

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

**Лабораторная работа**

*ОКЭИ 09.02.07 7026 07 У*

Тема: *«Функции получения системной информации»*

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.	Гадилыев И. И.				Практическая работа						Лит.	Лист	Листов
											У	2	
											Отделение программирования		

## 1 Цель работы

Цель работы: получение практических навыков по программированию в Win32 API.

## 2 Скриншот структуры готового проекта

На рисунке 1 представлен скриншот структуры готового проекта.

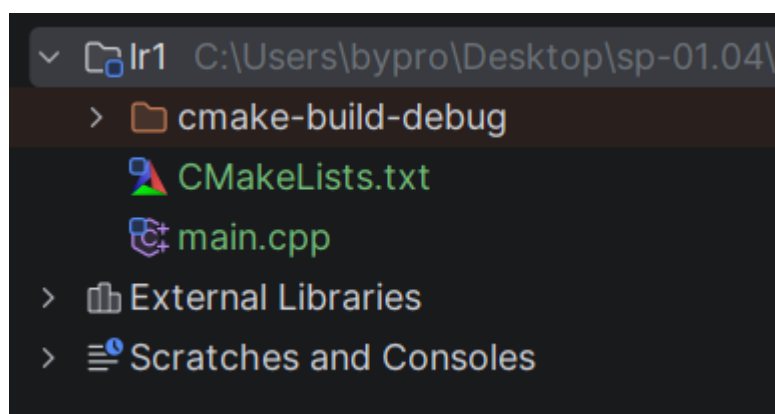


Рисунок 1 – Структура проекта

## 3 Листинг программы

Листинг файла main.cpp представлен ниже.

```
#include <windows.h>
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

string WideToUtf8(const wstring& wstr)
{
    if (wstr.empty()) return "";

    int sizeNeeded = WideCharToMultiByte(
        CP_UTF8,
        0,
        wstr.c_str(),
        (int)wstr.size(),
        nullptr,
        0,
        nullptr,
        nullptr
    );

    string result(sizeNeeded, 0);

    WideCharToMultiByte(
        CP_UTF8,
```

```

        0,
        wstr.c_str(),
        (int)wstr.size(),
        &result[0],
        sizeNeeded,
        nullptr,
        nullptr
    );

    return result;
}

string GetLastErrorAsString()
{
    DWORD errorMessageID = GetLastError();
    if (errorMessageID == 0)
        return "(no error)";

    LPWSTR messageBuffer = nullptr;

    size_t size = FormatMessageW(
        FORMAT_MESSAGE_ALLOCATE_BUFFER |
        FORMAT_MESSAGE_FROM_SYSTEM |
        FORMAT_MESSAGE_IGNORE_INSERTS,
        NULL,
        errorMessageID,
        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),
        (LPWSTR)&messageBuffer,
        0,
        NULL
    );

    wstring message(messageBuffer, size);
    LocalFree(messageBuffer);

    return WideToUtf8(message);
}

void PrintLine(const string& title)
{
    cout << "\n" << title << "\n\n";
}

int main()
{
    SetConsoleOutputCP(CP_UTF8);

    PrintLine("1) Имя компьютера и имя пользователя");

    WCHAR computerName[MAX_COMPUTERNAME_LENGTH + 1];
    DWORD size = MAX_COMPUTERNAME_LENGTH + 1;

    if (GetComputerNameW(computerName, &size))
        cout << "Имя компьютера: " << WideToUtf8(computerName) << "\n";
    else
        cout << "GetComputerNameW ошибка: " << GetLastErrorAsString() << "\n";

    WCHAR userName[256];
    DWORD userSize = 256;

    if (GetUserNameW(userName, &userSize))
        cout << "Имя пользователя: " << WideToUtf8(userName) << "\n";
    else
        cout << "GetUserNameW ошибка: " << GetLastErrorAsString() << "\n";

    PrintLine("2) Пути к системным каталогам Windows");

    WCHAR winDir[MAX_PATH];
    UINT winDirLen = GetWindowsDirectoryW(winDir, MAX_PATH);

