Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»

Факультет компьютерных технологий

Кафедра ПУРИС

Лабораторная работа №1

по дисциплине «Компьютерная графика»

Студент группы 0ВТб-1 Н.Д. Малышев

Преподаватель В.А. Тихомиров

2023

1 Разработка простейшего приложения

Вывести график функции, согласно формуле на рисунке 1.1.

$$y = \frac{2x - 1}{x + 2}$$

Рисунок 1.1 – Функция

В листинге 1.1 показан код, который реализует простое графическое приложение с использованием WinAPI для отображения графика математической функции y = (2x - 1) / (x + 2) и дополнительных элементов, таких как оси, сетка и текст.

Листинг 1.1 – Lab_1.cpp

```
#include "framework.h"
#include "Lab_1.h"
#include <windows.h>
#define _USE_MATH_DEFINES
#include <math.h>
#include <cmath>

double pixelPerUnit = 1.0;

// Глобальные переменные
HINSTANCE hInst;
LPCTSTR szClassName = L"WinAPICosineGraph"; // Имя класса окна
int screenWidth = 900; // Ширина окна
int screenHeight = 950; // Высота окна
double scale = 1.0; // Масштаб графика
double offsetX = 0.0;
double offsetY = 0.0;
const double epsilon = 0.0000001;
const double pi = 3.14159265;
const double x_start = -2 * pi;
const double x_end = 2 * pi;
const double dlt = 30;


// Прототипы функций
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam,
LPARAM lParam);
```

```
void DrawGraph(HDC hdc, RECT rectClient);
void DrawAxis(HDC hdc, RECT rectClient);
void DrawGrid(HDC hdc, RECT rectClient);
void DrawText(HDC hdc, RECT rectClient);

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
LPSTR lpCmdLine, int nCmdShow)
{
    hInst = hInstance;
    WNDCLASSEX wcex;

    // Заполнение структуры wcex данными о классе окна
    wcex.cbSize = sizeof(WNDCLASSEX);
    wcex.style = CS_HREDRAW | CS_VREDRAW;
    wcex.lpfnWndProc = WndProc; // Указатель на оконную
процедуру
    wcex.cbClsExtra = 0;
    wcex.cbWndExtra = 0;
    wcex.hInstance = hInstance;
    wcex.hIcon = LoadIcon(NULL, IDI_APPLICATION);
    wcex.hCursor = LoadCursor(NULL, IDC_ARROW);
    wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1);
    wcex.lpszMenuName = NULL;
    wcex.lpszClassName = szClassName;
    wcex.hIconSm = LoadIcon(NULL, IDI_APPLICATION);

    // Регистрация класса окна
    if (!RegisterClassEx(&wcex))
    {
        MessageBox(NULL, L"Failed to register the window
class.", L"Error", MB_ICONERROR);
        return 1;
    }

    // Создание окна
    HWND hWnd = CreateWindow(szClassName, L"График функции",
WS_OVERLAPPEDWINDOW & ~WS_SIZEBOX,
        CW_USEDEFAULT, CW_USEDEFAULT, 950, 900, NULL, NULL,
hInstance, NULL);

    if (!hWnd)
    {
        MessageBox(NULL, L"Failed to create the window.",
L"Error", MB_ICONERROR);
        return 1;
    }

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

    MSG msg;
    while (GetMessage(&msg, NULL, 0, 0))
    {
```

```
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }

    return (int)msg.wParam;
}

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam,
LPARAM lParam)
{
    switch (message)
    {
    case WM_PAINT:
    {
        PAINTSTRUCT ps;
        HDC hdc = BeginPaint(hWnd, &ps);
        RECT clientRect; // Создайте переменную для хранения
прямоугольника клиентской области
        GetClientRect(hWnd, &clientRect); // Получите
прямоугольник клиентской области окна
        DrawGraph(hdc, clientRect); // Передайте его в функцию
DrawGraph
        EndPaint(hWnd, &ps);
        break;
    }
    case WM_SIZE:
    {
        // Обработка изменения размеров окна
        screenWidth = LOWORD(lParam);
        screenHeight = HIWORD(lParam);
        InvalidateRect(hWnd, NULL, TRUE);
        break;
    }
    case WM_CLOSE:
        DestroyWindow(hWnd);
        break;
    case WM_DESTROY:
        PostQuitMessage(0);
        break;
    default:
        return DefWindowProc(hWnd, message, wParam, lParam);
    }
    return 0;
}

double f(double x) {
    return (2 * x - 1) / (x + 2);
}

// Функция для рисования графика
void DrawGraph(HDC hdc, RECT rectClient) {

    HPEN penGraph = CreatePen(PS_SOLID, 2, RGB(255, 0, 0));
```

```
    HGDIOBJ gdiOld = SelectObject(hdc, penGraph);
    HPEN hPenAxis = CreatePen(PS_SOLID, 2, RGB(0, 0, 0)); //
Перо для осей
    HPEN hPenGrid = CreatePen(PS_DOT, 1, RGB(0, 0, 0)); // Перо
для сетки
    HFONT hFont = CreateFont(16, 0, 0, 0, FW_NORMAL, FALSE,
FALSE, FALSE, DEFAULT_CHARSET, OUT_OUTLINE_PRECIS,
        CLIP_DEFAULT_PRECIS, CLEARTYPE_QUALITY, DEFAULT_PITCH |
FF_DONTCARE, L"Arial"); // Шрифт для текста

    double x_current = x_start;
    double step = (x_end - x_start) / (rectClient.right -
rectClient.left);

    // Рассчитываем значение функции для начальной точки
    double y_next;
    // Перемещаемся к начальной точке
