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

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Алгоритмы и структуры данных»

Тема: Деревья

Студентка гр. 8381 _____ Ивлева О.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2019

Цель работы.

Ознакомиться с основными характеристиками и особенностями такой структуры данных, как бинарное дерево, изучить особенности ее реализации на языке программирования C++.

Задание.

Формулу вида

< формула > ::= < терминал > | (< формула > < знак > < формула >)

< знак > ::= + | – | *

< терминал > ::= 0 | 1 | ... | 9 | *a* | *b* | ... | *z*

можно представить в виде бинарного дерева («***дерева-формулы***») с элементами типа *Elem=char* согласно следующим правилам:

- формула из одного терминала представляется деревом из одной вершины с этим терминалом;

- формула вида (f_1 *s* f_2) представляется деревом, в котором корень – это знак *s*, а левое и правое поддеревья – соответствующие представления формул f_1 и f_2.

Требуется:

- если в дереве-формуле *t* терминалами являются только цифры, то вычислить (как целое число) значение дерева-формулы *t*;

- построить дерево-формулу *t*1 – производную дерева-формулы *t* по заданной переменной.

Основные теоретические положения.

Арифметическое выражение с бинарными операциями можно представить в виде бинарного дерева. Пусть, например, дано арифметическое выражение в инфиксной записи: (*a* + *b*) * *c* – *d* / (*e* + *f* * *g*). На рис. 1 представлено соответствующее ему бинарное дерево.

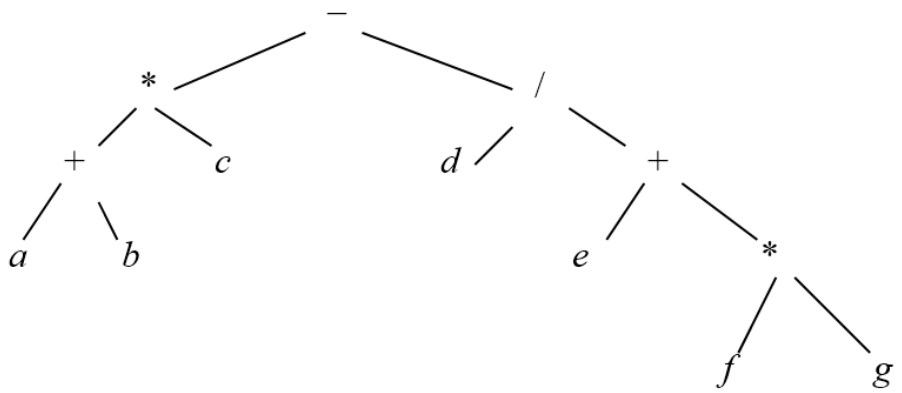

Рисунок 1 - Бинарное дерево, представляющее выражение

Выполнение работы.

Написание работы производилось на базе операционной системы Windows 10 в среде разработки QtCreator с использованием фреймворка Qt.

Для реализации программы был разработан графический интерфейс с помощью встроенного в QtCreator UI-редактора. Была добавлена кнопка, при нажатии на которую, происходит преобразование исходного выражения в бинарное дерево.

Для реализации бинарного дерева была создана структура tree, в которой хранится очередной символ(char name), указатель на предыдущий элемент(tree *parent), и указатель на правый и левый элементы(tree* childLeft and tree* childRight), а так же вычисленное значение в данный момент(int var).

Также были реализованы функции, создающие и изменяющие бинарное дерево.

Функция share(), которая принимает на вход заданную строку и ссылку на две строки, делит строку по точке “перегиба” на две подстроки и возвращает знак, по которому произошло деление.

Функция maketree(), которая получает на вход структуру tree, флаг, является ли данная структура первым элементом и входную строку, создает бинарное дерево. Она запускается рекурсивно для правого и левого поддерева и

заканчивает работу, когда strLeft и strRight становятся одним символом после разбиения на подстрочки.

Функция calc(), которая получает указатель на структуру, считает значение дерева, если все терминалы были равны числам.

Программа имеет возможность графического отображения полученного бинарного дерева с помощью виджета QGraphicsView.

Оценка сложности алгоритма

Операция разделения массива на две части относительно опорного элемента занимает время O($\log_2 n$). Поскольку все операции разделения, проделываемые на одной глубине рекурсии, обрабатывают разные части исходного массива, размер которого постоянен, суммарно на каждом уровне рекурсии потребуется также O(n) операций. В итоге получим в каждой строке по одному элементу. Асимптотика: O($n \log_2 n$)

Тестирование программы.

