МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Алгоритмы и структуры данных»

Тема: Деревья

Студент гр. 9384 _____ Соседков К.С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Получить практический навык написания структуры данных бинарное дерево.

Задание. (Вариант № 9)

Рассматриваются бинарные деревья с элементами типа char. Заданы перечисления узлов некоторого дерева b в порядке КЛП и ЛКП. Требуется:

а) восстановить дерево b и вывести его изображение;

б) перечислить узлы дерева b в порядке ЛПК.

Выполнение работы.

Для выполнения работы был разработан шаблонный класс *BinaryTree* со следующими методами:

setData — устанавливает значение текущего узла дерева

getData — возвращает значение узла дерева

height — вычисляет высоту дерева

print — выводит дерево на консоль

При написании программы использовалась библиотека *Qt*. С ее помощью были разработаны 3 класса:

InputWidget — класс для ввода дерева и проверки ввода на корректность. Для ввода используется форма *QlineEdit*, для подтверждения ввода *QPushButton*. Проверка ввода происходит в функции *isValid*. Если дерево введено правильно оно передается в одну из двух функций *readTreeRLR*(считывание дерева в порядке КЛП) или *readTreeLRR*(порядок ЛКП). Выбор этих функций зависит от нажатия на конкретную кнопку *QradioButton*. После считывания дерево передается в классы *TreeViewWidget* и *ListWidget*.

ListWidget — отображает в виде вертикального списка узлы дерева в порядке ЛПК. Для отображения используется *Qlabel*.

TreeViewWidget — рисует дерево на экране используя *QgraphicsView* и *QgraphicsScene*.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.		Invalid tree	Ошибка Ввод пустой строки
2.	(a (b) (c (d) (e)))	Valid Tree	Порядок КЛП
3.	(((q) a (y)) b ((z)c(o)))	Vaild Tree	Порядок ЛКП
4.	((w)	Invalid tree	Лишняя скобка на первой позиции
5.	(a (b) (cd))	Invalid tree	Лишняя буква

Выводы.

В ходе выполнения лабораторной работы была изучена и реализована динамическая структура данных — бинарное дерево, а так же были разработаны методы для работы с ней.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: binarytree.h

```
#ifndef BINARYTREE_H
#define BINARYTREE_H

#include <QString>
#include <QDebug>

template <class T>
class BinaryTree
{
public:
    BinaryTree() {}

    BinaryTree(T data) {
        this->data = data;
    }

    ~BinaryTree() {
        if(left) delete left;
        if(right) delete right;
    }

    void setData(T data) {
        this->data = data;
    }

    T getData() {
        return this->data;
    }

    int height() {
        int left_h = 0;
        int right_h = 0;
        if(this->left) left_h = left->height();
        if(this->right) right_h = right->height();
        return left_h>right_h? left_h+1 : right_h+1;
    }
```

```
    void print(int offset=0) {
        qDebug() << QString(' ').repeated(offset) << (this->data);
        if(this->left)
            this->left->print(offset+2);
        if(this->right)
            this->right->print(offset+2);
    }

    BinaryTree* left = nullptr;
    BinaryTree* right = nullptr;

private:
    T data;

};

#endif // BINARYTREE_H
```

