|  |  |
| --- | --- |
| Министерство науки и высшего образования Российской Федерации  Федеральное государственное автономное образовательное  учреждение высшего образования  «Южно-Уральский государственный университет  (национальный исследовательский университет)»  Институт естественных и точных наук  Кафедра прикладной математики и программирования | |
| Инженерный калькулятор | |
| Пояснительная записка к курсовой работе  по дисциплине «Методы программирования»  ЮУрГУ– 010302.2023.089.ПЗ КР | |
|  | Автор работы,  студент группы ЕТ-112  \_\_\_\_\_\_\_\_\_\_ / Н.М. Романов  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г. |
|  | Руководитель работы,  старший преподаватель  \_\_\_\_\_\_\_\_\_\_ / А.К. Демидов  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г. |
|  | Работа защищена с оценкой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г. |
| Челябинск 2024 | |

Федеральное государственное автономное образовательное учреждение высшего образования

«Южно-Уральский государственный университет   
(национальный исследовательский университет)»

Институт естественных и точных наук

Кафедра «Прикладная математика и программирование»

Направление Прикладная математика и информатика

УТВЕРЖДАЮ

Заведующий кафедрой ПМиП

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А.Замышляева

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

**ЗАДАНИЕ**

**на курсовую работу студента**

\_\_\_\_\_\_\_\_*Романова Н.М.*\_\_\_\_\_\_\_\_\_

Группа \_\_*ЕТ-112*\_\_\_

1. Дисциплина  *Методы программирования*

2. Тема работы \_\_\_\_*Инженерный калькулятор* \_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Срок сдачи студентом законченной работы *20 июня 2024 г.*

4. Перечень вопросов, подлежащих разработке

1. постановка задачи и алгоритм программы;
2. реализация программы на языке С;
3. оформление программной документации (руководство пользователя, листинг кода) и отчета по курсовой работе;
4. презентация проектных решений для защиты КР (постановка задачи, разработка алгоритма, особенности реализации)

5. Календарный план

|  |  |  |
| --- | --- | --- |
| **Наименование разделов**  **(этапов) курсовой работы** | **Срок выполнения**  **разделов (этапов) работы** | **Отметка**  **о выполнении**  **руководителя** |
| постановка задачи | 05.02.2024 – 20.02.2024 |  |
| разработка алгоритма | 29.02.2024 – 12.04.2024 |  |
| реализация программы | 14.03.2024 – 10.05.2024 |  |
| тестирование программы, улучшение и исправление ошибок | 31.03.2024 – 09.05.2024 |  |
| оформление программной документации и отчета по курсовой работе | 30.04.2024 – 30.05.2024 |  |
| защита курсовой работы | 14.05.2024 – 20.06.2024 |  |

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (расшифровка)

АННОТАЦИЯ

Романов Н.М. Инженерный калькулятор– Челябинск: ЮУрГУ, ЕТ-112, 2024. – 24с., 4 ил., библиографический список – 3 наим., 1 прил.

В курсовой работе описывается разработка инженерного калькулятора на языке программирования С.

Целью курсовой работы является получение навыков создания программного обеспечения на основе структурного подхода при использовании интегрированной среды разработки.

В процессе работы были выполнены все этапы разработки программы: постановка задачи, проектирование программы, включающее нисходящее проектирование и структурное программирование, реализация и отладка программы.

Пояснительная записка содержит результаты выполнения этих этапов, разработанные структуры данных и схемы алгоритмов, использованные математические модели, а также руководство пользователя программы, включающее примеры интерфейса.

В результате работы была разработан инженерный калькулятор, код которого приводится в приложении.

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 5](#__RefHeading___Toc4863_2799259349)

[1 ПОСТАНОВКА ЗАДАЧИ 6](#__RefHeading___Toc7484_2799259349)

[2 РАЗРАБОТКА АЛГОРИТМА 6](#__RefHeading___Toc7486_2799259349)

[3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 6](#__RefHeading___Toc7272_2799259349)

[ЗАКЛЮЧЕНИЕ 8](#__RefHeading___Toc7164_2799259349)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 9](#__RefHeading___Toc7166_2799259349)

[ПРИЛОЖЕНИЕ а 10](#__RefHeading___Toc7499_2626037720)

# ВВЕДЕНИЕ

**Актуальность темы**. Четкое соблюдение порядка этапов разработки программного обеспечения, понимание целей каждого из этих этапов, грамотное применение технологий и сред разработки обеспечивает реализацию сложных программных систем качественно и в срок.

**Цель работы** – разработать инженерный калькулятор, который способен выполнять, как базовые арифметические операции, так и сложные математические вычисления.

**Задачи работы**:

– научиться корректно осуществлять постановку задачи;

– изучить методики проектирования программ, включая нисходящее проектирование и структурное программирование;

– научиться разрабатывать и описывать алгоритмы на основе структурного подхода с применением метода пошаговой детализации и стандартных графических обозначений;

– научиться выполнять разработку и отладку программы для ее решения;

– получить навыки работы с различными средами программирования и прикладными библиотеками;

– овладеть способами планирования, организации, самоконтроля и самооценки деятельности;

– осуществлять целенаправленный поиск информации в сети Интернет по теории и практике программирования, математическим алгоритмам, библиотекам для разработки консольных и графических программ.

**Объект работы** – программа, предназначенная для выполнения различных математических вычислений.

**Предмет работы** – применение технологий разработки программного обеспечения на основе структурного подхода и языка С для разработки программы.

**Результаты работы** можно использовать в процессе последующего обучения в соответствии с учебным планом подготовки бакалавров по направлению «Прикладная математика и информатика»

## 1 ПОСТАНОВКА ЗАДАЧИ

Необходимо разработать программу, реализующую инженерный калькулятор. Для разработки ПО нужно использовать язык программирования С++, со встроенными библиотеками, и графическую библиотеку winBGIm с использованием средой разработки Visual Studio 2022.

После запуска программы должно появиться окно, с которой будет возможность взаимодействовать с кнопками.

Ниже предоставлен пример интерфейса инженерного калькулятора   
(Рисунок 1).

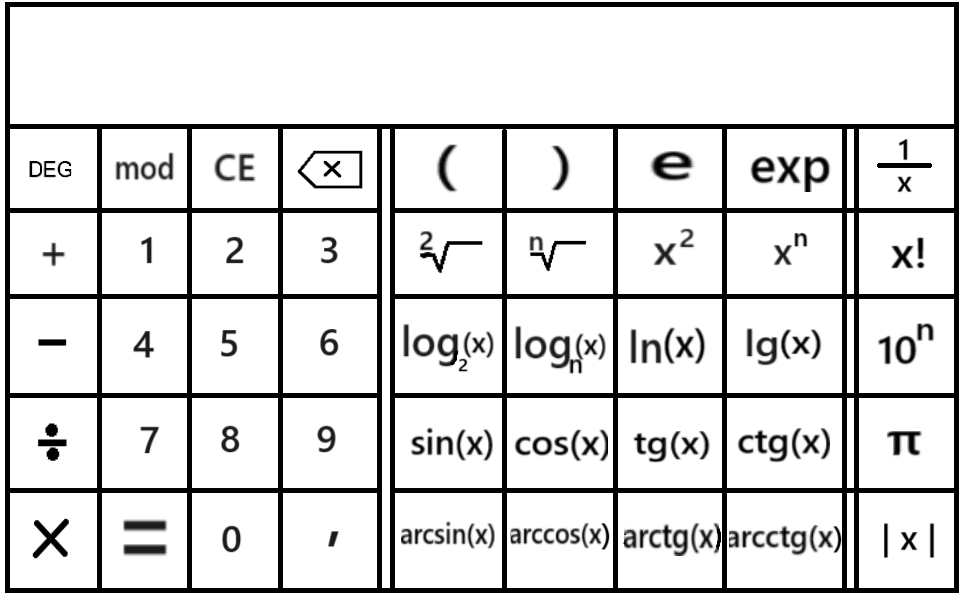


Рисунок 1 — Примерный интерфейс программного обеспечения.

Калькулятор должен иметь следующие возможности:

* Действующие кнопки на окне или действующие кнопки на клавиатуре.
* Действующая кнопка «СЕ», который полностью очищает ввод с экрана.
* Действующая кнопка удаление цифры из числа.
* Действующая кнопка «,», с помощью которой можно разделять целое число от десятичной части.
* Действующие кнопки «(» и «)» для расставление приоритетов.
* Получение результата числа на поле ввода числа.
* Действующие кнопки арифметических операций.
* Вычисление тригонометрических и обратных функций.
* Вычисление логарифмических функций.
* Вычисление квадратного корня и корня со степенью n.
* Вычисление возведения числа в квадрат и в степень n, а также числа 10 в степени n.
* Вычисление процента от числа и остатка от числа.
* Вычисление экспоненты, факториала числа, получение десятичного числа при делении на дробь и модуля числа.
* Действующие кнопки числа е и пи.

## 2 РАЗРАБОТКА АЛГОРИТМА

Основные сущности:

* Стек my\_stack содержит текущие вещественные результаты вычислений;
* Стек stack\_char содержит текущие символьные знаки (+, -, \*, /);
* Массив из кнопок buttons, каждая кнопка является структурой, содержащая координаты верхнего угла, размеры, имя файла с картинкой, загруженное изображение, тип кнопки;
* Тип кнопки typedef – перечисление {digit, arithmetic, funct};
* Строка результата result – строка из 150 символов;
* Строка error – строка из 100 символов для сообщения об ошибке.

Координаты кнопок можно вычислить по формуле:

Поэтому рисование этих кнопок и проверку нажатия можно проводить без определения вспомогательных структур данных.

Размеры кнопок можно посмотреть на рисунке 2.

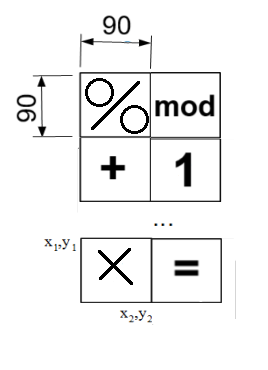


Рисунок 2­­ —­ Кнопки выбора функций

Схема основной алгоритм программы показана на рисунке 3. Схема обработка типа кнопки показана на рисунке 4. Алгоритм анализа и вычисления выражения со скобками показана на рисунке 5. Алгоритм анализа и вычисления выражения унарных математических функций показана на рисунке 6.