Вид программы после запуска представлен на рис. 1.

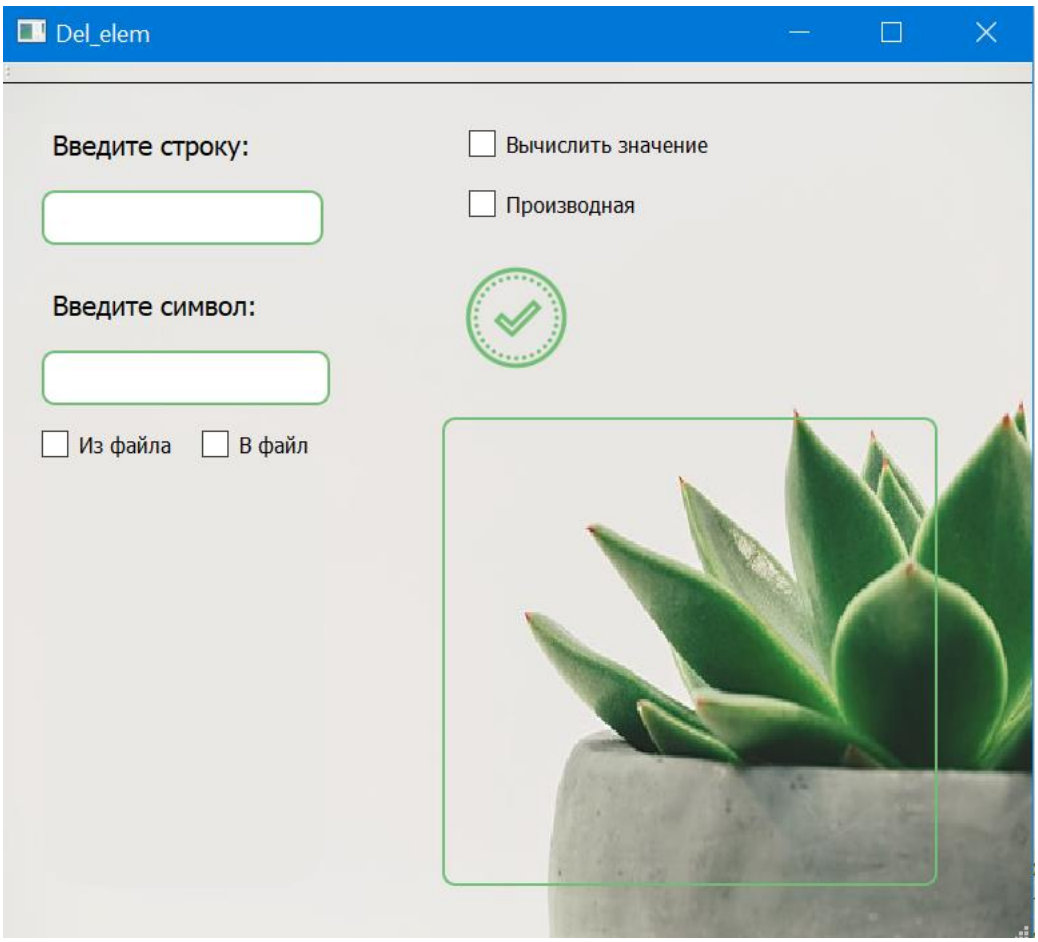

Рисунок 1 – Графический интерфейс программы

Работа программы представлена на рис. 2.

