МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)

Факультет Инфокоммуникационных сетей и систем
Кафедра Защищенных систем связи
Дисциплина: «Методы и технологии программирования»

Отчет по лабораторной работе №7:

«Работа с графиками»

Выполнил студент гр. ИБС-01:

Гребенников Тимофей

Проверил:

Ерофеев Сергей Анатольевич

доцент кафедры ПИВТ, кандидат физико-математических наук

Санкт-Петербург

Оглавление

Постановка задачи

Требуется разработать программу с графическим интерфейсом, строящую график заданной функции.

Программа должна включать в себя двумерную систему координат, на которой осуществляется построение графиков функций по заданным параметрам: формула функции с одним аргументом («x»), диапазон построения.

Проект реализован на языке C++ в среде разработки QT Creator (Community) версии 7.0.0.

Техническое задание

- **Список входных данных:**

Название переменной	Назначение	Тип данных
start	левая граница диапазона построения функции	double
stop	правая граница диапазона построения функции	double

- **Список выходных данных:**

Название переменной	Назначение	Тип данных
args	список всех аргументов функции	QVector <double>
vals	список всех значений функции	QVector <double>

- **Список промежуточных данных:**

Название переменной	Назначение	Тип данных
step	шаг смещения аргумента функции	double
warning_empty, warning_error, warning_syntax, warning_logic	информация об ошибках пользователя, допущенных при вводе	QString

	свойств добавляемого в базу данных объекта	
xPosition	текущее значение аргумента функции	double
xPositionText	текущее значение аргумента функции в строковом формате	QString
result	введённое пользователем выражение функции с заменёнными на численное значение буквами «x»	QString
expression	данные переменной result в динамическом формате	const char*
finalValue	текущее значение функции	double
error	переменная, хранящая информацию об ошибке при некорректной интерпретации выражения функции на текущем шаге	int
error_all	переменная, хранящая информацию о наличии некорректных интерпретаций	bool

	выражения функции на всех шагах в диапазоне	
infinity	наличие точки разрыва 2-го рода в окрестности текущей точки	bool
graphNum	нумерация графиков при разбиении в случае обнаружения точки разрыва	int
infinity_missed	итератор для приостановки разбиения графика в случае ошибочного обнаружения точки разрыва	int
infinity_counter	счётчик количества точек разрыва	int
yMin, yMax	минимальное и максимальное значение функции на промежутке построения	double
styleSheetFile	файл, содержащий настройки стиля программы	QFile
styleSheet	хранение информации из styleSheetFile для применения стиля	QString

- **Контроль входных данных:**

Ввод параметров функции осуществляется посредством записи информации в специальные поля, именуемые как *LineEdit*. Для осуществления контроля ввода применяется проверка каждой подобной записи с помощью конструкции условного оператора *If-Else*. В случае нахождения программой ошибки в какой-либо из записей, информация об этой ошибке заносится в одну из переменных warning и выводится на экран в виде всплывающего окна *QMessageBox*.

Ввод математического выражения, задающего график функции, проверяется на этапе расчёта значений функции. В случае возникновения ошибки (например, при вводе пользователем неизвестных интерпретатору выражений), информация о ней будет возвращена функцией интерпретации *te_interp(expression, error)* в выделенную переменную, после чего ошибка будет выведена на экран в виде всплывающего окна.

Реализация контроля ввода:

```
QString warning_empty = "", warning_error = "", warning_syntax = "",
warning_logic = "";

    if (ui->startEdit->text().isEmpty() || ui->stopEdit-
>text().isEmpty() || ui->funcEdit->text().isEmpty())
        warning_empty = " *все поля должны быть заполнены";

    if (ui->startEdit->text().toInt() == false && ui->startEdit-
>text() != "0" && ui->startEdit->text() != "")
        warning_syntax = " *диапазон значений функции должен быть
задан целыми числами";

    if (ui->stopEdit->text().toInt() == false && ui->stopEdit->text()
!= "0" && ui->stopEdit->text() != "")
        warning_syntax = " *диапазон значений функции должен быть
задан целыми числами";

    if (ui->startEdit->text().toInt() >= ui->stopEdit->text().toInt()
&& warning_syntax == "" && !(ui->stopEdit->text().isEmpty() || ui-
>funcEdit->text().isEmpty()))
        warning_logic = " *верхняя граница диапазона значений функции
не должна превосходить нижнюю";
...
```

```
if (!(warning_empty == "" && warning_error == "" && warning_logic ==
"" && warning_syntax == "")) QMessageBox::warning(this, "Неверные
данные",
                    "При заполнении параметров функции были допущены
следующие ошибки: "
                        + warning_empty + warning_error + warning_logic
+ warning_syntax +
                          ". Пожалуйста, исправьте ошибки и повторите
ввод.", QMessageBox::Ok);
```

- **Список функций:**

функция	описание
`void on_drawButton_clicked();`	Функции обработки нажатия кнопки «Нарисовать»
`void on_clearButton_clicked();`	Функции обработки нажатия кнопки «Очистить»
`void on_defaultButton_clicked();`	Функции обработки нажатия кнопки «Центрировать»
`void on_exitButton_clicked();`	Функции обработки нажатия кнопки «Выход»

Структурное описание

При запуске программы открывается окно построения графиков функций.

В центре окна расположен виджет с двумерной координатной плоскостью. С ним можно взаимодействовать при помощи мыши: изменять масштаб колёсиком и перемещаться при зажатой левой кнопке мыши.

Снизу слева находятся поля для ввода математического выражения функции и параметров построения (диапазона). Снизу справа расположен блок кнопок управления программой, состоящий из кнопок «Нарисовать», «Центрировать», «Очистить» и «Выход».

При нажатии на кнопку «Нарисовать» возможны следующие исходы:

- если введённое выражение и параметры функции были интерпретированы успешно, то на виджете произойдёт построение функции;
- если при обработке введённых данных будут обнаружены ошибки, то на экране появится всплывающее окно, информирующее пользователя о наличии ошибок ввода.

При нажатии на кнопку «Центрировать» на координатной плоскости произойдёт смещение к точке пересечения осей, а масштаб плоскости изменится до стандартных значений.

При нажатии на кнопку «Очистить» произойдёт очистка координатной плоскости от предыдущего графика.

При нажатии на кнопку «Выход» произойдёт завершение работы программы.

