Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Курский государственный университет»

Кафедра программного обеспечения и администрирования информационных систем

Направление подготовки математическое обеспечение и администрирование информационных систем

Форма обучения очная

**Отчет**

**по лабораторной работе №7**

«Введение в Windows API»

дисциплина «Операционные системы и оболочки»

Выполнил:

студент группы 313.1 Козявин М.С.

Проверил:

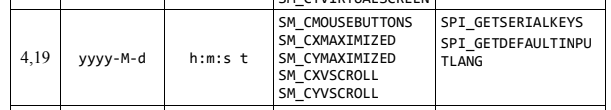
к.т.н., доцент кафедры ПОиАИС Кривонос А.В.

Курск, 2023

**Цели работы:**

Знакомство с понятием интерфейс прикладного программирования и получение практических навыков программирования для Windows на языке C/C++ с применением Win32 API.

**Вариант 4**

****

**Практическая часть**

1. Разработал в Visual C++ статическую библиотеку, в которой реализованы функции вывода в стандартный поток следующей информации:

* имя локального компьютера;
* доменное имя, назначенное локальному компьютеру;
* имя пользователя в текущем сеансе;
* полное имя пользователя в текущем сеансе.

**Исходный код задания 1:**

// 1.h

#pragma once

#if !defined (\_\_1\_H\_\_)

#define \_\_1\_H\_\_

void printComputerName();

void printComputerDomain();

void printUserName();

void printFullUserName();

#endif

// 1.cpp

#include <Windows.h>

#define SECURITY\_WIN32

#include <Security.h>

#include <iostream>

#include "1.h"

using namespace std;

void printComputerName() {

DWORD length = 100;

TCHAR buffer[100];

GetComputerName(buffer, &length);

wcout << L"Computer name: " << buffer << endl;

}

void printComputerDomain() {

DWORD length = 0;

TCHAR buffer[100];

GetComputerNameEx(ComputerNamePhysicalDnsHostname, NULL, &length);

GetComputerNameEx(ComputerNamePhysicalDnsHostname, buffer, &length);

wcout << L"Domain name: " << buffer << endl;

}

void printUserName() {

DWORD length = 100;

TCHAR buffer[100];

GetUserName(buffer, &length);

wcout << L"Username: " << buffer << endl;

}

void printFullUserName() {

DWORD length = 0;

TCHAR buffer[100];

GetUserNameEx(NameSamCompatible, NULL, &length);

GetUserNameEx(NameSamCompatible, buffer, &length);

wcout << L"Full username: " << buffer << endl;

}

1. Разработал в Visual C++ библиотеку DLL, в которой реализованы функции вывода в стандартный поток следующей информации:

* пути к системным каталогам;
* версия операционной системы;
* текущая дата и время (выводятся согласно варианту задания).

**Исходный код задания 2:**

// 2.h

#pragma once

#if !defined(\_\_2\_H\_\_)

#define \_\_2\_H\_\_

#ifdef SAMPLEDLL\_EXPORTS

#define SAMPLEDLL\_API \_\_declspec(dllexport)

#else

#define SAMPLEDLL\_API \_\_declspec(dllimport)

#endif

#ifdef \_\_cplusplus

extern "C" {

#endif

SAMPLEDLL\_API void printAllPaths();

SAMPLEDLL\_API void printWindowsVersion();

SAMPLEDLL\_API void printTime();

#ifdef \_\_cplusplus

}

#endif

#endif

// 2.cpp

#pragma warning(disable : 4996)

#include <Windows.h>

#include <ShlObj.h>

#include <iostream>

#include "2.h"

using namespace std;

SAMPLEDLL\_API void printAllPaths() {

TCHAR buffer[MAX\_PATH + 1];

SHGetFolderPath(NULL, CSIDL\_APPDATA, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_COMMON\_APPDATA, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_COMMON\_DOCUMENTS, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_HISTORY, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_INTERNET\_CACHE, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_LOCAL\_APPDATA, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_PERSONAL, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_PROGRAM\_FILES, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_PROGRAM\_FILES\_COMMON, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_SYSTEM, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

SHGetFolderPath(NULL, CSIDL\_WINDOWS, NULL, SHGFP\_TYPE\_DEFAULT, buffer);

wcout << buffer << endl;

}

SAMPLEDLL\_API void printWindowsVersion() {

OSVERSIONINFO osvi;

ZeroMemory(&osvi, sizeof(OSVERSIONINFO));

osvi.dwOSVersionInfoSize = sizeof(OSVERSIONINFO);

GetVersionEx((OSVERSIONINFO\*)&osvi);

wcout << "Major ver: " << osvi.dwMajorVersion << "\nMinor ver: " << osvi.dwMinorVersion << "\nBuild: " << osvi.dwBuildNumber << "\nPlatform: " << osvi.dwPlatformId << endl;

}

SAMPLEDLL\_API void printTime() {

TCHAR buffer[100];

TCHAR format[20] = L"ddd',' MMM dd yy";

GetDateFormatEx(LOCALE\_NAME\_SYSTEM\_DEFAULT, NULL, NULL, TEXT("yyyy'-'M'-'d"), buffer, 100, NULL);

wcout << buffer << endl;

GetTimeFormatEx(LOCALE\_NAME\_INVARIANT, NULL, NULL, TEXT("h':'m':'s t"), buffer, 100);

wcout << buffer << endl;

}

1. Изучил системные метрики и параметры, указанные в варианте задания.