```

```

if (winDirLen > 0)
    cout << "Каталог Windows: " << WideToUtf8(winDir) << "\n";
else
    cout << "GetWindowsDirectoryW ошибка: " << GetLastErrorAsString() << "\n";

WCHAR sysDir[MAX_PATH];
UINT sysDirLen = GetSystemDirectoryW(sysDir, MAX_PATH);
if (sysDirLen > 0)
    cout << "Системный каталог: " << WideToUtf8(sysDir) << "\n";
else
    cout << "GetSystemDirectoryW ошибка: " << GetLastErrorAsString() << "\n";

WCHAR tempPath[MAX_PATH];
DWORD tempLen = GetTempPathW(MAX_PATH, tempPath);
if (tempLen > 0)
    cout << "Каталог TEMP: " << WideToUtf8(tempPath) << "\n";
else
    cout << "GetTempPathW ошибка: " << GetLastErrorAsString() << "\n";

PrintLine("3) Версия операционной системы (GetVersionEx)");

OSVERSIONINFOW osvi{};
osvi.dwOSVersionInfoSize = sizeof(OSVERSIONINFOW);

if (GetVersionExW(&osvi))
{
    cout << "Major: " << osvi.dwMajorVersion << "\n";
    cout << "Minor: " << osvi.dwMinorVersion << "\n";
    cout << "Build: " << osvi.dwBuildNumber << "\n";
    cout << "PlatformId: " << osvi.dwPlatformId << "\n";
    cout << "Service Pack: " << WideToUtf8(osvi.szCSDVersion) << "\n";
}
else
{
    cout << "GetVersionExW ошибка: " << GetLastErrorAsString() << "\n";
}

PrintLine("4) Системные метрики (GetSystemMetrics)");

int screenW = GetSystemMetrics(SM_CXSCREEN);
int screenH = GetSystemMetrics(SM_CYSCREEN);
int mouseButtons = GetSystemMetrics(SM_CMOUSEBUTTONS);
int isMouseWheel = GetSystemMetrics(SM_MOUSEWHEELPRESENT);

cout << "Размер экрана: " << screenW << "x" << screenH << "\n";
cout << "Кнопки мыши: " << mouseButtons << "\n";
cout << "Колесо мыши есть? " << (isMouseWheel ? "Да" : "Нет") << "\n";

PrintLine("5) Системные параметры (SystemParametersInfo)");

BOOL dragFullWindows = FALSE;
if (SystemParametersInfoW(SPI_GETDRAGFULLWINDOWS, 0, &dragFullWindows, 0))
    cout << "SPI_GETDRAGFULLWINDOWS: " << (dragFullWindows ? "Да" : "Нет") <<
"\n";
else
    cout << "SPI_GETDRAGFULLWINDOWS ошибка: " << GetLastErrorAsString() << "\n";

BOOL fontSmoothing = FALSE;
if (SystemParametersInfoW(SPI_GETFONTSMOOTHING, 0, &fontSmoothing, 0))
    cout << "SPI_GETFONTSMOOTHING: " << (fontSmoothing ? "Включено" : "Выключено")
<< "\n";
else
    cout << "SPI_GETFONTSMOOTHING ошибка: " << GetLastErrorAsString() << "\n";

PrintLine("6) Системные цвета (GetSysColor)");

int element = COLOR_WINDOW;
COLORREF oldColor = GetSysColor(element);

```

```

int r = GetRValue(oldColor);
int g = GetGValue(oldColor);
int b = GetBValue(oldColor);

cout << "COLOR_WINDOW текущий: RGB(" << r << ", " << g << ", " << b << ") \n";

PrintLine("7) Функции для работы со временем");

SYSTEMTIME localTime{};
SYSTEMTIME systemTime{};

GetLocalTime(&localTime);
GetSystemTime(&systemTime);

cout << "LocalTime: "
    << localTime.wDay << "." << localTime.wMonth << "." << localTime.wYear
    << " " << setw(2) << setfill('0') << localTime.wHour
    << ":" << setw(2) << localTime.wMinute
    << ":" << setw(2) << localTime.wSecond << " \n";

cout << "SystemTime (UTC): "
    << systemTime.wDay << "." << systemTime.wMonth << "." << systemTime.wYear
    << " " << setw(2) << setfill('0') << systemTime.wHour
    << ":" << setw(2) << systemTime.wMinute
    << ":" << setw(2) << systemTime.wSecond << " \n";

DWORD tick = GetTickCount();
cout << "GetTickCount(): " << tick << " ms (время с запуска системы) \n";

PrintLine("8) Дополнительные API-функции");

cout << "Командная строка: " << WideToUtf8(GetCommandLineW()) << " \n";

UINT dblClick = GetDoubleClickTime();
cout << "GetDoubleClickTime(): " << dblClick << " ms \n";

SetLastError(0);
BOOL fake = DeleteFileW(L"Z:\\this_file_does_not_exist_12345.txt");
if (!fake)
{
    cout << "Пример ошибки DeleteFileW: " << GetLastErrorAsString() << " \n";
}

cin.get();

return 0;
}

```

## 4 Результаты работы программы

Результат выполнения программы показан на рисунках 2 и 3.

1) Имя компьютера и имя пользователя

Имя компьютера: T00WEE

Имя пользователя: bypro

2) Пути к системным каталогам Windows

Каталог Windows: C:\WINDOWS

Системный каталог: C:\WINDOWS\system32

Каталог TEMP: C:\Users\bypro\AppData\Local\Temp\

3) Версия операционной системы (GetVersionEx)

Major: 10

Minor: 0

Build: 26200

PlatformId: 2

Service Pack:

4) Системные метрики (GetSystemMetrics)

Размер экрана: 1536x864

Кнопок мыши: 5

Колесо мыши есть? Да

Рисунок 2 – Результат работы программы

					ОКЭИ 09.02.07 7026 07 У	Лист
Изм.	Лист	№ докум.	Подп.	Дата		7

```
5) Системные параметры (SystemParametersInfo)

SPI_GETDRAGFULLWINDOWS: Да
SPI_GETFONTSMOOTHING: Включено

6) Системные цвета (GetSysColor)

COLOR_WINDOW текущий: RGB(255, 255, 200)

7) Функции для работы со временем

LocalTime: 9.2.2026 01:27:52
SystemTime (UTC): 8.2.2026 20:27:52
GetTickCount(): 954847953 ms (время с запуска системы)

8) Дополнительные API-функции

Командная строка: C:\Users\bypro\Desktop\sp-01.04\lr1\cmake-build-debug\lr1.exe
GetDoubleClickTime(): 500 ms
Пример ошибки DeleteFileW: Системе не удастся найти указанный путь.
```

Рисунок 3 – Результат работы программы