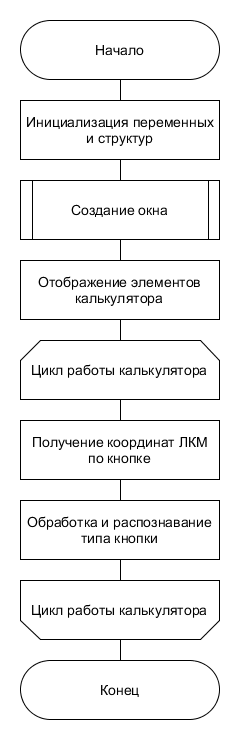


Рисунок 3 – Основной алгоритм программы

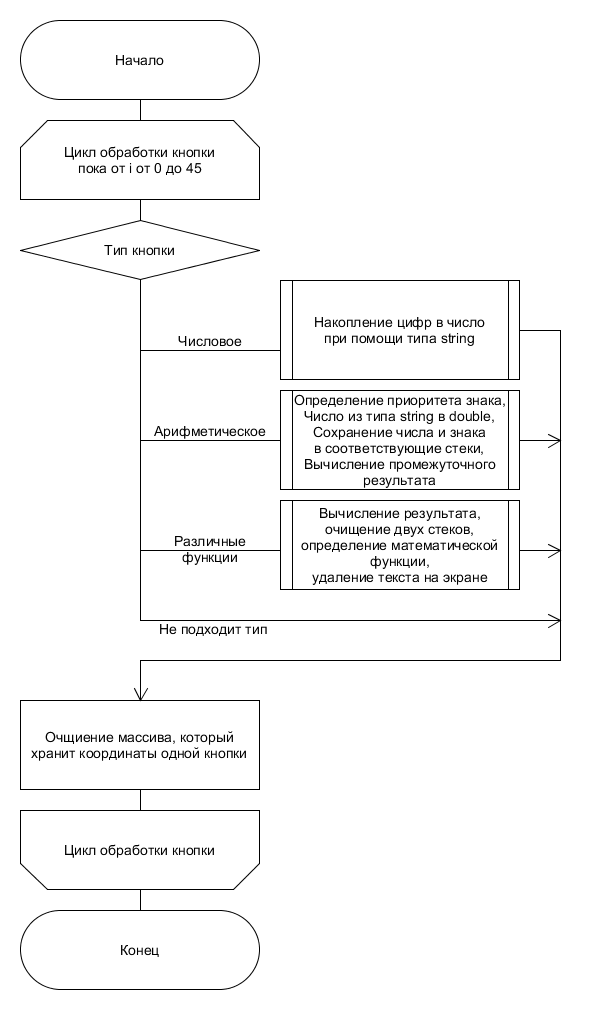


Рисунок 4 – Схема обработка типа кнопки

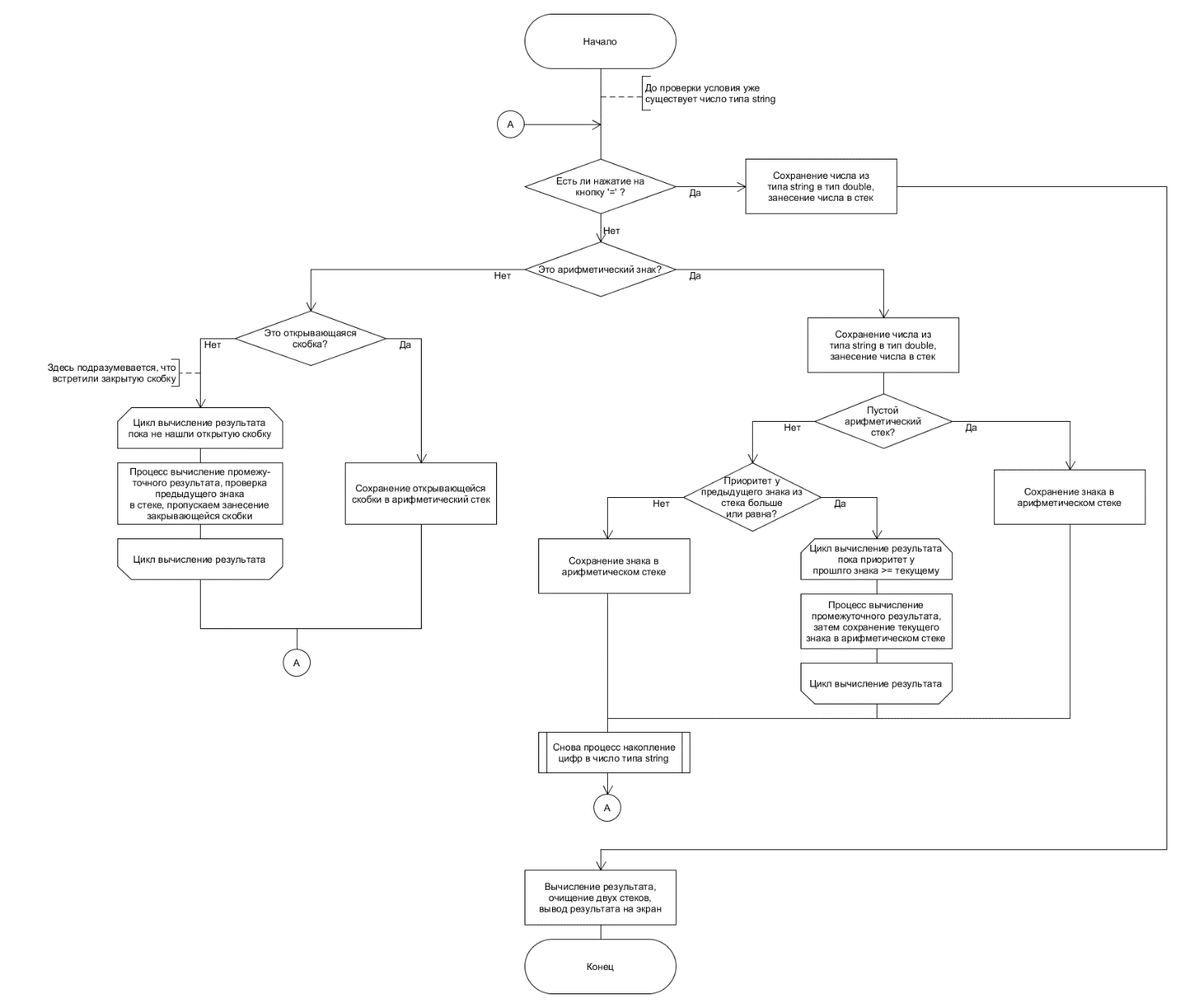


Рисунок 5 – Алгоритм анализа и вычисления выражения со скобками

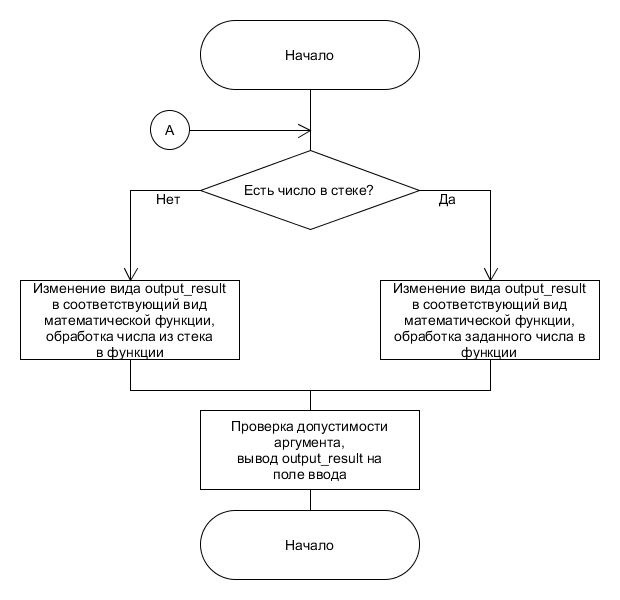


Рисунок 6 – Алгоритм анализа и вычисления выражения унарных математических функций

## 3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

При запуске программы откроется окно калькулятора, как было показано на рисунке 1.

На рисунке 7 продемонстрированы области кнопок для их взаимодействия с пользователем.

