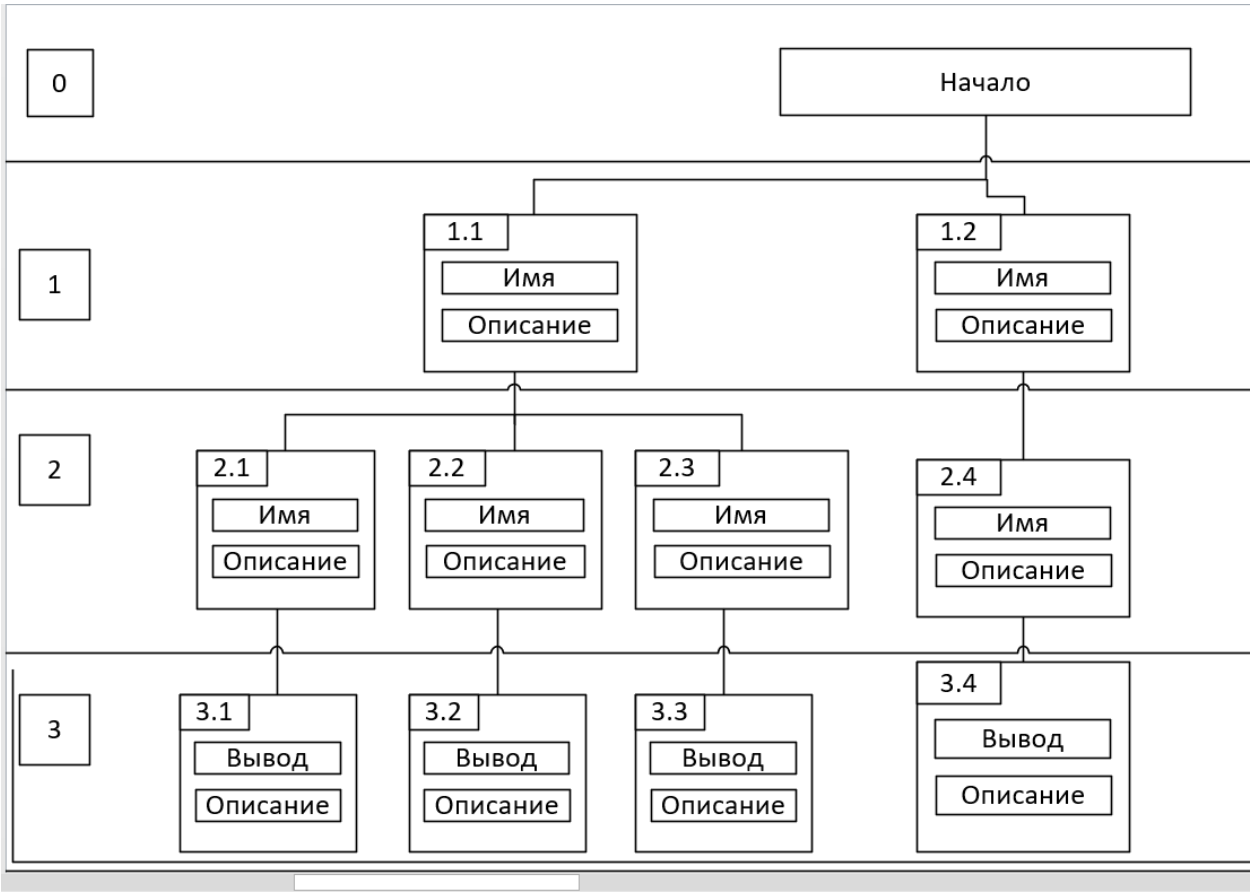


0 Начало

1

2

3



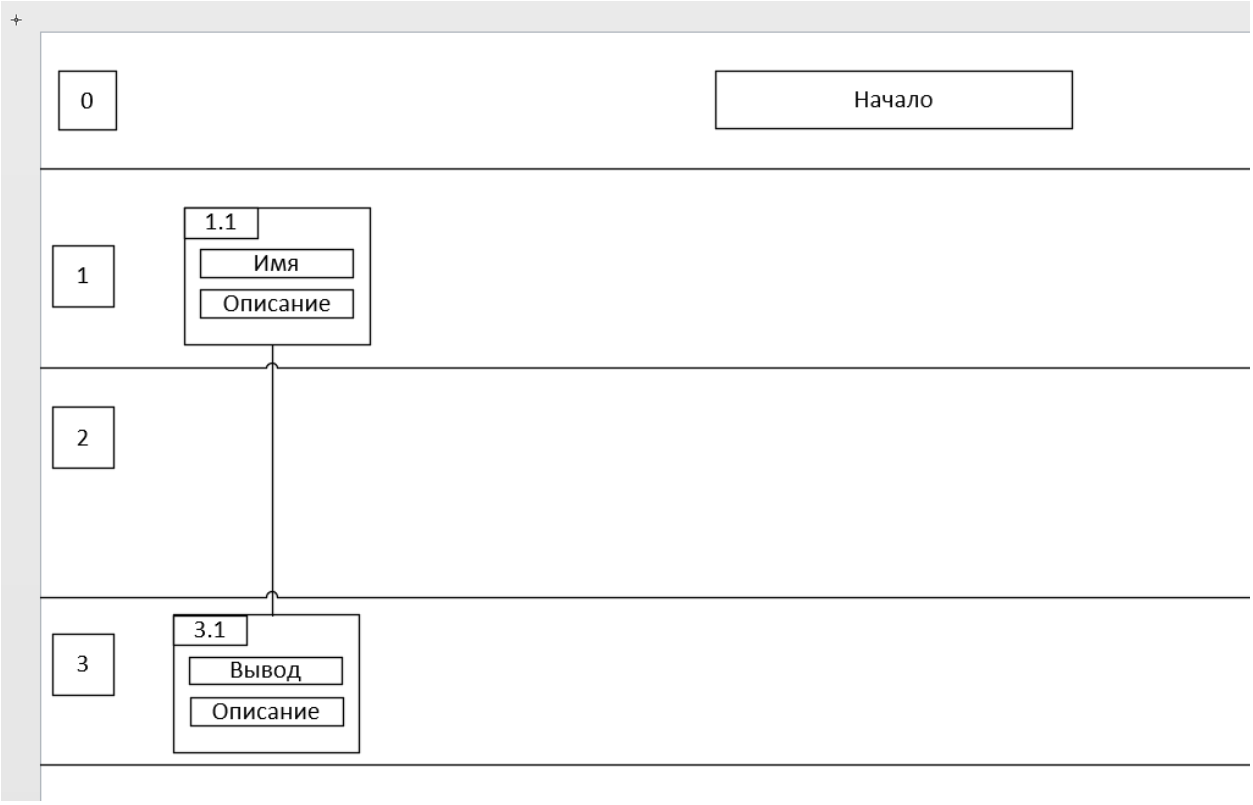
Примеры использования со стороны пользователя:

У нас есть сцена с уровнями. После запуска программы Нода с именем “начало” на уровне 0. По умолчанию при запуске программы программа запрашивает максимальный уровень. (Максимально может быть уровней 255). На последнем уровне, например 3, как видно на картинке ниже, могут быть ноды только типа вывод.

Нажав 2 раза в пустом поле в пустом поле на 1 уровне создаётся нода на 1 уровне (прямоугольник с 2 полями имя и описание) и номер уровня и горизонтальный порядковый номер. При вбивании текста, текст должен переноситься и нода расширяться. Так же с нодой должна расширяться линия разделяющая уровни.