Листинг программы

1. Заголовочные файлы:

- mainwindow.h:

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QVector>
#include <QString>

#include "tinyexpr.h"
#include "qcustomplot.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

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

private slots:
    void on_drawButton_clicked();

    void on_clearButton_clicked();

    void on_defaultButton_clicked();

    void on_exitButton_clicked();

private:
    Ui::MainWindow *ui;

    double start, stop, step = 0.01, xPosition;
    QVector<double> args, vals;
};
#endif // MAINWINDOW_H
```

- qcustomplot.h – библиотека для работы с графиками функций

- tinyexpr.h – библиотека для интерпретации математических выражений

2. Файлы исходного кода:

- main.cpp:

```
#include "mainwindow.h"

#include <QApplication>
#include <QFile>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);

    //добавление и применение файла со стилем проекта
    QFile styleSheetFile("./Combinear.qss");
    styleSheetFile.open(QFile::ReadOnly);
    QString styleSheet = QLatin1String(styleSheetFile.readAll());
    qApp->setStyleSheet(styleSheet);

    MainWindow w;
    w.show();

    return a.exec();
}
```

- mainwindow.cpp:

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

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    //изменение имени окна
    this->setWindowTitle("Графики функций");

    //установка стандартного масштаба при запуске программы
    ui->graphicWidget->xAxis->setRange(-5, 5);
    ui->graphicWidget->yAxis->setRange(-5, 5);

    //установка возможности масштабирования и смещения графика при помощи
мыши
    ui->graphicWidget->setInteraction(QCP::iRangeZoom, true);
    ui->graphicWidget->setInteraction(QCP::iRangeDrag, true);

    //настройка цветовой палитры графика
    QFont font("Fixedsys", 9, QFont::Bold);
    ui->graphicWidget->setBackground(QBrush(QColor(75, 75, 75)));
    ui->graphicWidget->xAxis->grid()->setPen(QPen(QColor(204, 129, 0), 1,
Qt::SolidLine));
    ui->graphicWidget->yAxis->grid()->setPen(QPen(QColor(204, 129, 0), 1,
Qt::SolidLine));
```

```
    ui->graphicWidget->xAxis->grid()->setZeroLinePen(QPen(QColor(255, 190,
77), 2, Qt::SolidLine));
    ui->graphicWidget->yAxis->grid()->setZeroLinePen(QPen(QColor(255, 190,
77), 2, Qt::SolidLine));
    ui->graphicWidget->xAxis->setLabelFont(font);
    ui->graphicWidget->xAxis->setTickLabelColor(QColor(255, 190, 77));
    ui->graphicWidget->yAxis->setTickLabelColor(QColor(255, 190, 77));
}

MainWindow::~MainWindow()
{
    delete ui;
}

//функция нажатия кнопки "Нарисовать"
void MainWindow::on_drawButton_clicked()
{
    //контроль входных данных
    QString warning_empty = "", warning_error = "", warning_syntax = "",
warning_logic = "";

    if (ui->startEdit->text().isEmpty() || ui->stopEdit->text().isEmpty() ||
ui->funcEdit->text().isEmpty())
        warning_empty = " *все поля должны быть заполнены";

    if (ui->startEdit->text().toInt() == false && ui->startEdit->text() !=
"0" && ui->startEdit->text() != "")
        warning_syntax = " *диапазон значений функции должен быть задан
целыми числами";

    if (ui->stopEdit->text().toInt() == false && ui->stopEdit->text() != "0"
&& ui->stopEdit->text() != "")
        warning_syntax = " *диапазон значений функции должен быть задан
целыми числами";

    if (ui->startEdit->text().toInt() >= ui->stopEdit->text().toInt() &&
warning_syntax == "" && !(ui->stopEdit->text().isEmpty() || ui->funcEdit-
>text().isEmpty()))
        warning_logic = " *верхняя граница диапазона значений функции не
должна превосходить нижнюю";

    //блок построения графика в случае отсутствия ошибок ввода
    if (warning_empty == "" && warning_syntax == "" && warning_logic == "")
    {

        //инициализация промежуточных переменных для расчёта значений функции
и обработки точек разрыва 2-го рода
        QString result;
        QString xPositionText;
        int error = 0;
        bool infinity = false, error_all = false;
        int graphNum = 0;
        int infinity_missed = 0, infinity_counter = 0;
        double yMin = INFINITY, yMax = -INFINITY;
        const char* expression;

        //предварительная очистка виджета
        ui->graphicWidget->clearGraphs();
        args.clear();
        vals.clear();

        //присваивание начального и конечного значения промежутка расчёта
        start = ui->startEdit->text().toInt();
```

```
        stop = ui->stopEdit->text().toInt();

        xPosition = start;

        //расчёт векторных списков аргументов и значений функции и построение
графика
        while (xPosition <= stop && error == 0)
        {
            do
            {
                //сохранение в переменной введённой пользователем функции и
замена в ней аргумента "x" на текущее значение
                xPositionText.setNum(xPosition);
                result = ui->funcEdit->text();
                result.replace('x', xPositionText);
                expression = qPrintable(result);

                //расчёт значения функции, соответствующего текущему значению
аргумента
                double finalValue;
                finalValue = te_interp(expression, &error);

                if (error != 0) error_all = true;   //проверка возможности
интерпретации введённого текста в математическое выражение

                //расчёт значения функции, соответствующего следующему
значению аргумента (смещение на один шаг) для обработки точек разрыва
                xPositionText.setNum(xPosition + step);
                result = ui->funcEdit->text();
                result.replace('x', xPositionText);
                expression = qPrintable(result);

                //проверка наличия точки разрыва 2-го рода путём нахождения
модуля мгновенной производной функции в текущей точке
                if (abs((finalValue - te_interp(expression, 0))/step) > 1000)
                {
                    infinity = true;
                    infinity_counter++;
                }
                    else infinity_missed = 0;

                if (finalValue < yMin) yMin = finalValue;
                if (finalValue > yMax) yMax = finalValue;

                //добавление в векторный массив текущих значений аргумента и
функции
                args.push_back(xPosition);
                vals.push_back(finalValue);

                //смещение аргумента на 1 шаг
                xPosition += step;
            }
            while (infinity == false && xPosition <= stop);  //завершение
вложенного цикла при возникновении точки разрыва или при окончании промежутка
построения

            //смещение аргумента на 1 шаг, выполняемое только в случае
обнаружения точки разрыва или при окончании промежутка построения
            xPosition += step;

            //организация проверки для исключения возможности бесконечного
добавления частей графика при неверном обнаружении точки разрыва
            if (infinity_missed < 3)
```

```
            {
                //добавление новой части графика после обхода точки разрыва
                ui->graphicWidget->addGraph();
                ui->graphicWidget->graph(graphNum)->addData(args, vals);
                ui->graphicWidget->graph(graphNum)->setPen(QPen(QColor(255,
31, 31), 2, Qt::SolidLine));
                infinity_missed += 1;

                //итерация переменной, хранящей текущий номер части графика
                graphNum += 1;

                //очистка векторных массивов значений агрументов и функций
для последующего занесения данных, соответствующих новой части графика
                args.clear();
                vals.clear();
            }
            else infinity_counter--;

            infinity = false;
        }

        //сохранение информации о некорректности ввода в случае возврата
ошибки при попытке вычисления текущего значения функции
        if (error_all == true)
            warning_error = " *поле <функция> содержит некорректную запись";

        //установка масштаба графика в зависимости от наличия точек разрыва
        if (infinity_counter <= 0)
        {
            ui->graphicWidget->yAxis->setRange(yMin, yMax);
            ui->graphicWidget->xAxis->setRange((stop + start)/2 - (abs(yMax-
yMin)/2), (stop + start)/2 + (abs(yMax-yMin)/2));
        }
        else
        {
            ui->graphicWidget->xAxis->setRange(-5, 5);
            ui->graphicWidget->yAxis->setRange(-5, 5);
        }

        //построение графика в случае отсутствия ошибок ввода
        if (warning_empty == "" && warning_error == "" && warning_logic == ""
&& warning_syntax == "") ui->graphicWidget->replot();

    }

    //вывод сообщения об ошибках ввода в случае их наличия
    if (!(warning_empty == "" && warning_error == "" && warning_logic == ""
&& warning_syntax == "")) QMessageBox::warning(this, "Неверные данные",
                    "При заполнении параметров функции были допущены следующие
ошибки: "
                        + warning_empty + warning_error + warning_logic +
warning_syntax +
                            ". Пожалуйста, исправьте ошибки и повторите ввод.",
QMessageBox::Ok);
}


//функция нажатия кнопки "Очистить"
void MainWindow::on_clearButton_clicked()
{
    ui->graphicWidget->clearGraphs();

    ui->graphicWidget->replot();
```

```
    ui->funcEdit->clear();
    ui->startEdit->clear();
    ui->stopEdit->clear();
}


//функция нажатия кнопки "Центрировать"
void MainWindow::on_defaultButton_clicked()
{
    ui->graphicWidget->xAxis->setRange(-5, 5);
    ui->graphicWidget->yAxis->setRange(-5, 5);
    ui->graphicWidget->replot();
}


//функция нажатия кнопки "Выход"
void MainWindow::on_exitButton_clicked()
{
    qApp->exit();
}
```