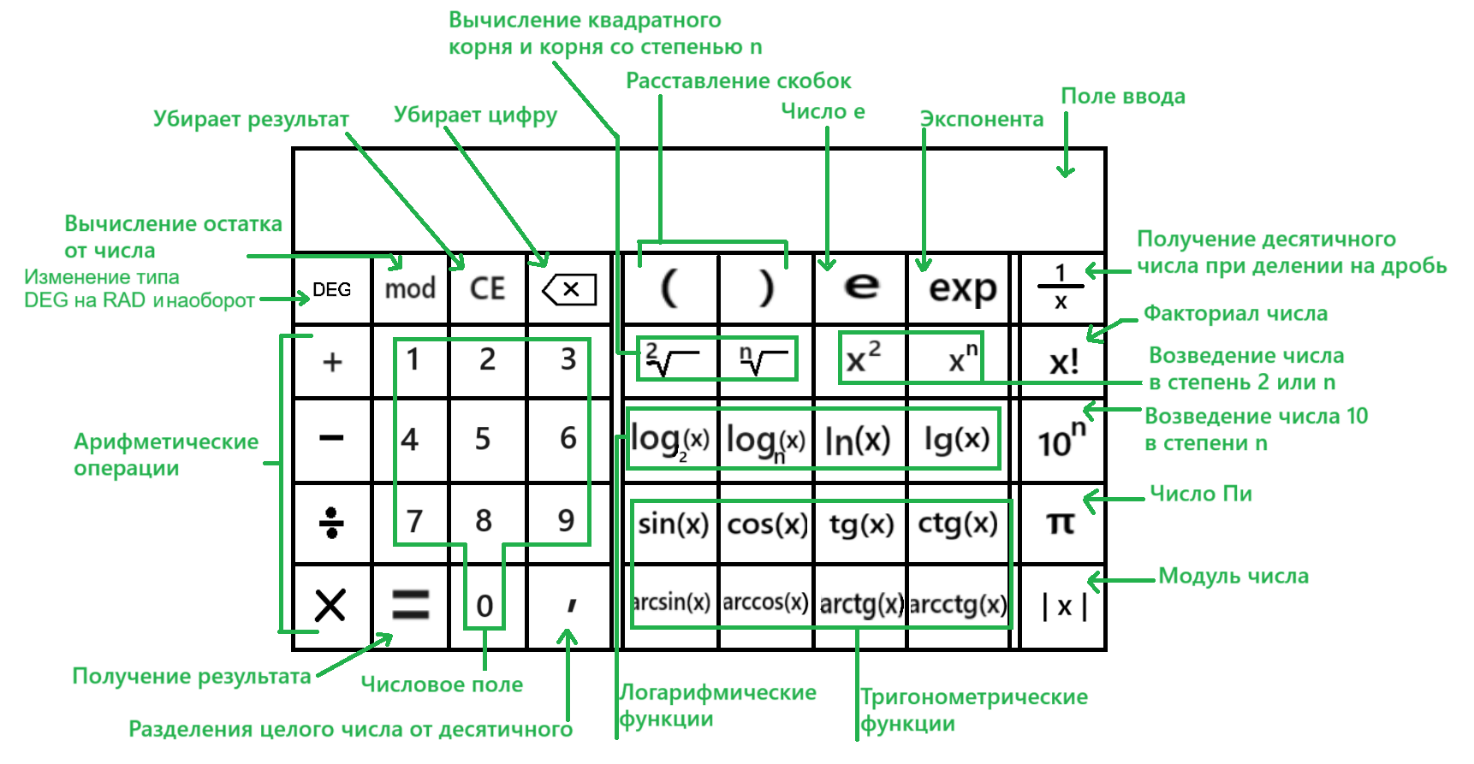


Рисунок 7 — Указания на отдельные области кнопок

Сверху над кнопками пользователь видит пустое поле ввода или диалоговое поле.

Для взаимодействия с полем необходимо нажать на любую кнопку из доступных в окне.

Каждая кнопка выводит свой символ или краткое словосочетание на поле для ввода.

В левой части окна расположены базовые кнопки для стандартного калькулятора такие, как: числовое поле – цифры от 0 до 9, арифметические операции, получение результата – знак равенства, разделение целого числа от десятичного – знак точки, вычисление остатка от числа, удаление результата – знак CE, удаление части выражения из поля ввода – знак крестика, а также смена типа из DEG в RAD и наоборот для тригонометрических функций.

В правой части окна расположены дополнительные математические функции, такие как: тригонометрические, логарифмические, корни и степени функций, кроме этого, существуют кнопки нахождения модуля числа, возведение числа 10 в степени n, нахождение факториала числа, вычисление экспоненты и десятичного числа.

Также существует две кнопки, которые обозначают константные значения – числа Пифагора и Эйлера.

Кроме этого, пользователь может воспользоваться скобками для составления более сложных математических выражений. Следует сделать замечание, что программное обеспечение может выдавать в поле для ввода текст с ошибкой, если пользователь не будет соблюдать одинаковое количество открытых и закрытых скобок.

Программа также может выдавать текст с ошибкой, если не соблюдена область допустимых значений (ОДЗ) в математических функциях, так как для них существуют условия перед выполнением результата операции.

Следует сделать важное замечание, что существует разделение математических функций на унарные и бинарные операции.

Различие в том, что бинарные принимают в себе два аргумента типа double, а унарные принимают только один аргумент такого же типа.

К унарным относятся такие, как: некоторые части логарифмических функций, вычисление квадратного корня, возведение числа в квадрат или числа 10 в степени n, вычисление факториала, модуля и получение десятичного числа при делении на дробь.

К бинарным относятся такие, как: остальная часть логарифмических функций, вычисление корня от числа в степени n, возведение в степени n, вычисление остатка и процента от двух чисел.

Важно, что перед использованием этих функций, обязательно, нужно записать первое число в качестве первого аргумента – степени или основания, второго числа в качестве второго аргумента – параметр математического функции. К примеру, 2MOD3, 2log8, 2sqrt4, 2pow3.

Для использования унарных математических функций сначала задаётся аргумент (число), а затем выбираться функция. После этого программа сразу же меняет вид к привычному виду выражения.

При использовании тригонометрических функций можно использовать кнопку для переключения типа из DEG в RAD и наоборот. Таким образом, аргументы в таких функциях могут быть в радианах или градусах.

Рассмотрим, как обозначаются математические в поле для ввода:

1. Бинарные:

* MOD ­– вычисление остатка
* sqrt – вычисление корня от степени n
* ^ – вычисление числа от степени n
* log – вычисление логарифма с основание n

1. Унарные:

* exp – вычисление экспоненты
* 1/ - вычисление десятичного числа при делении на n
* sqrt(2) – вычисление квадратного корня
* 2^ – вычисление квадрата
* 10^ – вычисление числа 10 от степени n
* fact – вычисление факториала
* abs – вычисление абсолютного значения числа
* log(2) – вычисление логарифма с основанием 2
* ln – вычисление натурального логарифма
* lg – вычисление десятичного логарифма
* sin, cos, tg, ctg – вычисление тригонометрических функций
* arcsin, arccos, arctg, arcctg – вычисление обратных тригонометрических функций

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы были поставлены точные требования к программе, затем были выявлены элементы интерфейса пользователя, разработаны необходимые математические модели, определены и детализированы структуры данных и алгоритмы. После завершения проектирования алгоритмы были реализованы на языке С. Разработанный код был проверен на контрольных тестах и в код были внесены необходимые исправления. Для программы было разработано руководство пользователя. Таким образом, цель работы была достигнута, задачи – решены.

Результаты работы можно использовать в процессе последующего обучения в форме навыков практического применения структурного программирования для разработки сложных программных систем, понимания порядка этапов разработки программного обеспечения и достигаемых на каждом этапе результатов.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Подбельский В. В. Программирование на языке Си : учеб. пособие для вузов по направлениям : "Приклад. математика и информатика", "Информатика и вычисл. техника" / В. В. Подбельский, С. С. Фомин. - 2-е изд., доп.. - М. : Финансы и статистика, 2007. - 600 с.

2 Графическая библиотека WinBGIm. – URL: https://ipc.susu.ru/20786.html (дата обращения: 02.02.2022).

3 Графическая библиотека WinBGLm. – URL: https://home.cs.colorado.edu/~main/bgi/doc/ (дата обращения: 04.04.2024)

4 Руководство по программирования на языке С. – URL: https://metanit.com/c/tutorial/?ysclid=lxdmhddpgf195866871 (дата обращения: 20.02.2024)

5 Калькулятор со скобками и приоритетами. – URL: https://www.youtube.com/watch?v=Vk-tGND2bfc&t=370s (дата обращения: 14.03.2024)

# ПРИЛОЖЕНИЕ а

А.1 Файл main\_calc.cpp

#include <iostream>

#include <string>

#include <cmath>

#include <locale.h>

#include "Button.h"

#include "Math\_for\_calc.h"

#include "Button\_processing.h"

#include "graphics.h"

using namespace std;

//Задан размер диалогового окна

#define WIDTH 810

#define HEIGHT 530

int main() {

//-----Массив\_с\_координатами\_клика\_мышью-----

//Каждый клин мышью имеет координаты по оси Ох и оси Oy

//Эти два ниже объекта используются в заголовочном и .cpp файлах, названная как Button

int mass[2] = { 0 }; //Хранение координат одного клика в массиве

int count\_mass = 0; //Даст ответ количество уже знаесенных в массив

//-----Массив\_с\_координатами\_клика\_мышью-----

//----Объявление двух стеков и строкового массива - нужно для функции about\_but()----

my\_stack m;

stack\_char b;

stack\_init(&m);

stack\_init(&b);

//Промежуточный массив перед output\_result{}

string result\_str{}; //Нужна после нажатия пользователем на кнопку '='

//Массив для вывода результата и выражения на диалоговое окно

string output\_result;

setlocale(LC\_ALL, "rus"); //Необходимо для того, чтобы получился вывод русских символов на диалоговое окно

initwindow(WIDTH, HEIGHT, "calculat", true, true); //Создание диалогового окна

draw\_buts();

if (kbhit() == 1) {

closegraph();

}

//функция по замене типа DEG на RAG и наоборот

replace\_deg\_rag(&is\_check\_deg\_rag);

//В цикле происходит работа калькулятора, в заголовочном файле Button.h расписаны комментарии к этим функциям

while (true) {

//Получаем координаты из нажатия ЛКМ:

int x = 0, y = 0;

getmouseclick(WM\_LBUTTONDOWN, x, y); // Получаем координаты мыши

if (x > 0 && y > 0) {

// Заносим координаты в массив

mass[count\_mass] = x; // Записываем x

mass[count\_mass + 1] = y; // Записываем y

// Увеличиваем счетчик на 2, так как мы записали два элемента (x и y)

count\_mass += 2;

clearmouseclick(WM\_LBUTTONDOWN); // Очищаем событие нажатия мыши

about\_but(mass, &count\_mass, output\_result, result\_str, m, b);

}

}

}

А.2 Файл Button.cpp

#include <iostream>

#include "Button.h"

#include "Math\_for\_calc.h"

#include "Button\_processing.h"

#include "graphics.h"

//Структура кнопки, в которой хранятся координаты, приоритеты, названия и типы

typedef struct Button {

int x1;

int y1;

int x2;

int y2;

int priority;

char name[20];

char symb[20];

std::string type\_str;

int type;

IMAGE\* bmp;

};