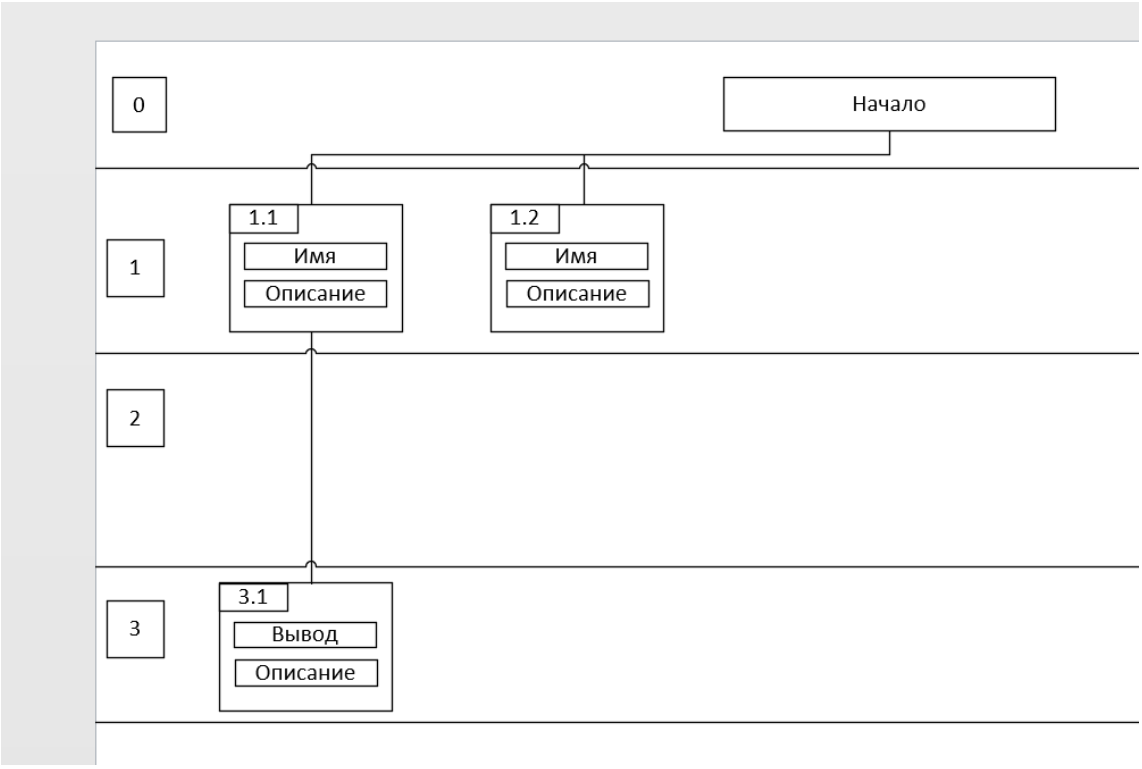
При клике на последнем уровне (который задаётся) создаётся нода другого типа “вывод”

Если добавляем ноду 1.1 на уровень 1,а затем уровень 3, то автоматически ставится стрелка от ноды 1.1 до ноды 3.