Название файла: InputWidget.cpp

```
#include "inputwidget.h"

InputWidget::InputWidget(QWidget *parent) : QWidget(parent) {
    QLabel* rootLeftRightLabel = new QLabel("Root-Left-Right (a(Tree)(Tree))");
    QLabel* leftRootRightLabel = new QLabel("Left-Root-Right ((Tree)a(Tree))");
    QRadioButton* rootLeftRightBtn = new QRadioButton();
    QRadioButton* leftRootRightBtn = new QRadioButton();
    input = new QLineEdit();
    QPushButton* accept = new QPushButton("Accept");

    rootLeftRightBtn->setChecked(true);
     input->setPlaceholderText("Root-Left-Right: ( a (b) (c) )        Left-Root-Right: ( (b) a
(c) )");

    main_layout = new QVBoxLayout;
    labels = new QHBoxLayout;
    radio_buttons = new QHBoxLayout;
    input_form = new QHBoxLayout;
```

```
    labels->addWidget(rootLeftRightLabel);
    labels->addWidget(leftRootRightLabel);

    labels->itemAt(0)->setAlignment(Qt::AlignCenter);
    labels->itemAt(1)->setAlignment(Qt::AlignCenter);

    radio_buttons->addWidget(rootLeftRightBtn);
    radio_buttons->addWidget(leftRootRightBtn);

    radio_buttons->itemAt(0)->setAlignment(Qt::AlignCenter);
    radio_buttons->itemAt(1)->setAlignment(Qt::AlignCenter);

    input_form->addWidget(input);
    input_form->addWidget(accept);

    main_layout->addLayout(labels);
    main_layout->addLayout(radio_buttons);
    main_layout->addLayout(input_form);
    main_layout->setAlignment(Qt::AlignTop);

    this->setLayout(main_layout);

    connect(accept, &QPushButton::pressed, this, &InputWidget::isValid);
    connect(rootLeftRightBtn, &QRadioButton::clicked,  [=](){
        this->type = RootLeftRight;
    });
    connect(leftRootRightBtn, &QRadioButton::clicked,  [=](){
        this->type  = LeftRootRight;
    });
}

InputWidget::~InputWidget() {
    QLayoutItem * item;
    QWidget * widget;
    while ((item = main_layout->takeAt(0))) {
        if ((widget = item->widget()) != 0) {widget->hide(); delete widget;}
        else {delete item;}
    }
    delete main_layout;
}
```

```
void InputWidget::isValid() {
    QString text = input->text();
    text.replace(" ", "");
    int index = 0;
    bool ok = (type == RootLeftRight)?
              readTreeRLR(text, tree, index) :
              readTreeLRR(text, tree, index);

    if(ok && index == text.size()) {
        qDebug() << "Valid tree";
        tree->print();
        qDebug() << tree->height();
        emit treeIsValid(tree, tree->height());
    }
    else qDebug() << "Invalid tree";

    if(tree) {
        delete tree;
        tree = nullptr;
    }
}



//KLP
bool InputWidget::readTreeRLR(QString text, BinaryTree<char>*& node, int& index) {
    if(isList(text, index)) {
        if(isRoot(text, index))
            node = new BinaryTree<char>(text.toStdString()[index-1]);
        else if(isEmptyList(text, index))
            return true;
        else
            return false;
```

```
        if(isEmptyList(text,index))
            return true;

        return readTreeRLR(text, node->left, index) &&
               readTreeRLR(text, node->right, index) &&
               isEmptyList(text,index);

    }
    else return false;
}



//LKP
bool InputWidget::readTreeLRR(QString text, BinaryTree<char>*& node, int& index) {
    if(isList(text,index)) {
        if(isEmptyList(text, index)) //EMPTY LIST
            return true;

        node = new BinaryTree<char>();
        if(!readTreeLRR(text, node->left, index)) //ISLIST
            return false;

        if(isRoot(text, index)) // IS ROOT
            node->setData(text.toStdString()[index-1]);

        if(isEmptyList(text, index))
            return true;

        return readTreeLRR(text, node->right, index) && isEmptyList(text,index);

    }
    else if(text[index].isLetterOrNumber())
        return true;

    else return false;
}



bool InputWidget::isList(QString text, int& index) {
```

```
    if(text[index] == '(') {
        index++;
        return true;
    }
    return false;
}

bool InputWidget::isRoot(QString text, int& index) {
    if(index == 0) return false;
    if(text[index].isLetterOrNumber()) {
        index++;
        return true;
    }
    return false;
}

bool InputWidget::isEmptyList(QString text, int& index) {
    if(text[index] == ')') {
        if(index==1) return false;
        index++;
        return true;
    }
    return false;
}
```

Название файла: TreeViewWidget.cpp

```
#include "treeviewwidget.h"
#include <QDebug>
TreeViewWidget::TreeViewWidget(QWidget *parent) : QWidget(parent) {
    view = new QGraphicsView(this);
    scene = new QGraphicsScene(this);
    view->setMinimumSize(this->width(), this->height());
    view->setScene(scene);
    view->setAlignment(Qt::AlignTop);
}

TreeViewWidget::~TreeViewWidget() {
    delete view;
    delete scene;
```

```
}

void TreeViewWidget::drawTree(BinaryTree<char>* tree, int power, int offset_x, int offset_y) {
    if(offset_y == 0) scene->clear();
    QGraphicsTextItem * io = new QGraphicsTextItem();
    io->setPos(view->width()/2+offset_x,offset_y*20);

    io->setPlainText(QString(tree->getData()));
    scene->addItem(io);

    int pow = qPow(2, power)*1.5;

    if(tree->left) {
        drawTree(tree->left, power-1, offset_x-pow, offset_y+1);
    }

    if(tree->right) {
        drawTree(tree->right, power-1, offset_x+pow, offset_y+1);
    }

}
```

Название файла: MainWidget.cpp

```
#include "mainwidget.h"
#include <QtMath>
#include "listwidget.h"


MainWidget::MainWidget(QWidget *parent) : QWidget(parent) {
    this->resize(800,600);

    view = new TreeViewWidget();
    input = new InputWidget();
    list = new ListWidget();
    main_layout = new QVBoxLayout();
    hbox = new QHBoxLayout();

    hbox->addWidget(view,3);
    hbox->addWidget(list,1);
```

```
    main_layout->addWidget(input);
    main_layout->addLayout(hbox, Qt::AlignTop);

    this->setLayout(main_layout);

    connect(input, &InputWidget::treeIsValid, view, [=](BinaryTree<char>* tree, int power){
        view->drawTree(tree, power);
        list->drawTreeElements(tree);
    });
}

MainWidget::~MainWidget() {
    delete view;
    delete input;
    delete list;
    delete hbox;
    delete main_layout;
}
```