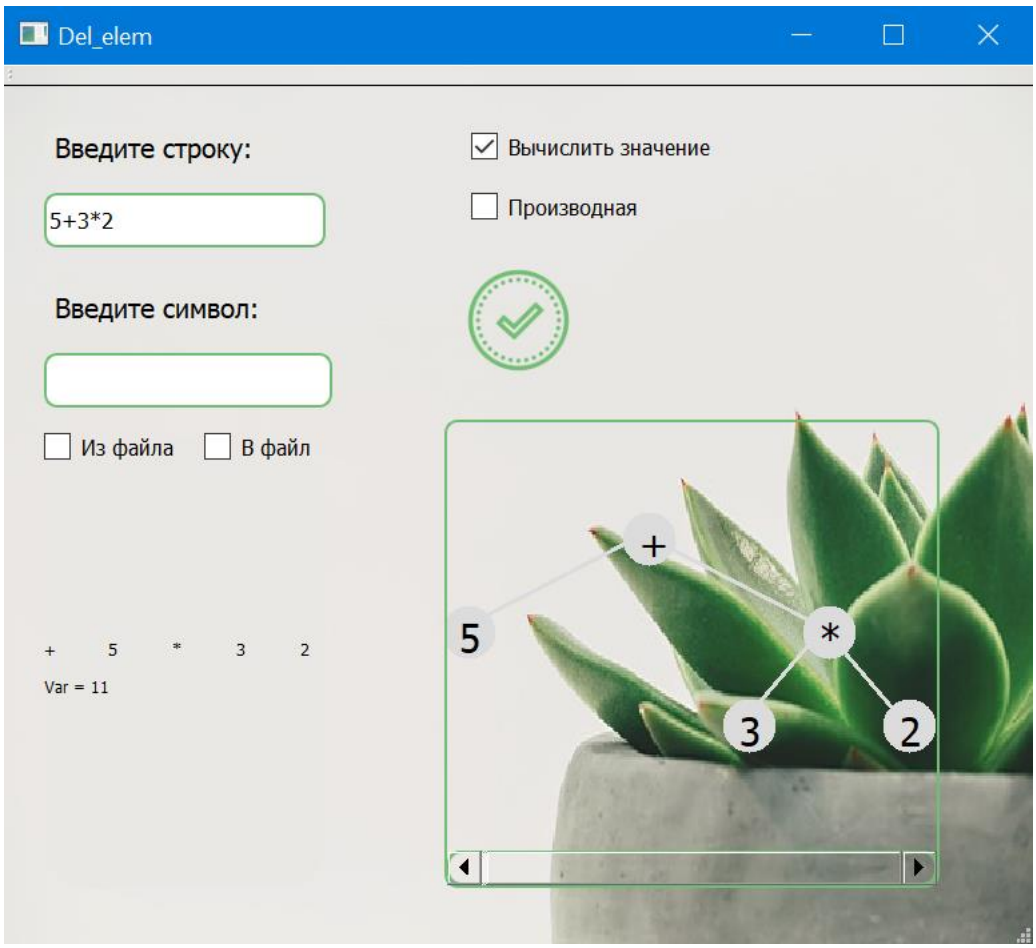

Рисунок 2 –Работа программы

Был проведен ряд тестов, проверяющих корректность работы программы. Результаты тестирование приведены в табл. 1.

Таблица 1 – Тестирование программы

Входная строка	Вывод (в инфиксной записи)
5+3	+ 5 3 var = 8
(a+b)*c	*+a b c
(2+3)*5-2	-*+2 3 5 2 var = 23
(a+b)*c-d/(e+f*g)	-*+a b c / d + e * f g
(2+3)*5-8/(1+1*1)	-*+2 3 5 / 8 + 1 * 1 1 var = 21

Выводы.

В ходе выполнения лабораторной работы была написана программа, создающая бинарное дерево согласно заданному выражению и считающая его значение, а также находящая производную выражения.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();

    return a.exec();
}
```

Название файла: mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include "functions.h"

namespace Ui {
class MainWindow;
}

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

private slots:
    void on_Hello_clicked();

private:
    Ui::MainWindow *ui;
};

#endif // MAINWINDOW_H
```

Название файла: functions.h

```
#include <iostream>
#include <string>
#include <cctype>
```

```
using namespace std;

struct tree;
char share(string str, string &str1, string &str2);
tree* maketree(string str, int flag, tree *s);
void printTree(tree *symb, string &output);
void calc(tree *symb);
void derivative(string &str, string symbol);

struct tree{
    char name;
    int var = 0;
    tree *parent;
    tree *childLeft;
    tree *childRight;
};
```