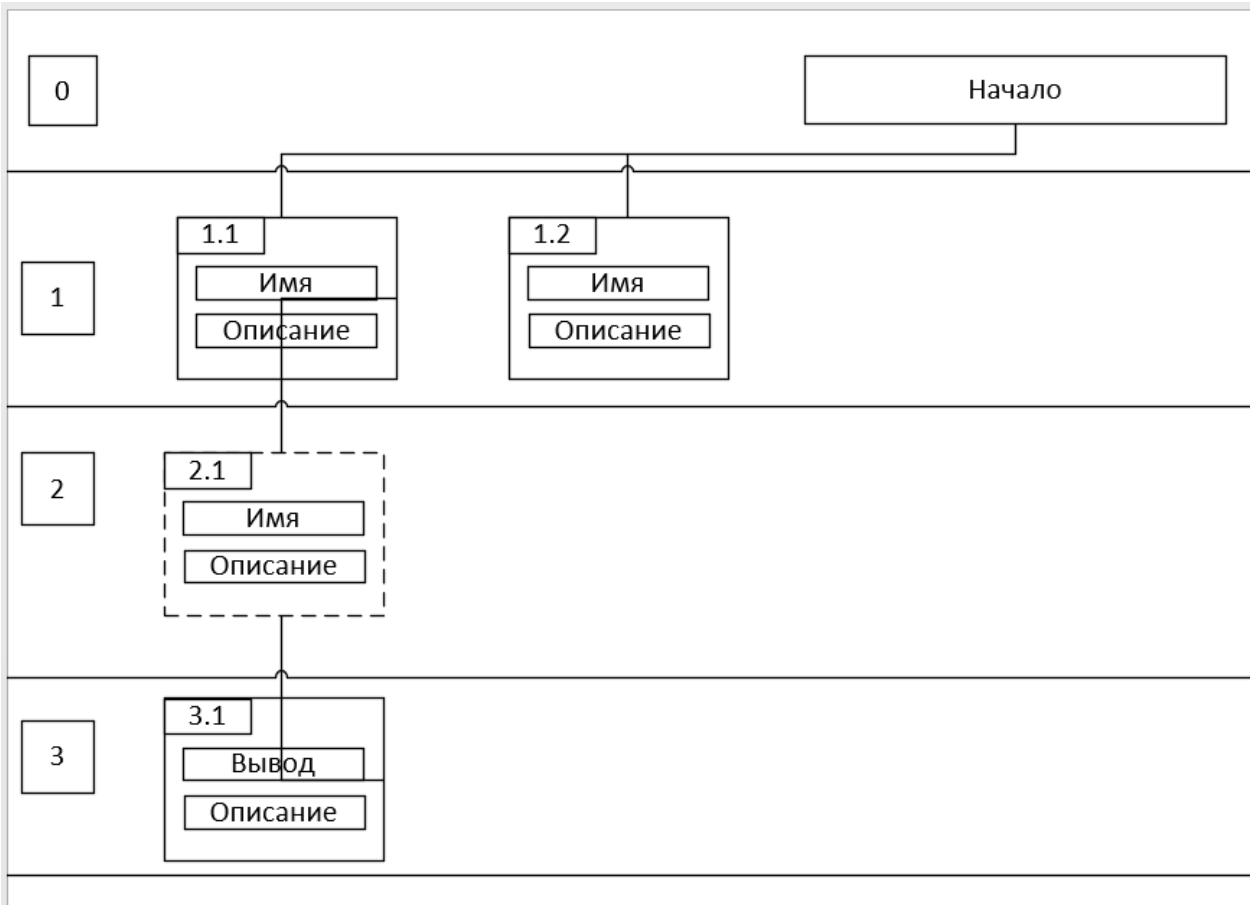


Функционал перетаскивания:

Так же реализовать перемещение ноды на другой уровень + зависимость от другого “parent”.



Хватаем ноду 1.2. На мыши появляется фантомное штриховое отображение ноды 1.2.Перетаскиваем фантомную ноду на уровень 2 под ноду 1.1 (там ещё проходит стрелка).На фантомной ноде нумерация которая после отпускания маши будет установлена (2.1)



После отпускания мыши на старом месте пропадает нода 1.2 и появляется на уровне 2 с измененной нумерацией.

