Пермский национальный исследовательский политехнический университет ПНИПУ

Отчёт по лабораторной работе на тему “Бинарные деревья”

Выполнил студент группы РИС-23-3Б:

Смирнов Андрей Сергеевич

Проверила:

Доцент кафедры ИТАС

О.А. Полякова

2024

1. Постановка задачи:

Необходимо реализовать структуру бинарного дерева с методами удаления, добавки, вывода обходов дерева (Прямой, центрированный, обратный) Выполнить визуализацию в фреймворке Qt.

1. Анализ задачи:

Определение структуры узла: каждый узел бинарного дерева содержит данные (или ключ) и ссылки на два дочерних узла – левый и правый.

Вставка элементов: новые элементы добавляются в дерево путем последовательного сравнения их значений со значениями уже существующих узлов. Если значение нового элемента меньше, чем значение текущего узла, он помещается в левое поддерево, если больше – в правое. Если значение нового элемента равно значению текущего узла, то элемент не добавляется в дерево.

Поиск элементов: поиск элемента в бинарном дереве осуществляется путем последовательного сравнения значений искомого элемента со значениями узлов дерева. Начинается поиск с корня дерева и продолжается либо влево, если значение искомого элемента меньше значения текущего узла, либо вправо, если значение искомого элемента больше значения текущего узла.

Удаление элементов: удаление элемента из бинарного дерева также происходит путем последовательного обхода узлов дерева. Сначала находится узел, который нужно удалить, затем проверяется, есть ли у этого узла потомки. Если потомков нет, то узел просто удаляется. Если же потомки есть, то необходимо найти узел, который заменит удаляемый узел. Это может быть последний узел в поддереве, который будет перемещен вверх, или другой узел, который будет перемещен вниз.

1. UML-диаграмма:

(След. страница)

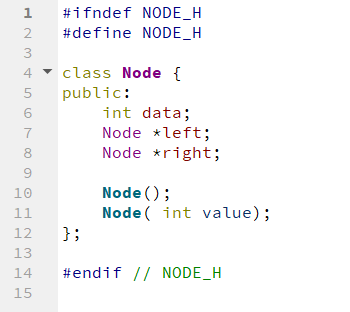


1. Код:

(След. Страница)