Название файла: functions.cpp

```
#include "functions.h"

char share(string str, string &str1, string &str2){
    int now = 1;
    int ind_now = 0;
    int flag = 0;
    for(unsigned int i = 0; i < str.length(); i++) {
        if (str[i] == '(')
            flag = 1;
        if (str[i] == ')')
            flag = 0;
        if (flag == 0) {
            if (str[i] == '-' || str[i] == '+') {
                now = 2;
                ind_now = i;
            }
            if (str[i] == '*' || str[i] == '/') {
                if (now < 2)
                    ind_now = i;
            }
        }
    }
    for(unsigned int i = 0; i < str.length(); i++) {
        if (i < ind_now)
            str1 += str[i];
        if (i > ind_now)
            str2 += str[i];
    }
    return str[ind_now];
```

```
}

tree* maketree(string str, int flag, tree *s){
    tree *symb = new tree;
    string str1 = "";
    string str2 =  "";
    if (str[1] != '\0') {
        symb->name = share(str, str1, str2);
        if (str1[0] == '(' && str1[str1.length()-1] == ')'){
            for(int i = 1; i < str1.length()-1; i++)
                str1[i-1] = str1[i];
            //str1.pop_back();
            //str1.pop_back();
            str1[str1.length()-2] = '\0';
        }
        if (str2[0] == '(' && str2[str2.length()-1] == ')'){
            for(int i = 1; i < str2.length()-1; i++)
                str2[i-1] = str2[i];
            //str2.pop_back();
            //str2.pop_back();
            str2[str2.length()-2] = '\0';
        }
    }
    else {
        symb->name = str[0];
        symb->childLeft = nullptr;
        symb->childRight = nullptr;
        symb->parent = s;
        return symb;
    }
    if (flag == 1)
        symb->parent = s;
    symb->childLeft = maketree(str1, 1, symb);
    symb->childRight = maketree(str2, 1, symb);
    return symb;
}

void printTree(tree *symb, string &output){
    output += symb->name;
    output +="\t";
    if (symb->childLeft == nullptr)
        return;
    printTree(symb->childLeft, output);
    printTree(symb->childRight, output);
}

void calc(tree *symb){
    if (symb->childLeft == nullptr){
        symb->var = int(symb->name) - 48;
```

```
        return;
    }
    calc(symb->childLeft);
    calc(symb->childRight);
    if (symb->name == '+')
        symb->var          =          int(symb->childLeft->var)          +
int(symb->childRight->var);
    if (symb->name == '-')
        symb->var          =          int(symb->childLeft->var)          -
int(symb->childRight->var);
    if (symb->name == '*')
        symb->var          =          int(symb->childLeft->var)          *
int(symb->childRight->var);
    if (symb->name == '/')
        symb->var          =          int(symb->childLeft->var)          /
int(symb->childRight->var);
}

void derivative(string &str, string symbol){
    for(int i = 0; i < str.length(); i++){
        if(str[i] == symbol[0])
            str[i] = 1;
    }
}
```

Название файла: mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "printtree.h"

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->output->setWordWrap(true);
    QPixmap bkgnd("D:\\prog\\Styles\\plant-2004483_640.jpg");
    bkgnd = bkgnd.scaled(this->size(), Qt::IgnoreAspectRatio);
    QPalette palette;
    palette.setBrush(QPalette::Background, bkgnd);
    this->setPalette(palette);
    //ui->graphicsView->hide();
    scene = new QGraphicsScene;
    ui->graphicsView->setScene(scene);
}