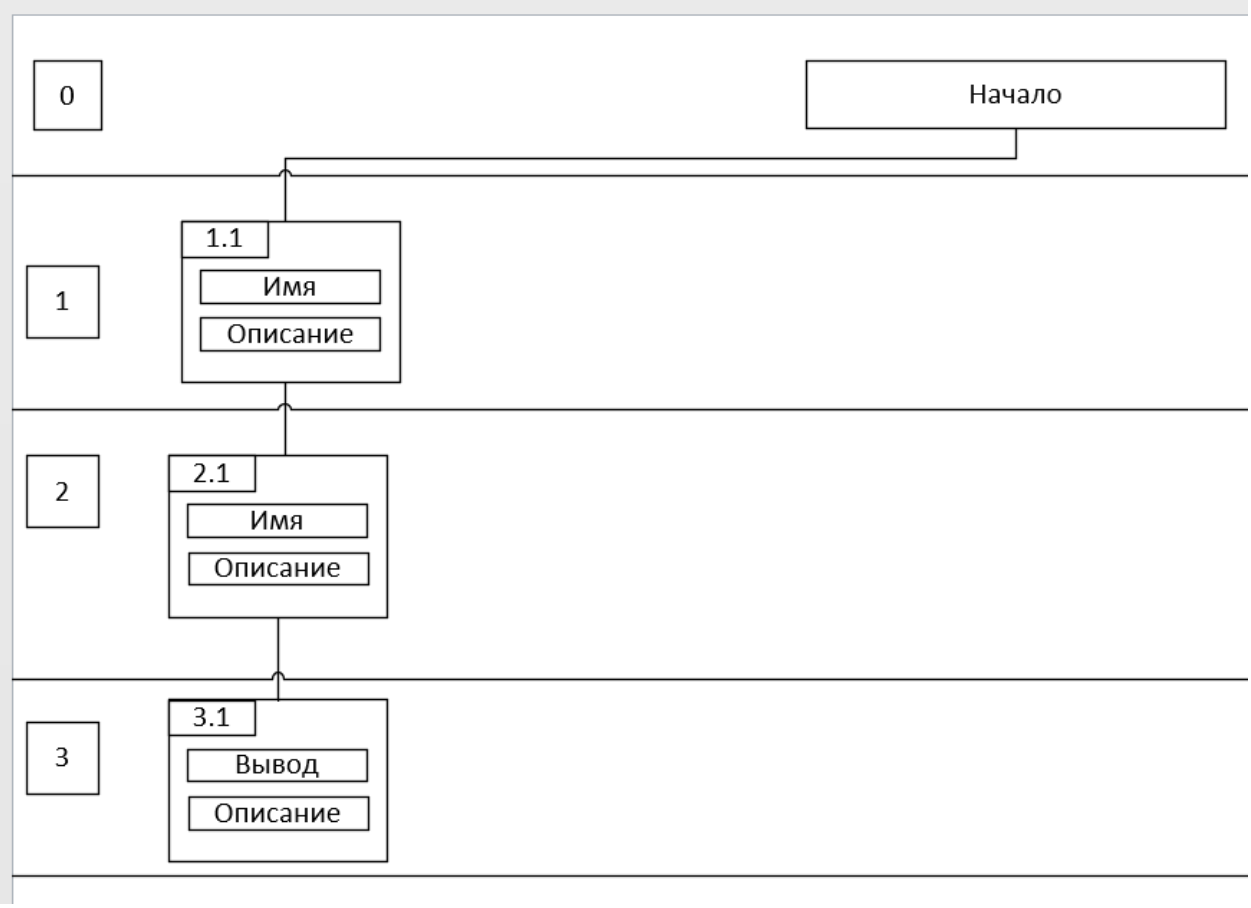
При хватании ноды создаётся фантомный вид ноды (штриховой как показано на рисунке). При отпускании мыши происходит перемещение ноды на другую вертикальную группу.

Пример переноса ноды с изменением уровня:

Так же как описано выше

При хватании ноды создаётся фантомный вид ноды (штриховой как показано на рисунке). При отпускании мыши происходит перемещение ноды на

другую вертикальную группу и меняется уровень. На старом месте нода исчезает.