//Перечисление, которая нужна для определение типа кнопки (Используется в switch)

enum type {

BINARY\_OP, //Бинарные математические функции или арифметические операции

UNARY\_OP, //Унарные математические функции

PARENTHESIS, //Скобки

DIGIT, //Цифры

CONSTANT, //Константы

EDITION, //Очищение экрана или посимвольное удаление

RESULT, //Вычисление результата

DEGRAD //Перевод из типа DEG в RAG (и наоборот | по умолчанию DEG)

};

//Структура кнопки

struct Button but[N] = { {0, 80, 89, 169, 3, "buttons\\DEGRAD.bmp", "", "DEGRAD", DEGRAD}, {90, 80, 179, 169, 3, "buttons\\mod.bmp", "V", "MOD", BINARY\_OP}, {180, 80, 269, 169, 0, "buttons\\CE.bmp", "", "CE", EDITION}, {270, 80, 359, 169, 0, "buttons\\DELETE.bmp", "", "delete", EDITION}, {360, 80, 449, 169, 0, "buttons\\(.bmp", "(", "(", PARENTHESIS}, {450, 80, 539, 169, 0,"buttons\\).bmp", ")", ")", PARENTHESIS}, {540, 80, 629, 169, 0,"buttons\\e.bmp", "e", "2,718281828459045", CONSTANT}, {630, 80, 719, 169, 3,"buttons\\exp.bmp", "W", "exp", UNARY\_OP}, {720, 80, 810, 169, 3,"buttons\\del1.bmp", "X", "1/", UNARY\_OP} ,

{0, 170, 89, 259, 1, "buttons\\+.bmp", "+", "+", BINARY\_OP}, {90, 170, 179, 259, 0, "buttons\\1.bmp", "1", "1", DIGIT}, {180, 170, 269, 259, 0, "buttons\\2.bmp", "2", "2", DIGIT}, {270, 170, 359, 259, 0, "buttons\\3.bmp", "3", "3", DIGIT}, {360, 170, 449, 259, 3, "buttons\\sqrt2.bmp", "A", "sqrt2", UNARY\_OP}, {450, 170, 539, 259, 3, "buttons\\sqrt.bmp", "B", "sqrt", BINARY\_OP}, {540, 170, 629, 259, 3, "buttons\\x2.bmp", "C", "2^", UNARY\_OP}, {630, 170, 719, 259, 3, "buttons\\xn.bmp", "D", "^", BINARY\_OP}, {720, 170, 810, 259, 3, "buttons\\x!.bmp", "E", "fact", UNARY\_OP},

{0, 260, 89, 349, 1, "buttons\\-.bmp", "-", "-", BINARY\_OP}, {90, 260, 179, 349, 0, "buttons\\4.bmp", "4", "4", DIGIT}, {180, 260, 269, 349, 0, "buttons\\5.bmp", "5", "5", DIGIT}, {270, 260, 359, 349, 0, "buttons\\6.bmp", "6", "6", DIGIT}, {360, 260, 449, 349, 3, "buttons\\log2.bmp", "F", "log2", UNARY\_OP}, {450, 260, 539, 349, 3, "buttons\\logn.bmp", "G", "log", BINARY\_OP}, {540, 260, 629, 349, 3, "buttons\\ln.bmp", "H", "ln", UNARY\_OP}, {630, 260, 719, 349, 3, "buttons\\lg.bmp", "I", "lg", UNARY\_OP}, {720, 260, 810, 349, 3, "buttons\\10n.bmp", "K", "10^", UNARY\_OP},

{0, 350, 89, 439, 2, "buttons\\del.bmp", "/", "/", BINARY\_OP}, {90, 350, 179, 439, 0, "buttons\\7.bmp", "7", "7", DIGIT}, {180, 350, 269, 439, 0, "buttons\\8.bmp", "8", "8", DIGIT}, {270, 350, 359, 439, 0, "buttons\\9.bmp", "9", "9", DIGIT}, {360, 350, 449, 439, 3, "buttons\\sin.bmp", "L", "sin", UNARY\_OP}, {450, 350, 539, 439, 3, "buttons\\cos.bmp", "M", "cos", UNARY\_OP}, {540, 350, 629, 439, 3, "buttons\\tg.bmp", "N", "tg", UNARY\_OP}, {630, 350, 719, 439, 3, "buttons\\ctg.bmp", "O", "ctg", UNARY\_OP}, {720, 350, 810, 439, 0, "buttons\\pi.bmp", "pi", "3,14159265358979323846", CONSTANT},

{0, 440, 89, 530, 2, "buttons\\X.bmp", "\*", "\*", BINARY\_OP}, {90, 440, 179, 530, 0, "buttons\\=.bmp", "=", "=", RESULT}, {180, 440, 269, 530, 0, "buttons\\0.bmp", "0", "0", DIGIT}, {270, 440, 359, 530, 0, "buttons\\point.bmp", ",", "point", DIGIT}, {360, 440, 449, 530, 3, "buttons\\arcsin.bmp", "P", "arcsin", UNARY\_OP}, {450, 440, 539, 530, 3, "buttons\\arccos.bmp", "Q", "arccos", UNARY\_OP}, {540, 440, 629, 530, 3, "buttons\\arctg.bmp", "R", "arctg", UNARY\_OP}, {630, 440, 719, 530, 3, "buttons\\arcctg.bmp", "S", "arcctg", UNARY\_OP}, {720, 440, 810, 530, 3, "buttons\\modulx.bmp", "T", "abs", UNARY\_OP}};

int priority; //Нужен для запоминания приоритета у знака

int priority\_skb\_o; //Запоминает количество открытых скобок

int check\_skb\_z = false; //Проверка, что пользователь нажал на закрытую скобку

int is\_break\_one = false; //Для завершения внешнего цикла

int special\_skb = false; // Проверка на то, что пользователь пользуется скобками при включении мат.функций

int use\_arith = false; // Запрет на использование какой-либо арифметики для бинарных функций

int is\_check\_constant = false; //Проверка на то, что пользователь нажал на константу

int is\_check\_del = false; //Проверка на делимость двух чисел

int is\_check\_deg\_rag = false; //Проверка на замену из DEG в RAG и наоборот (По умолчанию DEG)

int is\_finish = false; //Проверка, что нажата кнопка =

int is\_check\_error = false; // Проверка, если математическая функция дала ошибку

int is\_check\_binary = false; //Проверка, что пользователь пользуется бинарной функцией

char is\_check\_binary\_symb; //Проверка, какая была нажата бинарная функция

double num\_1 = 0, num\_2 = 0;//Переменные для бин.функции

std::string num\_str{};//Накопление цифр в строке

//Отображение картинок

void draw\_buts() {

setcolor(LIGHTGRAY);

for (int i = 0; i < N; i++) {

but[i].bmp = loadBMP(but[i].name);

if (but[i].bmp == nullptr) {

std::cout << "Ошибка загрузки изображения!" << "\n";

closegraph();

return; // Выйдем из функции при возникновении ошибки загрузки изображения

}

putimage(but[i].x1, but[i].y1, but[i].bmp, COPY\_PUT);

rectangle(but[i].x1, but[i].y1, but[i].x2, but[i].y2);

}

}

void clear\_global\_obj(){

priority = 0; //Нужен для запоминания приоритета у знака

priority\_skb\_o = 0; //Запоминает количество открытых скобок

check\_skb\_z = false; //Проверка, что пользователь нажал на закрытую скобку

is\_break\_one = true; //Для завершения внешнего цикла

use\_arith = false; // Запрет на использование какой-либо арифметики для бинарных функций

special\_skb = false; // Проверка на то, что пользователь пользуется скобками при включении мат.функций

is\_check\_constant = false; //Если объявляли ранее константу, то уберем флажок

is\_check\_del = false; //Проверка на делимость двух чисел

is\_finish = false; //Проверка, что нажата кнопка =

is\_check\_error = false; // Проверка, если математическая функция дала ошибку

is\_check\_binary = false; //Проверка, что пользователь пользуется бинарной функцией

//Переменные для бинарной функции

num\_1 = 0;

num\_2 = 0;

}

//Очищение двух стеков

void clear\_the\_input(my\_stack& m, stack\_char& b) {

//Очистка двух стеков

while (stack\_isempty(&b) == false || stack\_isempty(&m) == false) {

if (stack\_isempty(&m) == false) {

stack\_pop(&m);

}

else if (stack\_isempty(&b) == false) {

stack\_pop(&b);

}

}

}

//Очищение буфера, который хранит координаты нажатия ЛКМ

void clear\_bufer(int\* mass, int\* jndex, int\* count\_mass) {

//-----Очищает память-----

mass[(\*jndex)] = 0;

mass[(\*jndex) + 1] = 0;

\*count\_mass = 0;

//-----Очищает память-----

}

//Изменение типа из DEG в RAD и наоборот

void replace\_deg\_rag(int\* is\_check\_deg\_rag) {

if (\*is\_check\_deg\_rag == false) {

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 1);

outtextxy(15, 114, "DEG"); // 15, 114

}

else if (\*is\_check\_deg\_rag == true) {

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 1);

outtextxy(15, 114, "RAG");

}

}

//Функция по удалению временного вычисления

void clear\_interim\_solution() {

//Делаем черный прямоугольник для удаление числа

setfillstyle(SOLID\_FILL, BLACK);

bar(10, 35, 810, 70);

//Показываем на экран цифры

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 4);

outtextxy(10, 35, num\_str.c\_str());

}

//Функция по замене знака '-' на '+' для мат.функ. abs

void draw\_template\_abs(std::string& output\_result ){

int s1 = 0, s2 = 0;

s2 = output\_result.length();

output\_result.erase(s2-1);

output\_result += '+';

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 3);

outtextxy(10, 10, output\_result.c\_str());

}

//Рисование шаблона для унарнх функций

void draw\_template(std::string& output\_result, int index, double num) {

output\_result += but[index].type\_str + '(' + std::to\_string(num) + ')';

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 3);

outtextxy(10, 10, output\_result.c\_str());

}

//Рисования шаблона для бинарных функций

void draw\_temlate\_buniary(std::string& output\_result, int index, double num\_1) {

output\_result += std::to\_string(num\_1) + but[index].type\_str;