**Метрики:**

SM\_CMOUSEBUTTONS – Число кнопок мыши, если мышь не подключена выводит 0.

SM\_CXMAXIMIZED – Ширина в пикселях развёрнутого на весь экран окна.

SM\_CYMAXIMIZED – Высота в пикселях развёрнутого на весь экран окна.

SM\_CXVSCROLL – Ширина в пикселях вертикальной полосы прокрутки.

SM\_CYVSCROLL – Высота в пикселях бокового маркера-стрелки вертикальной полосы прокрутки.

**Параметры:**

SPI\_GETSERIALKEYS – Параметры последовательного порта. Данные параметры поддерживаются только на ОС Windows 95/98.

SPI\_GETDEFAULTINPUTLANG – Языковые параметры, настроенные в системном вводе по умолчанию.

1. Разработал в Visual C++ библиотеку DLL, в которой реализованы функции вывода в стандартный поток значений системных метрик, изученных в п. 3.

**Исходный код задания 4:**

// 3.h

#pragma once

#if !defined(\_\_3\_H\_\_)

#define \_\_3\_H\_\_

#ifdef DLL3\_EXPORTS

#define DLL3\_API \_\_declspec(dllexport)

#else

#define DLL3\_API \_\_declspec(dllimport)

#endif

#ifdef \_\_cplusplus

extern "C" {

#endif

DLL3\_API void printMetrics();

#ifdef \_\_cplusplus

}

#endif

#endif

// 3.cpp

#include <Windows.h>

#include <iostream>

#include "3.h"

using namespace std;

DLL3\_API void printMetrics() {

int n = GetSystemMetrics(SM\_CMOUSEBUTTONS);

wcout << "Number of mouse buttons: " << n << endl;

n = GetSystemMetrics(SM\_CXMAXIMIZED);

wcout << "Width of a maximized window: " << n << endl;

n = GetSystemMetrics(SM\_CYMAXIMIZED);

wcout << "Height of a maximized window: " << n << endl;

n = GetSystemMetrics(SM\_CXVSCROLL);

wcout << "Width of a vertical scroll bar: " << n << endl;

n = GetSystemMetrics(SM\_CYVSCROLL);

wcout << "Height of the arrow bitmap on a vertical scroll bar: " << n << endl;

}

1. Разработал в Visual C++ библиотеку DLL, в которой реализованы функции вывода в стандартный поток значений системных параметров, изученных в п. 3.

**Исходный код задания 5:**

// 4.h

#pragma once

#if !defined(\_\_4\_H\_\_)

#define \_\_4\_H\_\_

#ifdef DLL4\_EXPORTS

#define DLL4\_API \_\_declspec(dllexport)

#else

#define DLL4\_API \_\_declspec(dllimport)

#endif

#ifdef \_\_cplusplus

extern "C" {

#endif

DLL4\_API void printParameters();

#ifdef \_\_cplusplus

}

#endif

#endif

// 4.cpp

#include <Windows.h>

#include <iostream>

#include "4.h"

using namespace std;

int main() {

printParameters();

}

DLL4\_API void printParameters() {

SERIALKEYS param1;

ZeroMemory(&param1, sizeof(SERIALKEYS));

param1.cbSize = sizeof(SERIALKEYS);

SystemParametersInfo(SPI\_GETSERIALKEYS, sizeof(SERIALKEYS), &param1, 0);

wcout << "State of serial port (only on Win95/98): " << param1.iPortState << endl;

HKL param2;

SystemParametersInfo(SPI\_GETDEFAULTINPUTLANG, sizeof(HKL), &param2, 0);

wcout << "Input locale identifier for the system default input language: " << param2 << endl;

}

1. Разработал в Visual C++ консольное приложение Win32, в котором используются функции, реализованные в следующих библиотеках:
   * статическая библиотека, созданная в п.1;
   * DLL, созданная в п.2 (использовать неявное подключение);
   * DLL, созданная в п.4 (использовать явное подключение);
   * DLL, созданная в п.5 (использовать отложенную загрузку).

