

МИНИСТЕРСТВО ОБРАЗОВАНИЯ ОРЕНБУРГСКОЙ ОБЛАСТИ
Государственное автономное профессиональное образовательное
учреждение
**«ОРЕНБУРГСКИЙ КОЛЛЕДЖ ЭКОНОМИКИ И ИНФОРМАТИКИ»
(ГАПОУ ОКЭИ)**

Практическая работа

ОКЭИ 09.02.07 7026 07 У

Тема: *«Разработка оконного приложения. Добавление функционала на форму»*

Выполнил: Гадияев Ислам Ильгамович

Оренбург 2026

Содержание

1 Цель работы 3

2 Скриншот структуры готового проекта..... 3

3 Листинг программы 3

4 Результаты работы программы..... 6

					ОКЭИ 09.02.07 7026 07 У										
Изм.	Лист	№ докум.	Подп.	Дата											
Разраб.		Гадияев И. И.				Практическая работа					Лит.	Лист	Листов		
											У		2		
											Отделение программирования				

1 Цель работы

Задание: в раннее разработанное приложение(пз_14-15)добавить статическую библиотеку LIB, разработанную в ПЗ 12. добавить элементы управления: кнопки, поля ввода и вывода текста. На кнопки привязать функционал библиотеки по вычислению значений. Реализовать считывание из полей ввода и запись результата в поле вывода

2 Скриншот структуры готового проекта

На рисунке 1 представлен скриншот структуры готового проекта.

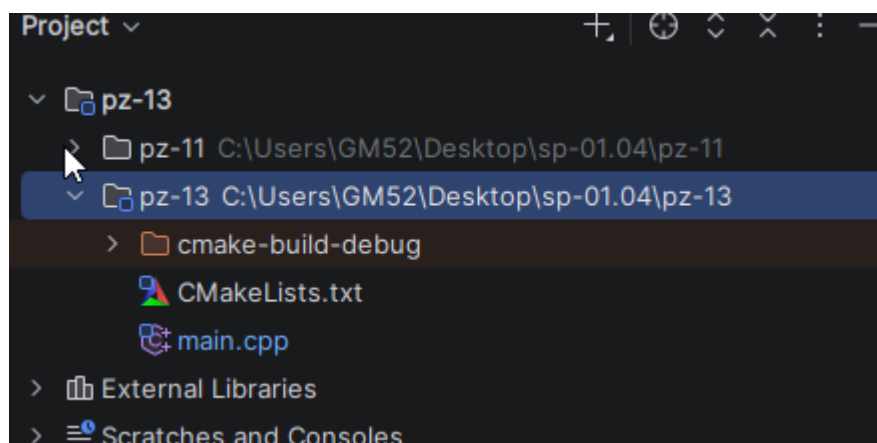


Рисунок 1 – Структура проекта

3 Листинг программы

Листинг файла main.cpp представлен ниже.

```
#define STRICT
#define WIN32_LEAN_AND_MEAN

#include <windows.h>
#include <stdio>
#include <stdexcept>
#include "Mathlib.h"

constexpr int ID_EDIT_X      = 1001;
constexpr int ID_EDIT_Y      = 1002;
constexpr int ID_EDIT_N      = 1003;
constexpr int ID_EDIT_RESULT = 1004;

constexpr int ID_BTN_ADD      = 2001;
constexpr int ID_BTN_SUB      = 2002;
constexpr int ID_BTN_MUL      = 2003;
constexpr int ID_BTN_DIV      = 2004;
constexpr int ID_BTN_POW      = 2005;
constexpr int ID_BTN_SQRT     = 2006;
```

```

LRESULT CALLBACK WindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

static HWND hEditX, hEditY, hEditN, hEditResult;

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int
nCmdShow)
{
    const char* CLASS_NAME = "MyWindowsClass";

    WNDCLASSEX wc = {};
    wc.cbSize        = sizeof(WNDCLASSEX);
    wc.style          = CS_HREDRAW | CS_VREDRAW | CS_DBLCLKS;
    wc.lpfnWndProc    = WindowProc;
    wc.hInstance      = hInstance;
    wc.hIcon          = LoadIconA(nullptr, IDI_APPLICATION);
    wc.hCursor        = LoadCursorA(nullptr, IDC_ARROW);
    wc.hbrBackground  = (HBRUSH)GetStockObject(WHITE_BRUSH);
    wc.lpszClassName  = CLASS_NAME;
    wc.hIconSm        = LoadIconA(nullptr, IDI_APPLICATION);

    if (!RegisterClassExA(&wc)) {
        MessageBoxA(nullptr, "Register failed", "Error", MB_OK);
        return 0;
    }

    HWND hWnd = CreateWindowExA(
        0,
        CLASS_NAME,
        "naaa",
        WS_OVERLAPPEDWINDOW,
        CW_USEDEFAULT, CW_USEDEFAULT,
        600, 400,
        nullptr, nullptr, hInstance, nullptr
    );

    if (!hWnd) {
        MessageBoxA(nullptr, "CreateWindow failed", "Error", MB_OK);
        return 0;
    }

    ShowWindow(hWnd, nCmdShow);
    UpdateWindow(hWnd);

    MSG msg = {};
    while (GetMessageA(&msg, nullptr, 0, 0)) {
        TranslateMessage(&msg);
        DispatchMessageA(&msg);
    }

    return (int)msg.wParam;
}

LRESULT CALLBACK WindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    static HBRUSH currentBrush = (HBRUSH)GetStockObject(WHITE_BRUSH);

    switch (uMsg) {
        case WM_CREATE:
        {
            CreateWindowA("STATIC", "X:", WS_VISIBLE | WS_CHILD | SS_RIGHT,
                20, 20, 100, 20, hWnd, nullptr, nullptr, nullptr);
            CreateWindowA("STATIC", "Y:", WS_VISIBLE | WS_CHILD | SS_RIGHT,
                20, 60, 180, 20, hWnd, nullptr, nullptr, nullptr);
            CreateWindowA("STATIC", "Result:", WS_VISIBLE | WS_CHILD | SS_RIGHT,
                20, 140, 100, 20, hWnd, nullptr, nullptr, nullptr);

            hEditX = CreateWindowExA(WS_EX_CLIENTEDGE, "EDIT", "", WS_VISIBLE |
WS_CHILD | WS_BORDER | ES_AUTOHSCROLL,