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 3);

outtextxy(10, 10, output\_result.c\_str());

}

//В этой функции обрабатывается тип кнопки

void about\_but(int\* mass, int\* count\_mass, std::string& output\_result, std::string& result\_str, my\_stack& m, stack\_char& b) {

if (mass[0] == 0 && mass[1] == 0) {

return;

}

for (int i = 0; i < N; i++) { //Проходимся по структуре

//Для завершения внешнего цикла

if (is\_break\_one == true) {

is\_break\_one = false;

break;

}

for (int j = 0; j < (\*count\_mass); j += 2) { //Проходимся по массиву пар "кнопок"

//Проверка по Х и проверка по Y

if ((but[i].x1 <= mass[j] && mass[j] <= but[i].x2) && (but[i].y1 <= mass[j + 1] && mass[j + 1] <= but[i].y2)) {

switch (but[i].type) {

case BINARY\_OP: {

use\_arith = true; // Использовалось арифметика

is\_check\_constant = false; //Если объявляли ранее константу, то уберем флажок

clear\_bufer(mass, &j, count\_mass);

//Если пользователь нажал на бинарную функцию:

if (num\_str != "" && num\_1 == 0 && num\_2 == 0 && (but[i].type\_str == "MOD" || but[i].type\_str == "^" || but[i].type\_str == "sqrt" || but[i].type\_str == "log")) {

is\_check\_binary = true;

is\_check\_binary\_symb = but[i].symb[0]; // передаем специальный символ бин.мат.функции

num\_1 = std::stod(num\_str);

num\_str.clear();

draw\_temlate\_buniary(output\_result, i, num\_1); // Отображем выражение на экране

is\_break\_one = true;

break;

}

else if (num\_str != "" && is\_check\_binary == true && (but[i].symb[0] == '+' || but[i].symb[0] == '-' || but[i].symb[0] == '\*' || but[i].symb[0] == '/')) {

num\_2 = std::stod(num\_str);

out\_expression\_str(output\_result, num\_str); //Отображаем число

out\_expression(output\_result, but[i].symb[0]); // Отображаем знак

num\_str.clear();

switch\_binary\_func(is\_check\_binary\_symb, num\_1, num\_2, &is\_check\_error, output\_result, m); //Определение бин.функции

if (is\_check\_error == true) {

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

break;

}

setfillstyle(SOLID\_FILL, BLACK);

bar(10, 35, 810, 70);

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 4);

outtextxy(10, 35, std::to\_string(stack\_top(&m)).c\_str()); //Достаем число в качестве промежуточного вычисления

is\_break\_one = true;

break;

}

//Если не ввели число, то пропускаем ход

else if (num\_str == "" && is\_check\_binary == true) {

is\_break\_one = true;

break;

}

//Если пользователь опять нажал на арифметический знак, то ничего не делаем, должно поступить от пользователя число

if (num\_str == "" && check\_skb\_z == false && stack\_isempty(&m) == false && stack\_isempty(&b) == false) {

is\_break\_one = true;

break; //Досрочно уходим

}

//Если пользователь изначально хочет ввести знаки, которые написаны в условии ниже, то пропускаем

//output\_result.length() == 0

if (num\_str == "" && check\_skb\_z == false &&(but[i].symb[0] == '+' || but[i].symb[0] == '\*' || but[i].symb[0] == '/')) {

is\_break\_one = true;

}

//Если пользователь с самого начала нажал на знак '-' | result\_out == ""

else if (output\_result.length() == 0 && num\_str == "" && but[i].symb[0] == '-' && check\_skb\_z == false) {

is\_break\_one = true;

out\_expression(output\_result, '0'); //Показываем число 0

out\_expression(output\_result, but[i].symb[0]); // По символьно выводим арифметический знак на диалоговое окно

arith\_switch(but[i].symb[0], &priority, b); //Сначала рассмотрим какая приоритетность у знака

switch\_arith\_push(but[i].symb[0], m, b); //Затем заносим в стек и знаем, что за приоритет у знака

}

else {

out\_expression\_str(output\_result, num\_str); //Сначала покажем число

savestr(num\_str, m); // Затем сохраним число

out\_expression(output\_result, but[i].symb[0]); //Потом покажем знак

//Рассмотрим изменение выражения, если 5-4\*,4-3-, 4-2+, 4-1/ . -> ?

if (stack\_top(&b) == '-' && (but[i].symb[0] == '+' || but[i].symb[0] == '-' || but[i].symb[0] == '\*' || but[i].symb[0] == '/')) { // Проверить вот этот случай

delete\_sub(&priority, m, b);

}

if (stack\_isempty(&b) == true) {

is\_break\_one = true;

arith\_switch(but[i].symb[0], &priority, b);

switch\_arith\_push(but[i].symb[0], m, b);

}

else if (priority == but[i].priority || priority > but[i].priority) {

while (priority == but[i].priority || priority > but[i].priority) {

merging\_two\_numbers(&priority, m, b, output\_result);

}

is\_break\_one = true;

//Затем новый знак заносим (1)-(2)

arith\_switch(but[i].symb[0], &priority, b);

switch\_arith\_push(but[i].symb[0], m, b);

//Делаем черный прямоугольник для удаление числа

setfillstyle(SOLID\_FILL, BLACK);

bar(10, 35, 810, 70);

out\_result(num\_str, m); //Здесь мы заного добавили число в num\_str, который ранее очищали

}

else {

is\_break\_one = true;

arith\_switch(but[i].symb[0], &priority, b);

switch\_arith\_push(but[i].symb[0], m, b);

}

}

break;

}

case UNARY\_OP: {

double num = 0;

//Проверка на то, что если пользователь использовал abs

if (but[i].type\_str == "abs" && stack\_top(&b) == '-') {

draw\_template\_abs(output\_result);

delete\_sub\_unary(&priority, m, b);

//draw\_buts();

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

break;

}//Иначе пользователь используем другой арифметический знак

else if (but[i].type\_str == "abs" && (stack\_top(&b) == '+' || stack\_top(&b) == '\*' || stack\_top(&b) == '/')) {

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

break;

}

//Условие необходимо, если пользователь сразу использует унарную операцию

if (stack\_isempty(&m) == true && num\_str.length() == 0) {

switch\_unary\_func(but[i].symb[0], num, &is\_check\_error, &is\_check\_deg\_rag, output\_result, m);

if (is\_check\_error == true) {

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

break;

}

draw\_template(output\_result, i, num); //Рисуем шаблон

delete\_expression(num\_str, m);

check\_skb\_z = true;

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

}

//Обработаем случай, когда пользователь забыл ввести число, то вытаскиваем из стека число

else if (output\_result.length() != 0 && num\_str == "") {

//Когда появляется ответ на экране и пользователь хочет найти функци. от этого числа, то выолняется это условие

if (stack\_isempty(&b) == true) {

cleardevice();

draw\_buts();

output\_result.clear();

}

num = stack\_top(&m); //Достаем из стека число и передаем в качестве аргумента в функцию ниже

stack\_pop(&m);//Удаляем число

switch\_unary\_func(but[i].symb[0], num,&is\_check\_error, &is\_check\_deg\_rag, output\_result, m);

if (is\_check\_error == true) {

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

break;

}

draw\_template(output\_result, i, num); //рисуем шаблон

delete\_expression(num\_str, m);

check\_skb\_z = true;

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

}

//Иначе все нормально

else {

savestr(num\_str, m); //Сохраняем число в стек

draw\_template(output\_result, i, stack\_top(&m)); // рисуем шаблон

num = stack\_top(&m); //Достаем из стека число и передаем в качестве аргумента в функцию ниже

stack\_pop(&m);//Удаляем число

//Объявляется функция по обработке кнопки

switch\_unary\_func(but[i].symb[0], num,&is\_check\_error, &is\_check\_deg\_rag, output\_result, m);

if (is\_check\_error == true) {

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

break;

}

delete\_expression(num\_str, m);

check\_skb\_z = true;

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

}

replace\_deg\_rag(&is\_check\_deg\_rag);

break;

}

case PARENTHESIS: {

//Если есть арифметические знаки перед скобками, то

if (but[i].symb[0] == '(') {

out\_expression(output\_result, but[i].symb[0]);

priority = 0;

priority\_skb\_o += 1;

arith\_switch(but[i].symb[0], &priority, b);

switch\_arith\_push(but[i].symb[0], m, b);

clear\_bufer(mass, &j, count\_mass);

is\_break\_one = true;

}

else if (but[i].symb[0] == ')') {

check\_skb\_z = true;

//Если пользователь ни разу не нажимал ‘(‘

if (priority\_skb\_o > 0) {

//Отобразим число, а затем скобку

out\_expression\_str(output\_result, num\_str);

out\_expression(output\_result, but[i].symb[0]);

savestr(num\_str, m);

is\_singular(m, b, &priority, &priority\_skb\_o, output\_result);

//Делаем черный прямоугольник для удаление числа

setfillstyle(SOLID\_FILL, BLACK);

bar(10, 35, 810, 70);

out\_result(result\_str, m);

result\_str.clear();

if (stack\_top(&b) == '(' || stack\_top(&b) == '+' || stack\_top(&b) == '-' || stack\_top(&b) == '\*' || stack\_top(&b) == '/' || stack\_isempty(&b) == true) {

//Очищаем буфер

clear\_bufer(mass, &j, count\_mass);

is\_break\_one = true;

break;

}

}

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

}

break;

}

case DIGIT: {

//Если не было цифры 0 от пользователя

if (result\_str[0] == '0') result\_str.clear();

if (use\_arith == true) {

use\_arith = false;

num\_str.clear();

}

//Если было обработана кнопка ‘=’

if (is\_finish == true) {

is\_finish = false;

cleardevice();

num\_str.clear();

result\_str.clear();

output\_result.clear();

}

//Основная часть сохранения цифр в тип string

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

//Обязательно num\_str пустой!

num\_str = num\_str + but[i].symb[0];

clear\_interim\_solution();

break;