    MoveToEx(hdc, rectClient.left, int(-f(x_start) / step) +
rectClient.bottom / 2, NULL);

    while (x_current <= x_end)
    {
        x_current += step;
        if (abs(x_current + 2) <= 0.1) {
            x_current += step;

            MoveToEx(hdc, int(x_current / step) +
(rectClient.right - rectClient.left) / 2 + dlt, int(-
f(x_current) / step) + rectClient.bottom / 2, NULL);
            continue;
        }
        y_next = f(x_current);

        LineTo(hdc, int(x_current / step) + (rectClient.right -
rectClient.left) / 2 + dlt, int(-y_next / step) +
rectClient.bottom / 2);
    }

    // Выбор других перьев и шрифта
    SelectObject(hdc, hPenAxis);
    DrawAxis(hdc, rectClient);

    SelectObject(hdc, hPenGrid);
    DrawGrid(hdc, rectClient);

    SelectObject(hdc, hPenAxis);
    SelectObject(hdc, hFont);
    DrawText(hdc, rectClient);

    SelectObject(hdc, gdiOld);

    // Освобождение ресурсов
    DeleteObject(hPenAxis);
```

```
    DeleteObject(hPenGrid);
    DeleteObject(hFont);
}

// Рисование осей
void DrawAxis(HDC hdc, RECT rectClient)
{
    // Горизонтальная ось
    MoveToEx(hdc, 0, screenHeight / 2, NULL);
    LineTo(hdc, screenWidth, screenHeight / 2);

    // Вертикальная ось
    MoveToEx(hdc, screenWidth / 2, 0, NULL);
    LineTo(hdc, screenWidth / 2, screenHeight);

    // Добавляем метки и цифры на осях
    HFONT hFont = CreateFont(16, 0, 0, 0, FW_BOLD, FALSE, FALSE,
FALSE, DEFAULT_CHARSET, OUT_OUTLINE_PRECIS,
        CLIP_DEFAULT_PRECIS, CLEARTYPE_QUALITY, DEFAULT_PITCH |
FF_DONTCARE, L"Tahoma");
    SelectObject(hdc, hFont);
    SetTextColor(hdc, RGB(0, 0, 0));
    SetBkMode(hdc, TRANSPARENT);

    // Метки и цифры на горизонтальной оси
    for (int x = -15; x <= 15; x++)
    {
        int xPos = screenWidth / 2 + x * 40;
        MoveToEx(hdc, xPos, screenHeight / 2 - 5, NULL); //
Рисуем метки над осью
        LineTo(hdc, xPos, screenHeight / 2 + 5);

        WCHAR strX[16];
        wsprintf(strX, L"%d", x);

        RECT textRect;
        textRect.left = xPos - 35;
        textRect.right = xPos + 35;
        textRect.top = screenHeight / 2 + 10;
        textRect.bottom = screenHeight / 2 + 35;

        DrawText(hdc, strX, -1, &textRect, DT_CENTER);
    }

    // Метки и цифры на вертикальной оси
    for (int y = -5; y <= 5; y++)
    {
        int yPos = screenHeight / 2 - y * 40 * 1.65; // Изменено
здесь
        MoveToEx(hdc, screenWidth / 2 - 5, yPos, NULL); //
Рисуем метки слева от оси
        LineTo(hdc, screenWidth / 2 + 5, yPos);
```

```
            WCHAR strY[16];
            wsprintf(strY, L"%d", y);

            RECT textRect;
            textRect.left = screenWidth / 2 - 30;
            textRect.right = screenWidth / 2 - 10;
            textRect.top = yPos - 15;
            textRect.bottom = yPos + 15;

            DrawText(hdc, strY, -1, &textRect, DT_CENTER);
        }

        // Освобождение ресурсов
        DeleteObject(hFont);
}

// Рисование сетки
void DrawGrid(HDC hdc, RECT rectClient){
        int stepX = 40; // Шаг сетки
        int stepY = 40*1.67; // Шаг сетки

        // Вертикальные линии
        for (int x = screenWidth / 2 + stepX; x < screenWidth; x +=
stepX)
        {
            MoveToEx(hdc, x, 0, NULL);
            LineTo(hdc, x, screenHeight);
            MoveToEx(hdc, screenWidth - x, 0, NULL);
            LineTo(hdc, screenWidth - x, screenHeight);
        }

        // Горизонтальные линии
        for (int y = screenHeight / 2 + stepY; y < screenHeight; y
+= stepY)
        {
            MoveToEx(hdc, 0, y, NULL);
            LineTo(hdc, screenWidth, y);
            MoveToEx(hdc, 0, screenHeight - y, NULL);
            LineTo(hdc, screenWidth, screenHeight - y);
        }
}

// Рисование текста
void DrawText(HDC hdc, RECT rectClient)
{
        HBRUSH hBrush = CreateSolidBrush(RGB(255, 255, 255)); //
Белая кисть
        HGDIOBJ hOldBrush = SelectObject(hdc, hBrush);

        RECT rectWhiteBox;
        rectWhiteBox.left = screenWidth / 2 - 100; // Левая граница
        rectWhiteBox.top = 0; // Верхняя граница
```

```
    rectWhiteBox.right = screenWidth / 2 + 100; // Правая
граница
    rectWhiteBox.bottom = 50; // Нижняя граница

    FillRect(hdc, &rectWhiteBox, hBrush); // Заливка белым
цветом

    SelectObject(hdc, hOldBrush);
    DeleteObject(hBrush);

    SetTextColor(hdc, RGB(0, 0, 0));
    SetBkMode(hdc, TRANSPARENT);
    HFONT hFont = CreateFont(24, 0, 0, 0, FW_NORMAL, FALSE,
FALSE, FALSE, DEFAULT_CHARSET, OUT_OUTLINE_PRECIS,
        CLIP_DEFAULT_PRECIS, CLEARTYPE_QUALITY, DEFAULT_PITCH |
FF_DONTCARE, L"Arial");
    SelectObject(hdc, hFont);
    TextOut(hdc, screenWidth / 2 - 60, 10, L"y = (2x-1)/(x+2)",
20); // Вывод текста
    DeleteObject(hFont);
}
```