```

```

210, 20, 200, 25, hWnd, (HMENU)ID_EDIT_X,
nullptr, nullptr);
    hEditY = CreateWindowExA(WS_EX_CLIENTEDGE, "EDIT", "", WS_VISIBLE |
WS_CHILD | WS_BORDER | ES_AUTOHSCROLL,
210, 60, 200, 25, hWnd, (HMENU)ID_EDIT_Y,
nullptr, nullptr);
    hEditResult = CreateWindowExA(WS_EX_CLIENTEDGE, "EDIT", "", WS_VISIBLE |
WS_CHILD | WS_BORDER | ES_READONLY,
210, 140, 200, 25, hWnd,
(HMENU)ID_EDIT_RESULT, nullptr, nullptr);

    CreateWindowA("BUTTON", "+", WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
450, 20, 100, 30, hWnd, (HMENU)ID_BTN_ADD, nullptr,
nullptr);
    CreateWindowA("BUTTON", "-", WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
450, 60, 100, 30, hWnd, (HMENU)ID_BTN_SUB, nullptr,
nullptr);
    CreateWindowA("BUTTON", "*", WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
450, 100, 100, 30, hWnd, (HMENU)ID_BTN_MUL, nullptr,
nullptr);
    CreateWindowA("BUTTON", "/", WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
450, 140, 100, 30, hWnd, (HMENU)ID_BTN_DIV, nullptr,
nullptr);

    return 0;
}

case WM_COMMAND:
{
    if (HIWORD(wParam) == BN_CLICKED) {
        char bufX[64] = {}, bufY[64] = {}, bufN[64] = {};
        char resBuf[128] = "Error";

        GetWindowTextA(hEditX, bufX, sizeof(bufX));
        GetWindowTextA(hEditY, bufY, sizeof(bufY));
        GetWindowTextA(hEditN, bufN, sizeof(bufN));

        double x = atof(bufX);
        double y = atof(bufY);
        int n = atoi(bufN);

        try {
            double result = 0.0;
            int id = LOWORD(wParam);

            if (id == ID_BTN_ADD) result = add(x, y);
            else if (id == ID_BTN_SUB) result = subtract(x, y);
            else if (id == ID_BTN_MUL) result = multiply(x, y);
            else if (id == ID_BTN_DIV) result = divide(x, y);
            else if (id == ID_BTN_POW) result = pow(x, n);
            else if (id == ID_BTN_SQRT) result = sqrt(x, n);

            sprintf(resBuf, "%.6g", result);
        } catch (const std::exception& e) {
            strcpy(resBuf, "Error");
        } catch (...) {
            strcpy(resBuf, "Error");
        }

        SetWindowTextA(hEditResult, resBuf);
    }
    return 0;
}

case WM_ACTIVATE:
{
    HBRUSH newBrush = (LOWORD(wParam) == WA_INACTIVE)
        ? CreateSolidBrush(RGB(100, 100, 100))
        : CreateSolidBrush(RGB(0, 255, 0));
}