}

case CONSTANT: {

if (is\_check\_constant == false) {

is\_check\_constant = true;

//Показываем константное число, а потом стерем

num\_str = but[i].type\_str;

clear\_interim\_solution();

clear\_bufer(mass, &j, count\_mass);

is\_break\_one = true;

}

break;

}

case EDITION: {

clear\_bufer(mass, &j, count\_mass);

if (but[i].type\_str == "CE" && (output\_result.length() != 0 || result\_str.length() != 0 || num\_str.length() != 0)) {

clear\_global\_obj();

clear\_the\_input(m, b);

output\_result.clear();

result\_str.clear();

num\_str.clear();

cleardevice();

draw\_buts();

replace\_deg\_rag(&is\_check\_deg\_rag);

}

else if (but[i].type\_str == "delete") {

if (result\_str != "" && is\_finish == true) {

output\_result.clear();

num\_str = result\_str

result\_str.clear();

is\_finish = false;

}

else if (num\_str != "") {

num\_str.erase(num\_str.length() - 1);

}

cleardevice();

draw\_buts();

//Задаем конкретное место удаления

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 4);

outtextxy(10, 35, num\_str.c\_str());

replace\_deg\_rag(&is\_check\_deg\_rag);

}

is\_break\_one = true;

break;

}

case RESULT: {

if (num\_str == "") {

is\_break\_one = true;

break;

}

if (priority\_skb\_o > 0) {

cleardevice();

out\_error(output\_result, "Не равное количество открытых и закрытых скобок!");

draw\_buts();

replace\_deg\_rag(&is\_check\_deg\_rag);

is\_break\_one = true;

break;

}

is\_finish = true;

check\_skb\_z = false;

clear\_bufer(mass, &j, count\_mass);

//В if eсли пользователь ничего не ввел, а нажимает на '='

if (num\_str == "" && output\_result.length() == 0) {

is\_break\_one = true;

}

//Может быть так, что пользователь нажал только на 2+, а нужно польностью 2+2

//((((5+4)+3)\*2)/1)+2 -> в итоге 22 + 2, которое нужно посчитать

else if (stack\_isempty(&b) == false) {

is\_check\_binary = false; // Забываем про мат.функцию

is\_break\_one = true;

out\_expression\_str(output\_result, num\_str); // Добавляем в выражение

out\_expression(output\_result, '='); // Добавляем знак =

savestr(num\_str, m);

while (stack\_isempty(&b) == false) {

operation\_result(stack\_top(&b), output\_result, &is\_check\_del, m, b);

}

//Удаляем число

setfillstyle(SOLID\_FILL, BLACK);

bar(10, 35, 810, 70);

out\_result(result\_str, m);

replace\_deg\_rag(&is\_check\_deg\_rag);

}

//(4+(5+(6\*(4/2)))) -> в итоге получится 21

else if(stack\_isempty(&b) == true){

is\_break\_one = true;

out\_expression\_str(output\_result, num\_str); // Добавляем в выражение

out\_expression(output\_result, '='); // Добавляем знак =

if (is\_check\_binary == true) {

is\_check\_binary = false; // Забываем про мат.функцию

switch\_binary\_func(is\_check\_binary\_symb, num\_1, std::stod(num\_str), &is\_check\_error, output\_result, m);

if (is\_check\_error == true) {

is\_break\_one = true;

clear\_bufer(mass, &j, count\_mass);

break;

}

}

else {

savestr(num\_str, m);

}

draw\_buts();

bar(10, 35, 810, 70);

out\_result(result\_str, m);

replace\_deg\_rag(&is\_check\_deg\_rag);

}

break;

}

case DEGRAD:{

if (is\_check\_deg\_rag == false) {

is\_check\_deg\_rag = true;

replace\_deg\_rag(&is\_check\_deg\_rag);

clear\_bufer(mass, &j, count\_mass);

is\_break\_one = true;

}

else {

is\_check\_deg\_rag = false;

replace\_deg\_rag(&is\_check\_deg\_rag);

clear\_bufer(mass, &j, count\_mass);

is\_break\_one = true;

}

break;

}

}

}else{

continue;

}

}

}

}

А.3 Файл Button.h

#pragma once

#ifndef \_BUTTON\_H\_

#define \_BUTTON\_H\_

#include "Math\_for\_calc.h"

const int N = 45; //Размер структурного массива | 45 - количество всего кнопок

extern int is\_check\_deg\_rag; //Проверка на замену из DEG в RAG и наоборот (По умолчанию DEG)

//Вывод изображения на экран

void draw\_buts();

//Очищение двух стеков

void clear\_the\_input(my\_stack& m, stack\_char& b);

//Очищение буфера, 'буфер' хранит в себе координаты кнопки после нажатия ЛКМ на диалоговое окно

void clear\_bufer(int\* mass, int\* jndex, int\* count\_mass);

//Проверка замены типа из DEG в RAG и наоборот (По умолчанию тип DEG)

void replace\_deg\_rag(int\* is\_check\_deg\_rag);

//Функция по замене знака '-' на '+' для мат.функ. abs

void draw\_template\_abs(std::string& output\_result);

//Рисование шаблона

void draw\_template(std::string& output\_result, int index, int num);

//"Сердце" - здесь происходит процесс работы

void about\_but(int\* mass, int\* count\_mass, std::string& output\_result, std::string& result\_str, my\_stack& m, stack\_char& b);

#endif // \_BUTTON\_H\_

А.4 Файл Button\_processing.cpp

#include "Math\_for\_calc.h"

#include "graphics.h"

//Функция отображает результат вычисления после нажатия пользователем конпки '='

void out\_result(std::string& result\_str, my\_stack& m) {

//Проверка на то, что если пользователь ввёл ((())) -> результат 0

if (stack\_isempty(&m) == true) {

result\_str.clear();

result\_str += std::to\_string(0);

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 4);

outtextxy(10, 35, result\_str.c\_str());

}

else {

result\_str.clear();

result\_str += std::to\_string(stack\_top(&m));

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 4);

outtextxy(10, 35, result\_str.c\_str());

}

}

//Функция отображает каждый символ, который пользователь может нажать на кнопку

void out\_expression(std::string& output\_result, char type) {

output\_result += (type);

//Если указатель спустился ниже своего уровня, а уровень должен начинаться с 0

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR,3);

outtextxy(10, 10, output\_result.c\_str());

}

//Это функция отвечает за отображение целых строк в поле для ввода

void out\_expression\_str(std::string& output\_result, const std::string type) {

output\_result += type;

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 3);

outtextxy(10, 10, output\_result.c\_str());

}

//Это функция отвевчает за отображение ошибки на экран

void out\_error(std::string& output\_result, std::string error) {

//cleardevice();

output\_result.clear();

output\_result += error;

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 3);

outtextxy(10, 20, output\_result.c\_str());

}

//Это функция выводит промежуточное вычилениe на экран

void out\_bottom\_result\_str(std::string& result\_str, char symb) {

result\_str += symb;

settextstyle(SIMPLEX\_FONT, HORIZ\_DIR, 4);

outtextxy(10, 35, result\_str.c\_str());

}

//Удаление части выражения для исправления вида | применяется для унаных математических функциях

void delete\_expression(std::string& num\_str, my\_stack& m) {

//Здесь должны показать промежуточное вычисление

//Делаем черный прямоугольник для удаление числа

setfillstyle(SOLID\_FILL, BLACK);

bar(10, 35, 810, 70);

out\_result(num\_str, m); //Здесь мы заного добавили число в num\_str, который ранее очищали

num\_str.clear();

}

//Функция, которая определяет какую бинарную функцию пользователь нажал

void switch\_binary\_func(char type, double a, double b,int\* is\_check\_error,std::string& output\_result, my\_stack& m) {

if (type == 'B') {

if (sqrtn(a, b, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, sqrtn(a, b, output\_result));

}

}

else if (type == 'D') {

stack\_push(&m, pown(a, b));

}

else if (type == 'G') {

if (logn(a, b, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, logn(a, b, output\_result));

}

}

else if (type == 'V') {

stack\_push(&m, MOD(a, b));

}

}

//Функция, которая определяет какую унарную функцию пользователь нажал на кнопку

void switch\_unary\_func(char type, double a,int\* is\_check\_error,int\* check\_DEG\_RAG, std::string& output\_result, my\_stack& m) {

if (type == 'A') {

if (sqrt2(a, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, sqrt2(a, output\_result));

}

}

else if (type == 'C') {

stack\_push(&m, pow2(a));

}

else if (type == 'E') {

if (fact(a, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, fact(a, output\_result));

}

}

else if (type == 'F') {

if (log\_2(a, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, log\_2(a, output\_result));

}

}

else if (type == 'H') {

if (ln(a, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, ln(a, output\_result));

}

}

else if (type == 'I') {

if (lg(a, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, lg(a, output\_result));

}

}

else if (type == 'K') {

stack\_push(&m, pow10n(a));

}

else if (type == 'L') {

stack\_push(&m, sinus(a, check\_DEG\_RAG));

}

else if (type == 'M') {

stack\_push(&m, cosinus(a, check\_DEG\_RAG));

}

else if (type == 'N') {

if (tangent(a, check\_DEG\_RAG, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, tangent(a, check\_DEG\_RAG, output\_result));

}

}

else if (type == 'O') {

if (cotangent(a, check\_DEG\_RAG, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, cotangent(a, check\_DEG\_RAG, output\_result));

}

}

else if (type == 'P') { // ???????????????????

if (arcsinus(a, check\_DEG\_RAG, output\_result) == false) {

stack\_push(&m, arcsinus(a, check\_DEG\_RAG, output\_result));

}

else {

\*is\_check\_error = true;

}

}

else if (type == 'Q') {

if (arccosinus(a, check\_DEG\_RAG, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, arccosinus(a, check\_DEG\_RAG, output\_result));

}

}

else if (type == 'R') {

stack\_push(&m, arctangent(a, check\_DEG\_RAG));

}

else if (type == 'S') {

stack\_push(&m, arccotangent(a, check\_DEG\_RAG));