Результаты работа представлены на рисунке 1.2.

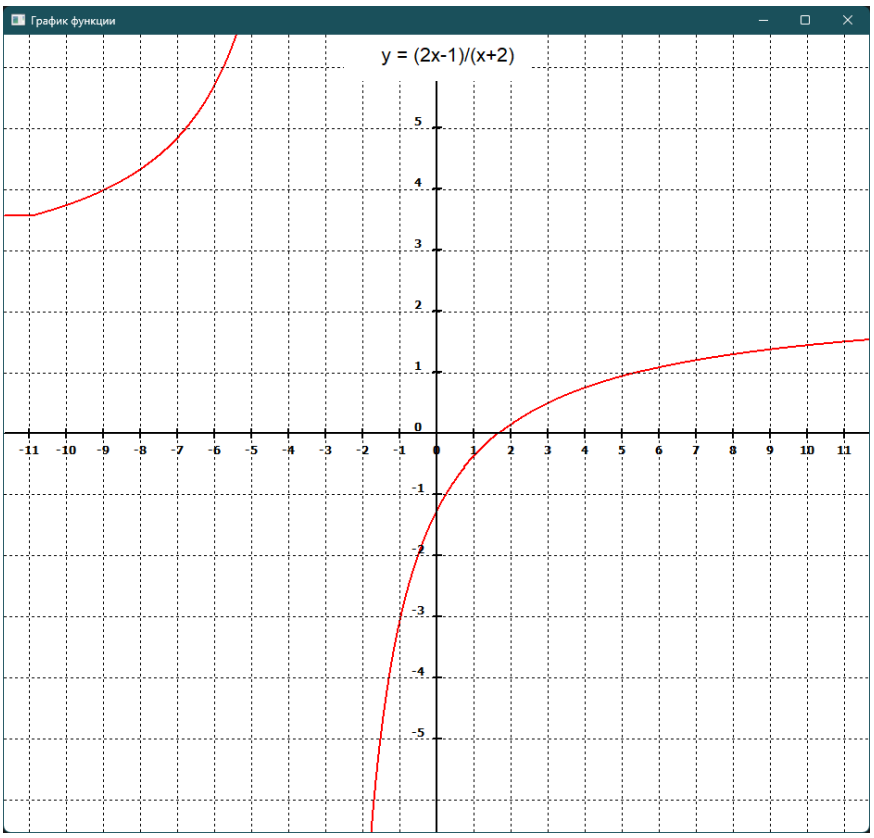

Рисунок 1.2 – Пример работы приложения

Список использованных источников

1 РД ФГБОУ ВО «КнАГТУ» 013-2016. Текстовые студенческие работы. Правила оформления. – Введ. 2016-03-10. – Комсомольск-на-Амуре: ФГБОУ ВО «КнАГТУ», 2016. – 55 с.