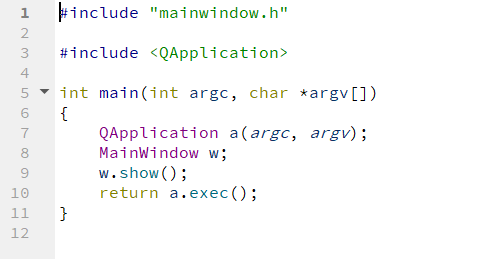
Файл Mainwindow.h

Файл Node.h



Файл Tree.h

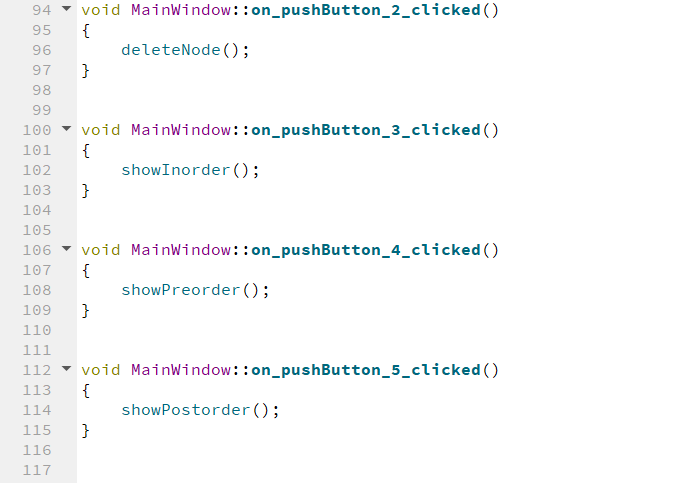
Файл main.cpp



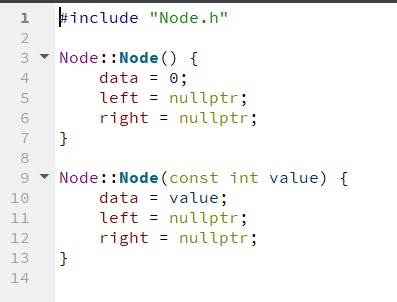
Файл mainwindow.cpp





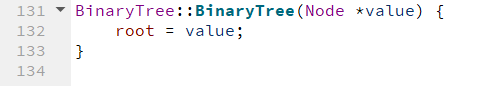


Файл Node.cpp

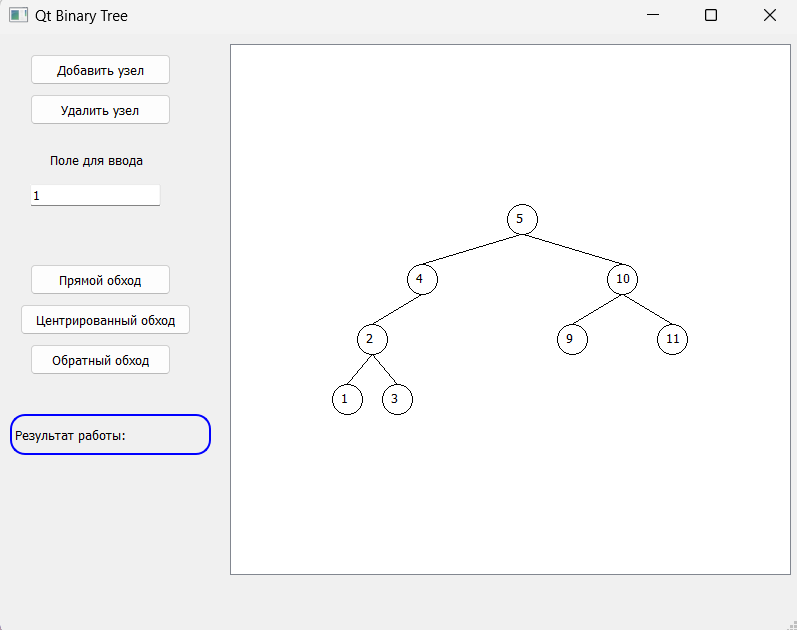


Файл Tree.cpp

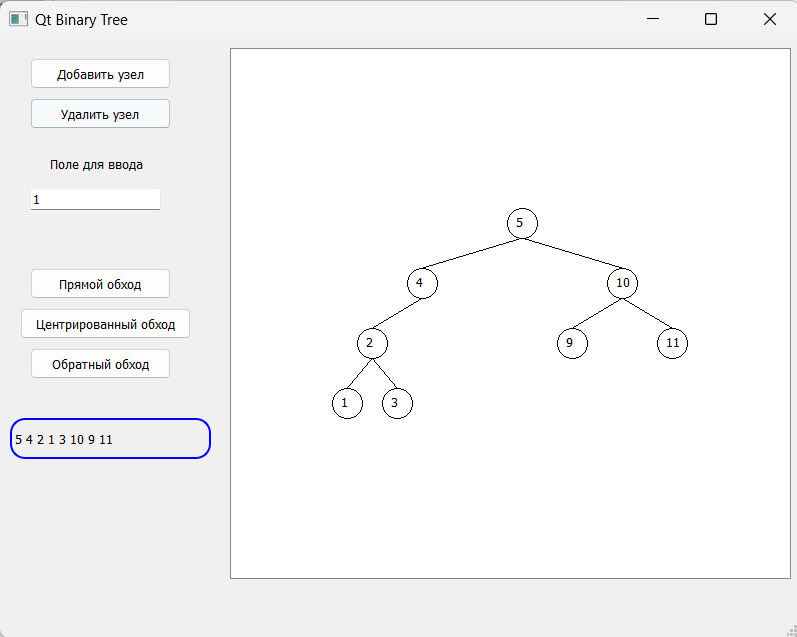




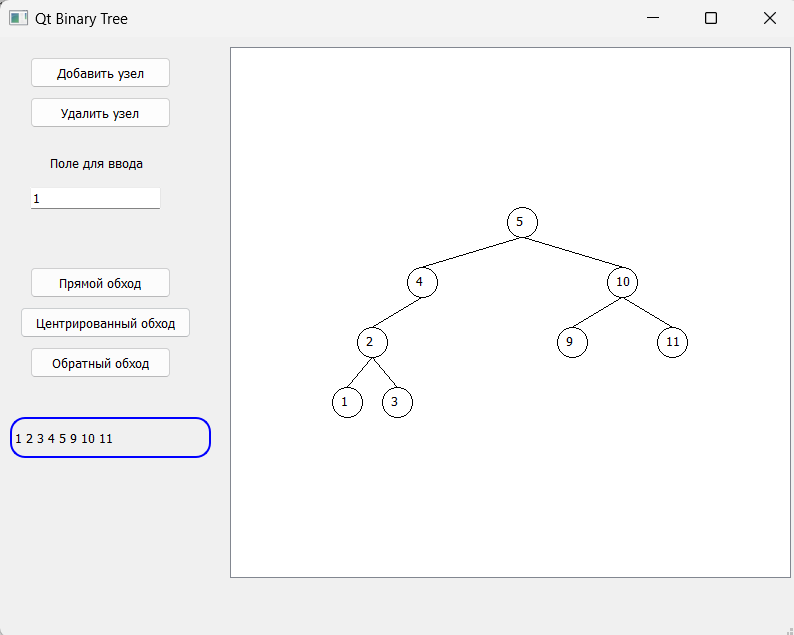
1. Результат работы:



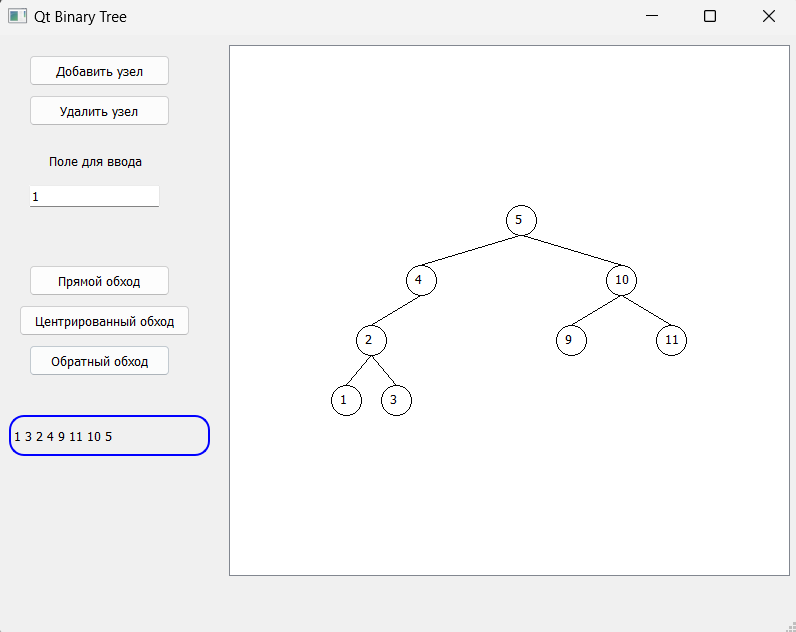
Прямой обход:



Центрированный обход:



Обратный обход:



1. Вывод:

Бинарные деревья позволяют быстро находить элементы, так как каждый узел содержит ссылку на два поддерева (левое и правое). Это особенно полезно при работе с большими объемами данных. Они могут использоваться для сортировки данных. Например, дерево поиска может быть использовано для упорядочивания элементов по определенному критерию. Бинарные деревья могут использоваться для хранения пар ключ-значение, где ключ может быть использован для быстрого доступа к соответствующему значению.

1. GitHub:

<https://github.com/Andr0medA007/Labs/tree/main/Tree>