}

else if (type == 'T') {

stack\_push(&m, modul(a));

}

else if (type == 'W') {

stack\_push(&m, exponenta(a));

}

else if (type == 'X') {

if (delete\_1(a, output\_result) == false) {

\*is\_check\_error = true;

}

else {

stack\_push(&m, delete\_1(a, output\_result));

}

}

}

//Функция определяет приоритет вычисления знака в выражении

void arith\_switch(char type, int\* priority, stack\_char& b) {

if (type == '+') {

\*priority = 1;

}

else if (type == '-') {

\*priority = 1;

}

else if (type == '\*') {

\*priority = 2;

}

else if (type == '/') {

\*priority = 2;

}

else if (type == '(' || type == ')') {

\*priority = 0;

}

else if ((65 <= static\_cast<int>(stack\_top(&b)) && static\_cast<int>(stack\_top(&b)) <= 90)) {

\*priority = 0;

}

else if (stack\_isempty(&b) == true) {

\*priority = 0;

}

}

//Это функция заносит арифметический знак в стек

void switch\_arith\_push(char type, my\_stack& m, stack\_char& b) {

if (type == '+') {

stack\_push(&b, type);

}

else if (type == '-') {

stack\_push(&b, type);

}

else if (type == '\*') {

stack\_push(&b, type);

}

else if (type == '/') {

stack\_push(&b, type);

}

else if (type == '(' || type == ')') {

stack\_push(&b, type);

}

}

//Функция, которая проводит арифметическую операцию, а затем проверяет наличие знака в глубине стека

void merging\_two\_numbers(int\* priority, my\_stack& m, stack\_char& b, std::string& output\_result) {

double n1 = 0.0, n2 = 0.0;

char symb;

n1 = stack\_top(&m);

stack\_pop(&m);

if (stack\_isempty(&m) == true) {

n2 = 0.0;

}

else {

n2 = stack\_top(&m);

stack\_pop(&m);

}

if (stack\_top(&b) == '+') {

stack\_push(&m, sum\_digits(n2, n1));

stack\_pop(&b);

//Далее смотрим в арифметическом стеке символ, если он есть, то берем и смотрим на приоритет, если нет, то приоритет 0

symb = stack\_top(&b);

arith\_switch(symb, priority, b);

}

else if (stack\_top(&b) == '-') {

stack\_push(&m, sub\_digits(n2, n1));

stack\_pop(&b);

symb = stack\_top(&b);

arith\_switch(symb, priority, b);

}

else if (stack\_top(&b) == '\*') {

stack\_push(&m, mul\_digits(n2, n1));

stack\_pop(&b);

symb = stack\_top(&b);

arith\_switch(symb, priority, b);

}

else if (stack\_top(&b) == '/') {

stack\_push(&m, div\_digits(n2, n1, output\_result));

stack\_pop(&b);

symb = stack\_top(&b);

arith\_switch(symb, priority, b);

}

//Ниже может быть нужно исправить случаи

else if (stack\_isempty(&b) == true) {

\*priority = 0;

}

}

//Это функция выполняет математические операции (+,-,\*,/) с двумя аргументами типа double

void operation\_result(char type\_symbol,std::string& output\_result, int\* is\_check\_del, my\_stack& m, stack\_char& b) {

double n1 = 0.0, n2 = 0.0;

n1 = stack\_top(&m);

stack\_pop(&m);

if (stack\_isempty(&m)) {

n2 = 0.0;

}

else {

n2 = stack\_top(&m);

stack\_pop(&m);

}

if (type\_symbol == '+') {

stack\_push(&m, sum\_digits(n2, n1));

stack\_pop(&b);

}

else if (type\_symbol == '-') {

stack\_push(&m, sub\_digits(n2, n1));

stack\_pop(&b);

}

else if (type\_symbol == '\*') {

stack\_push(&m, mul\_digits(n2, n1));

stack\_pop(&b);

}

else if (type\_symbol == '/') {

if (div\_digits(n2, n1, output\_result) != false) { // Если деление на ноль

stack\_push(&m, div\_digits(n2, n1, output\_result));

}

else {

n2 = 0.0;

n1 = 0.0;

out\_error(output\_result, "Деление на 0 запрещено!");

\*is\_check\_del = true;

}

stack\_pop(&b);

}

}

//Функция отвечает за явный перевод чисел из типа string в тип double для дальнейшего вычисления

void savestr(std::string& num\_str,my\_stack& m) {

if (num\_str != "") {

//Переводим явным образом из типа str в тип double

stack\_push(&m, std::stod(num\_str));

//Очистка строки

num\_str.clear();

}

}

//Это функция применяется только тогда, когда пользователь пользуется скобками (выражения со скобками)

void is\_singular(my\_stack& m, stack\_char& b,int\* priority, int\* priority\_skb\_o, std::string& output\_result) {

int its\_true = false;

char syb;

while (its\_true == false) {

if (stack\_top(&b) != '(') {

merging\_two\_numbers(priority, m, b, output\_result);

}

else if (stack\_top(&b) == '(') {

its\_true = true;

stack\_pop(&b); // Удаляем '('

syb = stack\_top(&b);

arith\_switch(syb, priority, b);

\*priority\_skb\_o -= 1;

}

}

}

//Это функция выполняет замену знака '-', на '+' | используется для функции abs

void delete\_sub\_unary(int\* priority, my\_stack& m, stack\_char& b) {

//Процесс

char plus = '+';

stack\_pop(&b); //Должно быть удаление '-'

arith\_switch(plus, priority, b);

switch\_arith\_push(plus, m, b);

}

//Это функция выполняет замену знака '-', на '+'

void delete\_sub(int\* priority, my\_stack& m, stack\_char& b) {

//Как работает функция: 5-3 -> 5+(-3), где (-3) - это уже double, а не отдельные два вида '-' - char и '3' - int

double n = 0.0;

char sub = stack\_top(&b); //Должно быть '-'

char plus = '+';

//Процесс

stack\_pop(&b); //Должно быть удаление '-'

arith\_switch(plus, priority, b);

switch\_arith\_push(plus, m, b);

n = n - stack\_top(&m);

stack\_pop(&m);

stack\_push(&m, n); //Заносим то же число, но уже с отрицанием

}

А.5 Файл Button.h

#pragma once

#ifndef \_BUTTON\_PROCESSING\_H\_

#define \_BUTTON\_PROCESSING\_H\_

//Вывод выражения или ошибки на экран

void out\_result(std::string& output\_result, my\_stack& m);

void out\_expression(std::string& output\_result, char type);

void out\_error(std::string& output\_result, std::string error);

void out\_expression\_str(std::string& output\_result, const std::string type);

//Вывод промежуточного вычиления на экран

void out\_bottom\_result\_str(std::string& result\_str,char symb);

//Удаление части выражения для исправления вида | применяется для унаных математических функциях

void delete\_expression(std::string& num\_str, my\_stack& m);

//Определение мат. функции

void switch\_binary\_func(char type, double a, double b, int\* is\_check\_error, std::string& output\_result, my\_stack& m);

void switch\_unary\_func(char type, double a, int\* is\_check\_error, int\* check\_DEG\_RAG, std::string& output\_result, my\_stack& m);

//Определение арифметического знака и добавление его в арифметический стек

void arith\_switch(char type, int\* priority, stack\_char& b);

void switch\_arith\_push(char type, my\_stack& m, stack\_char& b);

//После нажатия на кнопку '=', происходит процесс вычисления выражения

void operation\_result(char type\_symbol, std::string& output\_result, int\* is\_check\_del, my\_stack& m, stack\_char& b);

//Процесс вычисления

void is\_singular(my\_stack& m, stack\_char& b, int\* priority, int\* priority\_skb\_o, std::string& output\_result);

void merging\_two\_numbers(int\* priority, my\_stack& m, stack\_char& b, std::string& output\_result);

//Сохранение числа из строки

void savestr(std::string& num\_str, my\_stack& m);

//Замена знака '-' на '+'

void delete\_sub\_unary(int\* priority, my\_stack& m, stack\_char& b);

void delete\_sub(int\* priority, my\_stack& m, stack\_char& b);

#endif // \_BUTTON\_PROCESSING\_H\_

А.6 Файл Math\_for\_calc.cpp

#include <iostream>

#include <cmath>

#include "Math\_for\_calc.h"

#include "Button\_processing.h"

//--------stack\_digit--------

//Инициализация численного стека

void stack\_init(my\_stack\* a) {

a->top = -1;

}

//Добавление значения в численный стек

void stack\_push(my\_stack\* a, double v) {

if (a->top == 1000) {

return;

}

a->top += 1;

a->mass[a->top] = v;

}

//Удаление верхнего значения из численного стека

void stack\_pop(my\_stack\* a) {

a->mass[a->top] = 0;

a->top -= 1;

}

//Получение верхнего значения из численного стека

double stack\_top(my\_stack\* a) {

return a->mass[a->top];

}

//Проверка на пустоту численного стека

double stack\_isempty(my\_stack\* a) { // проверить, что стек пуст

if (a->top == -1) {

return true;

}

else {

return false;

}

}

//----------------stack\_digit----------------

//----------------stack\_char----------------

// Инициализация арифметичексого стека

void stack\_init(stack\_char\* b) {

b->top = -1;

}

//Добавление значения в арифметический стек

void stack\_push(stack\_char\* b, char v) {

if (b->top == 1000) {

return;

}

b->top += 1;

b->mass\_char[b->top] = v;

}

//Удаление верхнего значения из арифметического стека

void stack\_pop(stack\_char\* b) {

b->mass\_char[b->top] = NULL;

b->top -= 1;

}

//Получение верхнего значения из арифметического стека

int stack\_top(stack\_char\* b) {

return b->mass\_char[b->top];

}

//Проверка на пустоту арифметического стека

int stack\_isempty(stack\_char\* b) {

if (b->top == -1) {

return true;

}

else {

return false;

}

//return b->top == -1;

}

//----------------stack\_char----------------

//--------------Математические операции--------------

double sum\_digits(double a, double b) {

return a + b;

}

double sub\_digits(double a, double b) {

return a - b;