Практический пример добавления новой ноды на уровне 2 к группе нод привязанных к родителю с текущим номером 1.1

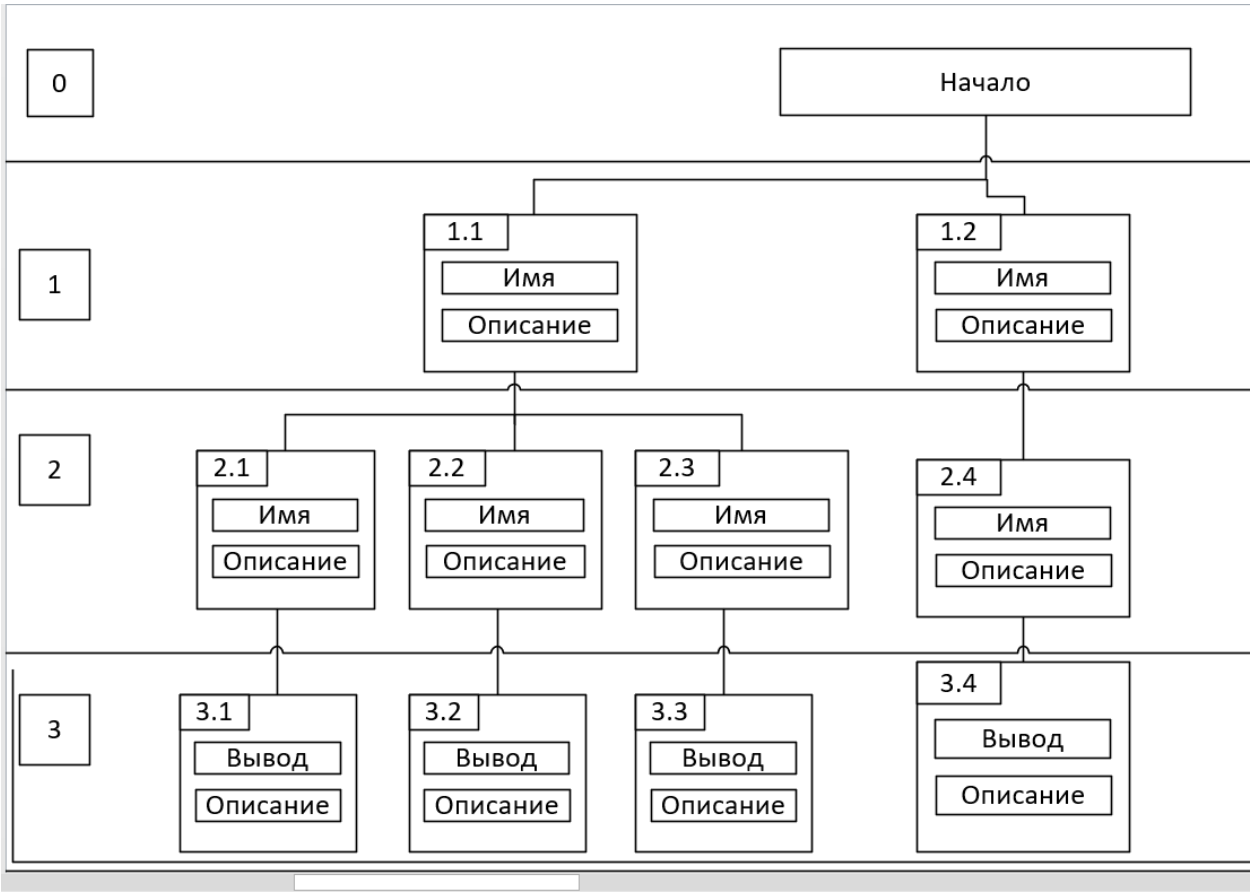


Рисунок 1 -

Далее кликаем по ноде 2.3 правой кнопкой и открывается меню:

