БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

ИИТ БГУИР

Программное обеспечение информационных технологий.

Дисциплина: «Сетевые технологии и администрирование операционных систем»

Отчет

По лабораторной работе №7

«Driver»

Выполнил студент группы 681072: Иванов К.Е.

Проверил преподаватель: Калитеня И.Л.

Минск 2019

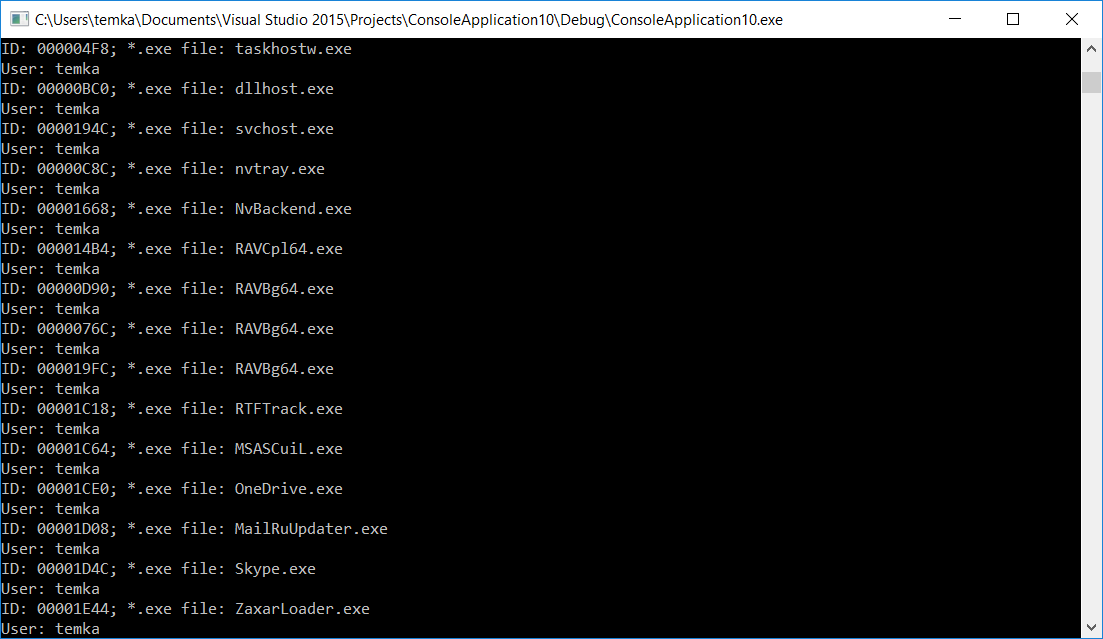
**Цель:**

Разработать программу, которая запускает приложение, для отслеживания процессов и потоков.

**Задание:**

Написать программу, выводящую на экран список запущенных процессов и потоков и их атрибутов.

**Скриншоты работающего приложения:**



**Код приложения:**

#include "stdafx.h"

#include "CONIO.H"

#include "stdafx.h"

#include <windows.h>

#include <tlhelp32.h>

#define MAX\_NAME 256

char\* getProcessUsername(HANDLE hSnapshot, PROCESSENTRY32 processList)

{

DWORD dwSize = 256;

HANDLE hProcess;

TOKEN\_USER \*pUserInfo;

static char staticName[MAX\_NAME];

char name[MAX\_NAME];

char domain[MAX\_NAME];

char \*result;

int iUse;

hProcess = OpenProcess(PROCESS\_ALL\_ACCESS, FALSE, processList.th32ProcessID);

OpenProcessToken(hProcess, TOKEN\_QUERY, &hSnapshot);

pUserInfo = (PTOKEN\_USER)GlobalAlloc(GPTR, dwSize);

GetTokenInformation(hSnapshot, TokenUser, pUserInfo, dwSize, &dwSize);

LookupAccountSidA(0, pUserInfo->User.Sid, name, &dwSize, domain, &dwSize, (PSID\_NAME\_USE)&iUse);

strncpy\_s(staticName, name, \_TRUNCATE);

//printf( "%s\n", lpDomain, lpName );

result = staticName;

return result;

}

VOID printProcessList(HANDLE CONST hStdOut)

{

PROCESSENTRY32 processList;

TCHAR szBuff[1024];

DWORD dwTemp;

HANDLE CONST hSnapshot = CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS, 0);

HANDLE hProcess;

HANDLE hCurrentProcess;

processList.dwSize = sizeof(PROCESSENTRY32);

Process32First(hSnapshot, &processList);

while (Process32Next(hSnapshot, &processList))

{

wsprintf(szBuff, L"ID: %08X; \*.exe file: %s\r\n",

processList.th32ProcessID, processList.szExeFile);

WriteConsole(hStdOut, szBuff, lstrlen(szBuff), &dwTemp, 0);

printf("User: %s \n", getProcessUsername(hSnapshot, processList));

hProcess = OpenProcess(PROCESS\_ALL\_ACCESS, FALSE, processList.th32ProcessID);

hCurrentProcess = GetCurrentProcess();

if (hProcess == hCurrentProcess)

{

printf(" <-- It's my process");

}

// getProcessUsername();

// printf("\n");

}

CloseHandle(hSnapshot);

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

HANDLE CONST hStdOut = GetStdHandle(STD\_OUTPUT\_HANDLE);

printProcessList(hStdOut);

getchar();

ExitProcess(0);

getchar();

return 0;

getchar();

}