- qcustomplot.cpp – библиотека для работы с графиками функций
- tinyexpr.c – библиотека для интерпретации математических выражений

Создание репозитория

- Первый коммит:

```
brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graphics_view (master)
$ git commit -m "Коммит"
[master (root-commit) 57a4f53] Коммит
 10 files changed, 44666 insertions(+)
 create mode 100644 graphics_view.pro
 create mode 100644 graphics_view.pro.user
 create mode 100644 main.cpp
 create mode 100644 mainwindow.cpp
 create mode 100644 mainwindow.h
 create mode 100644 mainwindow.ui
 create mode 100644 qcustomplot.cpp
 create mode 100644 qcustomplot.h
 create mode 100644 tinyexpr.c
 create mode 100644 tinyexpr.h
```

- Новая ветка с добавленным функционалом:

```
brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graph
ics_view (master)
$ git branch new_feature

brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graph
ics_view (master)
$ git checkout new-feature
error: pathspec 'new-feature' did not match any file(s) known to git

brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graph
ics_view (master)
$ git checkout new_feature
Switched to branch 'new_feature'
M       graphics_view.pro.user
M       mainwindow.cpp
```

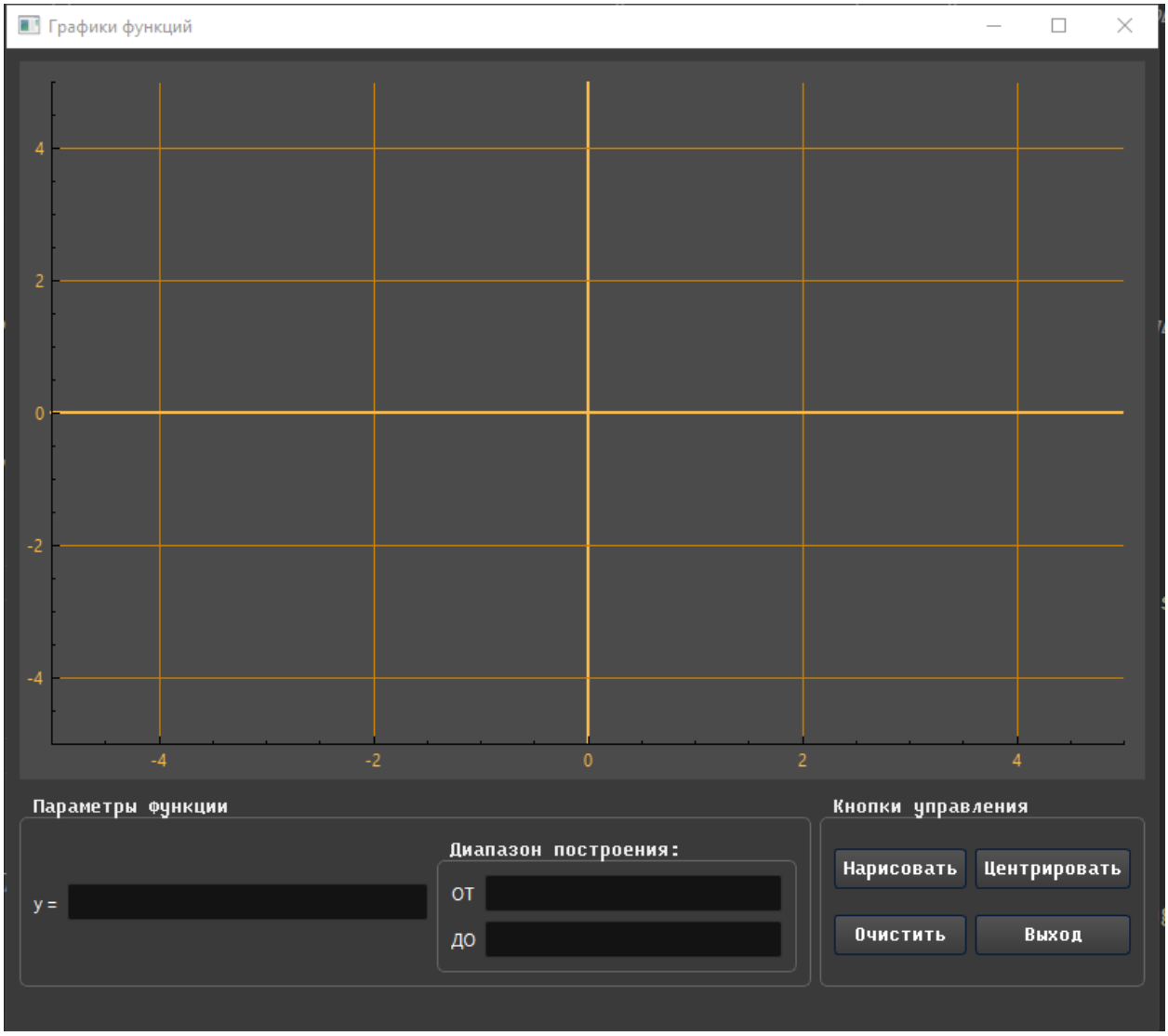
- Заполнение ветки и второй коммит:

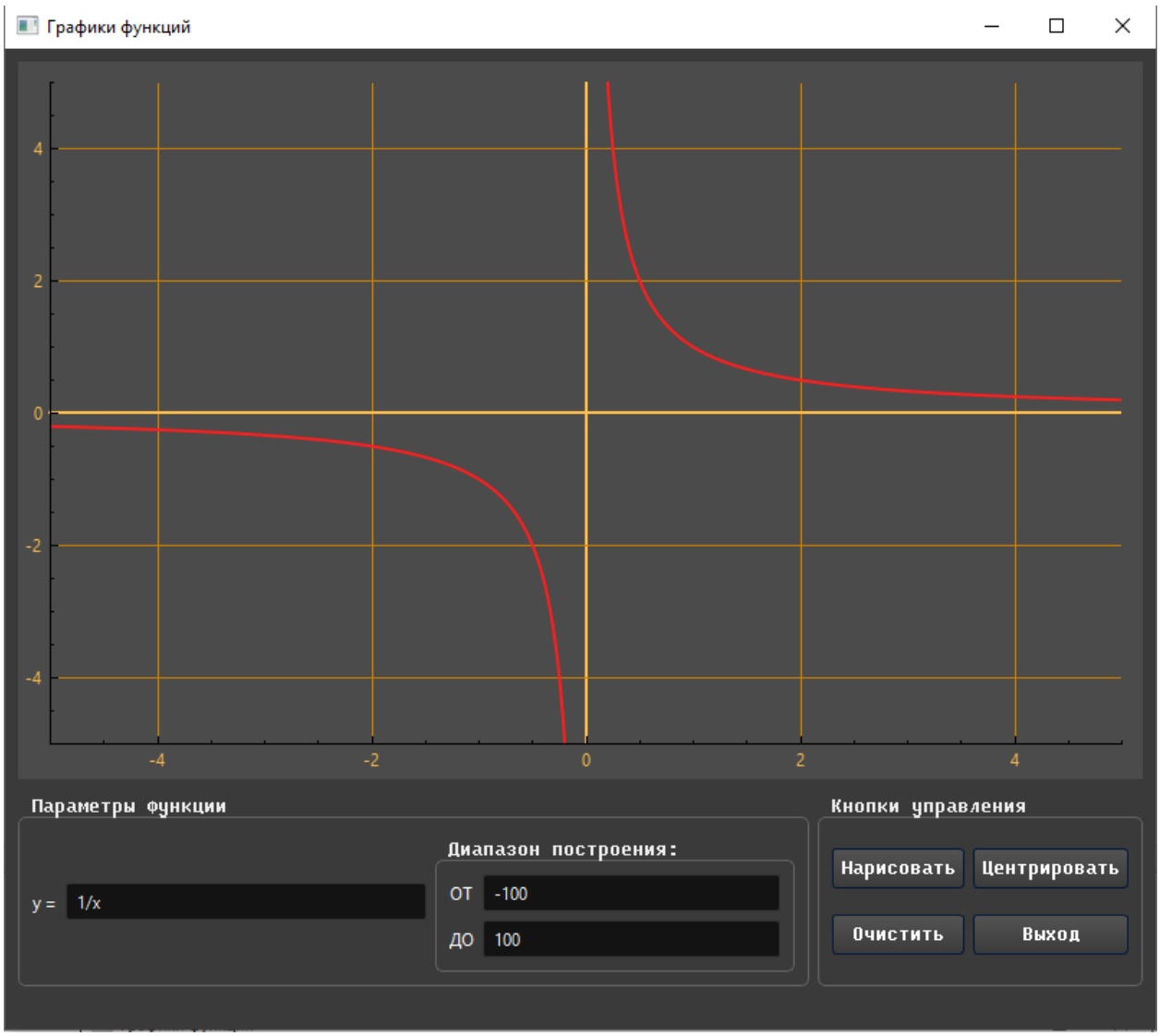
```
brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graph
ics_view (new_feature)
$ git status
On branch new_feature
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   graphics_view.pro.user
        modified:   mainwindow.cpp

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        commit.JPG

no changes added to commit (use "git add" and/or "git commit -a")

brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graph
ics_view (new_feature)
$ git add .
warning: CRLF will be replaced by LF in graphics_view.pro.user.
The file will have its original line endings in your working directory
warning: CRLF will be replaced by LF in mainwindow.cpp.
The file will have its original line endings in your working directory

brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graph
ics_view (new_feature)
$ git status
On branch new_feature
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   commit.JPG
        modified:   graphics_view.pro.user
        modified:   mainwindow.cpp


brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graph
$ git commit -m "Improved chart scalability, changed color of coordinate axes label"
[new_feature 7803100] Improved chart scalability, changed color of coordinate axes label
 3 files changed, 28 insertions(+), 5 deletions(-)
 create mode 100644 commit.JPG
```