**Исходный код задания 5:**

#pragma comment(lib, "1.lib")

#pragma comment(lib, "2.lib")

#include "1.h"

#include "2.h"

#include "4.h"

#include <iostream>

#include <Windows.h>

#include <delayimp.h>

using namespace std;

typedef void (\*PRINT\_METRICS)();

int main() {

cout << "Task 1" << endl;

printComputerName();

printComputerDomain();

printUserName();

printFullUserName();

cout << endl << "Task 2" << endl;

printAllPaths();

printWindowsVersion();

printTime();

cout << endl << "Task 3" << endl;

HMODULE hDLL = LoadLibrary(TEXT("3.dll"));

if (NULL != hDLL) {

PRINT\_METRICS func = (PRINT\_METRICS)GetProcAddress(hDLL, "printMetrics");

if (NULL != func) func();

FreeLibrary(hDLL);

}

cout << endl << "Task 4" << endl;

printParameters();

\_\_FUnloadDelayLoadedDLL2("4.dll");

}

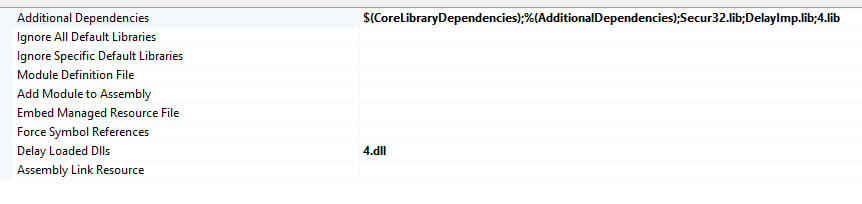


Рисунок 1 – Настройки проекта 1

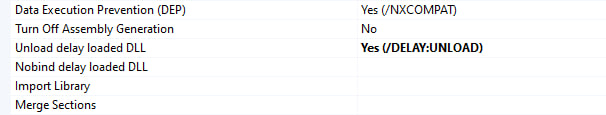


Рисунок 2 – Настройки проекта 2

1. Протестировал работу приложения, созданного в п. 6, на компьютере под управлением Windows XP (или выше). Результаты тестирования отразил в отчете.

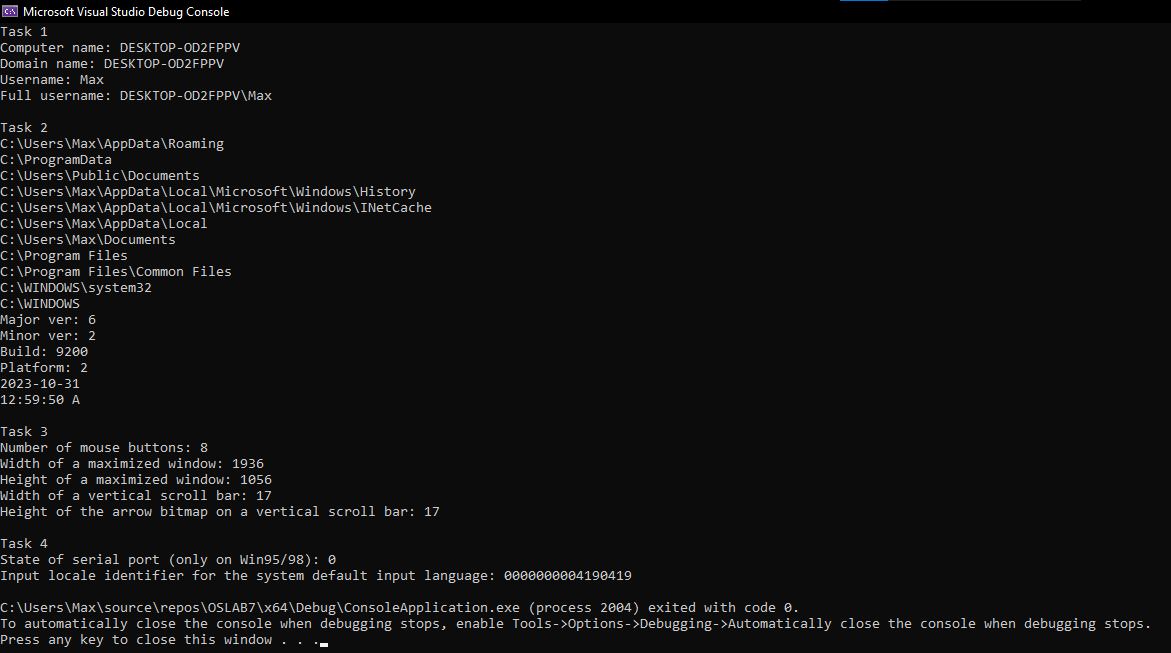


Рисунок 3 – Тест работы приложения

1. Включил в отчет исходный программный код и выводы о проделанной работе.

**Вывод**

В ходе лабораторной работы я ознакомился с понятием интерфейс прикладного программирования и получил практических навыков программирования для Windows на языке C++ с применением Win32 API. А также научился работать с DLL и LIB библиотеками.