```

```

        if (currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
            currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
            currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
            DeleteObject(currentBrush);
        }

        currentBrush = newBrush;
        SetClassLongPtrA(hWnd, GCLP_HBRBACKGROUND, (LONG_PTR)newBrush);
        InvalidateRect(hWnd, nullptr, TRUE);
        return 0;
    }

case WM_MOVE:
{
    HBRUSH newBrush = CreateSolidBrush(RGB(0, 0, 255));
    if (currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
        currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
        currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
        DeleteObject(currentBrush);
    }
    currentBrush = newBrush;
    SetClassLongPtrA(hWnd, GCLP_HBRBACKGROUND, (LONG_PTR)newBrush);
    InvalidateRect(hWnd, nullptr, TRUE);
    return 0;
}

case WM_RBUTTONDOWNBLCLK:
{
    HBRUSH newBrush = CreateSolidBrush(RGB(255, 0, 0));
    if (currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
        currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
        currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
        DeleteObject(currentBrush);
    }
    currentBrush = newBrush;
    SetClassLongPtrA(hWnd, GCLP_HBRBACKGROUND, (LONG_PTR)newBrush);
    InvalidateRect(hWnd, nullptr, TRUE);
    return 0;
}

case WM_DESTROY:
{
    if (currentBrush != (HBRUSH)GetStockObject(WHITE_BRUSH) &&
        currentBrush != (HBRUSH)GetStockObject(BLACK_BRUSH) &&
        currentBrush != (HBRUSH)GetStockObject(GRAY_BRUSH)) {
        DeleteObject(currentBrush);
    }
    PostQuitMessage(0);
    return 0;
}

default:
    return DefWindowProcA(hWnd, uMsg, wParam, lParam);
}
}

```

4 Результаты работы программы

Результат выполнения программы показан на рисунке 2.

					ОКЭИ 09.02.07 7026 07 У	Лист
Изм.	Лист	№ докум.	Подп.	Дата		6

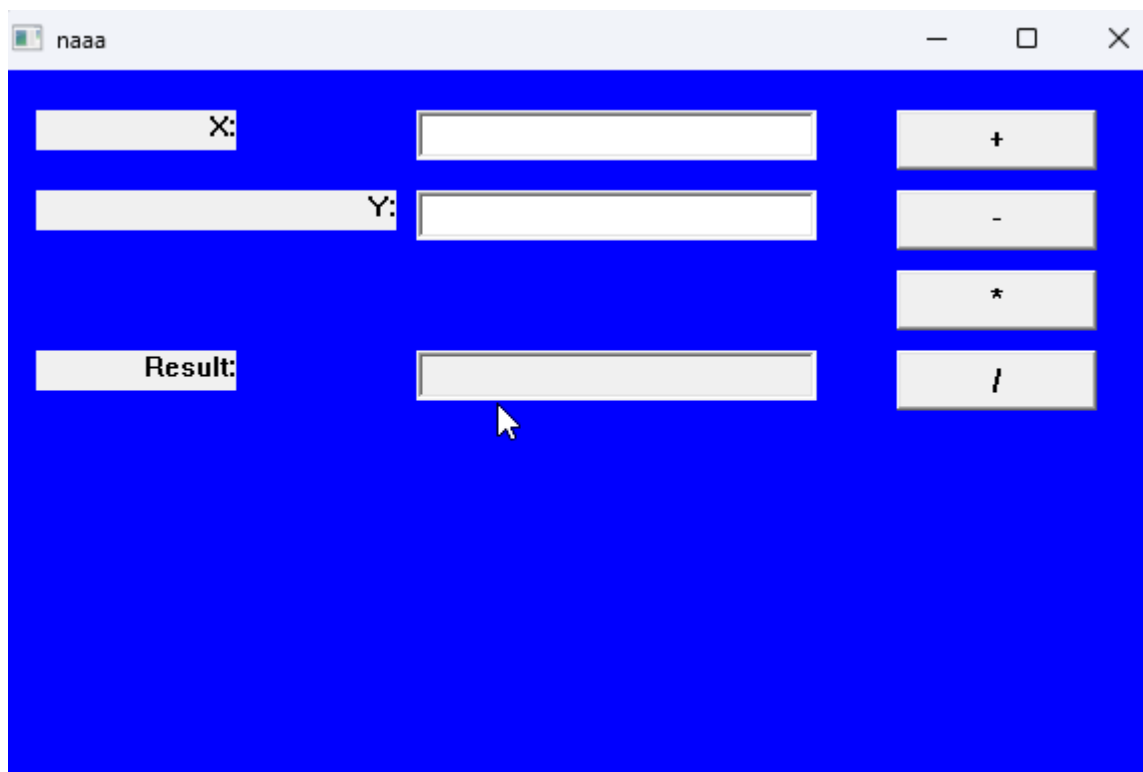


Рисунок 2 – Результат работы программы