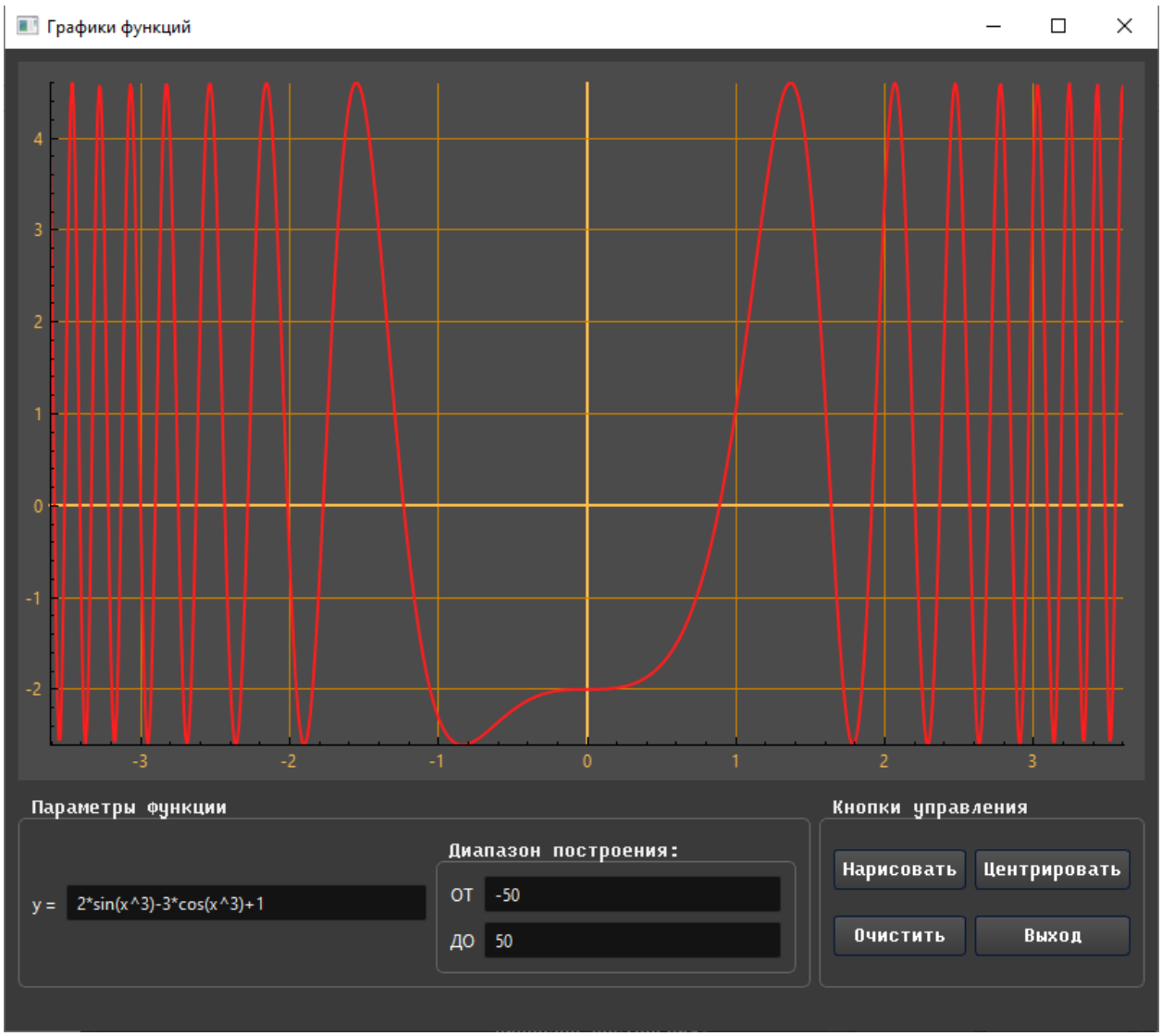
- Слияние веток репозитория:

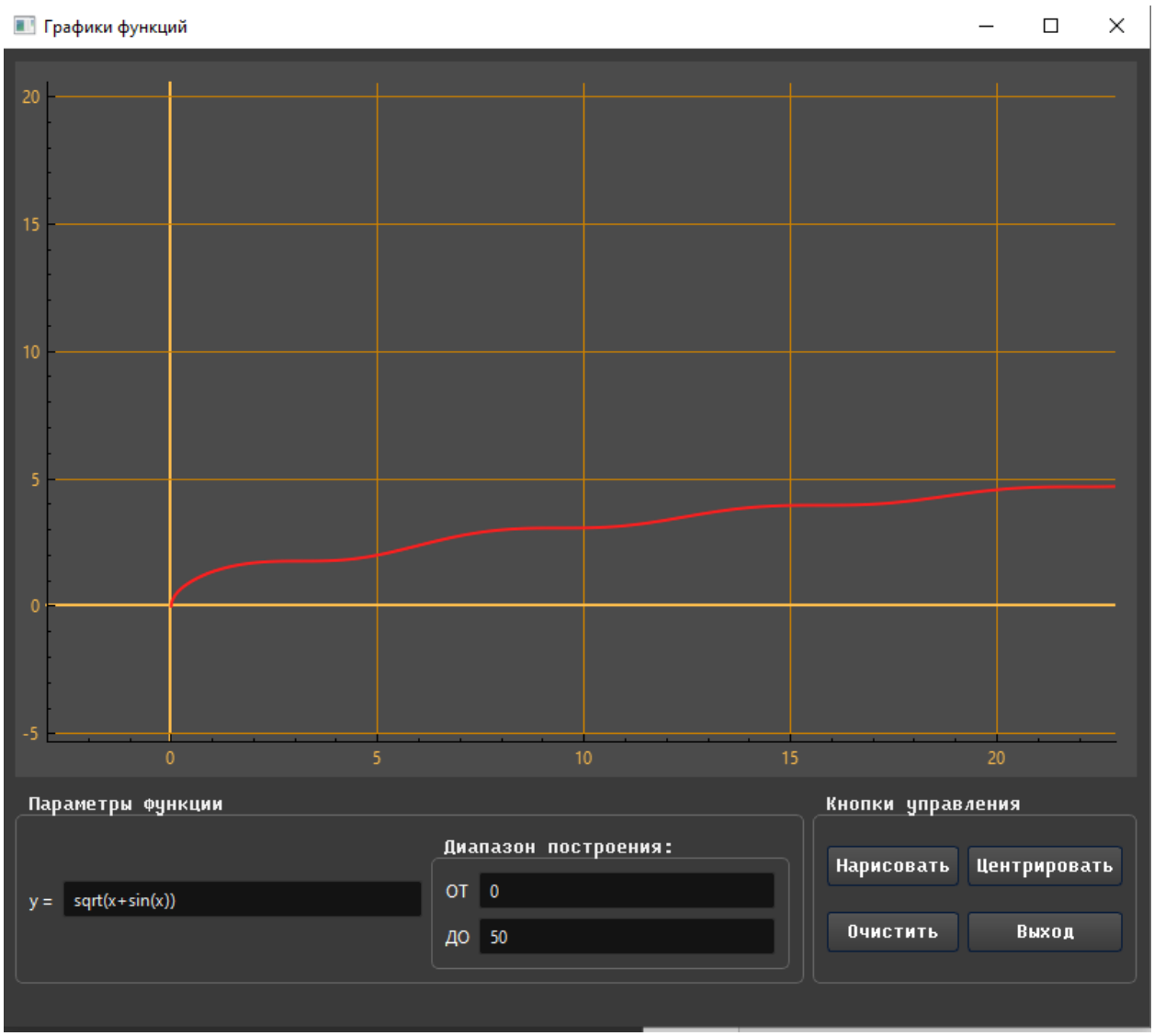
```
brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graphics_view (new_feature)
$ git checkout master
Switched to branch 'master'

brabi@DESKTOP-VCF9VKL MINGW64 ~/Desktop/Bonch.Zad/2 kurs/4 sem/C++/labs/QT/graphics_view (master)
$ git merge new_feature
Updating 57a4f53..35df99d
Fast-forward
 commit.JPG            | Bin 0 -> 34687 bytes
 graphics_view.pro.user |   6 +++---
 mainwindow.cpp        |  27 ++++++++++++++++++++++++---
 new_branch.JPG        | Bin 0 -> 35893 bytes
 new_commit.JPG        | Bin 0 -> 86155 bytes
 5 files changed, 28 insertions(+), 5 deletions(-)
 create mode 100644 commit.JPG
 create mode 100644 new_branch.JPG
 create mode 100644 new_commit.JPG
```