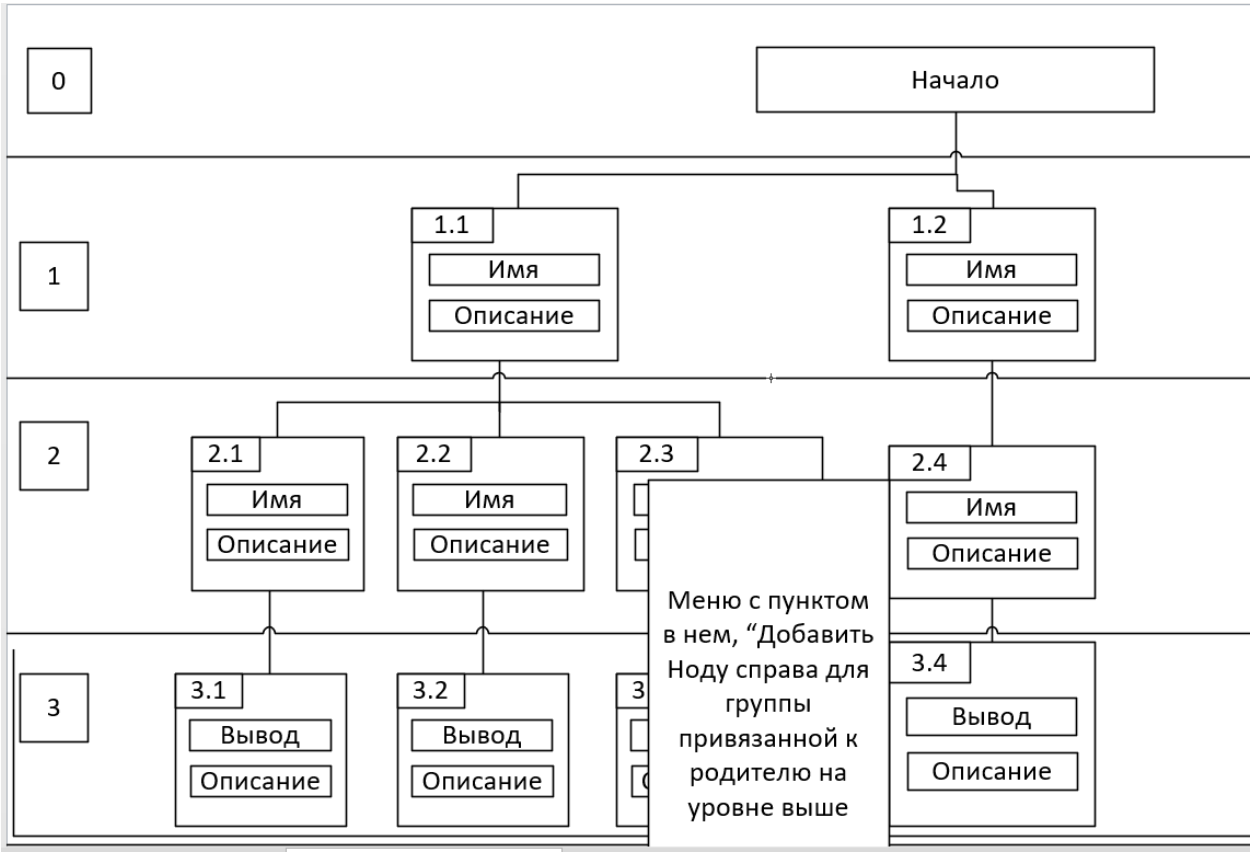


Рисунок 2 – открытие меню по правой кнопке

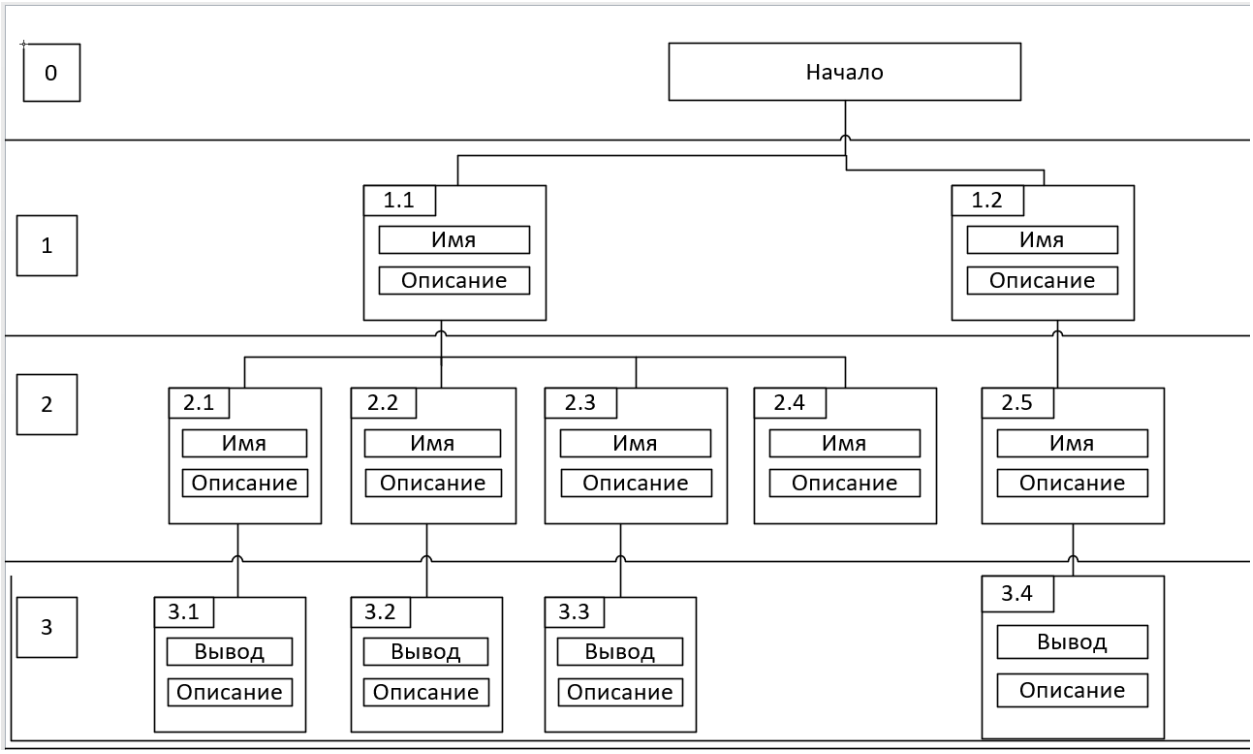


Рисунок 3 – добавленная нода и изменённое положение нод справа с измененной нумерацией так же

Технические пожелания по архитектуре программы:

Архитектура программы должна быть построена по архитектуре mvc.

Архитектура должна быть построена для удобного расширения функционала программы и доступа к данным нод для дальнейшей обработки. Что бы можно было с помощью обращения к классам, структурам обходить данные по дереву итераторами например, работать с данными из кода.

Класс GraphModel

В классе так же переменная с максимальным уровнем на данный момент (максимальный уровень 255).

В классе находится список со всеми нодами, связь нод между собой.

В GraphModel должны присутствовать сигналы,что бы GraphScene (Вид по терминологии mvc),
Был подписан на эти сигналы. При изменении

Например
```
void nodeRemoved(const QUuid& id);
void nodeAdded(const QUuid& newID);
```

Класс Node

Уникальный id ноды,внетренний идентификатор
```
QUuid _uid;
// Текущий уровень, используется только для отображения информации?
uint8_t level;
// порядок слева направо с 1 до ?, используется только для отображения информации?
//Изменяется при например добавлении новой ноды на уровне.
//uint16_t _older_lr;
std::string _title;
std::string _description;
```

GraphEditorView (Наследник от QGraphicsView)

Занимается отрисовкой, маштабированием, обработка перетаскивания)

GraphEditorScene