МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Полоцкий государственный университет»

Факультет информационных технологий Кафедра технологий программирования

# Лабораторная работа №14

по дисциплине: «Системное программирование»

на тему:

«Перехват функций ОС Windows» Вариант №12

Выполнил Студент группы 21-ИТ-1

Чиникайло А.П.

Проверил Преподаватель

Сергеев М.А.

Полоцк 2023

**Задания**

Установить фильтрующую функцию хука WH\_CBT.

**Задание:**#include <windows.h>

#include <iostream>

using namespace std;

LRESULT CALLBACK FilterProc(int nCode, WPARAM wParam, LPARAM lParam) {

if (nCode >= 0) {

switch (nCode)

{

case HCBT\_ACTIVATE:

cout << "HCBT\_ACTIVATE" << endl;

break;

case HCBT\_CLICKSKIPPED:

cout << "HCBT\_CLICKSKIPPED" << endl;

break;

case HCBT\_CREATEWND:

cout << "HCBT\_CREATEWND" << endl;

break;

case HCBT\_DESTROYWND:

cout << "HCBT\_DESTROYWND" << endl;

break;

case HCBT\_KEYSKIPPED:

cout << "HCBT\_KEYSKIPPED" << endl;

break;

case HCBT\_MINMAX:

cout << "HCBT\_MINMAX" << endl;

break;

case HCBT\_MOVESIZE:

cout << "HCBT\_MOVESIZE" << endl;

break;

case HCBT\_QS:

cout << "HCBT\_QS (WM\_QUEUESYNC)" << endl;

break;

case HCBT\_SETFOCUS:

cout << "HCBT\_SETFOCUS" << endl;

break;

case HCBT\_SYSCOMMAND:

cout << "HCBT\_SYSCOMMAND" << endl;

break;

default:

cout << "Default (kak??????)" << endl;

break;

}

}

return CallNextHookEx(NULL, nCode, wParam, lParam);

}

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {

switch (uMsg) {

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hwnd, uMsg, wParam, lParam);

}

return 0;

}

int main() {

WNDCLASS windowClass = {};

windowClass.lpfnWndProc = WindowProc;

windowClass.hInstance = GetModuleHandle(NULL);

windowClass.lpszClassName = L"HookedWindowClass";

RegisterClass(&windowClass);

HWND HookedWindow = CreateWindowEx(

0,

L"HookedWindowClass",

L"Hooked window",

WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, CW\_USEDEFAULT, 200, 200,

NULL,

NULL,

GetModuleHandle(NULL),

NULL

);

if (HookedWindow == NULL) {

cerr << "Failed to create window!" << endl;

return 1;

}

cout << "Hooked Window created sucessfully!" << endl;

HHOOK \_hook = SetWindowsHookEx(WH\_CBT, FilterProc, NULL, GetCurrentThreadId());

if (\_hook == NULL) {

DWORD error = GetLastError();

cerr << "Failed to install hook! Error code: " << error << endl;

return 1;

}

cout << "Hook was set sucessfully for: " << GetCurrentThreadId() << endl;

ShowWindow(HookedWindow, SW\_SHOWNORMAL);

UpdateWindow(HookedWindow);

cout << "Waiting for messages!" << endl;

MSG msg;

while (GetMessage(&msg, NULL, 0, 0)) {

TranslateMessage(&msg);

DispatchMessage(&msg);

}

UnhookWindowsHookEx(\_hook);

return 0;

}

**Вывод:** в результате выполнения данной лабораторной работы были изучены способы создания хуков для оконных приложений и был написан хук согласно варианту задания