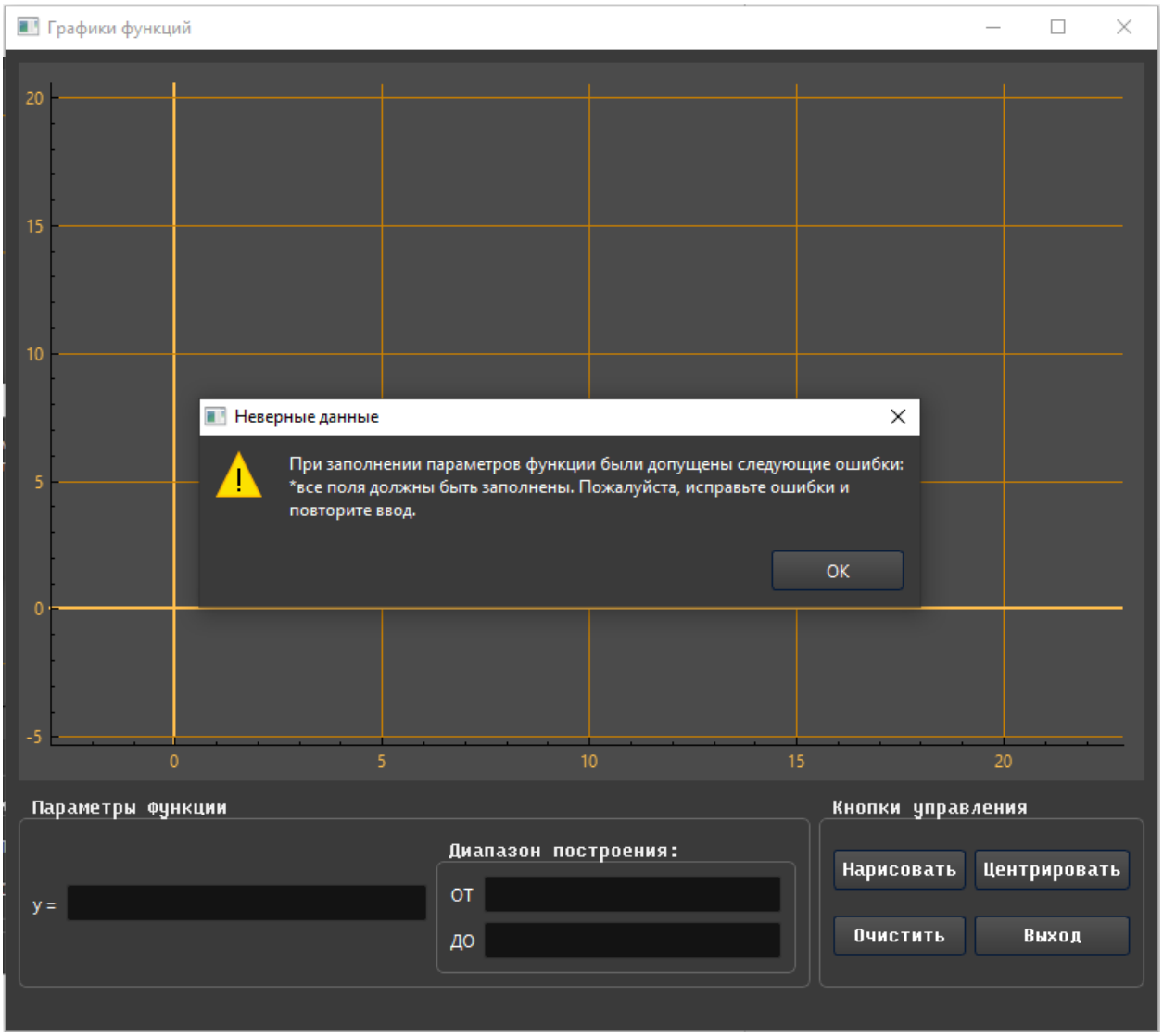
Тестирование

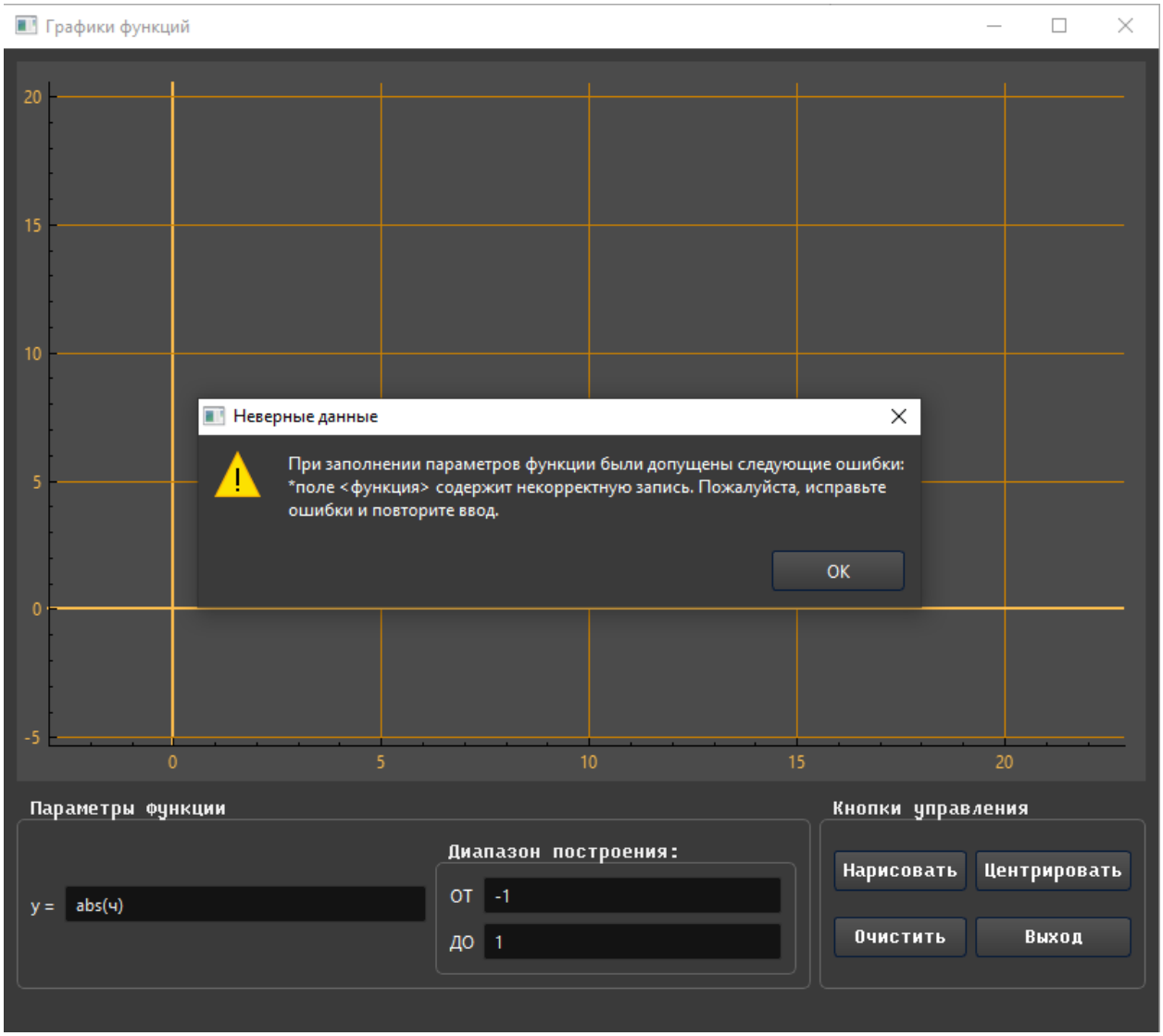


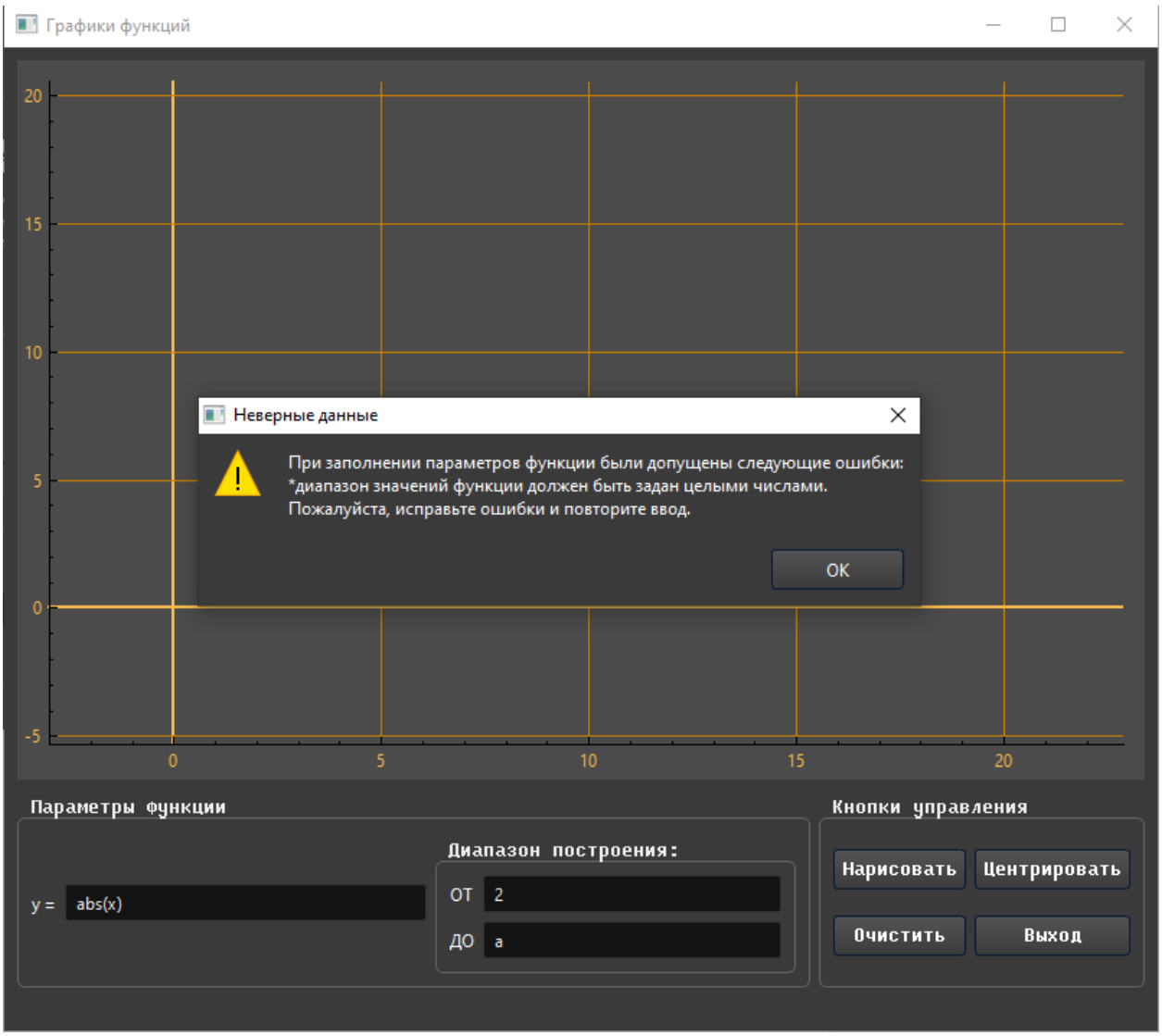

Графики функций
4
2
0
-2
-4
-4
-2
0
2
4
Параметры функции
y =
1/x
Диапазон построения:
ОТ
-100
ДО
100
Кнопки управления
Нарисовать
Центрировать
Очистить
Выход