}

double mul\_digits(double a, double b) {

return a \* b;

}

double div\_digits(double a, double b, std::string& output\_result) {

if (b == 0.0) {

out\_error(output\_result, "Деление на 0 запрещено!");

return NULL;

}

else {

return a / b;

}

}

//--------------Математические операции--------------

//--------------Математические функции--------------

//Функция вычисления факториала от числа a

double fact(double a, std::string& output\_result) {

double result = 1;

if (a < 0) {

out\_error(output\_result, "Неверный ввод. Диапазон начинается с нуля.");

return false;

}

for (int i = 1; i <= a; ++i) {

result \*= i;

}

return result;

}

//Функция синуса

double sinus(double a, int\* check\_DEG\_RAG) {

if (\*check\_DEG\_RAG == true) {

return sin(a); //Из радианов в градусы получаем значение

}

else {

return sin((a) \* (PI / 180.0)); //Из градусов в радианы получаем значение

}

}

//Функция косинуса

double cosinus(double a, int\* check\_DEG\_RAG) {

if (\*check\_DEG\_RAG == true) {

return cos(a);//Из радианов в градусы получаем значение

}

else {

return cos((a) \* (PI / 180.0)); //Из градусов в радианы получаем значение

}

}

//Функция тангенса

double tangent(double a, int\* check\_DEG\_RAG, std::string& output\_result) {

double num = 0;

//Если дано в градусах

if (\*check\_DEG\_RAG != true) {

for (int i = 1; i <= 10; i++) {

if (a == 90 \* i || a == 270 \* i) {

out\_error(output\_result, "Неверный ввод. Тангенс не существует");

return false;

}

else {

//Для быстрого выведения из цикла

if ((a) < 90 \* i || (a) < 270 \* i) break;

//т.е. число больше проверки, нужно добрать и проверить

continue;

}

}

return tan((a) \* (PI / 180.0)); //Из градусов к радианам и получаем число

}

else {

num = (PI / 2);

for (int i = 0; i < 10; i++) {

if (a == num + PI \* i || a == 3 \* num + PI \* i) {

out\_error(output\_result, "Неверный ввод. Тангенс не существует");

return false;

}

else {

if (a < num + PI \* i || a < 3 \* num + PI \* i) break;

continue;

}

}

return tan(a);

}

}

//Фунцкция котангенса

double cotangent(double a, int\* check\_DEG\_RAG, std::string& output\_result) {

//Если задана в градусах

if (\*check\_DEG\_RAG != true) {

for (int i = 0; i < 10; i++) {

if (a == 0 + 180 \* i || a == 180 + 180 \* i) {

out\_error(output\_result, "Неверный ввод. Котангенс не существует");

return false;

}

else {

//Для быстрого выведения из цикла

if (a < 0 + 180 \* i || a < 180 + 180 \* i) break;

//т.е. число больше проверки, нужно добрать и проверить

continue;

}

}

return tan((a) \* (PI / 180.0)); //Из градусов к радианам и получаем число

}

else {

for (int i = 0; i < 10; i++) {

if (a == PI \* i || a == PI + PI \* i) {

out\_error(output\_result, "Неверный ввод. Котангенс не существует");

return false;

}

else {

//Для быстрого выведения из цикла

if (a < 0 + 180 \* i || a < 180 + 180 \* i) break;

//т.е. число больше проверки, нужно добрать и проверить

continue;

}

}

}

}

//Функция обратного синуса

double arcsinus(double a, int\* check\_DEG\_RAG, std::string& output\_result) {

if (-1 <= a && a <= 1) {

return asin(a);

}

out\_error(output\_result, "Неверный ввод. Диапазон должен быть от -1 до 1");

return false;

}

//Функция обратного косинуа

double arccosinus(double a, int\* check\_DEG\_RAG, std::string& output\_result) {

if (-1 <= a && a <= 1) {

return acos(a);

}

out\_error(output\_result, "Неверный ввод. Диапазон должен быть от -1 до 1");

return false;

}

//Функция обратного тангенса

double arctangent(double a,int\* check\_DEG\_RAG) {

double num = 0;

//Если передано в градусах, то переделаем в радианах

if (\*check\_DEG\_RAG != true) num = a \* (PI / 180.0);

else num = a; //Передано в радианах

return atan(num);

}

//Функция обратного котангенса

double arccotangent(double a, int\* check\_DEG\_RAG) {

double num = 0;

//Если передано в градусах, то переделаем в радианах

if (\*check\_DEG\_RAG != true) num = a \* (PI / 180.0);

else num = a; //Передано в радианах

return atan(1.0 / num);

}

//Функция логарифма с основанием 2

double log\_2(double a, std::string& output\_result) {

if (a <= 0) {

out\_error(output\_result, "Неверный ввод. Диапазон начинатся от 0");

return false;

}

return log2(a);

}

//Функция десятичного логарифма

double lg(double a, std::string& output\_result) {

if (a <= 0) {

out\_error(output\_result, "Неверный ввод. Диапазон начинатся от 0");

return false;

}

return log10(a);

}

Функция натурального логарифма

double ln(double a, std::string& output\_result) {

if (a <= 0) {

out\_error(output\_result, "Неверный ввод. Диапазон начинатся от 0");

return false;

}

return log(a);

}

//Функиця вычисления логарифма с основанием b и числом a

double logn(double a, double b, std::string& output\_result) {

if (a < 0 || b <= 0 || b == 1.0) {

out\_error(output\_result, "Неверный ввод. Проверьте диапазон аргументов.");

return false;

}

return log(b) / log(a);

}

//Функция вычисления числа а степени 2.0

double pow2(double a) {

return pow(a, 2.0);

}

//Функция вычисления числа a от степени n

double pown(double a, double b) {

return pow(a, b);

}

//Функция вычисления числа 10 со степенью n, т.е. 10 ^ n

double pow10n(double a) {

return pow(10, a);

}

//Функция квадратного корня

double sqrt2(double a, std::string& output\_result) {

if (a < 0) {

out\_error(output\_result, "Неверный ввод. Диапазон начинатся от 0");

return false;

}

return sqrt(a);

}

//Функция sqrtn - функция корня с степенью b

double sqrtn(double a, double b, std::string& output\_result) {

double num = fmod(b, 2.0);

//Если остаток равен 0, то значит степень четная

if (isnan(num) == NAN && a < 0) {

out\_error(output\_result, "Неверный ввод. Диапазон начинатся от 0");

return false;

}

else if (isnan(num) == NAN && a >= 0) {

return pow(b, 1.0 / a);

}

else if (isnan(num) != NAN) {

return pow(b, 1.0 / a);

}

}

//Функция для вычисления остатка от двух чисел

double MOD(double a, double b) {

return fmod(a, b);

}

//Функция для вычисления абсолютного значения числа

double modul(double a) {

return fabs(a);

}

//Функция для вычисления выражения 1/x

double delete\_1(double a, std::string& output\_result) {

if (a == 0.0) { // По идеи, что вещественное число b должно быть равно 0.0

out\_error(output\_result, "Деление на 0 запрещено!");

return false;

}

else {

return 1.0 / a;

}

}

//Функция для вычисления экспоненты

double exponenta(double a) {

return exp(a);

}

А.7 Файл Math\_for\_calc.h

#pragma once

#ifndef \_MATH\_FOR\_CALC\_H\_

#define \_MATH\_FOR\_CALC\_H\_

//Костантные значения числа PI - Пифагора и E - Эйлера

const double PI = 3.14159265358979323846;

const double E = 2.718281828459045;

//----Стек, в котором хранится числа типа double----

typedef struct{

int top;

double mass[1001];

}my\_stack;

//----Стек, в котором хранится числа типа double----

//----Функции стека c типом double----

void stack\_init(my\_stack\* a);

void stack\_push(my\_stack\* a, double v);

void stack\_pop(my\_stack\* a);

double stack\_top(my\_stack\* a);

double stack\_isempty(my\_stack\* a);

//----Функции стека c типом double----

//-------stack\_char-------

typedef struct {

int top;

char mass\_char[1001];

}stack\_char;

//-------stack\_char-------

//-------Функции stack\_char-------

void stack\_init(stack\_char\* b);

void stack\_push(stack\_char\* b, char v);

void stack\_pop(stack\_char\* b);

int stack\_top(stack\_char\* b);

int stack\_isempty(stack\_char\* b);

//-------Функции stack\_char-------

double sum\_digits(double a, double b);

double sub\_digits(double a, double b);

double mul\_digits(double a, double b);

double div\_digits(double a, double b, std::string& output\_result);

//--------------Математические функции--------------

//Рядом с функциями обозначены на латинице в алфавитном порядке их краткое обозначение

//Нужно для сохранения этих мат.функций в арифметическом стеке

double sqrt2(double a, std::string& output\_result); // A

double sqrtn(double a, double b, std::string& output\_result); //B

double pow2(double a); //C

double pown(double a, double b); //D

double fact(double a, std::string& output\_result); // E

double log\_2(double a, std::string& output\_result); //F

double logn(double a, double b, std::string& output\_result); // G

double ln(double a, std::string& output\_result); //H

double lg(double a, std::string& output\_result); // I

double pow10n(double a); //K

double sinus(double a, int\* check\_DEG\_RAG); //L

double cosinus(double a, int\* check\_DEG\_RAG); //M

double tangent(double a, int\* check\_DEG\_RAG, std::string& output\_result); //N

double cotangent(double a, int\* check\_DEG\_RAG, std::string& output\_result); //O

double arcsinus(double a, int\* check\_DEG\_RAG, std::string& output\_result); //P

double arccosinus(double a, int\* check\_DEG\_RAG, std::string& output\_result); //Q

double arctangent(double a, int\* check\_DEG\_RAG); //R

double arccotangent(double a, int\* check\_DEG\_RAG); //S

double modul(double a); //T

double MOD(double a, double b); //V

double exponenta(double a); //W

double delete\_1(double a, std::string& output\_result); //X

//--------------Математические функции--------------

#endif // \_MATH\_FOR\_CALC\_H\_