MainWindow::~MainWindow()
{
```

```
    delete ui;
}

void MainWindow::on_Hello_clicked()
{
    string str;
    string symbol;
    if (!ui->checkBox_2->isChecked()){
        str = qPrintable(ui->input->text());
    }
    else {
        ifstream fin; // создаем объект класса ifstream (считать)
        fin.open("D:\\prog\\cpp\\lab1\\text.txt"); // открываем файл для
считывания
        fin >> str;
        fin.close(); // закрываем файл
    }
    tree *symb = new tree;
    if (ui->checkBox->isChecked()){
        symbol = qPrintable(ui->input_char->text());
        derivative(str, symbol);
    }
    symb = maketree(str, 0, nullptr);
    graphic(symb, scene);
    string output = "";
    printTree(symb, output);
    if (ui->checkBox_5->isChecked()){
        calc(symb);
        output += "\n";
        output += "Var = ";
        output += to_string(symb->var);
    }
    if (ui->checkBox_3->isChecked()){
        ofstream fin2; //дозаписать
        fin2.open("D:\\prog\\cpp\\lab1\\text.txt",     ios::app);     //
открываем файл и дозыписываем в него
        fin2 << endl;
        fin2 << output;
        fin2.close(); // закрываем файл
    }
    ui->output->setText(QString::fromStdString(output));
    //graphic(symb, scene);
}
```

Название файла: mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
```

```
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>600</width>
    <height>600</height>
   </rect>
  </property>
  <property name="focusPolicy">
   <enum>Qt::NoFocus</enum>
  </property>
  <property name="windowTitle">
   <string>Del_elem</string>
  </property>
  <property name="autoFillBackground">
   <bool>false</bool>
  </property>
  <property name="styleSheet">
   <string notr="true"/>
  </property>
  <widget class="QWidget" name="centralWidget">
   <widget class="QPushButton" name="Hello">
    <property name="geometry">
     <rect>
      <x>320</x>
      <y>100</y>
      <width>91</width>
      <height>91</height>
     </rect>
    </property>
    <property name="cursor">
     <cursorShape>PointingHandCursor</cursorShape>
    </property>
    <property name="styleSheet">
     <string                                        notr="true">border-image:
url(:/C:/Users/Олеся/Downloads/icons8-ok-100_1.png);</string>
    </property>
    <property name="text">
     <string/>
    </property>
    <property name="autoRepeat">
     <bool>false</bool>
    </property>
   </widget>
   <widget class="QLabel" name="output">
    <property name="geometry">
     <rect>
      <x>30</x>
      <y>250</y>
```

```
      <width>201</width>
      <height>301</height>
     </rect>
    </property>
    <property name="styleSheet">
     <string notr="true">font: 6pt &quot;MS Shell Dlg 2&quot;;
border-radius: 30%;
background-color: #e9e8e4;
</string>
    </property>
    <property name="text">
     <string/>
    </property>
   </widget>
   <widget class="QLineEdit" name="input">
    <property name="geometry">
     <rect>
      <x>30</x>
      <y>80</y>
      <width>211</width>
      <height>41</height>
     </rect>
    </property>
    <property name="styleSheet">
     <string notr="true">border-radius: 10%;
border: 2px solid #72be74;</string>
    </property>
   </widget>
   <widget class="QLineEdit" name="inputChar">
    <property name="geometry">
     <rect>
      <x>30</x>
      <y>190</y>
      <width>211</width>
      <height>41</height>
     </rect>
    </property>
    <property name="styleSheet">
     <string notr="true">border-radius: 10%;
border: 2px solid #72be74;</string>
    </property>
   </widget>
   <widget class="QCheckBox" name="checkBox">
    <property name="geometry">
     <rect>
      <x>320</x>
      <y>40</y>
      <width>191</width>
      <height>21</height>
```

```
     </rect>
    </property>
    <property name="cursor">
     <cursorShape>PointingHandCursor</cursorShape>
    </property>
    <property name="text">
     <string>Доп. Информация</string>
    </property>
   </widget>
   <widget class="QLineEdit" name="lineEdit">
    <property name="geometry">
     <rect>
      <x>30</x>
      <y>30</y>
      <width>181</width>
      <height>31</height>
     </rect>
    </property>
    <property name="styleSheet">
     <string notr="true">
font: 10pt &amp;quot;MS Shell Dlg 2&amp;quot;;
border-radius: 10%;
background-color: #e9e8e4;
</string>
    </property>
    <property name="text">
     <string> Введите строку:</string>
    </property>
   </widget>
   <widget class="QLineEdit" name="lineEdit_2">
    <property name="geometry">
     <rect>
      <x>30</x>
      <y>140</y>
      <width>181</width>
      <height>31</height>
     </rect>
    </property>
    <property name="styleSheet">
     <string notr="true">
font: 10pt &amp;quot;MS Shell Dlg 2&amp;quot;;
border-radius: 10%;
background-color: #e9e8e4;</string>
    </property>
    <property name="text">
     <string> Введите символ:</string>
    </property>
   </widget>
  </widget>
```

```
  <widget class="QMenuBar" name="menuBar">
   <property name="geometry">
    <rect>
     <x>0</x>
     <y>0</y>
     <width>600</width>
     <height>17</height>
    </rect>
   </property>
  </widget>
  <widget class="QToolBar" name="mainToolBar">
   <attribute name="toolBarArea">
    <enum>TopToolBarArea</enum>
   </attribute>
   <attribute name="toolBarBreak">
    <bool>false</bool>
   </attribute>
  </widget>
  <widget class="QStatusBar" name="statusBar"/>
 </widget>
 <layoutdefault spacing="6" margin="11"/>
 <resources/>
 <connections/>
</ui>
```