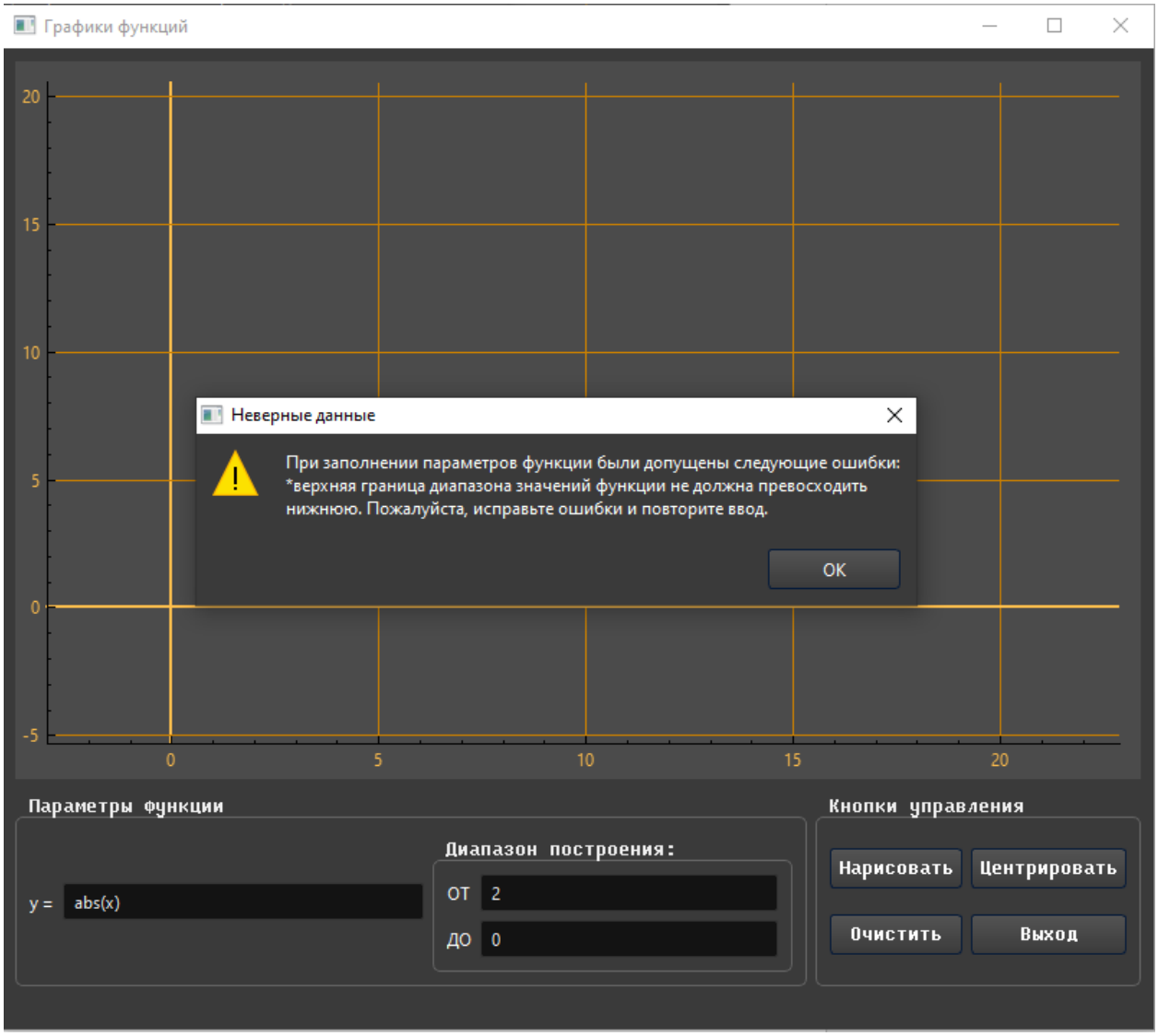

Графики функций
4
3
2
1
0
-1
-2
-3
-2
-1
0
1
2
3
Параметры функции
y =
2*sin(x^3)-3*cos(x^3)+1
Диапазон построения:
ОТ
-50
ДО
50
Кнопки управления
Нарисовать
Центрировать
Очистить
Выход


Графики функций
20
15
10
5
0
-5
0
5
10
15
20
Параметры функции
y =
sqrt(x+sin(x))
Диапазон построения:
ОТ
0
ДО
50
Кнопки управления
Нарисовать
Центрировать
Очистить
Выход


Графики функций
20
15
10
5
0
-5
0
5
10
15
20
Неверные данные
При заполнении параметров функции были допущены следующие ошибки:
*все поля должны быть заполнены. Пожалуйста, исправьте ошибки и повторите ввод.
OK
Параметры функции
y =
Диапазон построения:
ОТ
ДО
Кнопки управления
Нарисовать
Центрировать
Очистить
Выход


Графики функций
20
15
10
5
0
-5
0
5
10
15
20
Неверные данные
При заполнении параметров функции были допущены следующие ошибки:
*поле <функция> содержит некорректную запись. Пожалуйста, исправьте ошибки и повторите ввод.
OK
Параметры функции
y =
abs(ч)
Диапазон построения:
ОТ
-1
ДО
1
Кнопки управления
Нарисовать
Центрировать
Очистить
Выход


Графики функций
20
15
10
5
0
-5
0
5
10
15
20
Неверные данные
При заполнении параметров функции были допущены следующие ошибки:
*диапазон значений функции должен быть задан целыми числами.
Пожалуйста, исправьте ошибки и повторите ввод.
OK
Параметры функции
y =
abs(x)
Диапазон построения:
ОТ
2
ДО
a
Кнопки управления
Нарисовать
Центрировать
Очистить
Выход


Графики функций
20
15
10
5
0
-5
0
5
10
15
20
Неверные данные
При заполнении параметров функции были допущены следующие ошибки:
*верхняя граница диапазона значений функции не должна превосходить нижнюю. Пожалуйста, исправьте ошибки и повторите ввод.
OK
Параметры функции
y = abs(x)
Диапазон построения:
ОТ 2
ДО 0
Кнопки управления
Нарисовать
Центрировать
Очистить
Выход

Вывод

При получении задания по лабораторной работе №7 была поставлена задача написать на языке C++ программу с графическим интерфейсом, организующую построение графиков функций по заданным параметрам. Поставленная задача была успешно выполнена в среде разработки QT Creator (Community) версии 7.0.0.

Выполнение лабораторной работы помогло ознакомиться с процессом разработки графических приложений, а также с принципами работы с библиотеками.

Весь функционал протестирован при различных сценариях, программа работает корректно.